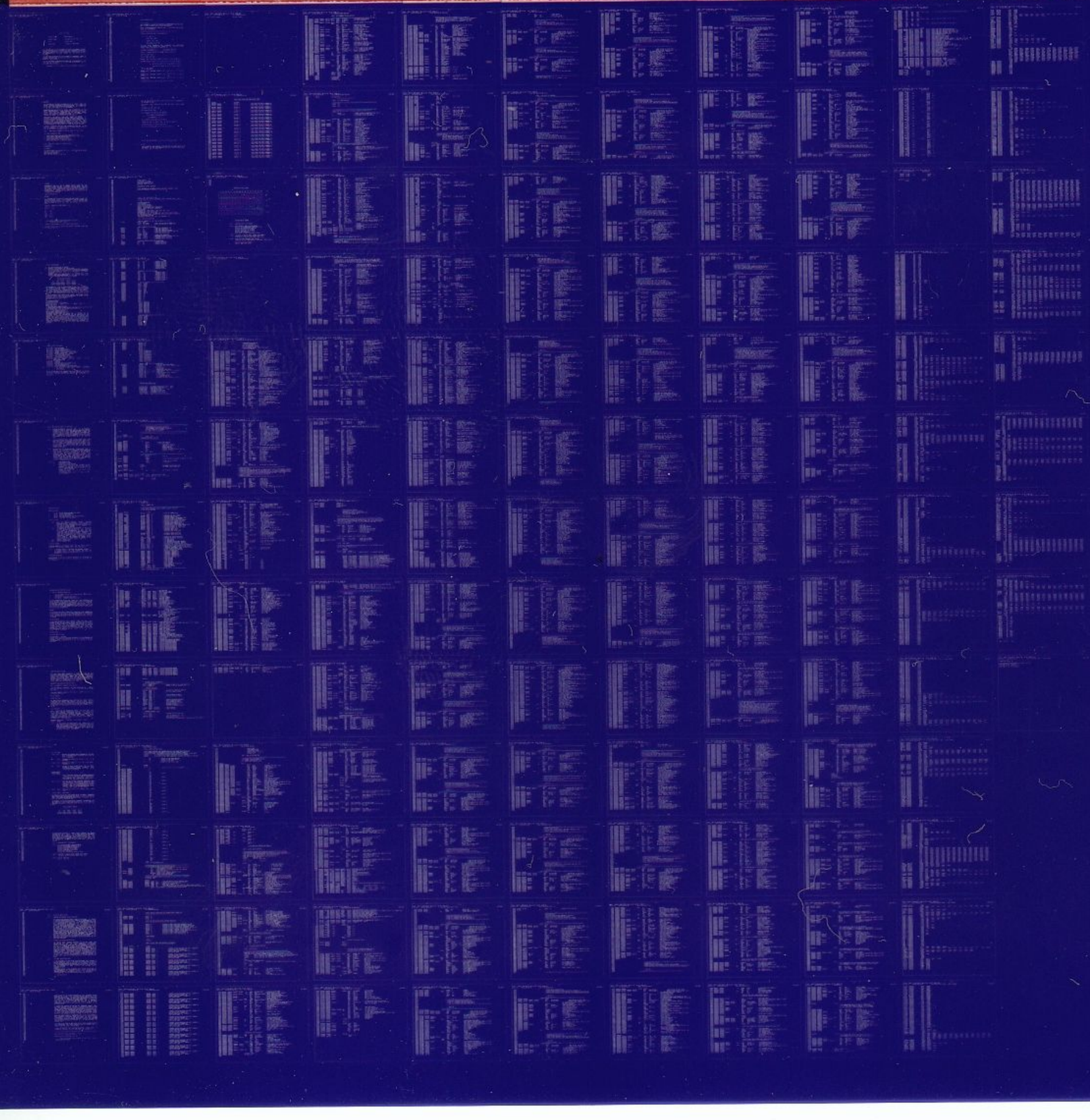
KMC11-A BITSTUFF MD L TSTS AH-A914B-MC
CZKCFB0 FICHE 1 OF 1



```
1                                   .REM    $
2
3
4
5
6
7
8
9
10
11
12
13
14                                                  IDENTIFICATION
15
16
17                                          PRODUCT CODE:           AC-A913B-MC
18
19                                          PRODUCT NAME:           CZKCFB0 BITSTUFF MD L TSTS
20
21                                          DATE:                   AUGUST 1980
22
23                                          MAINTAINER:             DIAGNOSTICS-MERRIMACK
24
25
26
27
28                                  THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE  WITHOUT
29                                  NOTICE  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
30                                  EQUIPMENT CORPORATION.  DIGITAL EQUIPMENT CORPORATION  ASSUMES
31                                  NO  RESPONSIBILITY  FOR  ANY  ERRORS  THAT  MAY APPEAR IN THIS
32                                  DOCUMENT.
33
34                                  THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED  UNDER  A
35                                  LICENSE  AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE
36                                  TERMS OF SUCH LICENSE.
37
38                                  DIGITAL EQUIPMENT CORPORATION ASSUMES  NO  RESPONSIBILITY  FOR
39                                  THE  USE  OR  RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS
40                                  NOT SUPPLIED BY DIGITAL.
41
42                                       COPYRIGHT (C) 1977, 1980 BY DIGITAL EQUIPMENT CORPORATION
```

1. ABSTRACT

THE FUNCTION OF THE KMC11 DIAGNOSTICS IS TO VERIFY THAT THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS VERFIY THAT THERE ARE NO MALFUNCTIONS AND THE ALL OPERATIONS OF THE KMC11 ARE CORRECT IN ITS ENVIRONMENT.

PARAMETERS MUST BE SET UP TO ALERT THE DIAGNOSTICS TO THE KMC11 CONFIGURATION. THESE PARAMETERS ARE CONTAINED IN THE STATUS TABLE AND ARE GENERATED IN TWO WAYS: 1) MANUAL INPUT - THE OPERATOR ANSWERS QUESTIONS. 2) AUTOSIZING - THE PROGRAM DETERMINES THE PARAMETERS AUTOMATICALLY.

CZKCF TESTS THE KMC-11 LINE UNIT (M8201 OR M8202). IT PERFORMS WRITE/READ TESTS ON THE KMC LINE UNIT REGISTERS. IT CHECKS FOR PROPER TRANSMITTER, RECEIVER, AND BCC OPERATION IN BITSTUFF MODE. THE MODEM SIGNALS ARE ALSO CHECKED. DZKCF REQUIRES A KMC MICRO-PROCESSOR (M8204) TO RUN. FOR BEST DIAGNOSIS A TURN-AROUND CONNECTOR SHOULD BE INSTALLED, HOWEVER THE DIAGNOSTIC WILL RUN WITHOUT IT (SOME TESTS ARE SKIPPED).

CURRENTLY THERE ARE FOUR OFF LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO INSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE.

NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE FOUR DIAGNOSTICS ARE:

1. DZKCC [REV] BASIC W/R AND MICRO-PROCESSOR TESTS
2. DZKCD [REV] JUMP AND MAIN MEMORYTESTS (HEAT TEST TAPE)
3. DZKCE [REV] DDCMP LINE UNIT TESTS
4. DZKCF [REV] BITSTUFF LINE UNIT TESTS
5. DZKCA [REV] KMC11 CPU MICRO-DIGNOSTICS.

NOTE: NAMES MAY CHANGE AS UPDATES OCCUR.

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (EXCEPT AN LSI-11) WITH MINIMUM 8K MEMORY
ASR 33 (OR EQUILIVALENT)
KMC11-AN IOP (M8204)
KMC11-DA OR KMC11-MD OR KMC11-MA

```
    92
    93
    94                          2.2     STORAGE
    95
    96                                  PROGRAM WILL USE  ALL  8K  OF  MEMORY  EXCEPT  WHERE  ABL  AND
    97                                  BOOTSTRAP  LOADER  RESIDE.  LOCATIONS 2100 THRU 2300;  CONTAIN
    98                                  THE "STATUS TABLE" INFORMATION WHICH IS GENERATED AT START  OF
    99                                  DIAGNOSTICS  BY  MANUAL  INPUT  (QUESTIONS)  OR  AUTOMATICALLY
   100                                  (AUTO-SIZING).  THIS AREA IS AN OVERLAY AREA AND SHOULD NOT BE
   101                                  ALTERED BY THE OPERATOR.
   102
   103                          3       LOADING PROCEEDURE
   104
   105                          3.1     METHOD
   106
   107                                  ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED  USING  THE
   108                                  ABSOLUTE LOADER.  NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH
   109                                  AS DISK ,MAGTAPE,DECTAPE, OR  CASSETTE;  FOLLOW  INSTRUCTIONS
   110                                  FOR  THE  MONITOR  WHICH  HAS  BEEN  PROVIDED ON THAT SPECIFIC
   111                                  MEDIA.
   112
   113                                  ABSOLUTE LOADER STARTING ADDRESS *500
   114
   115                                  MEMORY  * SIZE
   116
   117                                  4K      17
   118                                  8K      37
   119                                  12K     57
   120                                  16K     77
   121                                  20K     117
   122                                  24K     137
   123                                  28K     157
   124
   125
   126                          3.1.1   PLACE ADDRESS OF ABS LOADER INTO SWITCH REGISTER.
   127                                          (ALSO PLACE 'HALT' SW UP)
   128
   129                          3.1.2   DEPRESS 'LOAD ADDRESS' KEY ON CONSOLE AND RELEASE.
   130
   131                          3.1.3   DEPRESS 'START KEY' ON CONSOLE AND RELEASE (PROGRAM SHOULD NOW
   132                                  BE LOADING INTO CPU)
   133
   134
```

4. STARTING PROCEEDURE

A. SET SWITCH REGISTER TO 000200
B. DEPRESS 'LOAD ADDRESS' KEY AND RELEASE
C. SET SWR TO ZERO FOR 'AUTO SIZING' OR SWR BIT0=1 FOR MANUAL INPUT (QUESTIONS) OR SWR BIT7=1 TO USE EXISTING PARAMETERS SET UP BY A PREVIOUS START OR A PREVIOUSLY RUN KMC11 DIAGNOSTIC.
D. DEPRESS 'START KEY' AND RELEASE. THE PROGRAM WILL TYPE MAINDEC NAME AND PROGRAM NAME (IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING:

MAP OF KMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
002100	160010	045310	177777	000000
002110	160020	045320	177777	000000

THE PROGRAM WILL TYPE 'R' AND PROCEED TO RUN THE DIAGNOSTIC. THE ABOVE IS ONLY AN EXAMPLE. THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 2100 IN THE PROGRAM. IN THIS EXAMPLE THE TABLE CONTAINS THE INFORMATION AND STATUS OF TWO KMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.
IF THE DIAGNOSTIC WAS STARTED WITH SW00=1 INDICATING MANUAL PARAMETER INPUT THEN THE FOLLOWING SHOWS AN EXAMPLE OF THE QUESTIONS ASKED AND SOME EXAMPLE ANSWERS:

```
HOW MANY KMC11'S TO BE TESTED?1
01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL?  (4,5,6,7)?5
WHICH LINE UNIT?  IF NONE TYPE "N",  IF  M8201  TYPE  "1",  IF
M8202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
(IF M8201 AND LOOP BACK CONNECTOR ON THEN YOU GET THE NEXT QUESTION)
WHICH MODEM TYPE, TYPE "D" FOR KMC11-DA (RS232),OR
TYPE "F" FOR KMC11-FA (V.35) ? D
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377
```

FOLLOWING THE QUESTIONS THE STATUS MAP IS PRINTED OUT AS DESCRIBED ABOVE, THE INFORMATION IN THE MAP REFLECTS THE ANSWERS TO THE QUESTIONS. IF THE DIAGNOSTIC WAS STARTED WITH SW00=0 AND SW07=0 (AUTO-SIZING) THEN NO QUESTIONS ARE ASKED AND ONLY THE STATUS-MAP IS PRINTED OUT. IF AUTO-SIZING IS USED THE STATUS INFORMATION MUST BE VERIFIED TO BE CORRECT (MATCH THE HARDWARE). IF IT DOES NOT MATCH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED WITH SW00=1 AND THE QUESTIONS ANSWERED.

```
188
189
190                          4.1     CONTROL SWITCH SETTINGS
191
192                                  SW 15 SET:  HALT ON ERROR
193                                  SW 14 SET:  LOOP ON CURRENT TEST
194                                  SW 13 SET:  INHIBIT ERROR PRINT OUT
195                                  SW 12 SET:  INHIBIT TYPE OUT ABELL ON ERROR.
196                                  SW 11 SET:  INHIBIT ITERATIONS.  (QUICK PASS)
197                                  SW 10 SET:  ESCAPE TO NEXT TEST ON ERROR
198                                  SW 09 SET:  LOOP WITH CURRENT DATA
199                                  SW 08 SET:  CATCH ERROR AND LOOP ON IT
200                                  SW 07 SET:  USE PREVIOUS STATUS TABLE.
201                                  SW 06 SET:  HALT IN    ROMCLK    ROUTINE    BEFORE    CLOCKING
202                                              MICRO-PROCESSOR
203                                  SW 05 SET:  RESERVED
204                                  SW 04 SET:  RESERVED
205                                  SW 03 SET:  RESELECT KMC11'S DESIRED ACTIVE
206                                  SW 02 SET:  LOCK ON SELECTED TEST
207                                  SW 01 SET:  RESTART PROGRAM AT SELECTED TEST
208                                  SW 00 SET:  BUILD NEW STATUS TABLE FROM QUESTIONS.  (IF SW07=0
209                                                AND  SW00=0  A  NEW  STATUS  TABLE  IS  BUILT BY
210                                                AUTO-SIZING)
211
212                                  SWITCH 06 AND 08-15 ARE DYNAMIC AND CAN BE CHANGED  AS  NEEDED
213                                  WHILE THE DIAGNOSTIC IS RUNNING.  SWITCHES 00-03 AND SWITCH 07
214                                  ARE STATIC, AND ARE USED ONLY ON STARTING  OR  RESTARTING  THE
215                                  DIAGNOSTIC.
```

4.1.2 SWITCH REGISTER OPTIONS (AT START UP)

SW 01 RESTART PROGRAM AT SELECTED TEST. IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A TEST, THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS. WHEN THIS SWITCH IS USED THE DIAGNOSTIC WILL ASK TEST NO.? ANSWER BY TYPING THE NUMBER OF THE TEST DESIRED AND CARRIGE RETURN TO BEGIN EXECUTION AT THE SELECTED TEST.

SW 02 LOCK ON SELECTED TEST. THIS SWITCH WHEN USED WITH SW01 WILL CAUSE THE PROGRAM TO CONSTANTLY LOOP ON THE SELECTED TEST. HITTING ANY KEY ON THE CONSOLE WILL LET IT ADVANCE TO THE NEXT TEST AND LOOP UNTIL A KEY IS HIT AGAIN. IF SW02=0 WHEN SW01 IS USED. THE PROGRAM WILL BEGIN AT THE SELECTED TEST AND CONTINUE NORMAL OPERATIONS.

SW 03 RESELECT KMC11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO KMC11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS FOUR KMC11S; BITS 00,01,02,03 WILL BE SET IN LOC 'KMACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW00) ALTERS THAT LOCATION;THEREFORE IF FOUR KMC11S ARE IN THE SYSTEM ***DO NOT*** SET SWITCHS GREATER THAN SW 03 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE KMC11S THAN THERE IS INFORMATION ON IN THE STATUS TABLE.

METHOD:
- A: LOAD ADDRESS 200
- B: START WITH SW 00=1
- C: PROGRAM WILL TYPE MESSAGE
- D: SET A SWITCH FOR EACH KMC DESIRED ACTIVE. EXAMPLE: IF YOU HAVE 4 KMC'S BUT ONLY WANT TO RUN THE FIRST AND THE LAST SET SWR BITS 0 AND 3 = 1. PRESS CONTINUE
- E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)
- F: SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST(ON ERROR).
5. SW 10 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW06 HALT IN ROMCLK ROUTINE BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION. THIS ALLOWS THE OPERATOR TO SCOPE A MICRO-PROCESSOR INSTRUCTION IN THE STATIC STATE BEFORE IT IS CLOCKED. HIT CONTINUE TO RESUME RUNNING.
2. SW09 (IF ENABLED BY 'SCOP1') ON AN ERROR; IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. (EX. *TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS USUALLY THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0). IF SW09 IS NOT ENABELED; AND THERE IS A HARD ERROR (CONSTANT); SW08 IS BEST. (SW14=1,0, SW10=0, SW09=0, SW08=1). FOR INTERMITTEMT ERRORS; SW14=1 WILL LOOP ON TEST REGUARDLESS OF ERROR OR NOT ERROR. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 INHIBIT INTERATIONS.
4. SW14 LOOP ON CURRENT TEST.

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200 THERE ARE NO OTHER STARTING ADDRESSES FOR THE KMC11 DIAGNOSTICS. (SEE SECTION 4.0)

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY AFTER ALL AVAILABLE KMC11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION 4.0 WILL BE PRINTED, AND PROGRAM WILL BEGIN RUNNING THE DIAGNOSTIC

```
307
308
309                                 5.2     PROGRAM AND/OR OPERATOR ACTION
310
311                                         THE TYPICAL APPROACH SHOULD BE
312
313                                         1.      HALT ON ERROR (VIA SW 15=1) WHEN EVER AN ERROR OCCURS.
314                                         2.      CLEAR SW 15.
315                                         3.      SET SW 14:  (LOOP ON THIS TEST)
316                                         4.      SET SW 13:  (INHIBIT ERROR PRINT OUT)
317
318                                         THE TEST NUMBER AND PC WILL BE  TYPED  OUT  AND  POSSIBILY  AN
319                                         ERROR  MESSAGE (THIS DEPENDS ON THE TEST) TO GIVE THE OPERATOR
320                                         AN IDEA AS TO THE SOURCE OF THE PROBLEM.  IF IT  IS  NECESSARY
321                                         TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT;  LOOK IN
322                                         THE LISTING FOR THAT TEST NUMBER WHICH WAS TYPED OUT AND  THEN
323                                         NOTE THE PC OF THE ERROR REPORT THIS WAY THE EXACT FUNCTION OF
324                                         THE TEST CAN BE DETERMINED.
325
326                                 6.      ERRORS
327
328                                         AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE A TEST NUMBER AND
329                                         PC TYPED OUT AT THE TIME OF AN ERROR (PROVIDING SW 13=0 AND Sw
330                                         12=0).  IN MOST CASES ADDITIONAL INFORMATION WILL BE  SUPPLIED
331                                         IN THE THE ERROR MESSAGE TO GIVE THE OPERATOR AN INDICATION OF
332                                         THE ERROR.
333
334                                 6.2     ERROR RECOVERY
335
336                                         IF FOR SOME REASON THE  KMC11  SHOULD  'HANG  THE  BUS'  (GAIN
337                                         CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED)
338                                         AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR  TO  REGAIN
339                                         CONTROL  OF  CPU.   IF  THIS  SHOULD HAPPEN;  LOOK IN LOCATION
340                                         'TSTNM' (ADDRESS 1202)FOR THE NUMBER OF  THE  TEST  THAT  WAS
341                                         RUNNING  AT  THE  TIME OF THE CATASTROPHIC ERROR.  IN THIS WAY
342                                         THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE KMC11 WAS  DOING
343                                         AT THE TIME OF THE ERROR.
344
345                                 7.      RESTRICTIONS
346
347                                 7.1     STARTING RESTRICTIONS
348
349                                         SEE SECTION 4.  (PLEASE)
350                                         STATUS TABLE SHOULD BE VERIFIED REGUARDLESS OF HOW PROGRAM WAS
351                                         STARTED.   ALSO IT IS IMPORTANT TO USE THIS LISTING ALONG WITH
352                                         THE INFORMATION  PRINTED  ON  THE  TTY  TO  COMPLETLY  ISOLATE
353                                         PROBLEMS.
354
```

```
355
356
357                          7.2     OPERATING RESTRICTIONS
358
359                                  THE FIRST TIME A KMC11 DIAGNOSTIC IS LOADED INTO CORE AND  RUN
360                                  THE STATUS TABLE MUST BE SET UP.  THIS IS DONE BY MANUAL INPUT
361                                  (SW00=1) OR BY AUTOSIZING  (SW00=0  AND  SW07=0).  THEREAFTER
362                                  HOWEVER  THE  STATUS  TABLE  NEED  NOT  BE SETUP BY SUBSEQUENT
363                                  RESTARTS OR EVEN LOADING THE NEXT KMC DIAGNOSTIC  BECAUSE  THE
364                                  STATUS  TABLE  IS  OVERLAYED.  THE  CURRENT PARAMETERS IN THE
365                                  STATUS TABLE ARE USED WHEN SW07=1 ON START UP.
366
367                          7.3     HARDWARE CONFIGURATION RESTRICTIONS
368
369                                  KMC11 IOP(M8204)- JUMPER W1 MUST BE IN,
370
371                                  LINE UNIT(M8201)- JUMPERS W1, W2, AND W4 MUST BE IN.  JUMPERS
372                                  W3,  AND  W5  MUST  BE  OUT.  SW8  OF  E26  MUST BE IN THE ON
373                                  POSITION.
374
375                                  LINE UNIT (M8202)- JUMPER W1 MUST BE IN.  SW8 OF E26  MUST  BE
376                                  IN THE OFF POSITION.
377
378                          8.      MISCELLANEOUS
379
380                          8.1     EXECUTION TIME
381
382                                  ALL KMC11 DEVICE DIAGNOSTICS WILL GIVE AN 'END  PASS'  MESSAGE
383                                  (PROVIDING  NO  ERRORS  AND  SW12=0)  WITHIN  4 MINS.  THIS IS
384                                  ASSUMING SW11=1 (DELETE ITERATIONS) IS SET TO GIVE THE FASTEST
385                                  POSSIBLE EXECUTION.  THE ACTUAL EXECUTION TIME DEPENDS GREATLY
386                                  ON THE PDP11 CPU CONFIGURATION AND THE AMOUNT OF MEMORY IN THE
387                                  SYSTEM.
388
389                          8.2     PASS COMPLETE
390
391                                  NOTE:  EVERY TIME THE PROGRAM IS STARTED;  THE TESTS WILL  RUN
392                                  AS  IF  SW11  (DELETE  ITERATIONS)  WAS  UP  (=1).  THIS IS TO
393                                  'VERIFY NO HARD ERRORS' AS SOON AS  POSSIBLE.  THEREFORE  THE
394                                  FIRST  PASS  -EACH  TIME  PROGRAM IS STARTED- WILL BE A 'QUICK
395                                  PASS' UNTIL ALL  KMC11'S  IN  SYSTEM  ARE  TESTED.  WHEN  THE
396                                  DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF
397                                  THE PRINT OUT TO BE EXPECTED.
398
399                                  END PASS CZKCF CSR:  175000 VEC:  0300 PASSES:  000001
400                                  ERRORS:  000000
401
402                                  NOTE:    THE PASS COUNT AND ERROR COUNTS  ARE  CUMMULITIVE  FOR
403                                           EACH  KMC11  THAT IS RUNNING, AND ARE SET TO ZERO ONLY
404                                           WHEN THE DIAGNOSTIC IS STARTED.  THEREFORE  AFTER  AN
405                                           OVERNIGHT RUN FOR EXAMPLE, THE TOTAL PASSES AND ERRORS
406                                           FOR EACH KMC11 SINCE THE DIAGNOSTIC  WAS  STARTED  ARE
407                                           REFLECTED IN PASSES:  AND ERRORS:.
```

8.4 KEY LOCATIONS

IPADR (1206) CONTAINS THE ADDRESS WHERE PROGRAM WILL RETURN WHEN ITERATION COUNT IS REACHED OR IF LOOP ON TEST IS ASSERTED.

NEXT (1442) CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PEFORMED.

TSTNM (1202) CONTAINS THE NUMBER OF THE TEST NOW BEING PEFORMED.

RUN (1500) THE BIT IN 'RUN' ALWAYS POINTS TO THE KMC11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1500/0000000001000000 MEANS THAT KMC11 NO.06 IS THE KMC11 NOW RUNNING.

KMCR00-KMCR17
KMST00-KMST17
(2100)-(2300)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) KMC11S SEQUENTIALY. THEY CONTAIN THE CSR,VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH KMC11.

KMACTV (1470) EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED KMC11 WILL BE TESTED IN TURN.
EXAMPLE: (KMACTV) 1470/0000000000011111 MEANS THAT KMC11 NO. 00,01,02,03,04 WILL BE TESTED.
EXAMPLE: (KMACTV) 1470/0000000000010001 MEANS THAT KMC11 NO. 00,04 WILL BE TESTED.

KMCSR (2066) CONTAINS THE CSR OF THE CURRENT KMC11 UNDER TEST.

8.4A 'STATUS TABLE' (2100-2300)

THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT (QUESTIONS) AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION.

THE EXAMPLE STATUS MAP SHOWN BELOW CONTAINS INFORMATION FOR TWO KMC11'S. THE TABLE CAN CONTAIN UP TO 16 KMC11'S. FOLLOWING THE MAP IS A DESCRIPTION OF THE BITS FOR EACH MAP ENTRY

MAP OF KMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
002100	160010	045310	177777	000000
002110	160020	016320	000000	000000

```
464
465
466                     EACH MAP ENTRY CONTAINS  4  WORDS  WHICH  CONTAIN  THE  STATUS
467                     INFORMATION  FOR  1  KMC11.  THE PC SHOWS WHERE IN CORE MEMORY
468                     THE FIRST OF THE 4 WORDS IS.  IN THE EXAMPLE ABOVE  THE  FIRST
469                     KMC'S STATUS IS IN LOCATIONS, 2100, 2102, 2104, AND 2106.  THE
470                     SECOND KMC STATUS IS LOCATED AT 2110, 2112,  2114,  AND  2116.
471                     THE  INFORMATION  CONTAINED IN EACH 4 WORD ENTRY IS DEFINED AS
472                     FOLLOWS:
473
474                     CSR:    CONTAINS KMC11 CSR ADDRESS
475
476                     STAT1:  BITS 00-08 IS KMC11 VECTOR ADDRESS
477                             BIT14=1 TURNAROUND CONNECTOR IS ON
478                             BIT14=0 NO TURNAROUND CONNECTOR
479                             BIT13=0 LINE UNIT IS AN M8201
480                             BIT13=1 LINE UNIT IS AN M8202
481                             BIT12=1 NO LINE UNIT
482                             BITS 09-11 IS KMC11 BR PRIORITY LEVEL
483
484                     STAT2:  LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
485                             HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)
486
487                     STAT3:  BIT2=0  KMC11-DA
488                             BIT2=1  KMC11-FA
489
```

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE AUTO-SIZING ROUTINE FINDS A KMC11 AS FOLLOWS: IT STARTS AT ADDRESS 160000 AND TESTS ALL ADDRESS IN INCREMENTS OF 10 UP TO AND INCLUDING ADDRESS 167760. IF THE ADDRESS DOES NOT TIME OUT, THE FOLLOWING IS DONE, THE FIRST CRAM ADDRESS IS WRITTEN TO A 125252 THEN IT IS READ BACK. IF IT CONTAINS A -1 OR 125252 A KMC11 HAS BEEN FOUND, IF NOT, THE ADDRESS IS UPDATED BY 10 AND THE SEARCH CONTINUES. A -1 INDICATES A KMC11 WITH NO CRAM, A 125252 INDICATES A KMC11 WITH CRAM FURTHER TESTS ARE PERFORMED AT THIS POINT TO DETERMINE WHICH LINE UNIT, IF ANY, IS INSTALLED, IF A LOOP-BACK CONNECTOR IS INSTALLED AND VARIOUS SWITCH SETTINGS ON THE LINE UNIT. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. ALL KMC11'S IN THE SYSTEM WILL BE FOUND BY THE AUTO-SIZER. IF IT DOES NOT FIND A KMC11 THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS ANSWERED.

8.5.2 FINDING THE VECTOR AND BR LEVEL

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '.+2' (NEXT ADDRESS). THE PROCESSOR STATUS IS STARTED AT 7 AND THE KMC IS PROGRAMMED TO INTERRUPT. THE PS IS LOWERED BY 1 UNTIL THE KMC INTERRUPTS, A DELAY IS MADE AND IF NO INTERUPT OCCURES AT PS LEVEL 3 (BECAUSE OF A BAD KMC11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AT BR LEVEL 5 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED; THE PROGRAM SHOULD BE RE-SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERUPT OCCURED; THE ADDRESS TO WHICH THE KMC11 INTERUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU; THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.6 SOFTWARE SWITCH REGISTER

IF THE DIAGNOSTIC IS RUN ON AN 11/04 OR OTHER CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED TO ALLOW USER THE SAME SWITCH OPTIONS AS DESCRIBED PREVIOUSLY. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THIS SOFTWARE SWITCH REGISTER IS USED.
CONTROL:
TO OBTAIN CONTROL AT ANY ALLOWABLE TIME DURING EXECUTION OF THE DIAGNOSTIC THE OPERATOR TYPES A CTRL G ON THE CONSOLE TERMINAL KEYBOARD. AS SOON AS THE CTRL G IS RECOGNIZED, BY THE DIAGNOSTIC, THE FOLLOWING MESSAGE WILL BE DISPLAYED:

    SWR=XXXXXX NEW?

```
545
546
547                     WHERE XXXXXX IS THE CURRENT CONTENTS OF  THE  SOFTWARE  SWITCH
548                     REGISTER  IN  OCTAL.  THE  SOFTWARE CONTROL ROUTINE WILL THEN
549                     AWAIT OPERATOR ACTION.  AT WHICH TIME THE OPERATOR IS REQUIRED
550                     TO  TYPE  ONE  OR  MORE OF THE LEGAL CHARACTERS:  1) 0 - 7, 2)
551                     LINE FEED(<LF>), 3) CARRIAGE  RETURN(<CR>),  OR  4)  CONTROL-U
552                     (CTRL  U).  NO  CHECK  IS  MADE  FOR  LEGALITY.  IF THE INPUT
553                     CHARACTER IS NOT A <LF>, <CR>, OR CTRL U IT IS ASSUMED  TO  BE
554                     AN OCTAL DIGIT.
555
556                     TO CHANGE THE CONTENTS OF THE SSR THE  OPERATOR  SIMPLY  TYPES
557                     THE  NEW  DESIRED  VALUE  IN OCTAL - LEADING ZEROS NEED NOT BE
558                     TYPED.  AND TERMINATES THE INPUT STRING WITH A  <CR>  OR  <LF>
559                     DEPENDING  ON  THE  PROGRAM ACTION DESIRED AS DESCRIBED BELOW.
560                     THE INPUT VALUE WILL BE TRUNCATED TO THE LAST 6 DIGITS  TYPED.
561                     AT  LEAST  ONE  DIGIT  MUST BE TYPED ON ANY GIVEN INPUT STRING
562                     PRIOR TO THE TERMINATOR BEFORE A CHANGE TO THE SSR WILL OCCUR.
563
564                     WHEN THE INPUT STRING IS TERMINATED WITH A <CR> THE DIAGNOSTIC
565                     WILL  CONTINUE  EXECUTION  FROM  THE  POINT  AT  WHICH  IT WAS
566                     INTERRUPTED.  IF A <CR> IS THE ONLY THING  TYPED  THE  PROGRAM
567                     WILL CONTINUE WITHOUT CHANGING THE SSR.  THE <LF> DIFFERS FROM
568                     THE <CR> BY RESTARTING THE PROGRAM AS IF IT WERE RESTARTED  AT
569                     ADDRESS 200.
570
571                     IF A CTRL U IS TYPED AT ANY POINT IN THE INPUT STRING PRIOR TO
572                     THE  TERMINATOR  THE  INPUT  VALUE WILL BE DISREGARDED AND THE
573                     PROMPT DISPLAYED (SWR = XXXXXX NEW?).
574
575                     TO SET THE SSR FOR  THE  STARTING  SWITCHES,  FIRST  LOAD  THE
576                     DIAGNOSTIC, THEN HIT CTRL G, THEN START THE DIAGNOSTIC.
577
578                     ************************************************************
579
580                     NOTE:FOR IPG'S LINE UNIT M8202-YE USERS.
581
582                     CABLE DATA TEST:[TEST 60, TEST 61]
583
584                             THESE TESTS WON'T RUN RELIABLY ON LINE  UNITS  WITHOUT
585                     TERMINATING RESISTENCE.
```

```
586
587
588                                     APT/ACT/XXDP/SLIDE
589
590
591                                     ----------
592                                     THIS DIAGNOSTIC IS APT/ACT/XXDP/SLIDE COMPATIBLE USER WOULD BE
593                                     ABLE TO RUN IT UNDER APT/ACT/XXDP ENVIRONMENT.
594
595                                     NOTE:  FOR MANUFACTURING PURPOSE ONLY ITS DESCRIBED HOW TO RUN
596                                     UNDER APT ENVIRONMENT.
597
598                                     ************************************************************
599
600                                     ETABLE SETTING FOR APT TO RUN UNDER APT
601
602
603                                     --------------------
604                                             FIRST PASS TIME:
605
606                                             LONGEST TEST TIME:
607
608                                             ADDITIONAL TEST TIME:
609
610                                     ALL  THE  ABOVE  PARAMETERS  ARE   DEPENDENT   ON   PARTICULAR
611                                     DIAGNOSTICS  AND  SHOULD  BE  LOADED  AT  THE  TIME OF SETTING
612                                     ETABLE.THERE IS NO DEFAULT TIME SET UP.
613
614                                     SOFTWARE ENVIRONMENT:001          ENVIRONMENT MODE:200
615
616                                     SWITCH 1:-SHOULD BE USED AS NORMAL SWITCH REGISTER.
617
618                                     SWITCH 2:-NOT USED.
619
620                                     CPU OPTIONS:-NOT USED.
621
622                                     MEMORY TYPE 1:-BITS<2:4>:=BITS <12:14> OF STAT1 OF DEV:0.
623
624                                     MAXIMUM ADDRESS:-BITS<17:19>:=BITS<12:14> OF STAT1 OF DEV:1
625
626                                                     BITS<2:4>:=BITS <12:14> OF STAT1 OF DEV:2
627
628                                                     BITS<10:12>:=BITS<12:14> OF STAT1 OF DEV:3
629
630                                     IN THE SAME MANNER
631
632                                     MEMORY TYPE 2 MAXIMUM  ADDRESS:-GETS  STAT1<12:14>  OF  DEVICE
633                                     4,5,6,7.
634
635                                     MEMORY TYPE 3 MAXIMUM  ADDRESS:-GETS  STAT1<12:14>  OF  DEVICE
636                                     8,9,10,11.
637
638                                     MEMORY TYPE 4 MAXIMUM  ADDRESS:-GETS  STAT1<12:14>  OF  DEVICE
639                                     12,13,14,15.
640
641                                     INTERRUPT VECTOR 1:FIRST DEVICE RECEIVE VECTOR.
```

```
642
643
644                         REST OF THE DEVICE(KMC'S) VECTOR SHOULD BE SET UP SEQUENTIALLY
645                         IN INCREMENTS OF 10.
646
647                         BUS PRIORITY:KMC'S PRIORITY(SHOULD BE SAME FOR ALL KMC'S UNDER
648                         TEST).
649
650                         INTERRUPT VECTOR 2:NOT USED.
651
652                         BUS PRIORITY:NOT USED.
653
654                         BASE ADDRESS:FIRST DEVICE CSR ADDRESS.
655
656                                 REST SHOULD FOLLOW SEQUENTIALLY
657
658                                 IN INCREMENTS OF 10.
659
660                         DEVICE MAP:AS DESCRIBED IN APT MANUAL.
661
662                         CONTROLLER SPECIFIC CODE 1:-NO.  OF DEVICES UNDER TEST.
663
664                         CONTROLLER SPECIFIC CODE 2:-NOT USED.
665
666                         DEVICE DESCRIPTOR WORD 0:STAT2 OF FIRST DEVICE.
667
668                         .  .
669
670                         .  .
671
672                         TO
673
674                         .  .
675
676                         .  .
677
678                         DEVICE DESCRIPTOR WORD 15:STAT2 OF 16TH DEVICE.(KMC)
679
680
681               9.0   HISTORY
682
683                         THIS DIAGNOSTIC WAS UPDATED TO DETECT FOR THE CONDITION OF V.35
684                         AND M8201.  IN THIS CONIFURATION, RING WILL NOT BE LOOPED BACK
685                         AND SHOULD NOT BE TESTED FOR.
686               $
```

```
                        .TITLE  CZKCF
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DINESH GORADIA
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        ;*CZKCF   BIT STUFF MD L TSTS
                        ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                        ;*-----------------------------------------------------------------

                        ;STARTING PROCEDURE
                        ;LOAD PROGRAM
                        ;LOAD ADDRESS 000200
                        ;SWR=0  AUTOSIZE KMC11
                        ;SW07=1  USE CURRENT KMC11 PARAMETERS
                        ;SW00=1  INPUT NEW KMC11 PARAMETERS
                        ;PRESS START
                        ;PROGRAM WILL TYPE "CZKCF   BIT STUFF MD L TSTS"
                        ;PROGRAM WILL TYPE STATUS MAP
                        ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                        ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
                        ;AND THEN RESUME TESTING
                        ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
001200                  STACK=  1200
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
```

```
                        R3=     %3              ;;GENERAL REGISTER
                        R4=     %4              ;;GENERAL REGISTER
                        R5=     %5              ;;GENERAL REGISTER
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
                        PP0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
```

```
                        000100          BIT06=  100
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          BIT00=  1
                                        .EQUIV  BIT09,BIT9
                                        .EQUIV  BIT08,BIT8
                                        .EQUIV  BIT07,BIT7
                                        .EQUIV  BIT06,BIT6
                                        .EQUIV  BIT05,BIT5
                                        .EQUIV  BIT04,BIT4
                                        .EQUIV  BIT03,BIT3
                                        .EQUIV  BIT02,BIT2
                                        .EQUIV  BIT01,BIT1
                                        .EQUIV  BIT00,BIT0

                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014          TBITVEC=14              ;;"T" BIT
                        000014          TRTVEC= 14              ;;TRACE TRAP
                        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024          PWRVEC= 24              ;;POWER FAIL
                        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034          TRAPVEC=34              ;;"TRAP" TRAP
                        000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064          TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR



                                        ;INSTRUCTION DEFINITIONS
                                        ;----------------------

                        005746          PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
                        005726          POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
                        010046          PUSHR0=10046    ;SAVE R0 ON STACK
                        012600          POPR0=12600     ;RESTORE R0 FROM STACK
                        024646          PUSH2SP=24646   ;DECREMENT STACK TWICE
                        022626          POP2SP=22626    ;INCREMENT STACK TWICE
                                        .EQUIV  EMT,HLT ;BASIC DEFINITION OF ERROR CALL
```

```
                                ;:********************************* *****************************************
                                ;-------------------------------------------------------------------------
                                        ;TRAPCATCAER FOR ILLEGAL INTERRUPTS
                                        ;THE STANDARD "TRAP CATCHER" IS PLACED
                                        ;BETWEEN ADDRESS 0 TO ADDRESS 776.
                                        ;IT LOOKS LIKE "PC+2 HALT".
                                ;-------------------------------------------------------------------------
                                ;:*************************************************************************

            000000              .=0
000000  000000  000000                  .WORD   0,0
                                ;STANDARD INTERRUPT VECTORS
                                ;-------------------------

            000020              .=20
000020  004134                          $SCOPE          ; SCOPE LOOP HANDLER.
000022  000340                          PR7             ; SERVICE AT LEVEL 7.
000024  007126                          $PWRDN                  ;POWER FAIL HANDLER
000026  000340                          PR7                     ;SERVICE AT LEVEL 7
000030  006512                          $ERROR                  ;ERROR HANDLER
000032  000340                          PR7                     ;SERVICE AT LEVEL 7
000034  006414                          $TRAP                   ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                          PR7                     ;SERVICE AT LEVEL 7
                                .SBTTL  ACT11 HOOKS

                                ;;*************************************************************************
                                ;HOOKS REQUIRED BY ACT11
            000040                      $SVPC=.                 ;SAVE PC
            000046                      .=46
000046  004070                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
            000052                      .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
            000040                      .=$SVPC                 ;; RESTORE PC

            000174              .=174
000174  000000              DISPREG:0                           ;SOFTWARE DISPLAY REGISTER
000176  000000              SWREG:  0                           ;SOFTWARE SWITCH REGISTER

            000200              .=200
000200  000137  002402                  JMP     .START          ;GO TO START OF PROGRAM


            001000              .=1000
001000  005200  055103  041513  MTITLE: .ASCII  <200><12>/CZKCF/<200>
(2) 001010  044502  020124  052123          .ASCIZ  /BIT STUFF MD L TSTS/<200>
(2)

            177570              DSWR    =       177570
            177570              DDISP   =       177570
```

```
                                .SBTTL  COMMON TAGS

                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

001200                                  .=1200
001200                          $CMTAG:                         ;;START OF COMMON TAGS
001200  000000                          .WORD   0
001202  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001203  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001204  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001206  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001210  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001212  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001214  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001215  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001216  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001220  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001222  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001224  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001226  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001230  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001232  000000                          .WORD   0
001234  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001235  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001236  000000                          .WORD   0
001240  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001242  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001244  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001246  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001250  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001252  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001254  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001255  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001256  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001257  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001260  000000                  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REG0) WAS OBTAINED
001262  000000                  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001264  000000                  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001266  000000                  $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001270  000000                  $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001272  000000                  $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001274  000000                  $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001276  000000                  $TMP0:  .WORD   0               ;;USER DEFINED
001300  000000                  $TMP1:  .WORD   0               ;;USER DEFINED
001302  000000                  $TMP2:  .WORD   0               ;;USER DEFINED
001304  000000                  $TMP3:  .WORD   0               ;;USER DEFINED
001306  000000                  $TMP4:  .WORD   0               ;;USER DEFINED
001310  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001312  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001313  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001314  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE
```

```
                                ;;*****************************************************************
                                .EVEN
001316                          $MAIL:                          ;;APT MAILBOX
001316  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001320  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
001322  000000                  $TESTN: .WORD   ATESTN   ;;TEST NUMBER
001324  000000                  $PASS:  .WORD   APASS    ;;PASS COUNT
001326  000000                  $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
001330  000000                  $UNIT:  .WORD   AUNIT    ;;I/O UNIT NUMBER
001332  000000                  $MSGAD: .WORD   AMSGAD   ;;MESSAGE ADDRESS
001334  000000                  $MSGLG: .WORD   AMSGLG   ;;MESSAGE LENGTH
001336                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001336      002                 $ENV:   .BYTE   AENV     ;;ENVIRONMENT BYTE
001337      000                 $ENVM:  .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
001340  000000                  $SWREG: .WORD   ASWREG   ;;APT SWITCH REGISTER
001342  000000                  $USWR:  .WORD   AUSWR    ;;USER SWITCHES
001344  000000                  $CPUOP: .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001346      000                 $MAMS1: .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001347      000                 $MTYP1: .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001350  000000                  $MADR1: .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001352      000                 $MAMS2: .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001353      000                 $MTYP2: .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001354  000000                  $MADR2: .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001356      000                 $MAMS3: .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001357      000                 $MTYP3: .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001360  000000                  $MADR3: .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001362      000                 $MAMS4: .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001363      000                 $MTYP4: .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001364  000000                  $MADR4: .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001366  000000                  $VECT1: .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001370  000000                  $VECT2: .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001372  000000                  $BASE:  .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001374  000000                  $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
001376  000000                  $CDW1:  .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
001400  000000                  $CDW2:  .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
001402  000000                  $DDW0:  .WORD   ADDW0    ;;DEVICE DESCRIPTOR WORD#0
001404  000000                  $DDW1:  .WORD   ADDW1    ;;DEVICE DESCRIPTOR WORD#1
001406  000000                  $DDW2:  .WORD   ADDW2    ;;DEVICE DESCRIPTOR WORD#2
001410  000000                  $DDW3:  .WORD   ADDW3    ;;DEVICE DESCRIPTOR WORD#3
001412  000000                  $DDW4:  .WORD   ADDW4    ;;DEVICE DESCRIPTOR WORD#4
001414  000000                  $DDW5:  .WORD   ADDW5    ;;DEVICE DESCRIPTOR WORD#5
001416  000000                  $DDW6:  .WORD   ADDW6    ;;DEVICE DESCRIPTOR WORD#6
001420  000000                  $DDW7:  .WORD   ADDW7    ;;DEVICE DESCRIPTOR WORD#7
001422  000000                  $DDW8:  .WORD   ADDW8    ;;DEVICE DESCRIPTOR WORD#8
```

```
 1006  001424  000000                $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
 1007  001426  000000                $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
 1008  001430  000000                $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
 1009  001432  000000                $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
 1010  001434  000000                $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
 1011  001436  000000                $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
 1012  001440  000000                $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
 1013
 1014
 1015  001442                        $ETEND:
 1016
 1017                                ;
 1018                                ;  PROGRAM CONTROL PARAMETERS
 1019                                ;-------------------------------
 1020  001442  000000                NEXT:   .WORD   0               ; ADDRSS OF NEXT TEST TO BE EXECUTED
 1021  001444  000000                LOCK:   .WORD   0               ; ADDRESS FOR LOCK CURRENT DATA
 1022                                ;
 1023                                ;   PROGRAM VARIABLES
 1024                                ;----------------------
 1025  001446  000000                STRTSW: .WORD   0               ; SWITCHES AT START OF PROGRAM
 1026  001450  000000                STAT:   .WORD   0               ; KM STATUS WORD STORAGE
 1027  001452  000000                CLKX:   .WORD   0               ;
 1028  001454  000000                MASKX:  .WORD   0               ;
 1029  001456  000000                SAVSP:  .WORD   0               ; STACK POINTER STORAGE
 1030  001460  000000                SAVPC:  .WORD   0               ; PROGRAM COUNTER STORAGE
 1031  001462  000000                ZERO:   .WORD   0               ;
 1032  001464  000001                ONE:    .WORD   1               ;
 1033  001466  000000                MEMLIM: .WORD   0               ; HIGHEST LOCATION FOR NPR'S
 1034  001470  000001                KMACTV: .BLKW   1               ; KMC11 SELECTED ACTIVE
 1035  001472  000001                KMNUM:  .BLKW   1               ; OCTAL NUMBER OF KMC11'S
 1036  001474  000001                SAVACT: .BLKW   1               ; ORIGINAL ACTIVE DEVICES.
 1037  001476  000001                SAVNUM: .BLKW   1               ; WORKABLE NUMBER.
 1038  001500  000000                RUN:    .WORD   0               ; POINTER TO RUNNING DEVICES
 1039                                        .EVEN
 1040  001502  002072                CREAM:  .WORD   KM.MAP-6        ; TABLE POINTER
 1041  001504  002276                MILK:   .WORD   CNT.MAP-4       ; TABLE POINTER
 1042                                ;
 1043                                ;  PROGRAM CONTROL FLAGS
 1044                                ; ------------------------
 1045  001506     000                INIFLG: .BYTE   0               ; PROGRAM INITIALIZING FLAG
 1046          001510                        .EVEN
 1047  001510     000                LOKFLG: .BYTE   0               ; LOCK ON CURRENT TEST FLAG
 1048  001511     000                QV.FLG: .BYTE   0               ; QUICK VERIFY FLAG
 1049                                                                ; ON FIRST PASS OF EACH KMC11 ITERATIONS WILL BE SJPPRES
 1050                                        .EVEN
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


001512                          $ERRTB:
                                .EVEN
                                ;*      DF              ;; DOES NOT APPLY IN THIS DIAGNOSTIC.
001512  000000                          0
001514  000000                          0
001516  000000                          0
001520  032062                          EM1
001522  033224                          DH1             ; ERROR 1
001524  033550                          DT1
001526  032120                          EM2
001530  033245                          DH2             ; ERROR 2
001532  033562                          DT2
001534  032163                          EM3
001536  033303                          DH3             ; ERROR 3
001540  033600                          DT3
001542  032227                          EM4
001544  033341                          DH4             ; ERROR 4
001546  033612                          DT4
001550  032227                          EM4
001552  033401                          DH5             ; ERROR 5
001554  033624                          DT5
001556  032271                          EM5
001560  033303                          DH3             ; ERROR 6
001562  033642                          DT6
001564  032321                          EM6
001566  033433                          DH6             ; ERROR 7
001570  033624                          DT5
001572  032340                          EM7
001574  000000                          0               ; ERROR 10
001576  000000                          0
001600  032365                          EM10
001602  000000                          0               ; ERROR 11
001604  000000                          0
001606  032570                          EM11
001610  033303                          DH3             ; ERROR 12
001612  033600                          DT3
001614  032617                          EM12
001616  033433                          DH6             ; ERROR 13
001620  033624                          DT5
001622  032643                          EM13
001624  033433                          DH6             ; ERROR 14
001626  033624                          DT5
```

```
1107  001630  032703                          EM14
1108  001632  033303                          DH3             ; ERROR 15
1109  001634  033642                          DT6
1110  001636  032753                          EM15
1111  001640  033341                          DH4             ; ERROR 16
1112  001642  033612                          DT4
1113  001644  032774                          EM16
1114  001646  000000                          0               ; ERROR 17
1115  001650  000000                          0
1116  001652  033010                          EM17
1117  001654  000000                          0               ; ERROR  20
1118  001656  000000                          0
1119  001660  032570                          EM11
1120  001662  033433                          DH6             ; ERROR 21
1121  001664  033624                          DT5
1122  001666  033054                          EM20
1123  001670  033303                          DH3             ; ERROR 22
1124  001672  033660                          DT7
1125  001674  033075                          EM21
1126  001676  033303                          DH3             ; ERROR 23
1127  001700  033600                          DT3
1128  001702  033054                          EM20
1129  001704  000000                          0               ; ERROR 24
1130  001706  000000                          0
1131  001710  033112                          EM22
1132  001712  033303                          DH3             ; ERROR 25
1133  001714  033600                          DT3
1134          002034                  .=2034
1135                                  .SBTTL  APT PARAMETER BLOCK
1136
1137                                  ;;*****************************************************************
1138                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1139                                  ;;*****************************************************************
1140          002034                          .$X=.   ;;SAVE CURRENT LOCATION
1141          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1142  000024  000200                          200     ;;FOR APT START UP
1143          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1144  000044  002034                          $APTHDR ;;POINT TO APT HEADER BLOCK
1145          002034                          .=.$X   ;;RESET LOCATION COUNTER
1146                                  ;;*****************************************************************
1147                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1148                                  ;INTERFACE SPEC.
1149
1150  002034                          $APTHD:
1151  002034  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1152  002036  001316                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1153  002040  000132                  $TSTM:  .WORD   90.     ;;RUN TIM OF LONGEST TEST
1154  002042  000137                  $PASTM: .WORD   95.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1155  002044  000137                  $UNITM: .WORD   95.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1156  002046  000052                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1157
```

```
                                ;KMC11 CONTROL INDICATORS FOR CURRENT KMC11 UNDER TEST
                                ;-----------------------------------------------------

002050  000000                  STAT1:  0
002052  000000                  STAT2:  0
002054  000000                  STAT3:  0

                                ;KMC11 VECTOR AND REGISTER INDIRECT POINTERS
                                ;-------------------------------------------

002056  000000                  KMRVEC: 0               ;POINTER TO KMC11 RECEIVER INTERRUPT VECTOR
002060  000000                  KMRLVL: 0               ;POINTER TO KMC11 RECEIVER INTERRUPT SERVICE PS
002062  000000                  KMTVEC: 0               ;POINTER TO KMC11 TRANSMITTER INTERRUPT VECTOR
002064  000000                  KMTLVL: 0               ;POINTER TO KMC11 TRANSMITTER INTERRUPT SERVICE PS
002066  000000                  KMCSR:  0               ;POINTER TO KMC11 CONTROL STATUS REGISTER
002070  000000                  KMCSRH: 0               ;POINTER TO KMC11 CONTROL STATUS REGISTER HIGH BYTE.
002072  000000                  KMCTL:  0               ;POINTER TO KMC11 CONTOL OUT REGISTER
002074  000000                  KMPO4:  0               ;POINTER TO KMC11 PORT REGISTER(SEL 4)
002076  000000                  KMPO6:  0               ;POINTER TO KMC11 PORT REGISTER(SEL 6)

                                ;TEMP STORAGE
                                ;------------

                                ;TEMP:  0
                                ;.=.+40

                                ;KMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
                                ;------------------------------------------

        002100                  .=2100
002100                          KM.MAP:
002100  000001                  KMCR00: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 00
002102  000001                  KMS100: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 00
002104  000001                  KMS200: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 00
002106  000001                  KMS300: .BLKW   1               ;3RD STATUS WORD

002110  000001                  KMCR01: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 01
002112  000001                  KMS101: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 01
002114  000001                  KMS201: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 01
002116  000001                  KMS301: .BLKW   1               ;3RD STATUS WORD

002120  000001                  KMCR02: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 02
002122  000001                  KMS102: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 02
002124  000001                  KMS202: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 02
002126  000001                  KMS302: .BLKW   1               ;3RD STATUS WORD

002130  000001                  KMCR03: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 03
002132  000001                  KMS103: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 03
002134  000001                  KMS203: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 03
002136  000001                  KMS303: .BLKW   1               ;3RD STATUS WORD

002140  000001                  KMCR04: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 04
002142  000001                  KMS104: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 04
002144  000001                  KMS204: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 04
002146  000001                  KMS304: .BLKW   1               ;3RD STATUS WORD
```

```
1214
1215  002150  000001                KMCR05: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 05
1216  002152  000001                KMS105: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 05
1217  002154  000001                KMS205: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 05
1218  002156  000001                KMS305: .BLKW   1               ;3RD STATUS WORD
1219
1220  002160  000001                KMCR06: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 06
1221  002162  000001                KMS106: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 06
1222  002164  000001                KMS206: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 06
1223  002166  000001                KMS306: .BLKW   1               ;3RD STATUS WORD
1224
1225  002170  000001                KMCR07: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 07
1226  002172  000001                KMS107: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 07
1227  002174  000001                KMS207: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 07
1228  002176  000001                KMS307: .BLKW   1               ;3RD STATUS WORD
1229
1230  002200  000001                KMCR10: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 10
1231  002202  000001                KMS110: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 10
1232  002204  000001                KMS210: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 10
1233  002206  000001                KMS310: .BLKW   1               ;3RD STATUS WORD
1234
1235  002210  000001                KMCR11: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 11
1236  002212  000001                KMS111: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 11
1237  002214  000001                KMS211: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 11
1238  002216  000001                KMS311: .BLKW   1               ;3RD STATUS WORD
1239
1240  002220  000001                KMCR12: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 12
1241  002222  000001                KMS112: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 12
1242  002224  000001                KMS212: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 12
1243  002226  000001                KMS312: .BLKW   1               ;3RD STATUS WORD
1244
1245  002230  000001                KMCR13: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 13
1246  002232  000001                KMS113: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 13
1247  002234  000001                KMS213: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 13
1248  002236  000001                KMS313: .BLKW   1               ;3RD STATUS WORD
1249
1250  002240  000001                KMCR14: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 14
1251  002242  000001                KMS114: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 14
1252  002244  000001                KMS214: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 14
1253  002246  000001                KMS314: .BLKW   1               ;3RD STATUS WORD
1254
1255  002250  000001                KMCR15: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 15
1256  002252  000001                KMS115: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 15
1257  002254  000001                KMS215: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 15
1258  002256  000001                KMS315: .BLKW   1               ;3RD STATUS WORD
1259
1260  002260  000001                KMCR16: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 16
1261  002262  000001                KMS116: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 16
1262  002264  000001                KMS216: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 16
1263  002266  000001                KMS316: .BLKW   1               ;3RD STATUS WORD
1264
1265  002270  000001                KMCR17: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 17
1266  002272  000001                KMS117: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 17
1267  002274  000001                KMS217: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 17
1268  002276  000001                KMS317: .BLKW   1               ;3RD STATUS WORD
1269
```

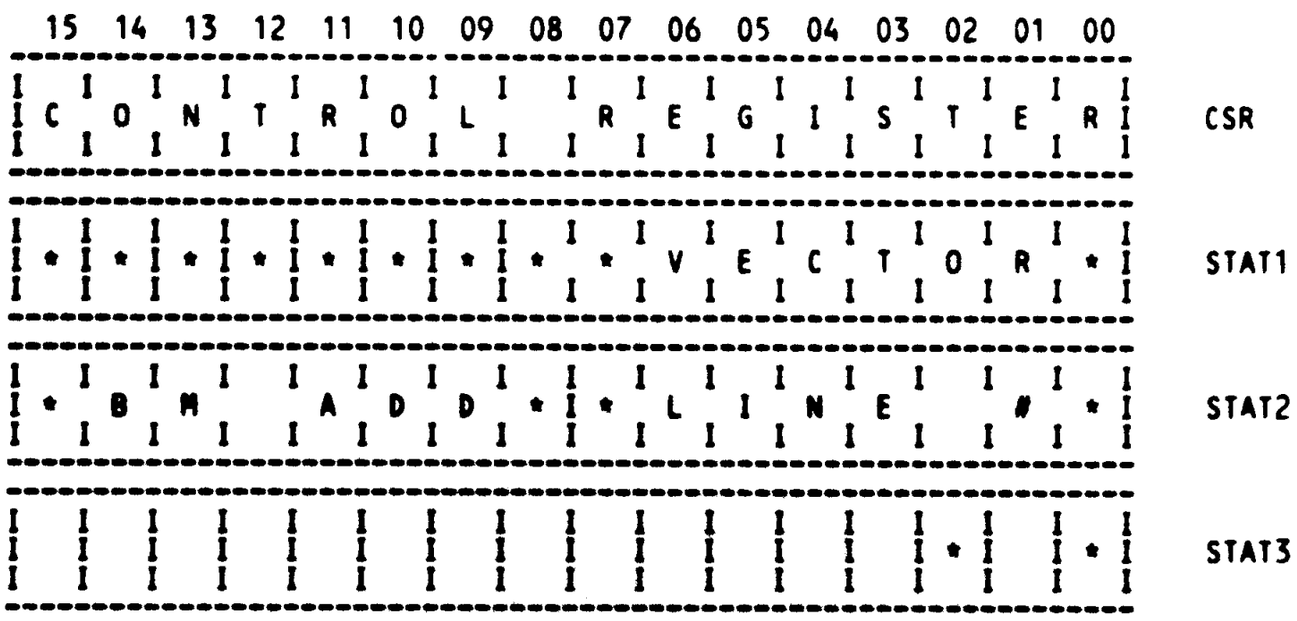
```
CZKCF    MACY11 30A(1052)  08-JUL-80  08:27  PAGE 28
CZKCF.P11     08-JUL-80 08:24          APT PARAMETER BLOCK                    SEQ 0027

  1270  002300  000000                 KM.END: 000000
```

```
1271
1272                                    ;KMC11 PASS COUNT AND ERROR COUNT TABLE
1273                                    ;--------------------------------------
1274
1275  002302                            CNT.MAP:
1276  002302  000000                    PACT00: 0               ;PASS COUNT FOR KMC11 NUMBER 00
1277  002304  000000                    ERCT00: 0               ;ERROR COUNT FOR KMC11 NUMBER 00
1278
1279  002306  000000                    PACT01: 0               ;PASS COUNT FOR KMC11 NUMBER 01
1280  002310  000000                    ERCT01: 0               ;ERROR COUNT FOR KMC11 NUMBER 01
1281
1282  002312  000000                    PACT02: 0               ;PASS COUNT FOR KMC11 NUMBER 02
1283  002314  000000                    ERCT02: 0               ;ERROR COUNT FOR KMC11 NUMBER 02
1284
1285  002316  000000                    PACT03: 0               ;PASS COUNT FOR KMC11 NUMBER 03
1286  002320  000000                    ERCT03: 0               ;ERROR COUNT FOR KMC11 NUMBER 03
1287
1288  002322  000000                    PACT04: 0               ;PASS COUNT FOR KMC11 NUMBER 04
1289  002324  000000                    ERCT04: 0               ;ERROR COUNT FOR KMC11 NUMBER 04
1290
1291  002326  000000                    PACT05: 0               ;PASS COUNT FOR KMC11 NUMBER 05
1292  002330  000000                    ERCT05: 0               ;ERROR COUNT FOR KMC11 NUMBER 05
1293
1294  002332  000000                    PACT06: 0               ;PASS COUNT FOR KMC11 NUMBER 06
1295  002334  000000                    ERCT06: 0               ;ERROR COUNT FOR KMC11 NUMBER 06
1296
1297  002336  000000                    PACT07: 0               ;PASS COUNT FOR KMC11 NUMBER 07
1298  002340  000000                    ERCT07: 0               ;ERROR COUNT FOR KMC11 NUMBER 07
1299
1300  002342  000000                    PACT10: 0               ;PASS COUNT FOR KMC11 NUMBER 10
1301  002344  000000                    ERCT10: 0               ;ERROR COUNT FOR KMC11 NUMBER 10
1302
1303  002346  000000                    PACT11: 0               ;PASS COUNT FOR KMC11 NUMBER 11
1304  002350  000000                    ERCT11: 0               ;ERROR COUNT FOR KMC11 NUMBER 11
1305
1306  002352  000000                    PACT12: 0               ;PASS COUNT FOR KMC11 NUMBER 12
1307  002354  000000                    ERCT12: 0               ;ERROR COUNT FOR KMC11 NUMBER 12
1308
1309  002356  000000                    PACT13: 0               ;PASS COUNT FOR KMC11 NUMBER 13
1310  002360  000000                    ERCT13: 0               ;ERROR COUNT FOR KMC11 NUMBER 13
1311
1312  002362  000000                    PACT14: 0               ;PASS COUNT FOR KMC11 NUMBER 14
1313  002364  000000                    ERCT14: 0               ;ERROR COUNT FOR KMC11 NUMBER 14
1314
1315  002366  000000                    PACT15: 0               ;PASS COUNT FOR KMC11 NUMBER 15
1316  002370  000000                    ERCT15: 0               ;ERROR COUNT FOR KMC11 NUMBER 15
1317
1318  002372  000000                    PACT16: 0               ;PASS COUNT FOR KMC11 NUMBER 16
1319  002374  000000                    ERCT16: 0               ;ERROR COUNT FOR KMC11 NUMBER 16
1320
1321  002376  000000                    PACT17: 0               ;PASS COUNT FOR KMC11 NUMBER 17
1322  002400  000000                    ERCT17: 0               ;ERROR COUNT FOR KMC11 NUMBER 17
1323
```

1324
1325
1326
1327
1328
1329

FORMAT OF STATUS TABLE



DEFINITION OF FORMAT

CSR: CONTAINS KMC11 CSR ADDRESS

STAT1: BITS 00-08 IS KMC11 VECTOR ADDRESS
BIT14=1 ???? TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS KMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
(MUST BE SET TO A ONE MANUALLY [PROGRAMS G AND H ONLY])

BIT2=0 DMC11-DA (RS232C)
BIT2=1 DMC11-FA (V.35)

```
CZKCF    MACY11 30A(1052)  08-JUL-80  08:27  PAGE 31
CZKCF.P11     08-JUL-80 08:24            APT PARAMETER BLOCK                SEQ 0030

        $
```

```
 1381
 1382                                           ;PROGRAM INITIALIZATION
 1383                                           ;LOCK OUT INTERRUPTS
 1384                                           ;SET UP PROCESSOR STACK
 1385                                           ;SET UP POWER FAIL VECTOR
 1386                                           ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
 1387                                           ;TYPE TITLE MESSAGE
 1388
 1389  002402  012737  000340  177776  .START: MOV     #340,PS          ;LOCK OUT INTERRUPTS
 1390  002410  012706  001200                  MOV     #STACK,SP        ;SET UP STACK
 1391  002414  012737  007126  000024          MOV     #$PWRDN,@#24     ;SET UP POWER FAIL VECTOR
 1392  002422  013737  001472  001476          MOV     KMNUM,SAVNUM     ;SAVE NUMBER OF DEVICES IN SYSTEM.
 1393  002430  005037  011544                  CLR     SWFLG            ;CLEAR SOFT TYPEOUT FLAG
 1394  002434  105037  001203                  CLRB    $ERFLG           ;CLEAR ERROR FLAG
 1395  002440  105037  001511                  CLRB    QV.FLG           ;ZERO QUICK VERIFY FLAG
 1396  002444  012737  002070  001502          MOV     #KM.MAP-10,CREAM;GET MAP POINTER.
 1397  002452  012737  002276  001504          MOV     #CNT.MAP-4,MILK  ;GET PASS COUNT MAP POINTER
 1398  002460  012737  100000  001500          MOV     #BIT15,RUN       ;POINT POINTER TO FIRST DEVICE.
 1399  002466  012700  002302                  MOV     #CNT.MAP,R0      ;PASS COUNT POINTER TO R0
 1400  002472  005020                  23$:    CLR     (R0)+            ;CLEAR TABLE
 1401  002474  022700  002402                  CMP     #CNT.MAP+100,R0  ;DONE YET?
 1402  002500  001374                          BNE     23$              ;KEEP GOING
 1403  002502  005037  001216                  CLR     $ERRPC           ;CLEAR LAST ERROR POINTER
 1404  002506  012737  000001  001202          MOV     #1,$TSTNM        ;SET UP FOR TEST 1
 1405  002514  012737  002402  001206          MOV     #.START,$LPADR   ;SET UP FOR POWER FAIL BEFORE
 1406                                          ;TESTING STARTS
 1407  002522  132737  000001  001336          BITB    #1,$ENV          ; IS IT RUNNING UNDER APT?
 1408  002530  001404                          BEQ     3$               ; IF NOT CHECK FOR TYPE OF SWITCH REGISTER.
 1409  002532  013737  001340  000176          MOV     $SWREG,SWREG     ; LOAD SOFTWARE SWITCH REG.
 1410  002540  000423                          BR      6$+2             ; GO SET UP SOFTWARE SWITCH REG.
 1411  002542  013746  000006          3$:     MOV     @#6,-(SP)        ;SAVE CURRENT VECTORS
 1412  002546  013746  000004                  MOV     @#4,-(SP)        ;
 1413  002552  012737  002606  000004          MOV     #6$,@#4          ;SET UP FOR TIMEOUT
 1414  002560  012737  177570  001240          MOV     #177570,SWR      ;SET SWR TO HARD SWR ADDRESS
 1415  002566  012737  177570  001242          MOV     #177570,DISPLAY  ;SET DISPLAY TO HARD SWR ADDRESS
 1416  002574  022777  177777  176436          CMP     #-1,@SWR         ;REFERENCE HARDWARE SWITCH REGISTER
 1417  002602  001402                          BEQ     6$+2             ;IF = -1 USE SOFT SWR ANYWAY
 1418  002604  000407                          BR      7$               ;IF IT EXISTS AND NOT = -1 USE HARD SWR
 1419  002606  022626                  6$:     CMP     (SP)+,(SP)+      ;ADJUST STACK
 1420  002610  012737  000176  001240          MOV     #SWREG,SWR       ;POINTER TO SOFT SWR
 1421  002616  012737  000174  001242          MOV     #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
 1422  002624  012637  000004          7$:     MOV     (SP)+,@#4        ;RESTORE VECTORS
 1423  002630  012637  000006                  MOV     (SP)+,@#6        ;
 1424  002634  105737  001506                  TSTB    INIFLG           ;HAS INITIALIZATION BEEN PERFORMED
 1425  002640  001006                          BNE     20$              ;BR IF YES
 1426  002642  022737  004070  000042          CMP     #$ENDAD,@#42     ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
 1427  002650  001402                          BEQ     20$
 1428  002652  104401  001000                  TYPE    ,MTITLE          ;TYPE TITLE MESSAGE
 1429  002656  004737  011340          20$:    JSR     PC,CKSWR                 ;CHECK FOR SOFT SWR
 1430  002662  017737  176352  001446          MOV     @SWR,STRTSW      ;STORE STARTING SWITCHES
 1431  002670  005737  000042                  TST     @#42             ;IS IT RUNNING IN AUTO MODE?
 1432  002674  001402                          BEQ     .+6              ;BR IF NO
 1433  002676  005037  001446                  CLR     STRTSW           ;IF YES, CLEAR SWITCHES
 1434  002702  032737  000001  001446          BIT     #SW00,STRTSW     ;IF SW00=1, QUESTIONS ARE ASKED.
 1435  002710  001012                          BNE     17$              ;BR IF SW00=1
 1436  002712  105737  001446                  TSTB    STRTSW           ;BIT7=1??
```

```
1437  002716  100007                        BPL     17$             ;BR IF SW07=0
1438  002720  005737  001470                TST     KMACTV          ;ARE ANY DEVICES SELECTED?
1439  002724  001027                        BNE     16$             ;BR IF YES
1440  002726  104401  010731                TYPE,   NOACT           ;NO DEVICES SELECTED.
1441  002732  000000                        HALT                    ;STOP THE SHOW
1442  002734  000776                        BR      .-2             ;DISQUALIFY CONTINUE SWITCH
1443  002736  105737  001336        17$:    TSTB    $ENV            ; IS IT UNDER APT DUMP MODE?
1444  002742  001405                        BEQ     27$             ; YES, CHECK IF APT SIZED IT?
1445  002744  132737  000001  001336        BITB    #1,$ENV         ; IS IT UNDER Q,V OR RUN MODE?
1446  002752  001012                        BNE     30$             ; YES, NEEDS ONLY APT SIZING.
1447  002754  000406                        BR      33$             ; NO, NEEDS REGULAR AUTO.SIZE.
1448  002756  105737  001337        27$:    TSTB    $ENVM           ; IS IT SIZED BY APT?
1449  002762  100406                        BMI     30$             ; YES, NEEDS ONLY APT SIZING.
1450  002764  042737  000001  001446        BIC     #SW00,STRTSW    ; SIZE ONLY IN AUTO MODE.
1451  002772  004737  012236        33$:    JSR     PC,AUTO.SIZE    ; GO DO THE AUTO.SIZE.
1452  002776  000402                        BR      16$             ; GO PRINT THE MAP.
1453  003000  004737  013716        30$:    JSR     PC,APT.SIZE     ; GO DO THE APT SIZING.
1454  003004  105737  001506        16$:    TSTB    INIFLG          ;FIRST TIME?
1455  003010  001410                        BEQ     21$             ;BR IF YES
1456  003012  105737  001446                TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
1457  003016  100431                        BMI     1$
1458  003020  032737  000006  001446        BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
1459  003026  001403                        BEQ     24$             ;IF NO THEN TYPE STATUS
1460  003030  000424                        BR      1$              ;IF YES DO NOT TYPE STATUS
1461  003032  105137  001506        21$:    COMB    INIFLG          ;SET FLAG
1462  003036  104401  010077        24$:    TYPE    ,XHEAD          ;TYPE HEADER
1463  003042  012704  002100                MOV     #KM.MAP,R4      ;SET POINTER
1464  003046  010437  001276        5$:     MOV     R4,$TMP0        ;SET ADDRESS
1465  003052  012437  001300                MOV     (R4)+,$TMP1     ;SET CSR
1466  003056  001411                        BEQ     1$              ;ALL DONE IF ZERO
1467  003060  012437  001302                MOV     (R4)+,$TMP2     ;SET STAT1
1468  003064  012437  001304                MOV     (R4)+,$TMP3     ;SET STAT2
1469  003070  012437  001306                MOV     (R4)+,$TMP4     ;SET STAT3
1470  003074  104416                        CONVRT                  ;TYPE OUT STATUS MAP
1471  003076  011206                        XSTATQ                  ;
1472  003100  000762                        BR      5$
1473  003102  012700  002100        1$:     MOV     #KM.MAP,R0      ;R0 POINTS TO STATUS TABLE
1474
1475                                        ;;**************************************************************************
1476                                        ;;*AUTO SIZE TEST
1477                                        ;;*THIS TEST VERIFYS THAT THE KMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
1478                                        ;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
1479                                        ;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
1480                                        ;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE KMC11, THE FIRST
1481                                        ;;* KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
1482                                        ;;*ADDRESS 760000.
1483                                        ;;**************************************************************************
1484
1485  003106  013746  000004                MOV     @#4,-(SP)       ;SAVE LOC 4
1486  003112  013746  000006                MOV     @#6,-(SP)       ;SAVE LOC 6
1487  003116  005037  000006                CLR     @#6             ;CLEAR VEC+2
1488  003122  005037  001302                CLR     $TMP2           ;CLEAR FLAG
1489  003126  011037  002066        AUSTRT: MOV     (R0),KMCSR      ;GET NEXT KMC CSR
1490  003132  001510                        BEQ     AUDONE          ;BR IF DONE
1491  003134  012737  003240  000004  2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
1492  003142  012703  000010        3$:     MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
```

```
1493  003146  012702  003342          4$:     MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
1494  003152  012701  160010                  MOV     #160010,R1      ;START WITH ADDRESS 160010
1495  003156  005711                  FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
1496  003160  111204                          MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
1497  003162  060401                          ADD     R4,R1           ;IN R1
1498  003164  005201                          INC     R1              ;
1499  003166  040401                          BIC     R4,R1           ;
1500  003170  005703                          TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
1501  003172  001371                          BNE     FLOAT           ;BR IF YES
1502  003174  012737  003244  000004          MOV     #ERR,@#4        ;OK ONLY KMC'S ARE LEFT, SET UP FOR TIMEOUT
1503  003202  005711                  FY:     TST     (R1)            ;CHECK KMC ADDRESS
1504  003204  020137  002066                  CMP     R1,KMCSR        ;DOES IT MATCH
1505  003210  001403                          BEQ     OK              ;BR IF YES
1506  003212  062701  000010                  ADD     #10,R1          ;GET NEXT KMC ADDRESS
1507  003216  000771                          BR      FY              ;DO IT AGAIN
1508  003220  062700  000010          OK:     ADD     #10,R0          ;SKIP TO NEXT KMC CSR
1509  003224  062701  000010                  ADD     #10,R1          ; GET NEXT KMC ADDRESS
1510  003230  011037  002066                  MOV     (R0),KMCSR      ; GET NEXT KMC CSR
1511  003234  001447                          BEQ     AUDONE          ; BRANCH IF ALL DONE.
1512  003236  000761                          BR      FY              ; DO IT AGAIN.
1513  003240  122243                  NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
1514  003242  000002                          RTI                     ;$LPADR
1515  003244  005737  001302          ERR:    TST     $TMP2           ;CHECK FLAG IF = 0 TYPE HEADER
1516  003250  001014                          BNE     1$              ;SKIP HEADER
1517  003252  104401                          TYPE                    ;TYPEOUT HEADER MESSAGE
1518  003254  011107                          CONERR                  ;CONFIGURATION ERROR!!!!
1519  003256  012737  003244  001460          MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
1520  003264  104417                          CNVRT                   ;TYPE OUT ERROR PC
1521  003266  003322                          ERRPC                   ;
1522  003270  104401                          TYPE                    ;TYPE REST OF HEADER
1523  003272  011154                          CNERR                   ;
1524  003274  012737  177777  001302          MOV     #-1,$TMP2       ;SET FLAG SO IT ONLY GETS TYPED ONCE
1525  003302  010137  001264          1$:     MOV     R1,$REG1        ;SAVE R1 FOR TYPEOUT
1526  003306  104416                          CONVRT
1527  003310  003330                          CONTAB                  ;TYPE CSR VALUES
1528  003312  104401                  3$:     TYPE
1529  003314  011175                          KMCM
1530  003316  022626                  4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
1531  003320  000737                          BR      OK              ;BR TO GET OUT
1532  003322  000001                  ERRPC:  1
1533  003324     006     002                  .BYTE   6,2
1534  003326  001460                          SAVPC
1535  003330  000002                  CONTAB: 2
1536  003332     006     004                  .BYTE   6,4
1537  003334  001264                          $REG1
1538  003336     006     002                  .BYTE   6,2
1539  003340  002066                          KMCSR
1540  003342     007                  DEVTAB: .BYTE   7               ;DJ
1541  003343     017                          .BYTE   17              ;DH
1542  003344     007                          .BYTE   7               ;DQ
1543  003345     007                          .BYTE   7               ;DU
1544  003346     007                          .BYTE   7               ;DUP
1545  003347     007                          .BYTE   7               ;LK
1546  003350     007                          .BYTE   7               ;DMC
1547  003351     007                          .BYTE   7               ;DZ
1548  003352     007                          .BYTF   7               ;KMC
```

```
1549            003354                          .EVEN
1550  003354                                    AUDONE:
1551  003354    012637  000006                  1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
1552  003360    012637  000004                          MOV     (SP)+,@#4       ;RESTORE LOC 4
1553  003364    032737  000010  001446                  BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
1554  003372    001422                                  BEQ     3$              ;BR IF NO.
1555  003374    104401  010017                          TYPE    ,MNEW           ;TYPE THE MESSAGE.
1556  003400    005000                                  CLR     R0              ;ZERO DATA LIGHTS
1557  003402    000000                                  HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1558  003404    027737  175630  001474                  CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
1559  003412    101404                                  BLOS    2$              ;BR IF NUMBER IS OK.
1560  003414    104401  007672                          TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
1561  003420    000000                                  HALT                    ;STOP EVERY THING.
1562  003422    000776                                  BR      .-2             ;RESTART THE PROGRAM AGAIN.
1563  003424    017737  175610  001470  2$:     MOV     @SWR,KMACTV     ;GET NEW DEVICE PATTERN
1564  003432    013700  001470                          MOV     KMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
1565  003436    000000                                  HALT                    ;CONTINUE DYNAMIC SWITCHES.
1566  003440    012700  000300          3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
1567  003444    012701  000302                          MOV     #302,R1         ;VECTOR AREA. 300-776
1568  003450    010120                  4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
1569  003452    005021                          CLR     (R1)+           ;IN VECTOR AREA.
1570  003454    022021                          CMP     (R0)+,(R1)+     ;POP POINTERS
1571  003456    022700  001000                  CMP     #1000,R0        ;ALL DONE??
1572  003462    001372                          BNE     4$              ;BR IF NO.
1573
1574                                            ;TEST START AND RESTART
1575                                            ;----------------------
1576
1577  003464    012706  001200          .BEGIN: MOV     #STACK,SP       ;SET UP STACK
1578  003470    013746  000006                  MOV     @#6,-(SP)       ;SAVE LOC 6
1579  003474    013746  000004                  MOV     @#4,-(SP)       ;SAVE LOC 4
1580  003500    005000                          CLR     R0              ;START AT 0
1581  003502    012737  003546  000004          MOV     #2$,@#4         ;SET UP FOR TIME OUT
1582  003510    005037  000006                  CLR     @#6             ;TO AUTOSIZE MEMORY
1583  003514    005720                  6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
1584  003516    022700  157776                  CMP     #157776,R0      ;IS IT AT LEAST 28K
1585  003522    001374                          BNE     6$              ;BR IF NO
1586  003524    162700  007776                  SUB     #7776,R0        ;SAVE 2K FOR MONITORS
1587  003530    010037  001466          7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
1588  003534    012637  000004                  MOV     (SP)+,@#4       ;RESTORE LOC 4
1589  003540    012637  000006                  MOV     (SP)+,@#6       ;RESTORE LOC 6
1590  003544    000413                          BR      10$             ;CONTINUE
1591  003546    022626                  2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
1592  003550    162700  000004                  SUB     #4,R0           ;GET LAST GOOD ADDRESS
1593  003554    162700  007776                  SUB     #7776,R0        ;SAVE 2K FOR MONITORS
1594  003560    022700  030000                  CMP     #30000,R0       ;IS IT 8K?
1595  003564    001361                          BNE     7$              ;BR IF NO
1596  003566    012700  037400                  MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
1597  003572    000756                          BR      7$              ;
1598  003574    012737  000340  177776  10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
1599  003602    032737  ^00004  001446          BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
1600  003610    001406                          BEQ     1$              ;BR IF NO LOCK DESIRED.
1601  003612    104401  007716                  TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
1602  003616    012737  000240  004146          MOV     #NOP,TTST       ;SET UP TO LOCK
1603  003624    000403                          BR      3$              ;CONTINUE ALONG.
1604  003626    013737  004360  004146  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
```

```
 1605  003634  012737  011606  001206  3$:     MOV     #CYCLE,$LPADR   ;START AT ''CYCLE'' FIND WHICH DEVICE TO TEST
 1606  003642  032737  000002  001446  4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
 1607  003650  001002                          BNE     5$              ;BR IF YES
 1608  003652  104401  007642                  TYPE    ,MR             ;TYPE R
 1609  003656  000177  175324          5$:     JMP     @$LPADR         ;START TESTING
```

```
                                        ;END OF PASS
                                        ;TYPE NAME OF TEST
                                        ;UPDATE PASS COUNT
                                        ;CHECK FOR EXIT TO ACT-11
                                        ;RESTART TEST

                                .SBTTL  END OF PASS ROUTINE

                                ;;*********************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO CYCLE

003662                          $EOP:
003662  000005                          RESET
003664  005237  001324                  INC     $PASS           ; INCREMENT THE PASS COUNT
003670  105037  001203                  CLRB    $ERFLG          ; CLEAR ERROR FLAG
003674  104401  007620                  TYPE    ,MEPASS         ; TYPE END PASS.
003700  104401  007745                  TYPE    ,MCSRX          ; TYPE "CSR"
003704  104417  004104                  CNVRT   ,XCSR           ; SHOW IT.
003710  104401  007753                  TYPE    ,MVECX          ; TYPE VECTOR.
003714  104417  004112                  CNVRT   ,XVEC           ; SHOW IT.
003720  104401  007761                  TYPE    ,MPASSX         ; TYPE " PASSES "
003724  104417  004120                  CNVRT   ,XPASS          ; SHOW IT.
003730  104401  007772                  TYPE    ,MERRX          ; TYPE " ERRORS "
003734  104417  004126                  CNVRT   ,XERR           ; SHOW IT.
003740  013700  001504                  MOV     MILK,R0         ; SET POINTER TO PASSCNT.
003744  013720  001324                  MOV     $PASS,(R0)+     ; SAVE THE PASS COUNT.
003750  013720  001212                  MOV     $ERTTL,(R0)+    ; SAVE ERROR COUNT
003754  013777  002060  176074          MOV     KMRLVL,@KMRVEC  ; RESTORE THE RECEIVER INTERRUPT VECTOR.
003762  005077  176072                  CLR     @KMRLVL         ; RESTORE RECEIVER LEVEL
003766  013777  002064  176066          MOV     KMTLVL,@KMTVEC  ; RESTORE THE TRANSMIT INTERRUPT VECTOR.
003774  005077  176064                  CLR     @KMTLVL         ; RESTORE TRANSMITTER LEVEL
004000  005337  001476                  DEC     SAVNUM          ; ALL DEVICE TESTED?
004004  001035                          BNE     $DOAGN          ; BRANCH IF NO.
004006  112737  000377  001511          MOVB    #377,QV.FLG     ; SET QUICK VERIFY FLAG.
004014  013737  001472  001476          MOV     KMNUM,SAVNUM    ; RESTORE DEVICE COUNT.
004022  005037  001216                  CLR     $ERRPC          ; CLEAR LAST ERROR PC
004026  005037  001310                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
004032  005237  001324                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
004036  042737  100000  001324          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004044  005327                          DEC     (PC)+           ;;LOOP?
004046  000001          $EOPCT: .WORD   1
004050  003013                  BGT     $DOAGN                  ;;YES
004052  012737                  MOV     (PC)+,@(PC)+            ;;RESTORE COUNTER
004054  000001          $ENDCT: .WORD   1
004056  004046                  $EOPCT
004060  013700  000042  $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
004064  001405                  BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
004066  000005                  RESET                           ;;CLEAR THE WORLD
004070  004710          $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
004072  000240                  NOP                             ;;SAVE ROOM
004074  000240                  NOP                             ;;FOR
004076  000240                  NOP                             ;;ACT11
004100                  $DOAGN:
004100  000137                  JMP     @(PC)+                  ;;RETURN
```

```
 1666  004102  011606                  $RTNAD: .WORD   CYCLE
 1667  004104  000001                  XCSR:   1
 1668  004106     006      002                 .BYTE   6,2
 1669  004110  002066                          KMCSR
 1670  004112  000001                  XVEC:   1
 1671  004114     004      002                 .BYTE   4,2
 1672  004116  002056                          KMRVEC
 1673  004120  000001                  XPASS:  1
 1674  004122     006      002                 .BYTE   6,2
 1675  004124  001324                          $PASS
 1676  004126  000001                  XERR:   1
 1677  004130     006      002                 .BYTE   6,2
 1678  004132  001212                          $ERTTL
 1679
 1680                                          ;SCOPE LOOP AND INTERATION HANDLER
 1681                                          ;--------------------------------
 1682
 1683                                  .SBTTL  SCOPE HANDLER ROUTINE
 1684
 1685                                  ;;*****************************************************************
 1686                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 1687                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 1688                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 1689                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 1690                                  ;*SW14=1         LOOP ON TEST
 1691                                  ;*SW11=1         INHIBIT ITERATIONS
 1692                                  ;*CALL
 1693                                  ;*      SCOPE           ;;SCOPE=IOT
 1694
 1695  004134                          $SCOPE:
 1696  004134  005037  001216                 CLR     $ERRPC                  ; CLEAR LAST ERROR PC
 1697  004140  023716  014142                 CMP     TST1+2,(SP)             ; IS THIS TEST #1 ?
 1698  004144  001413                         BEQ     $XTSTR                  ; IF SO DON'T LOOP.
 1699  004146  000406                  TTST:  BR      1$                      ;
 1700  004150  105777  175070                 TSTB    @$TKS                   ; KEYBOARD DONE ?
 1701  004154  100067                         BPL     $OVER                   ; IF NO DONT WAIT.
 1702  004156  017766  175064  177776         MOV     @$TKB,-2(SP)            ;
 1703  004164  032777  040000  175046  1$:    BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 1704  004172  001060                         BNE     $OVER           ;;YES IF SW14=1
 1705                                  ;#####START OF CODE FOR THE XOR TESTER#####
 1706  004174  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
 1707                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 1708  004176  013746  000004                 MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 1709  004202  012737  004222  000004         MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 1710  004210  005737  177060                 TST     @#177060        ;;TIME OUT ON XOR?
 1711  004214  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 1712  004220  000436                         BR      $SVLAD          ;;GO TO THE NEXT TEST
 1713  004222  022626                  5$:    CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 1714  004224  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 1715  004230  000441                         BR      $OVER           ;;LOOP ON THE PRESENT TEST
 1716  004232                          6$:;#####END OF CODE FOR THE XOR TESTER#####
 1717  004232  105737  001203          2$:    TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 1718  004236  001404                         BEQ     3$              ;;BR IF NO
 1719  004240  105037  001203          4$:    CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 1720  004244  005037  001310                 CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 1721  004250  032777  004000  174762  3$:    BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
```

```
 1722  004256  001011                          BNE     1$              ;;BR IF YES
 1723  004260  005737  001324                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
 1724  004264  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
 1725  004266  005237  001204                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
 1726  004272  023737  001310  001204          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 1727  004300  002015                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 1728  004302  012737  000001  001204  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 1729  004310  013737  004362  001310          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 1730  004316  105237  001202          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 1731  004322  113737  001202  001322          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 1732  004330  011637  001206                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 1733  004334  013777  001202  174700  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 1734  004342  013716  001206                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 1735  004346  005037  001444                  CLR     LOCK                    ; RESET LOCK ON DATA.
 1736  004352  013701  002066                  MOV     KMCSR,R1                ; R1 CONTAINS BASE KMC ADDRESS.
 1737  004356  000002                          RTI
 1738  004360  000406                  BRW:    .WORD   406
 1739  004362  000020                  $MXCNT: 20                      ;;MAX. NUMBER OF ITERATIONS
 1740
 1741                                          ;CHECK FOR FREEZE ON CURRENT DATA
 1742                                          ;-------------------------------
 1743
 1744  004364  004737  011340          .SCOP1: JSR     PC,CKSWR                ;CHECK FOR SOFT SWR
 1745  004370  032777  001000  174642          BIT     #SW09,@SWR      ;IS SW09=1(SET)?
 1746  004376  001405                          BEQ     1$              ;BR IF NOT SET.
 1747  004400  005737  001444                  TST     LOCK
 1748  004404  001402                          BEQ     1$
 1749  004406  013716  001444                  MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
 1750  004412  000002                  1$:     RTI                     ;GO BACK.
 1751
 1752                                          ;TELETYPE OUTPUT ROUTINE
 1753                                          ;----------------------
 1754
 1755                                  .SBTTL  TYPE ROUTINE
 1756
 1757                                  ;;**********************************************************************
 1758                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 1759                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 1760                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 1761                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 1762                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 1763                                  ;*
 1764                                  ;*CALL:
 1765                                  ;*1) USING A TRAP INSTRUCTION
 1766                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 1767                                  ;*OR
 1768                                  ;*      TYPE
 1769                                  ;*      MESADR
 1770                                  ;*
 1771
 1772  004414  105737  001257          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 1773  004420  100002                          BPL     1$              ;;BR IF YES
 1774  004422  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 1775  004424  000430                          BR      3$              ;;LEAVE
 1776  004426  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 1777  004430  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
```

```
 1778  004434  122737  000001  001336          CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
 1779  004442  001011                          BNE     62$               ;;NO,GO CHECK FOR APT CONSOLE
 1780  004444  132737  000100  001337          BITB    #APTSPOOL,$ENVM   ;;SPOOL MESSAGE TO APT
 1781  004452  001405                          BEQ     62$               ;;NO,GO CHECK FOR CONSOLE
 1782  004454  010037  004464                  MOV     R0,61$            ;;SETUP MESSAGE ADDRESS FOR APT
 1783  004460  004737  004704                  JSR     PC,$ATY3          ;;SPOOL MESSAGE TO APT
 1784  004464  000000                  61$:    .WORD   0                 ;;MESSAGE ADDRESS
 1785  004466  132737  000040  001337  62$:    BITB    #APTCSUP,$ENVM    ;;APT CONSOLE SUPPRESSED
 1786  004474  001003                          BNE     60$               ;;YES,SKIP TYPE OUT
 1787  004476  112046                  2$:     MOVB    (R0)+,-(SP)       ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 1788  004500  001005                          BNE     4$                ;;BR IF IT ISN'T THE TERMINATOR
 1789  004502  005726                          TST     (SP)+             ;;IF TERMINATOR POP IT OFF THE STACK
 1790  004504  012600                  60$:    MOV     (SP)+,R0          ;;RESTORE R0
 1791  004506  062716  000002          3$:     ADD     #2,(SP)           ;;ADJUST RETURN PC
 1792  004512  000002                          RTI                       ;;RETURN
 1793  004514  122716  000011          4$:     CMPB    #HT,(SP)          ;;BRANCH IF <HT>
 1794  004520  001430                          BEQ     8$
 1795  004522  122716  000200                  CMPB    #CRLF,(SP)        ;;BRANCH IF NOT <CRLF>
 1796  004526  001006                          BNE     5$
 1797  004530  005726                          TST     (SP)+             ;;POP  <CR><LF> EQUIV
 1798  004532  104401                          TYPE                      ;;TYPE A CR AND LF
 1799  004534  001313                          $CRLF
 1800  004536  105037  004672                  CLRB    $CHARCNT          ;;CLEAR CHARACTER COUNT
 1801  004542  000755                          BR      2$                ;;GET NEXT CHARACTER
 1802  004544  004737  004626          5$:     JSR     PC,$TYPEC         ;;GO TYPE THIS CHARACTER
 1803  004550  123726  001256          6$:     CMPB    $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
 1804  004554  001350                          BNE     2$                ;;IF NO GO GET NEXT CHAR.
 1805  004556  013746  001254                  MOV     $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
 1806                                                                    ;;AND THE NULL CHAR.
 1807  004562  105366  000001          7$:     DECB    1(SP)             ;;DOES A NULL NEED TO BE TYPED?
 1808  004566  002770                          BLT     6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
 1809  004570  004737  004626                  JSR     PC,$TYPEC         ;;GO TYPE A NULL
 1810  004574  105337  004672                  DECB    $CHARCNT          ;;DO NOT COUNT AS A COUNT
 1811  004600  000770                          BR      7$                ;;LOOP
 1812
 1813                                  ;HORIZONTAL TAB PROCESSOR
 1814
 1815  004602  112716  000040          8$:     MOVB    #' ,(SP)          ;;REPLACE TAB WITH SPACE
 1816  004606  004737  004626          9$:     JSR     PC,$TYPEC         ;;TYPE A SPACE
 1817  004612  132737  000007  004672          BITB    #7,$CHARCNT       ;;BRANCH IF NOT AT
 1818  004620  001372                          BNE     9$                ;;TAB STOP
 1819  004622  005726                          TST     (SP)+             ;;POP SPACE OFF STACK
 1820  004624  000724                          BR      2$                ;;GET NEXT CHARACTER
 1821  004626  105777  174416          $TYPEC: TSTB    @$TPS             ;;WAIT UNTIL PRINTER IS READY
 1822  004632  100375                          BPL     $TYPEC
 1823  004634  116677  000002  174410          MOVB    2(SP),@$TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 1824  004642  122766  000015  000002          CMPB    #CR,2(SP)         ;;IS CHARACTER A CARRIAGE RETURN?
 1825  004650  001003                          BNE     1$                ;;BRANCH IF NO
 1826  004652  105037  004672                  CLRB    $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
 1827  004656  000406                          BR      $TYPEX            ;;EXIT
 1828  004660  122766  000012  000002  1$:     CMPB    #LF,2(SP)         ;;IS CHARACTER A LINE FEED?
 1829  004666  001402                          BEQ     $TYPEX            ;;BRANCH IF YES
 1830  004670  105227                          INCB    (PC)+             ;;COUNT THE CHARACTER
 1831  004672  000000          $CHARCNT:.WORD  0                         ;;CHARACTER COUNT STORAGE
 1832  004674  000207          $TYPEX: RTS     PC
 1833
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;*****************************************************************
004676  112737  000001  005142  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
004704  112737  000001  005140  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
004712  000403                        BR      $ATYC
004714  112737  000001  005142  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
004722                          $ATYC:
004722  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
004724  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
004726  105737  005140                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
004732  001450                        BEQ     5$              ;;IF NOT:  BR
004734  122737  000001  001336        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
004742  001031                        BNE     3$              ;;IF NOT:  BR
004744  132737  000100  001337        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
004752  001425                        BEQ     3$              ;;IF NOT:  BR
004754  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
004760  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
004766  005737  001316          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
004772  001375                        BNE     1$              ;;IF NOT:  WAIT
004774  010037  001332                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
005000  105720                  2$:   TSTB    (R0)+           ;;FIND END OF MESSAGE
005002  001376                        BNE     2$
005004  163700  001332                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
005010  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
005012  010037  001334                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
005016  012737  000004  001316        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
005024  000413                        BR      5$
005026  017637  000004  005052  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
005034  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
005042  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
005046  004737  004414                JSR     PC,$TYPE        ;;CALL TYPE MACRO
005052  000000                  4$:   .WORD   0
005054                          5$:
005054  105737  005142          10$:  TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
005060  001416                        BEQ     12$             ;;IF NOT:  BR
005062  005737  001336                TST     $ENV            ;;RUNNING UNDER APT?
005066  001413                        BEQ     12$             ;;IF NOT:  BR
005070  005737  001316          11$:  TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
005074  001375                        BNE     11$             ;;IF NOT:  WAIT
005076  017637  000004  001320        MOV     @4(SP),$FATAL   ;;GET ERROR #
005104  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
005112  005237  001316                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
005116  105037  005142          12$:  CLRB    $FFLG           ;;CLEAR FATAL FLAG
005122  105037  005141                CLRB    $LFLG           ;;CLEAR LOG FLAG
005126  105037  005140                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
005132  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
005134  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
005136  000207                        RTS     PC              ;;RETURN
005140     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
005141     000          $LFLG:  .BYTE   0               ;;LOG FLAG
005142     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
            005144                  .EVEN
            000200      APTSIZE=200
            000001      APTENV=001
            000100      APTSPOOL=100
```

```
                        000040                  APTCSUP=040
                                                        ;-------------------------
                                                .SBTTL  TTY INPUT ROUTINE
                                                ;;**********************************************************************
                                                .ENABL  LSB
                                                .DSABL  LSB
                                                ;;**********************************************************************
                                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*CALL:
                                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                                ;
005144  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
005146  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
005154  105777  174064          1$:     TSTB    @$TKS           ;;WAIT FOR
005160  100375                          BPL     1$              ;;A CHARACTER
005162  117766  174060  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
005170  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
005176  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
005204  001013                          BNE     3$              ;;BRANCH IF NO
005206  105777  174032          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
005212  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
005214  117746  174026                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
005220  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
005224  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
005230  001366                          BNE     2$              ;;IF NOT DISCARD IT
005232  000750                          BR      1$              ;;YES, RESUME
005234  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
005242  002407                          BLT     4$              ;;BRANCH IF YES
005244  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
005252  003003                          BGT     4$              ;;BRANCH IF YES
005254  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
005262  000002                  4$:     RTI                     ;;GO BACK TO USER
                                                ;;**********************************************************************
                                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                                ;*CALL:
                                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
005264  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
005266  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
005270  012703  005520          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
005274  022703  005527          2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
005300  101456                          BLOS    4$              ;;BR IF YES
005302  104402                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
005304  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
005306  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
005312  001022                          BNE     5$              ;;BR IF NO
```

```
1946  005314  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
1947  005316  001007                          BNE     6$              ;;BR IF NO
1948  005320  112737  000134  005516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
1949  005326  104401  005516                  TYPE    ,9$
1950  005332  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
1951  005336  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
1952  005340  020327  005520                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
1953  005344  103434                          BLO     4$              ;;BR IF YES
1954  005346  111337  005516                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
1955  005352  104401  005516                  TYPE    ,9$             ;;GO TYPE
1956  005356  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
1957  005360  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
1958  005362  001406                          BEQ     7$              ;;BR IF NO
1959  005364  112737  000134  005516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
1960  005372  104401  005516                  TYPE    ,9$
1961  005376  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
1962  005400  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
1963  005404  001003                          BNE     8$              ;;BR IF NO
1964  005406  104401  005527                  TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
1965  005412  000726                          BR      1$              ;;GO START OVER
1966  005414  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
1967  005420  001011                          BNE     3$              ;;BRANCH IF NO
1968  005422  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
1969  005424  104401  001313                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
1970  005430  104401  005520                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
1971  005434  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
1972  005436  104401  001312          4$:     TYPE    ,$QUES          ;;TYPE A '?'
1973  005442  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
1974  005444  111337  005516          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
1975  005450  104401  005516                  TYPE    ,9$
1976  005454  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
1977  005460  001305                          BNE     2$              ;;LOOP IF NOT RETURN
1978  005462  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
1979  005466  104401  001314                  TYPE    ,$LF            ;;TYPE A LINE FEED
1980  005472  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
1981  005474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1982  005476  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1983  005500  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
1984  005506  012766  005520  000004          MOV     #$TTYIN,4(SP)
1985  005514  000002                          RTI                     ;;RETURN
1986  005516     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
1987  005517     000                          .BYTE   0               ;;TERMINATOR
1988  005520  000007                  $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
1989  005527     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
1990  005534  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
1991  005541     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
1992  005546  036440  000040
1993  005552  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
1994  005560  020075     000
1995          005564                  .EVEN
1996                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
1997
1998                                  ;;**************************************************************************
1999                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2000                                  ;*CHANGE IT TO BINARY.
2001                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
```

```
                        ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                        ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                        ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                        ;*CALL:
                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

005564  011646         $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
005566  016666 000004 000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
005574  010046                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005576  010146                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005600  010246                MOV     R2,-(SP)        ;;PUSH R2 ON STACK
005602  104403          1$:   RDLIN                   ;;READ AN ASCIZ LINE
005604  012600                MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
005606  010037 005712         MOV     R0,5$           ;;AND SAVE IT
005612  005001                CLR     R1              ;;CLEAR DATA WORD
005614  005002                CLR     R2
005616  112046          2$:   MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
005620  001420                BEQ     3$              ;;IF ZERO GET OUT
005622  122716 000060         CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
005626  003026                BGT     4$              ;;IS AN OCTAL DIGIT
005630  122716 000067         CMPB    #'7,(SP)
005634  002423                BLT     4$
005636  006301                ASL     R1              ;;*2
005640  006102                ROL     R2
005642  006301                ASL     R1              ;;*4
005644  006102                ROL     R2
005646  006301                ASL     R1              ;;*8
005650  006102                ROL     R2
005652  042716 177770         BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
005656  062601                ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
005660  000756                BR      2$              ;;LOOP
005662  005726          3$:   TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
005664  010166 000012         MOV     R1,12(SP)       ;;SAVE THE RESULT
005670  010237 005722         MOV     R2,$HIOCT
005674  012602                MOV     (SP)+,R2        ;;POP STACK INTO R2
005676  012601                MOV     (SP)+,R1        ;;POP STACK INTO R1
005700  012600                MOV     (SP)+,R0        ;;POP STACK INTO R0
005702  000002                RTI                     ;;RETURN
005704  005726          4$:   TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
005706  105010                CLRB    (R0)            ;;SET A TERMINATOR
005710  104401                TYPE                    ;;TYPE UP THRU THE BAD CHAR.
005712  000000          5$:   .WORD   0
005714  104401 001312         TYPE    ,$QUES          ;;"?" "CR" & "LF"
005720  000730                BR      1$              ;;TRY AGAIN
005722  000000         $HIOCT: .WORD  0               ;;HIGH ORDER BITS GO HERE

                        ;
                        ;    INPUT OCTAL NUMBER ROUTINE
                        ;-----------------------------------
                        ;

005724  010546         $INPUT: MOV    R5,-(SP)                ; SAVE REGISTER R5.
005726  016605 000002         MOV    2(SP),R5                ; GET FIRST PARAMETER ADDRESS.
005732  012537 005770         MOV    (R5)+,WHAT              ; GET MESSAGE ADDRESS.
005736  012537 006050         MOV    (R5)+,LOLIM             ; GET LOW LIMIT FOR THE  #
005742  012537 006052         MOV    (R5)+,HILIM             ; GET HIGH LIMIT FOR THE #.
```

```
2058  005746  012537  006054            MOV     (R5)+,WHERE             ; GET ADDRESS OF INBUFFER
2059  005752  112537  006056            MOVB    (R5)+,LOBITS            ; GET LOWMASK BITS.
2060  005756  112537  006057            MOVB    (R5)+,ADRCNT            ; GET # OF #'S TO BE GENERATED.
2061  005762  010566  000002            MOV     R5,2(SP)                ; SAVE THE RETURN ADDRESS.
2062  005766  104401              INLP1: TYPE                            ; TYPE THE MESSAGE.
2063  005770  000000              WHAT:  .WORD   0
2064  005772  104404                     RDOCT                          ; READ OCTAL # FROM KEYBOARD.
2065  005774  021637  006052            CMP     (SP),HILIM              ; IS IT IN HIGH LIMIT?
2066  006000  003003                     BGT     2$                      ; BRANCH IF NO.
2067  006002  021637  006050            CMP     (SP),LOLIM              ; IS IT MORE THAN LOW LIMIT.
2068  006006  002005                     BGE     3$                      ; BRANCH IF YES.
2069  006010  104401  001312      2$:    TYPE    ,$QUES                  ; TYPE  '' ? ''
2070  006014  104401  001313            TYPE    ,$CRLF                  ; TYPE <CR>,<LF>
2071  006020  000762                     BR      INLP1                   ;
2072  006022  013705  006054      3$:    MOV     WHERE,R5                ; GET BUFFER ADDRESS.
2073  006026  011625               4$:    MOV     (SP),(R5)+              ; SAVE THE # IN RIGHT PLACE.
2074  006030  062716  000002            ADD     #2,(SP)                 ; NEXT SEQUENTIAL NUMBER.
2075  006034  105337  006057            DECB    ADRCNT                  ; COUNT BY 1.
2076  006040  001372                     BNE     4$                      ; BRANCH IF NOT DONE.
2077  006042  005726                     TST     (SP)+                   ; POP THE STACK POINTER.
2078  006044  012605                     MOV     (SP)+,R5                ; POP THE REG.5
2079  006046  000002                     RTI
2080  006050  000000              LOLIM: .WORD   0
2081  006052  000000              HILIM: .WORD   0
2082  006054  000000              WHERE: .WORD   0
2083  006056     000              LOBITS:.BYTE   0
2084  006057     000              ADRCNT:.BYTE   0
2085
2086                                     ;  ADVANCE TO NEXT TEST HANDLER
2087                                     ;---------------------------------
2088                                     ;
2089  006060  013716  001442      .ADVANCE:      MOV     NEXT,(SP)               ; CRUNCH STACK WITH ADDRESSOF SCOPE CALL
2090  006064  005037  001444              CLR     LOCK                    ; RESET TIGHT LOOP ADDRESS
2091  006070  000002                     RTI                             ; CHECK TO SEE IF OLD TEST GETS REPEATED
2092
2093                                     ;SAVE PC OF TEST THAT FAILED AND R0-R5
2094                                     ;-------------------------------------
2095
2096  006072  016637  000004  001460  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
2097
2098                                     ;SAVE R0-R5
2099
2100  006100  010537  001274      SV05:  MOV     R5,$REG5        ;SAVE R5
2101  006104  010437  001272             MOV     R4,$REG4        ;SAVE R4
2102  006110  010337  001270             MOV     R3,$REG3        ;SAVE R3
2103  006114  010237  001266             MOV     R2,$REG2        ;SAVE R2
2104  006120  010137  001264             MOV     R1,$REG1        ;SAVE R1
2105  006124  010037  001262             MOV     R0,$REG0        ;SAVE R0
2106  006130  000002                     RTI                     ;LEAVE.
2107
2108                                     ;RESTORE R0-R5
2109
2110  006132  013700  001262      .RES05: MOV     $REG0,R0        ;RESTORE R0
2111  006136  013701  001264             MOV     $REG1,R1        ;RESTORE R1
2112  006142  013702  001266             MOV     $REG2,R2        ;RESTORE R2
2113  006146  013703  001270             MOV     $REG3,R3        ;RESTORE R3
```

```
 2114  006152  013704  001272                  MOV     $REG4,R4        ;RESTORE R4
 2115  006156  013705  001274                  MOV     $REG5,R5        ;RESTORE R5
 2116  006162  000002                          RTI                     ;LEAVE
 2117
 2118                                  ;       ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
 2119                                  ;       ;-----------------------------------------------------
 2120                                  ;
 2121  006164  104401  001313          .CONVR: TYPE    ,$CRLF
 2122  006170  010046                  .CNVRT: MOV     R0,-(SP)
 2123  006172  010146                          MOV     R1,-(SP)
 2124  006174  010346                          MOV     R3,-(SP)
 2125  006176  010446                          MOV     R4,-(SP)
 2126  006200  010546                          MOV     R5,-(SP)
 2127  006202  017601  000012                  MOV     @12(SP),R1
 2128  006206  062766  000002  000012          ADD     #2,12(SP)
 2129  006214  012137  006406                  MOV     (R1)+,WRDCNT
 2130  006220  112137  006410          1$:     MOVB    (R1)+,CHRCNT
 2131  006224  112137  006411                  MOVB    (R1)+,SPACNT
 2132  006230  013137  006412                  MOV     @(R1)+,BINWRD
 2133  006234  122737  000003  006410          CMPB    #3,CHRCNT
 2134  006242  001003                          BNE     2$
 2135  006244  042737  177400  006412          BIC     #177400,BINWRD
 2136  006252  013704  006412          2$:     MOV     BINWRD,R4
 2137  006256  113705  006410                  MOVB    CHRCNT,R5
 2138  006262  012700  011234                  MOV     #TEMP,R0
 2139  006266  010403                  3$:     MOV     R4,R3
 2140  006270  042703  177770                  BIC     #177770,R3
 2141  006274  062703  000060                  ADD     #060,R3
 2142  006300  110320                          MOVB    R3,(R0)+
 2143  006302  000241                          CLC
 2144  006304  006004                          ROR     R4
 2145  006306  000241                          CLC
 2146  006310  006004                          ROR     R4
 2147  006312  000241                          CLC
 2148  006314  006004                          ROR     R4
 2149  006316  005305                          DEC     R5
 2150  006320  001362                          BNE     3$
 2151  006322  012703  011276                  MOV     #MDATA,R3
 2152  006326  114023                  4$:     MOVB    -(R0),(R3)+
 2153  006330  105337  006410                  DECB    CHRCNT
 2154  006334  001374                          BNE     4$
 2155  006336  105737  006411                  TSTB    SPACNT
 2156  006342  001405                          BEQ     6$
 2157  006344  112723  000040          5$:     MOVB    #040,(R3)+
 2158  006350  105337  006411                  DECB    SPACNT
 2159  006354  001373                          BNE     5$
 2160  006356  105013                  6$:     CLRB    (R3)
 2161  006360  104401  011276                  TYPE    ,MDATA
 2162  006364  005337  006406                  DEC     WRDCNT
 2163  006370  001313                          BNE     1$
 2164  006372  012605                          MOV     (SP)+,R5
 2165  006374  012604                          MOV     (SP)+,R4
 2166  006376  012603                          MOV     (SP)+,R3
 2167  006400  012601                          MOV     (SP)+,R1
 2168  006402  012600                          MOV     (SP)+,R0
 2169  006404  000002                          RTI
```

```
2170  006406  000000                  WRDCNT: 0
2171  006410  000000                  CHRCNT: 0
2172          006411                  SPACNT=CHRCNT+1
2173  006412  000000                  BINWRD: 0
2174
2175
2176                                          ;TRAP DISPATCH SERVICE
2177                                          ;ARGUMENT OF TRAP IS EXTRACTED
2178                                          ;AND USED AS OFFSET TO OBTAIN POINTER
2179                                          ;TO SELECTED SUBROUTINE
2180
2181                                  .SBTTL  TRAP DECODER
2182
2183                                  ;;**********************************************************************
2184                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2185                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2186                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2187                                  ;*GO TO THAT ROUTINE.
2188
2189  006414  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2190  006416  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
2191  006422  005740                          TST     -(R0)           ;;BACKUP BY 2
2192  006424  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2193  006426  006300                          ASL     R0              ;;POSITION FOR INDEXING
2194  006430  016000  006450                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2195  006434  000200                          RTS     R0              ;;GO TO ROUTINE
2196
2197
2198                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2199
2200  006436  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2201  006440  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2202  006446  000002                          RTI                     ;;RESTORE THE PSW
2203
2204                                  .SBTTL  TRAP TABLE
2205
2206                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2207                                  ;*BY THE "TRAP" INSTRUCTION.
2208
2209                                  ;       ROUTINE
2210                                  ;       -------
2211  006450  006436                  $TRPAD: .WORD   $TRAP2
2212  006452  004414                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2213
2214
2215  006454  005144                          $RDCHR  ;;CALL=RDCHR    TRAP+2(104402)  TTY TYPEIN CHARACTER ROUTINE
2216  006456  005264                          $RDLIN  ;;CALL=RDLIN    TRAP+3(104403)  TTY TYPEIN STRING ROUTINE
2217  006460  005564                          $RDOCT  ;;CALL=RDOCT    TRAP+4(104404)  READ AN OCTAL NUMBER FROM TTY
2218  006462  004364                          .SCOP1  ;;CALL=SLOP1    TRAP+5(104405)  CALL TO LOOP ON CURRENT DATA HANDLER
2219  006464  006072                          .SAV05  ;;CALL=SAV05    TRAP+6(104406)  CALL TO REGISTER SAVE ROUTINE
2220  006466  006132                          .RES05  ;;CALL=RES05    TRAP+7(104407)  CALL TO REGISTER RESTORE ROUTINE
2221  006470  007362                          .MSTCLR ;;CALL=MSTCLR   TRAP+10(104410) CALL TO ISSUE A MASTER CLEAR
2222  006472  007332                          .DELAY  ;;CALL=DELAY    TRAP+11(104411) CALL TO DELAY
2223  006474  007400                          .ROMCLK ;;CALL=ROMCLK   TRAP+12(104412) CALL TO CLOCK ROM ONCE
2224  006476  007446                          .DATACLK        ;;CALL=DATACLK  TRAP+13(104413) CALL TO CLOCK DATA
2225  006500  007512                          .TIMER  ;;CALL=TIMER    TRAP+14(104414) CALL TO DELAY A CLOCK TICK
```

```
2226  006502  005724                        $INPUT  ;;CALL=INPUT    TRAP+15(104415) CALL TO OCTAL # INPUT ROUTINE
2227  006504  006164                        .CONVRT ;;CALL=CONVRT   TRAP+16(104416) CALL TO    .....
2228  006506  006170                        .CNVRT  ;;CALL=CNVRT    TRAP+17(104417) CALL TO    ......
2229  006510  006060                        .ADVANCE        ;;CALL=ADVANCE  TRAP+20(104420) CALL TO ADVANCE TO NEXT TEST
2230                                  ;
2231                                  ;-----------------------------------------------------------------
2232                                  ;:*****************************************************************
2233                                          ;ERROR HANDLER
2234                                          ;-------------
2235
2236  006512  004737  011340          $ERROR: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
2237  006516  032777  010000  172514          BIT     #SW12,@SWR      ;BELL ON ERROR?
2238  006524  001406                          BEQ     XBX             ;BR IF NO BELL
2239  006526  105777  172516                  TSTB    @$TPS           ;TTY READY.
2240  006532  100003                          BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
2241  006534  112777  000207  172510          MOVB    #207,@$TPB      ;PUSH A BELL AT THE TTY.
2242  006542  032777  020000  172470  XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
2243  006550  001107                          BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
2244  006552  021637  001216                  CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
2245  006556  001404                          BEQ     1$              ;BR IF YES
2246  006560  011637  001216                  MOV     (SP),$ERRPC     ;RECORD BEING HERE
2247  006564  105037  001203                  CLRB    $ERFLG          ;PREPARE HEADER
2248  006570  104406                  1$:     SAV05                   ;SAVE ALL PROC REGISTERS
2249  006572  011605                          MOV     (SP),R5         ;GET THE PC OF ERROR
2250  006574  162705  000002                  SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
2251  006600  011504                          MOV     (R5),R4         ;GET ERROR INSTRUCTION
2252  006602  110437  001214                  MOVB    R4,$ITEMB       ; COPY ERROR # FOR APT HANDLING
2253  006606  006304                          ASL     R4              ;MULT BY TWO
2254  006610  061504                          ADD     (R5),R4         ;DOUBLE IT
2255  006612  006304                          ASL     R4              ;MULT AGAIN
2256  006614  042704  177001                  BIC     #177001,R4      ;CLEAR JUNK
2257  006620  062704  001512                  ADD     #$ERRTB,R4      ;GET POINTER
2258  006624  012437  006740                  MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
2259  006630  012437  006752                  MOV     (R4)+,DATAHD    ;GET DATA HEADRER
2260  006634  011437  006764                  MOV     (R4),DATABP     ;GET DATA TABLE
2261  006640  105737  001203                  TSTB    $ERFLG          ;TYPE HEADREER
2262  006644  001403                          BEQ     TYPMSG          ;BR IF YES
2263  006646  005737  006764                  TST     DATABP          ;DOES DATA TABLE EXIST?
2264  006652  001040                          BNE     TYPDAT          ;BR IF YES.
2265  006654  104401  001313          TYPMSG: TYPE    ,$CRLF
2266  006660  104401  001313                  TYPE    ,$CRLF
2267  006664  005737  001444                  TST     LOCK
2268  006670  001402                          BEQ     1$
2269  006672  104401  010015                  TYPE    ,MASTEK
2270  006676  104401  010003          1$:     TYPE    ,MTSTN
2271  006702  104417  007120                  CNVRT   ,XTSTN          ;SHOW IT
2272  006706  104401  010072                  TYPE    ,MERRPC         ;TYPE PC.
2273  006712  104417  007112                  CNVRT   ,ERTABO         ;SHOW IT
2274  006716  104401  001313                  TYPE    ,$CRLF          ;GIVE A CR/LF
2275  006722  112737  177777  001203          MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
2276  006730  005737  006740                  TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
2277  006734  001402                          BEQ     WRKO.FM         ;BR IF NO.
2278  006736  104401                          TYPE                    ;TYPE
2279  006740  000000                  ERRMSG: 0                       ;    ERROR MESSAGE
2280  006742                          WRKO.FM:                        ;
2281  006742  005737  006752                  TST     DATAHD          ;DATA HEADER?
```

```
2282  006746  001402                        BEQ     TYPDAT          ;BR IF NO
2283  006750  104401                        TYPE                    ;TYPE
2284  006752  000000              DATAHD:   0                       ;    DATA HEADER
2285  006754  005737  006764      TYPDAT:   TST     DATABP          ;DATA TABLE?
2286  006760  001402                        BEQ     RESREG          ;BR IF NO.
2287  006762  104416                        CONVRT                  ;SHOW
2288  006764  000000              DATABP:   0                       ;    DATA TABLE
2289  006766  104407              RESREG:   RES05                   ;RESTORE PROC REGISTERS
2290  006770  122737  000001  001336  HALTS:  CMPB    #APTENV,$ENV    ; IS APT RUNNING ?
2291  006776  001007                        BNE     3$              ; SKIP APT CALL IF NOT.
2292  007000  113737  001214  007012        MOVB    $ITEMB,6$       ; COPY ERROR #.
2293  007006  004737  004714                JSR     PC,$ATY4        ; CALL APT SERVICES.
2294  007012  000000              6$:       .WORD   0               ; ERROR # GOES HERE.
2295  007014  000777              9$:       BR      9$              ; LOCK HERE.
2296  007016  022737  004070  000042  3$:   CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
2297  007024  001403                        BEQ     1$
2298  007026  005777  172206                TST     @SWR            ;HALT ON ERROR?
2299  007032  100005                        BPL     EXITER          ;BR IF NO HALT ON ERROR
2300  007034  010046              1$:       PUSHR0                  ;SAVE R0
2301  007036  016600  000002                MOV     2(SP),R0        ;SHOW ERROR PC IN DATA LIGHTS
2302  007042  000000                        HALT                    ;HALT
2303  007044  012600                        POPR0                   ;GET R0
2304  007046  005237  001212      EXITER:   INC     $ERTTL          ;UPDATE ERROR COUNT
2305  007052  032777  000400  172160        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
2306  007060  001007                        BNE     1$              ;BR IF YES
2307  007062  032777  002000  172150        BIT     #SW10,@SWR      ;GOTO NEXT TEST?
2308  007070  001407                        BEQ     2$              ;BR IF NO
2309  007072  013737  001442  001206        MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
2310  007100  012706  001200              1$:       MOV     #STACK,SP       ;RESET SP
2311  007104  000177  172076                JMP     @$LPADR         ;GOTO SPECIFIED TEST
2312  007110  000002              2$:       RTI                     ;$LPADR
2313  007112  000001              ERTABO:   1
2314  007114     006      002              .BYTE   6,2
2315  007116  001460                        SAVPC
2316  007120  000001              XTSTN:    1
2317  007122     003      002              .BYTE   3,2
2318  007124  001202                        $TSTNM
2319                                        ;ENTER HERE ON POWER FAILURE
2320                                        ;--------------------------
2321
2322                                  .SBTTL  POWER DOWN AND UP ROUTINES
2323
2324                                  ;;**************************************************************
2325                                  ;POWER DOWN ROUTINE
2326  007126  012737  007316  000024  $PWRDN:  MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2327  007134  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2328  007142  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2329  007144  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2330  007146  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2331  007150  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2332  007152  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2333  007154  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2334  007156  017746  172056                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
2335  007162  010637  007322                 MOV     SP,$SAVR6       ;;SAVE SP
2336  007166  012737  007200  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2337  007174  000000                         HALT
```

```
                        BR      .-2             ;;HANG UP

                ;;*****************************************************************
                ;POWER UP ROUTINE
007200  012737  007316  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
007206  013706  007322          MOV     $SAVR6,SP       ;;GET SP
007212  005037  007322          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
007216  005237  007322  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
007222  001375                  BNE     1$              ;;OF  WORD
007224  104401  007562          TYPE    ,MPFAIL         ;
007230  104417  007324          CNVRT   ,PFTAB          ;
007234  105037  001203          CLRB    $ERFLG          ;CLEAR ERROR FLAG.
007240  005037  001216          CLR     $ERRPC          ; CLEAR LAST ERROR PC
007244  013701  002066          MOV     KMCSR,R1        ; RESTORE DEVICE ADDRESS.
007250  005011                  CLR     (R1)            ; CLEAR THE CSR.
007252  104410                  MSTCLR                  ;
007254  012677  171760          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
007260  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
007262  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
007264  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
007266  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
007270  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
007272  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
007274  012737  007126  000024  MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
007302  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;PRIO:7
007310  104401                  TYPE                    ;;REPORT THE POWER FAILURE
007312  007562          $PWRMG: .WORD   MPFAIL          ;;POWER FAIL MESSAGE POINTER
007314  000002                  RTI
007316  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
007320  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
007322  000000          $SAVR6: 0                       ;;PUT THE SP HERE

007324  000001          PFTAB:  1
007326     003     002  .BYTE   3,2
007330  001202                  $TSTNM

007332                  .DELAY:
007332  012777  000020  172534          MOV     #20,@KMP04
007340  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
007342  121111                          121111                  ;POKE CLOCK DELAY BIT
007344                  1$:
007344  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
007346  121224                          121224                  ;PORT4 IBUS+11
007350  032777  000020  172516          BIT     #BIT4,@KMP04    ;IS CLOCK BIT SET?
007356  001772                          BEQ     1$              ;BR IF NO
007360  000002                          RTI

007362                  .MSTCLR:
007362  152777  000100  172500          BISB    #BIT6,@KMCSRH   ;SET MASTER CLEAR
007370  142777  000300  172472          BICB    #BIT6!BIT7,@KMCSRH  ;CLEAR MASTER CLEAR AND RUN
007376  000002                          RTI                     ;RETURN

007400                  .ROMCLK:
007400  152777  000002  172462          BISB    #BIT1,@KMCSRH   ;SET ROMI
007406  013677  172464                  MOV     @(SP)+,@KMP06   ;LOAD INSTRUCTION IN SEL6
007412  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
```

```
2394  007416  032777  000100  171614          BIT     #SW06,@SWR      ;HALT IF SW06 =1
2395  007424  001401                          BEQ     1$              ;BR IF SW06 =0
2396  007426  000000                          HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
2397  007430  152777  000003  172432  1$:     BISB    #BIT1!BIT0,@KMCSRH ;CLOCK INSTRUCTION
2398  007436  142777  000007  172424          BICB    #BIT2.BIT1!BIT0,@KMCSRH  ;CLEAR ROMO, ROMI, STEP
2399  007444  000002                          RTI
2400
2401  007446                          .DATACLK:
2402  007446  013637  011234                  MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
2403  007452  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
2404  007456  152777  000020  172404  1$:     BISB    #BIT4,@KMCSRH   ;SET STEP LU
2405  007464  027777  172376  172374          CMP     @KMCSR,@KMCSR   ;WASTE TIME
2406  007472  142777  000020  172370          BICB    #BIT4,@KMCSRH   ;CLEAR STEP LU
2407  007500  005337  011234                  DEC     TEMP            ;DEC TICK COUNT
2408  007504  001364                          BNE     1$              ;BR IF NOT DONE
2409  007506  000002                          RTI                     ;RETURN
2410  007510  000001                  3$:     .BLKW 1
2411
2412  007512                          .TIMER:
2413  007512  013637  011234                  MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
2414  007516  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
2415  007522                          1$:
2416  007522  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2417  007524  021364                          021364                  ;PORT4 IBUS* REG11
2418  007526  032777  000002  172340          BIT     #2,@KMP04       ;IS PGM CLOCK BIT CLEAR?
2419  007534  001772                          BEQ     1$              ;BR IF YES
2420  007536                          2$:
2421  007536  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2422  007540  021364                          021364                  ;PORT4 IBUS* REG11
2423  007542  032777  000002  172324          BIT     #2,@KMP04       ;IS PGM CLOCK BIT SET?
2424  007550  001372                          BNE     2$              ;BR IF YES
2425  007552  005337  011234                  DEC     TEMP            ;DEC COUNT
2426  007556  001361                          BNE     1$              ;BR IF NOT DONE
2427  007560  000002                          RTI                     ;RETURN
2428
2429  007562  050200  051127  043040  MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT TEST /
 (2)  007620  042600  042116  050040  MEPASS: .ASCIZ  <200>/END PASS CZKCF  /
 (2)  007642  051200     000          MR:     .ASCIZ  <200>/R/
 (2)  007645     200  047516  042040  MERR2:  .ASCIZ  <200>/NO DEVICES PRESENT./
 (2)  007672  044600  051516  043125  MERR3:  .ASCIZ  <200>/INSUFFICIENT DATA!/
 (2)  007716  046200  041517  020113  MLOCK:  .ASCIZ  <200>/LOCK ON SELECTED TEST/
 (2)  007745     103  051123  020072  MCSRX:  .ASCIZ   /CSR: /
 (2)  007753     126  041505  020072  MVECX:  .ASCIZ  /VEC: /
 (2)  007761     120  051501  042523  MPASSX: .ASCIZ  /PASSES: /
 (2)  007772  051105  047522  051522  MERRX:  .ASCIZ  /ERRORS: /
 (2)  010003     124  051505  020124  MTSTN:  .ASCIZ  /TEST NO: /
 (2)  010015     052     000          MASTEK: .ASCIZ  /*/
 (2)  010017     200  042523  020124  MNEW:   .ASCIZ  <200>/SET SWITCH REG TO KMC11'S DESIRED ACTIVE./
 (2)  010072  041520  020072     000  MERRPC: .ASCIZ  /PC: /
 (2)  010077     200  020040  020040  XHEAD:  .ASCII  <200>/            MAP OF KMC11 STATUS/
 (2)  010136  020200  020040  020040          .ASCII  <200>/            --------------------/
 (2)  010175     200  020040  041520          .ASCII  <200>/  PC      CSR     STAT1    STAT2    STAT3/
 (2)  010247     200  026455  026455          .ASCIZ  <200>/------   ------   ------   ------   ------/
 (2)  010323     200  047510  020127  NUM:    .ASCIZ  <200>/HOW MANY KMC11'S TO BE TESTED?/
 (2)  010363     200  051503  020122  CSR:    .ASCIZ  <200>/CSR ADDRESS?/
 (2)  010401     200  042526  052103  VEC:    .ASCIZ  <200>/VECTOR ADDRESS?/
```

```
   (2)  010422  041200  020122  051120  PRIO:    .ASCIZ  <200>/BR PRIORITY LEVEL? (4,5,6,7)?/
   (2)  010461     200  044127  041511  MODU:    .ASCIZ  <200>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYP
   (2)  010573     200  053523  052111  LINE:    .ASCIZ  <200>/SWITCH PAC#1 (DDCMP LINE #)?/
   (2)  010631     200  053523  052111  BM:      .ASCIZ  <200>/SWITCH PAC#2 (BM873 BOOT ADD)?/
   (2)  010671     200  051511  052040  CONN:    .ASCIZ  <200>/IS THE LOOP BACK CONNECTOR ON?/
   (2)  010731     200  047516  042040  NOACT:   .ASCIZ  <200>/NO DEVICES ARE SELECTED/
   (2)  010762  053600  044510  044103  MV35:    .ASCII  <200>/WHICH MODEM TYPE, TYPE "D" FOR KMC11-DA (RS232C),OR/
   (2)  011046  052200  050131  020105           .ASCIZ  <200>/TYPE "F" FOR KMC11-FA (V.35) ? /
   (2)
   (2)  011107     200  045600  041515  CONERR:  .ASCIZ  <200><200>/KMC11 AT NONSTANDARD ADDRESS  PC: /
   (2)  011154  042600  050130  041505  CNERR:   .ASCIZ  <200>/EXPECTED  FOUND/
   (2)  011175     040  045450  041515  KMCM:    .ASCIZ  / (KMC) /
   (2)          011206                  .EVEN
   (2)  011206  000005                  XSTATQ:  5
  2430  011210     006     003                   .BYTE   6,3
  2431  011212  001276                           $TMP0
  2432  011214     006     003                   .BYTE   6,3
  2433  011216  001300                           $TMP1
  2434  011220     006     003                   .BYTE   6,3
  2435  011222  001302                           $TMP2
  2436  011224     006     003                   .BYTE   6,3
  2437  011226  001304                           $TMP3
  2438  011230     006     002                   .BYTE   6,2
  2439  011232  001306                           $TMP4
  2440                                  .EVEN
  2441
  2442                                          ;BUFFERS FOR INPUT-OUTPUT
  2443
  2444  011234  000000                  TEMP:    0
  2445          011276                  .=.+40
  2446  011276  000000                  MDATA:   0
  2447          011340                  .=.+40
  2448
  2449
  2450                                          ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
  2451                                          ;REGISTER USING THE CONSOLE TERMINAL
  2452                                          ;-------------------------------------
  2453
  2454  011340  022737  000176  001240  CKSWR:   CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
  2455  011346  001075                          BNE     CKSWR5          ;BR IF NO
  2456  011350  132737  000001  001336          BITB    #1,$ENV         ; IS IT RUNNING UNDER APT?
  2457  011356  001071                          BNE     CKSWR5          ; EXIT IF YES.
  2458  011360  022777  000007  167660          CMP     #7,@$TKB        ;WAS CTRL G TYPED? (7 BIT ASCII)
  2459  011366  001404                          BEQ     1$              ;BR IF YES
  2460  011370  022777  000207  167650          CMP     #207,@$TKB      ;WAS CTRL G TYPED? (8 BIT ASCII)
  2461  011376  001061                          BNE     CKSWR5          ;BR IF NO
  2462  011400  010246                  1$:     MOV     R2,-(SP)        ;STORE R2
  2463  011402  010346                          MOV     R3,-(SP)        ;STORE R3
  2464  011404  010446                          MOV     R4,-(SP)        ;STORE R4
  2465  011406  012737  177777  011544          MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
  2466  011414  005002                  CKSWR1: CLR     R2              ;CLEAR NEW SWR CONTENTS
  2467  011416  012704  177777                  MOV     #-1,R4          ;SET FLAG TO ALL ONES
  2468  011422  104401  005541                  TYPE    ,$MSWR          ;TYPE "SWR= "
  2469  011426  104417                  CKSWR2: CNVRT                   ;TYPE OUT PRESENT CONTENTS
  2470  011430  011600                          SOFTSW                  ;OF SOFT SWITCH REGISTER
  2471  011432  104401  005552          CKSWR3: TYPE    ,$MNEW          ;TYPE "NEW? "
```

```
 2472  011436  004737  011546          CKSWR4: JSR     PC,INCHAR       ;GET RESPONSE
 2473  011442  022703  000015                  CMP     #15,R3          ;WAS IT A CR?
 2474  011446  001424                          BEQ     5$              ;BR IF YES
 2475  011450  022703  000012                  CMP     #12,R3          ;WAS IT A LF?
 2476  011454  001416                          BEQ     4$              ;BR IF YES
 2477  011456  022703  000025                  CMP     #25,R3          ;WAS IT CTRL U?
 2478  011462  001754                          BEQ     CKSWR1          ;BR IF YES(START OVER)
 2479  011464  022703  000007                  CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
 2480  011470  001762                          BEQ     CKSWR4
 2481  011472  005004                          CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
 2482  011474  042703  177770                  BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
 2483  011500  006302                          ASL     R2              ;SHIFT R2 3 TIMES
 2484  011502  006302                          ASL     R2
 2485  011504  006302                          ASL     R2
 2486  011506  050302                          BIS     R3,R2           ;ADD LAST DIGIT
 2487  011510  000752                          BR      CKSWR4          ;GET NEXT CHARACTER
 2488  011512  012766  002402  000006  4$:     MOV     #.START,6(SP)   ;LF WAS TYPED SO GO TO START
 2489  011520  005704                  5$:     TST     R4              ;IS FLAG CLEAR?
 2490  011522  001002                          BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
 2491  011524  010277  167510                  MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
 2492  011530  005037  011544          6$:     CLR     SWFLG           ;CLEAR TYPEOUT FLAG
 2493  011534  012604                          MOV     (SP)+,R4        ;RESTORE R4
 2494  011536  012603                          MOV     (SP)+,R3        ;RESTORE R3
 2495  011540  012602                          MOV     (SP)+,R2        ;RESTORE R2
 2496  011542  000207          CKSWR5: RTS     PC                      ;RETURN
 2497
 2498  011544  000000          SWFLG:  0
 2499
 2500  011546  105777  167472  INCHAR: TSTB    @$TKS
 2501  011552  100375                  BPL     .-4
 2502  011554  017703  167466          MOV     @$TKB,R3
 2503  011560  105777  167464          TSTB    @$TPS
 2504  011564  100375                  BPL     .-4
 2505  011566  010377  167460          MOV     R3,@$TPB
 2506  011572  042703  000200          BIC     #BIT7,R3
 2507  011576  000207                  RTS     PC
 2508
 2509  011600  000001          SOFTSW: 1
 2510  011602     006     002          .BYTE   6,2
 2511  011604  000176                  SWREG
```

```
 2512
 2513                                           ;
 2514                                           ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 KMC11'S
 2515                                           ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
 2516                                           ;AND RUNS THE SPECIFIED KMC11'S.   THIS ROUTINE *MUST*
 2517                                           ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
 2518                                           ;SETUP NECESSARY.
 2519                                           ;
 2520
 2521  011606  005737  001470          CYCLE:  TST     KMACTV          ;ARE ANY KMC11'S TO BE TESTED?
 2522  011612  001004                          BNE     1$              ;BR IF OK.
 2523  011614  104401  010731                  TYPE    ,NOACT          ;NO KMC11'S SELECTED!!
 2524  011620  000000                          HALT                    ;STOP THE SHOW.
 2525  011622  000776                          BR      .-2             ;DISQUALIFY CONT. SW.
 2526  011624  000241                  1$:     CLC                     ;CLEAR PROC. CARRY BIT.
 2527  011626  006137  001500                  ROL     RUN             ;UPDATE POINTER
 2528  011632  005537  001500                  ADC     RUN             ;CATCH CARRY FROM RUN
 2529  011636  062737  000004  001504          ADD     #4,MILK         ;UPDATE POINTER
 2530  011644  062737  000010  001502          ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
 2531  011652  022737  002300  001502          CMP     #KM.MAP+200,CREAM
 2532  011660  001006                          BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
 2533  011662  012737  002100  001502          MOV     #KM.MAP,CREAM   ;RESET ADDRESS POINTER.
 2534  011670  012737  002302  001504          MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
 2535  011676  033737  001500  001470  2$:     BIT     RUN,KMACTV      ;IS THIS ONE ACTIVE?
 2536  011704  001747                          BEQ     1$              ;BR IF NO
 2537  011706  013700  001502                  MOV     CREAM,R0        ;GET ADDRESS POINTER
 2538  011712  013702  001504                  MOV     MILK,R2         ;GET PASS COUNT POINTER
 2539  011716  012037  002066                  MOV     (R0)+,KMCSR     ;LOAD SYSTEM CTRL. REG
 2540  011722  011037  002056                  MOV     (R0),KMRVEC     ;LOAD VECTOR
 2541  011726  042737  177000  002056          BIC     #177000,KMRVEC  ;CLEAR UNWANTED BITS
 2542  011734  012037  002050                  MOV     (R0)+,STAT1     ;LOAD STAT1
 2543  011740  012037  002052                  MOV     (R0)+,STAT2     ;LOAD STAT2
 2544  011744  012037  002054                  MOV     (R0)+,STAT3     ;LOAD STAT3
 2545  011750  012237  001324                  MOV     (R2)+,$PASS     ;LOAD PASS COUNT
 2546  011754  012237  001212                  MOV     (R2)+,$ERTTL    ;LOAD ERROR COUNT
 2547  011760  012700  000002                  MOV     #2,R0           ;SAVE CORE THIS WAY!
 2548  011764  013737  002066  002070          MOV     KMCSR,KMCSRH
 2549  011772  005237  002070                  INC     KMCSRH
 2550  011776  013737  002070  002072          MOV     KMCSRH,KMCTL
 2551  012004  005237  002072                  INC     KMCTL
 2552  012010  013737  002072  002074          MOV     KMCTL,KMP04
 2553  012016  060037  002074                  ADD     R0,KMP04
 2554  012022  013737  002074  002076          MOV     KMP04,KMP06
 2555  012030  060037  002076                  ADD     R0,KMP06
 2556
 2557  012034  013737  002056  002060          MOV     KMRVEC,KMRLVL   ;PTY LVL
 2558  012042  060037  002060                  ADD     R0,KMRLVL       ;
 2559  012046  013737  002060  002062          MOV     KMRLVL,KMTVEC   ;TX VEC
 2560  012054  060037  002062                  ADD     R0,KMTVEC       ;
 2561  012060  013737  002062  002064          MOV     KMTVEC,KMTLVL   ;TX LVL
 2562  012066  060037  002064                  ADD     R0,KMTLVL
 2563
 2564  012072  032737  000002  001446          BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
 2565  012100  001447                          BEQ     7$              ;BR IF NO
 2566  012102                          4$:
 2567  012102  005737  000042                  TST     @#42            ;RUNNING IN AUTO MODE?
```

```
2568  012106  001044                        BNE     7$              ;BR IF YES
2569  012110  104401  001313                TYPE    ,$CRLF
2570  012114  104415                       INPUT
2571  012116  010003                       MTSTN
2572  012120  000001                       1
2573  012122  001000                       1000
2574  012124  001202                       $TSTNM
2575  012126     000              .BYTE   0
2576  012127     001              .BYTE   1
2577  012130  012700  014140              MOV     #TST1,R0
2578  012134  022710              5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
2579  012136  012737                       MOV     (PC)+,@(PC)+
2580  012140  001020                       BNE     6$              ;BR IF NOT SAME
2581  012142  023760  001202  000002       CMP     $TSTNM,2(R0)    ;DOES $TSTNM MATCH?
2582  012150  001014                       BNE     6$              ;BR IF NO
2583  012152  022760  001202  000004       CMP     #$TSTNM,4(R0)   ;IS LAST WORD OK?
2584  012160  001010                       BNE     6$              ;BR IF NO
2585  012162  010037  001206              MOV     R0,$LPADR       ;IT IS A LEGAL TEST SO DO IT
2586  012166  104401  007642              TYPE    ,MR
2587  012172  042737  000002  001446       BIC     #SW01,STRTSW
2588  012200  000412                       BR      8$
2589  012202  005720              6$:     TST     (R0)+           ;POP R0
2590  012204  020027  027716              CMP     R0,#TLAST+10    ;AT END YET?
2591  012210  001351                       BNE     5$              ;BR IF NO
2592  012212  104401  001312              TYPE    ,$QUES          ;YES ILLEGAL TEST NO.
2593  012216  000731                       BR      4$              ;TRY AGAIN
2594
2595  012220  012737  014140  001206  7$:     MOV     #TST1,$LPADR    ;PREPARE $LPADR ADDRESS
2596  012226  013701  002066          8$:     MOV     KMCSR,R1        ;R1 = BASE KMC11 ADDRESS
2597  012232  000177  166750              JMP     @$LPADR         ;GO START TESTING.
2598
2599
2600                                        ;ROUTINE USED TO "AUTO SIZE" THE KMC11
2601                                        ;CSR AND VECTOR.
2602                                        ;NOTE:  THE CSR MAY BE ANY WHERE IN THE FLOATING
2603                                        ;       ADDRESS RANGE (160000:164000)
2604                                        ;       AND THE VECTOR MAY BE ANY WHERE IN THE
2605                                        ;       FLOATING VECTOR RANGE (300:770)
2606                                        ;
2607
2608  012236                        AUTO.SIZE:
2609  012236  000005                       RESET                   ;INSURE A BUS INIT.
2610  012240  012702  002100      CSRMAP: MOV     #KM.MAP,R2      ;LOAD MAP POINTER.
2611  012244  005022              1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
2612  012246  022702  002300              CMP     #KM.END,R2      ;ALL DONE?
2613  012252  001374                       BNE     1$              ;BR IF NO
2614  012254  005037  001472              CLR     KMNUM           ;SET OCTAL NUMBER OF KMC11'S TO 0
2615  012260  012702  002100              MOV     #KM.MAP,R2      ;R2 POINTS TO KMC MAP
2616  012264  005037  001470              CLR     KMACTV          ;CLEAR ACTIVE
2617  012270  032737  000001  001446       BIT     #SW00,STRTSW    ;QUESTIONS?
2618  012276  001002                       BNE     .+6             ;BR IF YES
2619  012300  000137  012740              JMP     7$              ;IF NO SKIP QUESTIONS
2620  012304  012737  000001  001306       MOV     #1,$TMP4        ;START WITH 1
2621  012312  104415                       INPUT
2622  012314  010323                       NUM
2623  012316  000001                       1
```

```
2624  012320  000020                        16.
2625  012322  001302                        $TMP2
2626  012324     000                        .BYTE   0
2627  012325     001                        .BYTE   1
2628  012326  013737  001302  001472        MOV     $TMP2,KMNUM     ;KMNUM = HOW MANY
2629  012334  104401  001313        12$:    TYPE    ,$CRLF
2630  012340  104416                        CONVRT                  ;TYPE WHICH KMC IS BEING DONE
2631  012342  013372                        WHICH                   ;$TMP4 IS WHICH KMC
2632  012344  005237  001306                INC     $TMP4
2633  012350  104415                        INPUT
2634  012352  010363                        CSR
2635  012354  160000                        160000
2636  012356  164000                        164000
2637  012360  001304                        $TMP3
2638  012362     000                        .BYTE   0
2639  012363     001                        .BYTE   1
2640  012364  013722  001304                MOV     $TMP3,(R2)+     ;STORE CSR IN MAP
2641  012370  104415                        INPUT
2642  012372  010401                        VEC
2643  012374  000000                        0
2644  012376  000776                        776
2645  012400  001304                        $TMP3
2646  012402     000                        .BYTE   0
2647  012403     001                        .BYTE   1
2648  012404  013712  001304                MOV     $TMP3,(R2)      ;STORE VECTOR IN MAP
2649  012410  104401        10$:    TYPE
2650  012412  010422                        PRIO                    ;ASK WHAT BR LEVEL
2651  012414  004737  013664                JSR     PC,INTTY        ;GET RESPONSE
2652  012420  022703  000024                CMP     #24,R3          ;
2653  012424  101014                        BHI     50$             ;BR IF LESS THAN 4
2654  012426  022703  000027                CMP     #27,R3          ;
2655  012432  103411                        BLO     50$             ;BR IF GREATER THAN 7
2656  012434  012704  000011                MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
2657  012440  006303                        ASL     R3              ;SHIFT R3 LEFT
2658  012442  005304                        DEC     R4              ;DEC SHIFT COUNT
2659  012444  001375                        BNE     .-4             ;BR IF NOT DONE
2660  012446  042703  170777                BIC     #170777,R3      ;BIC UNWANTED BITS
2661  012452  050312                        BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
2662  012454  000403                        BR      8$              ;CONTINUE
2663  012456  104401        50$:    TYPE
2664  012460  001312                        $QUES                   ;RESPONSE IS OUT OF LIMITS
2665  012462  000752                        BR      10$             ;TRY AGAIN
2666  012464                8$:
2667  012464                9$:
2668  012464                16$:
2669
2670  012464  104401                        TYPE
2671  012466  010461                        MODU                    ;ASK WHICH LINE UNIT
2672  012470  004737  013664                JSR     PC,INTTY        ;GET REPLY
2673  012474  022703  000021                CMP     #21,R3          ;''1''
2674  012500  001417                        BEQ     30$
2675  012502  022703  000022                CMP     #22,R3          ;''2''
2676  012506  001412                        BEQ     31$
2677  012510  022703  000116                CMP     #116,R3         ;''N''
2678  012514  001403                        BEQ     32$
2679  012516  104401                        TYPE
```

```
2680  012520  001312                          $QUES                   ;IF NOT A 1,2 OR N TYPE ''?''
2681  012522  000760                          BR      16$             ;TRY AGIAN
2682  012524  052722  010000          32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
2683  012530  022222                          CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
2684  012532  000475                          BR      33$
2685  012534  052712  020000          31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
2686  012540  104401                  30$:    TYPE
2687  012542  010671                          CONN                    ;ASK IF LOOP-BACK IS ON
2688  012544  004737  013664                  JSR     PC,INTTY        ;GET REPLY
2689  012550  022703  000131                  CMP     #131,R3         ;Y
2690  012554  001406                          BEQ     17$
2691  012556  022703  000116                  CMP     #116,R3         ;N
2692  012562  001436                          BEQ     18$
2693  012564  104401                          TYPE
2694  012566  001312                          $QUES                   ;IF NOT Y OR N TYPE ''?''
2695  012570  000763                          BR      30$             ;TRY AGAIN
2696  012572  052722  040000          17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
2697  012576  032762  020000  177776          BIT     #BIT13,-2(R2)   ; M8202?
2698  012604  001027                          BNE     19$             ; BR IF YES.
2699  012606                          440$:
2700
2701  012606  104401                          TYPE            ; ASK QUESTION
2702  012610  010762                          MV35            ; ABOUT MODEM TTYPE
2703  012612  004737  013664                  JSR     PC,INTTY        ; GET ANSWER.
2704  012616  122703  000104                  CMPB    #'D,R3          ; IS IT DMC11-DA?
2705  012622  001004                          BNE     442$            ; NO.
2706
2707  012624  042762  000004  000002          BIC     #BIT2,2(R2)     ; YES INDICATE IT IN STAT3
2708  012632  000411                          BR      441$
2709
2710  012634  122703  000106          442$:   CMPB    #'F,R3          ;IS IT A DMC11-FA (V.35)?
2711  012640  001403                          BEQ     443$            ; YES-TAKE CARE OF IT.
2712
2713  012642  104401                          TYPE                    ; NO ASK OPERATER WHATS GOING ON.
2714  012644  001312                  $QUES
2715  012646  000757                          BR      440$            ; REASK QUESTION
2716
2717  012650  052762  000004  000002  443$:   BIS     #BIT2,2(R2)     ; YES V.35, RECORD IN STAT3
2718
2719  012656                          441$:                           ; END DECISION POINT.
2720  012656  000402                          BR      19$
2721  012660  042722  040000          18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
2722  012664                          19$:
2723  012664  104415                          INPUT
2724  012666  010573                          LINE
2725  012670  000000                          0
2726  012672  000377                          377
2727  012674  001304                          $TMP3
2728  012676     000                          .BYTE   0
2729  012677     001                          .BYTE   1
2730  012700  113722  001304                  MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
2731  012704  104415                          INPUT
2732  012706  010631                          BM
2733  012710  000000                          0
2734  012712  000377                          377
2735  012714  001304                          $TMP3
```

```
 2736  012716  000                          .BYTE   0
 2737  012717  001                          .BYTE   1
 2738  012720  113722  001304               MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
 2739  012724  005722                       TST     (R2)+           ;POP OVER STAT3
 2740  012726  005337  001302       33$:    DEC     $TMP2           ;DEC KMC COUNT
 2741  012732  001200                       BNE     12$             ;BR IF MORE TO DO
 2742  012734  000137  013272               JMP     13$             ;CONTINUE
 2743  012740  012701  160000       7$:     MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
 2744  012744  012737  013364  000004       MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
 2745  012752  005011               2$:     CLR     (R1)            ;CLEAR SEL0
 2746  012754  005711                       TST     (R1)            ;IF KMC11 KMCSR S/B 0
 2747  012756  001135                       BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO KMC11
 2748  012760  005061  000006               CLR     6(R1)           ;CLEAR SEL6
 2749  012764  005761  000006               TST     6(R1)           ;IF KMC11 THEN KMRIC S/B -0!
 2750  012770  001130                       BNE     3$              ;BR IF NOT KMC11
 2751  012772  012711  002000               MOV     #BIT10,(R1)     ;SET ROMO
 2752  012776  005061  000004               CLR     4(R1)           ;CLEAR SEL4
 2753  013002  012761  125252  000006       MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
 2754  013010  052711  020000               BIS     #BIT13,(R1)     ;WRITE IT!
 2755  013014  022761  125252  000004       CMP     #125252,4(R1)   ;WAS IT WRITTEN?
 2756  013022  001113                       BNE     3$              ;IF NO IT IS NOT CRAM
 2757                                ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A KMC11 CSR ADDRESS.
 2758  013024                       21$:
 2759  013024  010122               22$:    MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
 2760  013026  012711  001000       15$:    MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
 2761  013032  005061  000004               CLR     4(R1)           ;CLEAR PORT4
 2762  013036  012761  122113  000006       MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
 2763  013044  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
 2764  013050  012761  021264  000006       MOV     #021264,6(R1)   ;LOAD INSTRUCTION
 2765  013056  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
 2766  013062  122761  000377  000004       CMPB    #377,4(R1)      ;IS IT ALL ONES?
 2767  013070  001003                       BNE     .+10            ;BR IF NO
 2768  013072  052712  010000               BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
 2769  013076  000436                       BR      20$
 2770  013100  032761  000002  000004       BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
 2771  013106  001403                       BEQ     .+10            ;BR IF M9201
 2772  013110  052712  060000               BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
 2773  013114  000427                       BR      20$             ;CONNECTOR ON)
 2774  013116  032761  000010  000004       BIT     #BIT3,4(R1)     ;IS MRDY SET
 2775  013124  001023                       BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
 2776  013126  012761  000100  000004       MOV     #BIT6,4(R1)     ;LOAD PORT4
 2777  013134  012761  122113  000006       MOV     #122113,6(R1)   ;LOAD INSTRUCTION
 2778  013142  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
 2779  013146  012761  021264  000006       MOV     #021264,6(R1)   ;LOAD INSTRUCTION
 2780  013154  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
 2781  013160  032761  000010  000004       BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
 2782  013166  001402                       BEQ     20$             ;BR IF NO CONNECTOR
 2783  013170  052712  040000               BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
 2784  013174  005722               20$:    TST     (R2)+           ;POP POINTER
 2785  013176  012761  021324  000006       MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
 2786  013204  012711  001400               MOV     #BIT9!BIT8,(R1) ;PORT4_LU 15
 2787  013210  156122  000004               BISB    4(R1),(R2)+     ;STORE DDCMP LINE # IN TABLE
 2788  013214  012761  021344  000006       MOV     #021344,6(R1)   ;PORT6_INSTRUCTION
 2789  013222  012711  001400               MOV     #BIT8!BIT9,(R1) ;CLOCK_INSTR.
 2790  013226  156122  000004               BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
 2791  013232  005722                       TST     (R2)+           ;POP OVER STAT3
```

```
2792  013234  005011                          CLR     (R1)            ;CLEAR ROMI
2793  013236  005237  001472                  INC     KMNUM           ;UPDATE DEVICE COUNTER
2794  013242  022737  000020  001472          CMP     #20,KMNUM       ;ARE MAX. NO. OF DEV FOUND?
2795  013250  001410                          BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
2796  013252  005011                  3$:     CLR     (R1)            ;CLEAR BIT 10
2797  013254  005061  000006                  CLR     6(R1)           ;CLEAR SEL 6
2798  013260  062701  000010          14$:    ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
2799  013264  022701  164000                  CMP     #164000,R1
2800  013270  001230                          BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
2801  013272  005037  001470          13$:    CLR     KMACTV
2802  013276  005737  001472                  TST     KMNUM           ;WERE ANY KMC11'S FOUND AT ALL?
2803  013302  001423                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO KMC11'S IN THIS SYS.
2804  013304  013701  001472                  MOV     KMNUM,R1
2805  013310  010137  001476                  MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
2806  013314  000241                  4$:     CLC
2807  013316  006137  001470                  ROL     KMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
2808  013322  005237  001470                  INC     KMACTV          ;SET THE BIT
2809  013326  005301                          DEC     R1
2810  013330  001371                          BNE     4$              ;BR IF MORE TO GENERATE
2811  013332  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAP VECTOR
2812  013340  013737  001470  001474          MOV     KMACTV,SAVACT   ;SAVE ACTIVE REGISTER
2813  013346  000137  013400                  JMP     VECMAP          ;GO FIND THE VECTOR NOW.
2814  013352  104401  007645          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO KMC11'S FOUND.
2815  013356  005000                          CLR     R0              ;MAKE DATA LIGHTS ZERO
2816  013360  000000                          HALT                    ;STOP THE SHOW
2817  013362  000776                          BR      .-2             ;DISABLE CONT. SW.
2818  013364  012716  013260          6$:     MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
2819  013370  000002                          RTI                     ;RETURN TO MAINSTREAM
2820
2821  013372  000001                  WHICH:  1
2822  013374     002     002                  .BYTE   2,2
2823  013376  001306                          $TMP4
2824
2825  013400  032737  000001  001446  VECMAP: BIT     #SW00,STRTSW
2826  013406  001114                          BNE     5$
2827  013410  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
2828  013416  012737  013572  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
2829  013424  012702  002100                  MOV     #KM.MAP,R2      ;SET SOFTWARE POINTER
2830  013430  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
2831  013434  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
2832  013440  010120                  1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
2833  013442  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
2834  013446  022021                          CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
2835  013450  020127  001000                  CMP     R1,#1000
2836  013454  101771                          BLOS    1$              ;BR IF MORE TO FILL
2837  013456  013737  001470  001276          MOV     KMACTV,$TMP0    ;STORE TEMPORALLY
2838  013464  006037  001276          2$:     ROR     $TMP0           ;BRING OUT A BIT
2839  013470  103063                          BCC     5$              ;BR IF ALL DONE
2840  013472  012704  000012                  MOV     #12,R4          ;R4 IS INDEX REGISTER
2841  013476  016437  013650  177776          MOV     BRLVL(R4),PS    ;SET PS TO 7
2842  013504  011201                          MOV     (R2),R1
2843  013506  012761  000200  000004          MOV     #200,4(R1)
2844  013514  012711  001000                  MOV     #BIT9,(R1)      ;SET ROMI
2845  013520  012761  121111  000006          MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
2846  013526  012711  001400                  MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
2847  013532  105200                  7$:     INCB    R0              ;STALL
```

```
2848  013534  001376                          BNE     .-2             ;FOR TIME TO INTERUPT
2849  013536  162704  000002                  SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
2850  013542  001404                          BEQ     6$              ;BR IF R4 = 0
2851  013544  016437  013650  177776          MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
2852  013552  000767                          BR      7$              ;BR TO DELAY
2853  013554  052762  005300  000002  6$:     BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX KMC11 LATER
2854  013562  005011                  3$:     CLR     (R1)            ;CLEAR FOMI
2855  013564  062702  000010                  ADD     #10,R2          ;POP SOFTWARE POINTER
2856  013570  000735                          BR      2$              ;KEEP GOING
2857  013572  051662  000002          4$:     BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
2858  013576  042762  000007  000002          BIC     #7,2(R2)        ;CLEAR JUNK
2859  013604  016405  013652                  MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF KMC11
2860  013610  006305                          ASL     R5              ;SHIFT LEVEL 4 PLACES
2861  013612  006305                          ASL     R5              ;TO THE LEFT FOR THE
2862  013614  006305                          ASL     R5              ;STATUS TABLE
2863  013616  006305                          ASL     R5
2864  013620  042705  170777                  BIC     #170777,R5      ;CLEAR UNWANTED BITS
2865  013624  050562  000002                  BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
2866  013630  022626                          CMP     (SP)+,(SP)+     ;POP ICT JUNK OFF STACK
2867  013632  012716  013562                  MOV     #3$,(SP)        ;SET FOR RETURN
2868  013636  000002                          RTI
2869  013640  012737  004134  000020  5$:     MOV     #$SCOPE,@#20    ; RESTORE SCOPE VECTOR
2870  013646  000207                          RTS     PC              ;ALL DONE WITH "AUTO SIZING"
2871
2872  013650  000000                  BRLVL:  PR0     ;LEVEL 0
2873  013652  000000                          PR0     ;LEVEL 0
2874  013654  000200                          PR4     ;LEVEL 4
2875  013656  000240                          PR5     ;LEVEL 5
2876  013660  000300                          PR6     ;LEVEL 6
2877  013662  000340                          PR7     ;LEVEL 7
2878
2879
2880  013664  105777  165354          INTTY:  TSTB    @$TKS           ;WAIT FOR DONE
2881  013670  100375                          BPL     .-4
2882  013672  017703  165350                  MOV     @$TKB,R3        ;PUT CHAR IN R3
2883  013676  105777  165346                  TSTB    @$TPS           ;WAIT UNTIL PRINTER IS READY
2884  013702  100375                          BPL     .-4
2885  013704  010377  165342                  MOV     R3,@$TPB        ;ECHO CHAR
2886  013710  042703  000240                  BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
2887  013714  000207                          RTS     PC              ;RETURN
2888
2889  013716                          APT.SIZE:
2890  013716  000005                          RESET
2891  013720  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2892  013722  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2893  013724  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2894  013726  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2895  013730  005037  014132                  CLR     VECTR           ; CLEAR THE LOCAL VARIABLE
2896  013734  005037  014136                  CLR     PRIRTY          ; CLEAN UP LOCAL VARIABLE
2897  013740  013700  001376                  MOV     $CDW1,R0        ; GET THE DEVICE COUNT
2898  013744  010037  001476                  MOV     R0,SAVNUM       ; SAVE THE NO. OF DEVICES
2899  013750  012701  001346                  MOV     #$MAMS1,R1      ; GET EXTRA INFO. BITS POINTER
2900  013754  013737  001372  014134          MOV     $BASE,BASE      ; GET BASE CSR ADDRESS
2901  013762  113737  001366  014132          MOVB    $VECT1,VECTR    ; GET THE VECTOR
2902  013770  113737  001367  014136          MOVB    $VECT1+1,PRIRTY ; GET THE PRIORITY
2903  013776  013737  001374  001470          MOV     $DEVM,KMACTV    ; SAVE THE KMC'S SELECTED ACTIVE
```

```
2904  014004  013737  001470  001474          MOV     KMACTV,SAVACT   ; SAVE THE ACTIVE REGISTER
2905  014012  012702  001402                  MOV     #$DDW0,R2       ; GET ADDRESS OF FIRST DEVICE DESCRIPTOR WORD
2906  014016  012703  002100                  MOV     #KM.MAP,R3      ; GET POINTER TO DEVICE MAP
2907  014022  005023                  3$:     CLR     (R3)+           ; CLEAR DEVICE MAP
2908  014024  022703  002300                  CMP     #KM.END,R3      ; IS WHOLE DEV.MAP CLEARED?
2909  014030  003374                          BGT     3$              ; NO, THEN GO ON.
2910  014032  012703  002100                  MOV     #KM.MAP,R3      ; RESTORE DEV.MAP POINTER.
2911  014036  013723  014134          1$:     MOV     BASE,(R3)+      ; LOAD CSR ADDRESS
2912  014042  112163  000001                  MOVB    (R1)+,1(R3)     ; GET EXTRA INFO. BITS
2913  014046  006213                          ASR     (R3)            ; SET IT IN RIGHT POSITION.
2914  014050  006213                          ASR     (R3)            ; SET IT IN RIGHT POSITION.
2915  014052  053713  014136                  BIS     PRIRTY,(R3)     ; GET PRIORITY IN STAT1
2916  014056  006313                          ASL     (R3)            ; SET THEM IN RIGHT POSITION
2917  014060  006313                          ASL     (R3)            ;  ''   ''   ''   ''   ''
2918  014062  006313                          ASL     (R3)            ;  ''   ''   ''   ''   ''
2919  014064  006313                          ASL     (R3)            ;  ''   ''   ''   ''   ''
2920  014066  053723  014132                  BIS     VECTR,(R3)+     ; GET THE VECTOR IN STAT1.
2921  014072  012223                          MOV     (R2)+,(R3)+     ; GET THE STAT2 FROM DDWXX
2922  014074  005723                          TST     (R3)+           ; SKIP OVER STAT3
2923  014076  005300                          DEC     R0              ; COUNT BY 1
2924  014100  001407                          BEQ     2$              ; ALL DONE?
2925  014102  062737  000010  014134          ADD     #10,BASE        ; INCREMENT BASE CSR ADDRESS BY 10
2926  014110  062737  000010  014132          ADD     #10,VECTR       ; INCREMENT VECTOR ADDRESS BY 10
2927  014116  000747                          BR      1$              ; SET THE NEXT MAP ENTRY
2928  014120                           2$:
2929  014120  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2930  014122  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2931  014124  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2932  014126  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2933  014130  000207                          RTS     PC              ; RETURN
2934  014132  000000                  VECTR:  .WORD   0
2935  014134  000000                  BASE:   .WORD   0
2936  014136  000000                  PRIRTY: .WORD   0
2937
2938
2939
2940                                          ;***************************** TEST 1 ****************************
2941                                          ;*OUT CONTROL REGISTER READ/ONLY TEST
2942                                          ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2943                                          ;*BITS ARE IN THE CORRECT STATE
2944                                          ;;*********************************************************************
2945
2946                                          ;  TEST 1
2947                                          ;-------------
2948                                  ;;*********************************************************************
2949  014140  000004                  TST1:   SCOPE
2950  014142  012737  000001  001202          MOV     #1,$TSTNM               ; LOAD THE NO. OF THIS TEST
2951  014150  012737  014214  001442          MOV     #TST2,NEXT              ; POINT TO THE START OF NEXT TEST.
2952                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2953  014156  005077  165704                  CLR     @KMCSR                  ;CLEAR SEL0
2954  014162  012702  000011                  MOV     #11,R2                  ;SAVE R2 FOR TYPEOUT
2955  014166  104412                          ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2956  014170  021224                          021004!<20*11>                  ;PORT4 LINE UNIT REG 11
2957  014172  016104  000004                  MOV     4(R1),R4                ;PUT "FOUND" IN R4
2958  014176  042704  000054                  BIC     #54,R4                  ;CLEAR UNKNOWN BITS
2959  014202  012705  000020                  MOV     #20,R5                  ;PUT "EXPECTED" IN R5
```

```
                                        CMPB    R5,R4           ;IS OUT READY SET?
                                        BEQ     1$              ;BR IF YES
                                        ERROR   2               ;ERROR IN LU 11
                                1$:

                                        ;*************************** TEST 2 ***************************
                                        ;*IN CONTROL REGISTER READ/ONLY TEST
                                        ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
                                        ;*BITS ARE IN THE CORRECT STATE
                                        ;;***********************************************************

                                        ;  TEST 2
                                        ;-------------
                                ;;;*************************************************************
014214  000004                  TST2:   SCOPE
014216  012737  000002  001202          MOV     #2,$TSTNM               ; LOAD THE NO. OF THIS TEST
014224  012737  014262  001442          MOV     #TST3,NEXT              ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
014232  012702  000012                  MOV     #12,R2          ;SAVE R2 FOR TYPEOUT
014236  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014240  021244                          021004!<20*12>          ;PORT4 LINE UNIT REG 12
014242  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
014246  042704  000017                  BIC     #17,R4          ;CLEAR UNKNOWN BITS
014252  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
014254  120504                          CMPB    R5,R4           ;ARE ALL BITS CLEARED?
014256  001401                          BEQ     1$              ;BR IF YES
014260  104002                          ERROR   2               ;ERROR IN LU 12
014262                          1$:

                                        ;*************************** TEST 3 ***************************
                                        ;*MODEM CONTROL REGISTER READ/ONLY TEST
                                        ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
                                        ;*BITS ARE IN THE CORRECT STATE
                                        ;;***********************************************************

                                        ;  TEST 3
                                        ;-------------
                                ;;;*************************************************************
014262  000004                  TST3:   SCOPE
014264  012737  000003  001202          MOV     #3,$TSTNM               ; LOAD THE NO. OF THIS TEST
014272  012737  014334  001442          MOV     #TST4,NEXT              ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
014300  104410                          MSTCLR                  ;MASTER CLEAR KMC11
014302  012702  000013                  MOV     #13,R2          ;SAVE R2 FOR TYPEOUT
014306  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014310  021264                          021004!<20*13>          ;PORT4 LINE UNIT REG 13
014312  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
014316  042704  000213                  BIC     #213,R4         ;CLEAR UNKNOWN BITS
014322  012705  000100                  MOV     #100,R5         ;PUT "EXPECTED" IN R5
014326  120504                          CMPB    R5,R4           ;ARE RING, DTR,  AND MODEM READY SET?
014330  001401                          BEQ     1$              ;BR IF YES
014332  104002                          ERROR   2               ;ERROR IN LU 13
014334                          1$:
```

```
                                        ;***************************** TEST 4 *****************************
                                        ;*MAINTENANCE REGISTER READ/ONLY TEST
                                        ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
                                        ;*BITS ARE IN THE CORRECT STATE
                                        ;:*************************************************************

                                        ;  TEST 4
                                        ;-------------
                                    ;;*************************************************************
014334  000004                      TST4:   SCOPE
014336  012737  000004  001202              MOV     #4,$TSTNM               ; LOAD THE NO. OF THIS TEST
014344  012737  014426  001442              MOV     #TST5,NEXT              ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
014352  104410                              MSTCLR                  ;MASTER CLEAR KMC11
014354  012702  000017                      MOV     #17,R2          ;SAVE R2 FOR TYPEOUT
014360  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014362  021364                              021004!<20*17>          ;PORT4 LINE UNIT REG 17
014364  016104  000004                      MOV     4(R1),R4        ;PUT "FOUND" IN R4
014370  042704  000206                      BIC     #206,R4         ;CLEAR UNKNOWN BITS
014374  012705  000051                      MOV     #51,R5          ;PUT "EXPECTED" IN R5
014400  032737  020000  002050              BIT     #BIT13,STAT1    ;IS LU AN M8202 OR M8201?
014406  001404                              BEQ     .+12            ;BR IF M8201
014410  042704  000040                      BIC     #40,R4          ;MASK OFF SI BIT IF M8202
014414  042705  000040                      BIC     #BIT5,R5        ;SI BIT IS UNKNOWN ON AN M8202
014420  120504                              CMPB    R5,R4           ;ARE SI AND ICIR SET?
014422  001401                              BEQ     1$              ;BR IF YES
014424  104002                              ERROR   2               ;ERROR IN LU 17
014426                                  1$:
                                        ***************************** TEST 5 *****************************
                                        ;*LINE UNIT REGISTER WRITE/READ TEST
                                        ;*SET BIT5 IN LU REGISTER 12, VERIFY IT IS SET
                                        ;*CLEAR BIT5 IN LU REGISTER 12, VERIFY IT IS CLEAR
                                        ;:*************************************************************

                                        ;  TEST 5
                                        ;-------------
                                    ;;*************************************************************
014426  000004                      TST5:   SCOPE
014430  012737  000005  001202              MOV     #5,$TSTNM               ; LOAD THE NO. OF THIS TEST
014436  012737  014566  001442              MOV     #TST6,NEXT              ; POINT TO THE START OF NEXT TEST.
014444  012737  014460  001444              MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
014452  104410                              MSTCLR                  ;MASTER CLEAR KMC11
014454  012702  000012                      MOV     #12,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
014460  012761  000040  000004      1$:     MOV     #40,4(R1)       ;LOAD PORT4
014466  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014470  122112                              122112                  ;SET BIT5 IN LU-12
014472  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014474  021245                              021245                  ;READ LU-12
014476  012705  000040                      MOV     #40,R5          ;PUT "EXPECTED" IN R5
014502  116104  000005                      MOVB    5(R1),R4        ;PUT "FOUND" IN R4
014506  042704  000337                      BIC     #337,R4         ;CLEAR UNWANTED BITS
014512  120504                              CMPB    R5,R4           ;IS BIT5 SET?
```

```
3072  014514  001401                          BEQ     2$              ;BR IF YES
3073  014516  104003                          ERROR   3               ;ERROR, BIT 5 IS NOT SET
3074  014520  104405                  2$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)
3075  014522  012737  014530  001444          MOV     #3$,LOCK        ;NEW SCOP1
3076  014530  005061  000004          3$:     CLR     4(R1)           ;LOAD PORT4
3077  014534  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3078  014536  122112                          122112                  ;CLEAR BIT 5 IN LU-12
3079  014540  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3080  014542  021245                          021245                  ;READ LU-12
3081  014544  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
3082  014546  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3083  014552  042704  000337                  BIC     #337,R4         ;CLEAR UNWANTED BITS
3084  014556  120504                          CMPB    R5,R4           ;IS BIT5 CLEAR?
3085  014560  001401                          BEQ     4$              ;BR IF YES
3086  014562  104003                          ERROR   3               ;ERROR, BIT5 IS NOT CLEAR
3087  014564  104405                  4$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)
3088
3089
3090                                          ;*************************** TEST 6 ***************************
3091                                          ;*LINE UNIT REGISTER WRITE/READ TEST
3092                                          ;*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
3093                                          ;*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR
3094                                          ;;*************************************************************
3095
3096                                          ;  TEST 6
3097                                          ;------------
3098                                  ;;*************************************************************
3099  014566  000004                  TST6:   SCOPE
3100  014570  012737  000006  001202          MOV     #6,$TSTNM               ; LOAD THE NO. OF THIS TEST
3101  014576  012737  014726  001442          MOV     #TST7,NEXT              ; POINT TO THE START OF NEXT TEST.
3102  014604  012737  014620  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
3103                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3104  014612  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3105  014614  012702  000017                  MOV     #17,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3106  014620  012761  000001  000004  1$:     MOV     #1,4(R1)        ;LOAD PORT4
3107  014626  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3108  014630  122117                          122117                  ;SET BIT1 IN LU-17
3109  014632  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3110  014634  021365                          021365                  ;READ LU-17
3111  014636  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
3112  014642  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3113  014646  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
3114  014652  120504                          CMPB    R5,R4           ;IS BIT1 SET?
3115  014654  001401                          BEQ     2$              ;BR IF YES
3116  014656  104003                          ERROR   3               ;ERROR, BIT 1 IS NOT SET
3117  014660  104405                  2$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)
3118  014662  012737  014670  001444          MOV     #3$,LOCK        ;NEW SCOP1
3119  014670  005061  000004          3$:     CLR     4(R1)           ;LOAD PORT4
3120  014674  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3121  014676  122117                          122117                  ;CLEAR BIT 1 IN LU-17
3122  014700  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3123  014702  021365                          021365                  ;READ LU-17
3124  014704  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
3125  014706  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3126  014712  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
3127  014716  120504                          CMPB    R5,R4           ;IS BIT1 CLEAR?
```

```
3128  014720  001401                        BEQ     4$              ;BR IF YES
3129  014722  104003                        ERROR   3               ;ERROR, BIT1 IS NOT CLEAR
3130  014724  104405                4$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)
3131
3132
3133                                        ;*************************** TEST 7 ***************************
3134                                        ;*LINE UNIT REGISTER WRITE/READ TEST
3135                                        ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 13
3136                                        ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 13
3137                                        ;;*********************************************************************
3138
3139                                        ;  TEST 7
3140                                        ;-------------
3141                                    ;;**********************************************************************
3142  014726  000004                TST7:   SCOPE
3143  014730  012737  000007  001202        MOV     #7,$TSTNM               ; LOAD THE NO. OF THIS TEST
3144  014736  012737  015136  001442        MOV     #TST10,NEXT             ; POINT TO THE START OF NEXT TEST.
3145  014744  012737  014764  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3146                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3147  014752  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3148  014754  012702  000013                MOV     #13,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3149  014760  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3150  014764                        64$:
3151  014764  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3152  014770  042761  000257  000004        BIC     #257,4(R1)      ;CLEAR UNWANTED BITS
3153  014776  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3154  015000  122113                        122100!13               ;MOV DATA TO IBUS REGISTER 13
3155  015002  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3156  015004  021265                        21005!<13*20>           ;READ FROM IBUS REGISTER 13
3157  015006  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3158  015010  042705  000257                BIC     #257,R5         ;CLEAR UNWANTED BITS
3159  015014  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3160  015020  042704  000257                BIC     #257,R4         ;CLEAR UNWANTED BITS
3161  015024  120504                        CMPB    R5,R4           ;DATA CORRECT?
3162  015026  001401                        BEQ     65$             ;BR IF YES
3163  015030  104003                        ERROR   3               ;ERROR
3164  015032  104405                65$:    SCOP1                   ;SW09=1?
3165  015034  000241                        CLC                     ;CLEAR CARRY
3166  015036  106100                        ROLB    R0              ;SHIFT BIT IN R0
3167  015040  001351                        BNE     64$             ;IF R0=0 THEN DONE
3168  015042  012737  015056  001444        MOV     #67$,LOCK       ;NEW SCOP1
3169  015050  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3170  015054  005100                69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3171  015056                        67$:
3172  015056  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3173  015062  042761  000257  000004        BIC     #257,4(R1)      ;CLEAR UNWANTED BITS
3174  015070  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3175  015072  122113                        122100!13               ;MOV DATA TO IBUS REGISTER 13
3176  015074  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3177  015076  021265                        21005!<13*20>           ;READ FROM IBUS REGISTER 13
3178  015100  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3179  015102  042705  000257                BIC     #257,R5         ;CLEAR UNWANTED BITS
3180  015106  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3181  015112  042704  000257                BIC     #257,R4         ;CLEAR UNWANTED BITS
3182  015116  120504                        CMPB    R5,R4           ;DATA CORRECT?
3183  015120  001401                        BEQ     68$             ;BR IF YES
```

```
3184  015122  104003                          ERROR   3               ;ERROR
3185  015124  104405                  68$:    SCOP1                   ;SW09=1?
3186  015126  005100                          COM     R0              ;CHANGE TO FLOATING 1
3187  015130  000241                          CLC                     ;CLEAR CARRY
3188  015132  106100                          ROLB    R0              ;SHIFT BIT IN R0
3189  015134  001347                          BNE     69$             ;IF R0=0 THEN DONE
3190
3191
3192                                          ;*************************** TEST 10 ************************
3193                                          ;*LINE UNIT REGISTER WRITE/READ TEST
3194                                          ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
3195                                          ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
3196                                          ;:*********************************************************************
3197
3198                                          ;  TEST 10
3199                                          ;-------------
3200                                  ;;*********************************************************************
3201  015136  000004                  TST10:  SCOPE
3202  015140  012737  000010  001202          MOV     #10,$TSTNM              ; LOAD THE NO. OF THIS TEST
3203  015146  012737  015312  001442          MOV     #TST11,NEXT             ; POINT TO THE START OF NEXT TEST.
3204  015154  012737  015174  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3205                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3206  015162  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3207  015164  012702  000014                  MOV     #14,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3208  015170  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3209  015174                          64$:
3210  015174  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3211  015200  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3212  015202  122114                          122100!14               ;MOV DATA TO IBUS REGISTER 14
3213  015204  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3214  015206  021305                          21005!<14*20>           ;READ FROM IBUS REGISTER 14
3215  015210  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3216  015212  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3217  015216  120504                          CMPB    R5,R4           ;DATA CORRECT?
3218  015220  001401                          BEQ     65$             ;BR IF YES
3219  015222  104003                          ERROR   3               ;ERROR
3220  015224  104405                  65$:    SCOP1                   ;SW09=1?
3221  015226  000241                          CLC                     ;CLEAR CARRY
3222  015230  106100                          ROLB    R0              ;SHIFT BIT IN R0
3223  015232  001360                          BNE     64$             ;IF R0=0 THEN DONE
3224  015234  012737  015250  001444          MOV     #67$,LOCK       ;NEW SCOP1
3225  015242  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3226  015246  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3227  015250                          67$:
3228  015250  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3229  015254  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3230  015256  122114                          122100!14               ;MOV DATA TO IBUS REGISTER 14
3231  015260  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3232  015262  021305                          21005!<14*20>           ;READ FROM IBUS REGISTER 14
3233  015264  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3234  015266  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3235  015272  120504                          CMPB    R5,R4           ;DATA CORRECT?
3236  015274  001401                          BEQ     68$             ;BR IF YES
3237  015276  104003                          ERROR   3               ;ERROR
3238  015300  104405                  68$:    SCOP1                   ;SW09=1?
3239  015302  005100                          COM     R0              ;CHANGE TO FLOATING 1
```

```
3240  015304  000241                          CLC                     ;CLEAR CARRY
3241  015306  106100                          ROLB    R0              ;SHIFT BIT IN R0
3242  015310  001356                          BNE     69$             ;IF R0=0 THEN DONE
3243
3244
3245                                          ;***************************** TEST 11 ****************************
3246                                          ;*SWITCH PAC TEST
3247                                          ;*THIS TEST READS SWITCH PAC#1
3248                                          ;*THIS SWITCH PAC CONTAINS THE DDCMP LINE #
3249                                          ;:*************************************************************
3250
3251                                          ;  TEST 11
3252                                          ;-------------
3253                                    ;;*************************************************************
3254  015312  000004                    TST11:  SCOPE
3255  015314  012737  000011  001202           MOV     #11,$TSTNM              ; LOAD THE NO. OF THIS TEST
3256  015322  012737  015354  001442           MOV     #TST12,NEXT             ; POINT TO THE START OF NEXT TEST.
3257                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3258  015330  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3259  015332  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3260  015334  021324                          021324                  ;PORT4 LU15
3261  015336  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3262  015342  113705  002052                  MOVB    STAT2,R5        ;PUT "EXPECTED" IN R5
3263  015346  120504                          CMPB    R5,R4           ;SW OK?
3264  015350  001401                          BEQ     1$              ;BR IF YES
3265  015352  104031                          ERROR   31              ;ERROR, SWITCH PAC READ ERROR
3266  015354                            1$:
3267
3268
3269                                          ;***************************** TEST 12 ****************************
3270                                          ;*SWITCH PAC TEST
3271                                          ;*THIS TEST READS SWITCH PAC#2
3272                                          ;*THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
3273                                          ;:*************************************************************
3274
3275                                          ;  TEST 12
3276                                          ;-------------
3277                                    ;;*************************************************************
3278  015354  000004                    TST12:  SCOPE
3279  015356  012737  000012  001202           MOV     #12,$TSTNM              ; LOAD THE NO. OF THIS TEST
3280  015364  012737  015416  001442           MOV     #TST13,NEXT             ; POINT TO THE START OF NEXT TEST.
3281                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3282  015372  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3283  015374  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3284  015376  021344                          021344                  ;PORT4 LU16
3285  015400  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3286  015404  113705  002053                  MOVB    STAT2+1,R5      ;PUT "EXPECTED" IN R5
3287  015410  120504                          CMPB    R5,R4           ;SW OK?
3288  015412  001401                          BEQ     1$              ;BR IF YES
3289  015414  104031                          ERROR   31              ;ERROR, SWITCH PAC READ ERROR
3290  015416                            1$:
3291
3292
3293                                          ;***************************** TEST 13 ****************************
3294                                          ;*LINE UNIT CLOCK TEST
3295                                          ;*THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
```

```
                                        ;*(BIT 1 IN LU-17) IS WORKING
                                        ;;*****************************************************************
                                        ;  TEST 13
                                        ;-------------
                                    ;;*****************************************************************
015416  000004                  TST13:  SCOPE
015420  012737  000013  001202          MOV     #13,$TSTNM              ; LOAD THE NO. OF THIS TEST
015426  012737  015516  001442          MOV     #TST14,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
015434  104410                          MSTCLR                  ;MASTER CLEAR KMC11
015436  005037  011234                  CLR     TEMP            ;PREPARE FOR DELAY
015442                          1$:
015442  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
015444  021364                          021364                  ;PORT4 LU-17
015446  032761  000002  000004          BIT     #2,4(R1)        ;IS CLOCK BIT SET?
015454  001004                          BNE     2$              ;BR IF YES
015456  005237  011234                  INC     TEMP            ;DELAY
015462  001367                          BNE     1$      ;DELAY FINISHED?
015464  104004                          ERROR   4               ;ERROR BIT IS STUCK CLEAR
015466  005037  011234          2$:     CLR     TEMP            ;PREPARE FOR DELAY
015472                          3$:
015472  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
015474  021364                          021364                  ;PORT4 LU-17
015476  032761  000002  000004          BIT     #2,4(R1)        ;IS CLOCK BIT CLEAR?
015504  001404                          BEQ     4$              ;BR IF YES
015506  005237  011234                  INC     TEMP            ;DELAY
015512  001367                          BNE     3$              ;BR IF DELAY NOT DONE
015514  104004                          ERROR   4               ;ERROR BIT IS STUCK SET
015516                          4$:


                                        ;************************** TEST 14 **************************
                                        ;*OUT DATA SILO TEST
                                        ;*SET SOM AND LOAD OUT DATA SILO
                                        ;*VERIFY THAT OCOR SET, INDICATING THAT THE
                                        ;*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
                                        ;;*****************************************************************
                                        ;  TEST 14
                                        ;-------------
                                    ;;*****************************************************************
015516  000004                  TST14:  SCOPE
015520  012737  000014  001202          MOV     #14,$TSTNM              ; LOAD THE NO. OF THIS TEST
015526  012737  015616  001442          MOV     #TST15,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
015534  104410                          MSTCLR                  ;MASTER CLEAR KMC11
015536  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
015542  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
015550  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
015552  122111                          122111                  ;SET SOM
015554  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
015556  122110                          122110                  ;LOAD OUT DATA SILO
015560  104414  000002                  TIMER,  2               ;WAIT FOR OCOR
015564  012702  000017                  MOV     #17,R2          ;SAVE ADDRESS FOR TYPEOUT
015570  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
3352  015572  021364                          021364                  ;PORT4 LU 17
3353  015574  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3354  015600  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
3355  015604  012705  000020                  MOV     #20,R5          ;PUT "EXPECTED" IN R5
3356  015610  120504                          CMPB    R5,R4           ;IS OCOR SET?
3357  015612  001401                          BEQ     1$              ;BR IF YES
3358  015614  104005                          ERROR   5
3359  015616                          1$:
3360
3361
3362                                          ;***************************** TEST 15 ****************************
3363                                          ;*DDCMP TEST OF RTS AND OUT ACTIVE
3364                                          ;*SET SOM AND LOAD OUT DATA SILO
3365                                          ;*SINGLE STEP 2 DATA CLOCKS, VERIFY
3366                                          ;*THAT RTS AND ACTIVE ARE SET
3367                                          ;;*************************************************************
3368
3369                                          ;  TEST 15
3370                                          ;-------------
3371                                  ;;*************************************************************
3372  015616  000004                  TST15:  SCOPE
3373  015620  012737  000015  001202          MOV     #15,$TSTNM              ; LOAD THE NO. OF THIS TEST
3374  015626  012737  015754  001442          MOV     #TST16,NEXT             ; POINT TO THE START OF NEXT TEST.
3375                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3376  015634  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3377  015636  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3378  015642  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
3379  015650  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3380  015652  122111                          122111                  ;SET SOM
3381  015654  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3382  015656  122110                          122110                  ;LOAD OUT DATA SILO
3383  015660  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
3384  015664  104413  000002                  DATACLK,        2       ;CLOCK DATA FOUR TIMES
3385  015670  012702  000011                  MOV     #11,R2          ;SAVE ADDRESS FOR TYPEOUT
3386  015674  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3387  015676  021224                          021224                  ;PORT4 LU 11
3388  015700  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3389  015704  042704  000257                  BIC     #257,R4         ;CLEAR UNWANTED BITS
3390  015710  012705  000120                  MOV     #120,R5         ;PUT "EXPECTED" IN R5
3391  015714  120504                          CMPB    R5,R4           ;IS ACTIVE SET?
3392  015716  001401                          BEQ     1$              ;BR IF YES
3393  015720  104005                          ERROR   5
3394  015722                          1$:
3395  015722  012702  000013                  MOV     #13,R2          ;SAVE ADDRESS FOR TYPEOUT
3396  015726  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3397  015730  021264                          021264                  ;PORT4 LU 13
3398  015732  016104  000004                  MOV     4(R1),R4        ;PUT EXPECTED IN R4
3399  015736  042704  000337                  BIC     #337,R4         ;CLEAR UNWANTED BITS
3400  015742  012705  000040                  MOV     #BIT5,R5        ;PUT "EXPECTED" IN R5, RTS SHOULD BE SET
3401  015746  120504                          CMPB    R5,R4           ;IS RTS OK?
3402  015750  001401                          BEQ     2$              ;BR IF YES
3403  015752  104005                          ERROR   5               ;RTS ERROR
3404  015754                          2$:
3405
3406
3407                                          ;***************************** TEST 16 ****************************
```

```
                                        ;*TEST OF OUT CLEAR
                                        ;*SET SOM AND LOAD OUT DATA SILO
                                        ;*SINGLE STEP DATA CLOCK, SET OUT CLEAR
                                        ;*VERIFY THAT OCOR,RTS, AND ACTIVE ARE CLEARED
                                        ;;*****************************************************************

                                        ;  TEST 16
                                        ;-------------
                                    ;;*****************************************************************
015754  000004                    TST16:  SCOPE
015756  012737  000016  001202            MOV     #16,$TSTNM              ; LOAD THE NO. OF THIS TEST
015764  012737  016152  001442            MOV     #TST17,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
015772  104410                            MSTCLR                  ;MASTER CLEAR KMC11
015774  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
016000  012761  000001  000004            MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
016006  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016010  122111                            122111                  ;SET SOM
016012  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016014  122110                            122110                  ;LOAD OUT DATA SILO
016016  004737  030260                    JSR     PC,OCOR         ;WAIT FOR OCOR
016022  104413  000002                    DATACLK,        2       ;CLOCK DATA FOUR TIMES
016026  012761  000200  000004            MOV     #BIT7,4(R1)     ;SET BIT7 IN PORT4
016034  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016036  122111                            122111                  ;SET OUT CLEAR
016040  104413  000001                    DATACLK,        1       ;GIVE A TICK TO CLEAR RTS
016044  012702  000017                    MOV     #17,R2          ;SAVE ADDRESS FOR TYPEOUT
016050  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016052  021364                            021364                  ;PORT4 LU 17
016054  016104  000004                    MOV     4(R1),R4        ;PUT "FOUND" IN R4
016060  042704  000357                    BIC     #357,R4         ;CLEAR UNWANTED BITS
016064  005005                            CLR     R5              ;PUT "EXPECTED" IN R5
016066  120504                            CMPB    R5,R4           ;IS OCOR CLEARED?
016070  001401                            BEQ     1$              ;BR IF YES
016072  104005                            ERROR   5
016074                              1$:
016074  012702  000013                    MOV     #13,R2          ;SAVE ADDRESS FOR TYPEOUT
016100  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016102  021264                            021264                  ;PORT4 LU 13
016104  016104  000004                    MOV     4(R1),R4        ;PUT EXPECTED IN R4
016110  042704  000 '                     BIC     #337,R4         ;CLEAR UNWANTED BITS
016114  005005                            CLR     R5              ;PUT "EXPECTED" IN R5, RTS SHOULD BE CLEARED
016116  120504                            CMPB    R5,R4           ;IS RTS OK?
016120  001401                            BEQ     2$              ;BR IF YES
016122  104005                            ERROR   5               ;RTS ERROR
016124                              2$:
016124  012702  000011                    MOV     #11,R2          ;SAVE ADDRESS FOR TYPEOUT
016130  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016132  021224                            021224                  ;PORT4 LU11
016134  016104  000004                    MOV     4(R1),R4        ;PUT "FOUND" IN R4
016140  012705  000020                    MOV     #BIT4,R5        ;ONLY OUT READY SHOULD BE SET
016144  120504                            CMPB    R5,R4           ;IS ACTIVE CLEAR?
016146  001401                            BEQ     3$              ;BR IF YES
016150  104005                            ERROR   5               ;ERROR ACTIVE NOT CLEARED
016152                              3$:
```

```
                                        ;*************************** TEST 17 ***************************
                                        ;*DDCMP TRANSMITTER TEST
                                        ;*SINGLE CLOCK THE CHARACTER 0
                                        ;*VERIFY EACH BIT POSITION AS IT
                                        ;*PASSES THE BIT WINDOW (SI BIT)
                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
                                        ;*************************************************************

                                        ;  TEST 17
                                        ;-------------
                                        ;*************************************************************
016152  000004                  TST17:  SCOPE
016154  012737  000017  001202          MOV     #17,$TSTNM              ; LOAD THE NO. OF THIS TEST
016162  012737  016334  001442          MOV     #TST20,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
016170  104410                          MSTCLR                  ;MASTER CLEAR KMC11
016172  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
016176  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016202  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
016210  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016212  122111                          122111                  ;SET SOM!
016214  012705  000000                  MOV     #0,R5   ;LOAD CHARACTER IN R5 FOR TYPEOUT
016220  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016224  010561  000004                  MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
016230  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016232  122110                          122110                  ;LOAD OUT DATA
016234  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
016240  005003                          CLR     R3              ;CLEAR BIT COUNTER
016242  010502                          MOV     R5,R2           ;LOAD CHARACTER IN R2
016244  104413  000002                  DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
016250  104413  000001          1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
016254  106002                          RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
016256  103005                          BCC     2$              ;BR IF CARRY CLEAR
016260  004737  030226                  JSR     PC,GETSI        ;GET THE WINDOW
016264  103406                          BCS     3$              ;BR IF BIT IS A MARK
016266  104006                          ERROR   6               ;ERROR BIT WAS A SPACE
016270  000404                          BR      3$              ;CONTINE WITH TEST
016272  004737  030226          2$:     JSR     PC,GETSI        ;GET THE WINDOW
016276  103001                          BCC     3$              ;BR IF BIT IS A SPACE
016300  104006                          ERROR   6               ;ERROR BIT WAS A MARK
016302                          3$:
016302  005203                          INC     R3              ;NEXT BIT
016304  022703  000010                  CMP     #10,R3          ;DONE YET?
016310  001357                          BNE     1$              ;BR IF NO
016312  104413  000014                  DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
016316  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016320  021264                          021264                  ;PORT4 LU-13
016322  032761  000040  000004          BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
016330  001401                          BEQ     4$              ;BR IF YES
016332  104034                          ERROR   34              ;ERROR, RTS NOT CLEAR
016334                          4$:


                                        ;*************************** TEST 20 ***************************
                                        ;*DDCMP TRANSMITTER TEST
```

```
                                        ;*SINGLE CLOCK THE CHARACTER 125
                                        ;*VERIFY EACH BIT POSITION AS IT
                                        ;*PASSES THE BIT WINDOW (SI BIT)
                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
                                        ;:*****************************************************************

                                        ;  TEST 20
                                        ;-------------
                                    ;;*****************************************************************
016334  000004                    TST20:  SCOPE
016336  012737  000020  001202            MOV     #20,$TSTNM              ; LOAD THE NO. OF THIS TEST
016344  012737  016516  001442            MOV     #TST21,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
016352  104410                          MSTCLR                      ;MASTER CLEAR KMC11
016354  012711  004000                  MOV     #BIT11,(R1)         ;SET LINE UNIT LOOP
016360  004737  030412                  JSR     PC,OUTRDY           ;WAIT FOR OUT-READY
016364  012761  000001  000004          MOV     #1,4(R1)            ;SET BITO IN PORT4
016372  104412                          ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016374  122111                          122111                      ;SET SOM!
016376  012705  000125                  MOV     #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
016402  004737  030412                  JSR     PC,OUTRDY           ;WAIT FOR OUT-READY
016406  010561  000004                  MOV     R5,4(R1)            ;LOAD PORT4 WITH CHARACTER
016412  104412                          ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016414  122110                          122110                      ;LOAD OUT DATA
016416  004737  030260                  JSR     PC,OCOR             ;WAIT FOR OCOR TO SET
016422  005003                          CLR     R3                  ;CLEAR BIT COUNTER
016424  010502                          MOV     R5,R2               ;LOAD CHARACTER IN R2
016426  104413  000002                  DATACLK,        2           ;2 TICKS TO SET UP TRANSMITTER
016432  104413  000001            1$:   DATACLK,        1           ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
016436  106002                          RORB    R2                  ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
016440  103005                          BCC     2$                  ;BR IF CARRY CLEAR
016442  004737  030226                  JSR     PC,GETSI            ;GET THE WINDOW
016446  103406                          BCS     3$                  ;BR IF BIT IS A MARK
016450  104006                          ERROR   6                   ;ERROR BIT WAS A SPACE
016452  000404                          BR      3$                  ;CONTINE WITH TEST
016454  004737  030226            2$:   JSR     PC,GETSI            ;GET THE WINDOW
016460  103001                          BCC     3$                  ;BR IF BIT IS A SPACE
016462  104006                          ERROR   6                   ;ERROR BIT WAS A MARK
016464                            3$:
016464  005203                          INC     R3                  ;NEXT BIT
016466  022703  000010                  CMP     #10,R3              ;DONE YET?
016472  001357                          BNE     1$                  ;BR IF NO
016474  104413  000014                  DATACLK,        14          ;CLOCK TRANSMITTER 14 MORE TICKS
016500  104412                          ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016502  021264                          021264                      ;PORT4 LU-13
016504  032761  000040  000004          BIT     #BIT5,4(R1)         ;RTS SHOULD BE CLEAR NOW
016512  001401                          BEQ     4$                  ;BR IF YES
016514  104034                          ERROR   34                  ;ERROR, RTS NOT CLEAR
016516                            4$:

                                        ;***************************** TEST 21 ****************************
                                        ;*DDCMP TRANSMITTER TEST
                                        ;*SINGLE CLOCK THE CHARACTER 252
                                        ;*VERIFY EACH BIT POSITION AS IT
                                        ;*PASSES THE BIT WINDOW (SI BIT)
```

```
                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
                                        ;:**********************************************************************
                                        ;  TEST 21
                                        ;-------------
                                        ;:**********************************************************************
016516  000004                  TST21:  SCOPE
016520  012737  000021  001202          MOV     #21,$TSTNM              ; LOAD THE NO. OF THIS TEST
016526  012737  016700  001442          MOV     #TST22,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
016534  104410                          MSTCLR                  ;MASTER CLEAR KMC11
016536  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
016542  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016546  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
016554  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016556  122111                          122111                  ;SET SOM!
016560  012705  000252                  MOV     #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
016564  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016570  010561  000004                  MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
016574  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016576  122110                          122110                  ;LOAD OUT DATA
016600  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
016604  005003                          CLR     R3              ;CLEAR BIT COUNTER
016606  010502                          MOV     R5,R2           ;LOAD CHARACTER IN R2
016610  104413  000002                  DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
016614  104413  000001          1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
016620  106002                          RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
016622  103005                          BCC     2$              ;BR IF CARRY CLEAR
016624  004737  030226                  JSR     PC,GETSI        ;GET THE WINDOW
016630  103406                          BCS     3$              ;BR IF BIT IS A MARK
016632  104006                          ERROR   6               ;ERROR BIT WAS A SPACE
016634  000404                          BR      3$              ;CONTINE WITH TEST
016636  004737  030226          2$:     JSR     PC,GETSI        ;GET THE WINDOW
016642  103001                          BCC     3$              ;BR IF BIT IS A SPACE
016644  104006                          ERROR   6               ;ERROR BIT WAS A MARK
016646                          3$:
016646  005203                          INC     R3              ;NEXT BIT
016650  022703  000010                  CMP     #10,R3          ;DONE YET?
016654  001357                          BNE     1$              ;BR IF NO
016656  104413  000014                  DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
016662  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
016664  021264                          021264                  ;PORT4 LU-13
016666  032761  000040  000004          BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
016674  001401                          BEQ     4$              ;BR IF YES
016676  104034                          ERROR   34              ;ERROR, RTS NOT CLEAR
016700                          4$:


                                        ;************************** TEST 22 *************************
                                        ;*DDCMP TRANSMITTER TEST
                                        ;*SINGLE CLOCK THE CHARACTER 377
                                        ;*VERIFY EACH BIT POSITION AS IT
                                        ;*PASSES THE BIT WINDOW (SI BIT)
                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
                                        ;:**********************************************************************
```

```
                                        ;  TEST 22
                                        ;-------------
                                        ;;*****************************************************************
016700  000004                  TST22:  SCOPE
016702  012737  000022  001202          MOV     #22,$TSTNM              ; LOAD THE NO. OF THIS TEST
016710  012737  017062  001442          MOV     #TST23,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
016716  104410                          MSTCLR                  ;MASTER CLEAR KMC'1
016720  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
016724  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016730  012761  000001  000004          MOV     #1,4(R1)        ;SET BITO IN PORT4
016736  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016740  122111                          122111                  ;SET SOM!
016742  012705  000377                  MOV     #377,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
016746  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
016752  010561  000004                  MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
016756  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016760  122110                          122110                  ;LOAD OUT DATA
016762  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
016766  005003                          CLR     R3              ;CLEAR BIT COUNTER
016770  010502                          MOV     R5,R2           ;LOAD CHARACTER IN R2
016772  104413  000002                  DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
016776  104413  000001          1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
017002  106002                          RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
017004  103005                          BCC     2$              ;BR IF CARRY CLEAR
017006  004737  030226                  JSR     PC,GETSI        ;GET THE WINDOW
017012  103406                          BCS     3$              ;BR IF BIT IS A MARK
017014  104006                          ERROR   6               ;ERROR BIT WAS A SPACE
017016  000404                          BR      3$              ;CONTINE WITH TEST
017020  004737  030226          2$:     JSR     PC,GETSI        ;GET THE WINDOW
017024  103001                          BCC     3$              ;BR IF BIT IS A SPACE
017026  104006                          ERROR   6               ;ERROR BIT WAS A MARK
017030                          3$:
017030  005203                          INC     R3              ;NEXT BIT
017032  022703  000010                  CMP     #10,R3          ;DONE YET?
017036  001357                          BNE     1$              ;BR IF NO
017040  104413  000014                  DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
017044  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
017046  021264                          021264                  ;PORT4 LU-13
017050  032761  000040  000004          BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
017056  001401                          BEQ     4$              ;BR IF YES
017060  104034                          ERROR   34              ;ERROR, RTS NOT CLEAR
017062                          4$:


                                        ;***************************** TEST 23 ************************
                                        ;*DDCMP TRANSMITTER TEST
                                        ;*SINGLE CLOCK A BINARY COUNT PATTERN
                                        ;*VERIFY EACH BIT POSITION AS IT
                                        ;*PASSES THE BIT WINDOW (SI BIT)
                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
                                        ;*AND R5 CONTAINS THE CHARACTER THAT FAILED
                                        ;;*****************************************************************

                                        ;  TEST 23
                                        ;-------------
```

```
3688                                    ;;*****************************************************************
3689  017062  000004                    TST23:  SCOPE
3690  017064  012737  000023  001202            MOV     #23,$TSTNM              ; LOAD THE NO. OF THIS TEST
3691  017072  012737  017270  001442            MOV     #TST24,NEXT             ; POINT TO THE START OF NEXT TEST.
3692                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3693  017100  104410                            MSTCLR                  ;MASTER CLEAR KMC11
3694  017102  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3695  017106  005003                            CLR     R3              ;R3 CONTAINS BIT COUNT
3696  017110  005004                            CLR     R4              ;R4 CONTAINS CHAR TO BE LOADED I.. SILO
3697  017112  005005                            CLR     R5              ;R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED OUT
3698  017114  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3699  017120  012761  000001  000004            MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3700  017126  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3701  017130  122111                            122111                  ;SET SOM!
3702  017132  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3703  017136  010461  000004                    MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
3704  017142  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3705  017144  122110                            122110                  ;LOAD OUT DATA
3706  017146  005204                            INC     R4              ;INCREMENT TO NEXT CHARACTER
3707  017150  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3708  017154  010461  000004                    MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
3709  017160  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3710  017162  122110                            122110                  ;LOAD OUT DATA
3711  017164  004737  030260                    JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
3712  017170  104413  000002                    DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
3713  017174  005003                    4$:     CLR     R3              ;CLEAR BIT COUNTER
3714  017176  010502                            MOV     R5,R2           ;LOAD CHARACTER IN R2
3715  017200  104413  000001            1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3716  017204  106002                            RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3717  017206  103005                            BCC     2$              ;BR IF CARRY CLEAR
3718  017210  004737  030226                    JSR     PC,GETSI        ;GET THE WINDOW
3719  017214  103406                            BCS     3$              ;BR IF BIT IS A MARK
3720  017216  104006                            ERROR   6               ;ERROR BIT WAS A SPACE
3721  017220  000404                            BR      3$              ;CONTINE WITH TEST
3722  017222  004737  030226            2$:     JSR     PC,GETSI        ;GET THE WINDOW
3723  017226  103001                            BCC     3$              ;BR IF BIT IS A SPACE
3724  017230  104006                            ERROR   6               ;ERROR BIT WAS A MARK
3725  017232                            3$:
3726  017232  005203                            INC     R3              ;NEXT BIT
3727  017234  022703  000010                    CMP     #10,R3          ;DONE YET?
3728  017240  001357                            BNE     1$              ;BR IF NO
3729  017242  005204                            INC     R4              ;NEXT CHARACTER
3730  017244  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3731  017250  010461  000004                    MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
3732  017254  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3733  017256  122110                            122110                  ;LOAD OUT DATA
3734  017260  005205                            INC     R5              ;NEXT CHARACTER
3735  017262  022705  000400                    CMP     #400,R5         ;DONE YET?
3736  017266  001342                            BNE     4$              ;BR IF NO
3737  017270                            5$:
3738
3739
3740                                            ;***************************** TEST 24 ****************************
3741                                            ;*DDCMP STRIP SYNC TEST
3742                                            ;*SET LU LOOP, SINGLE STEP 5 SYNCS,
3743                                            ;*VERIFY THAT IN ACTIVE DOES NOT SET
```

```
                                        ;;*****************************************************************
                                        ;  TEST 24
                                        ;-------------
                                        ;;*****************************************************************
017270  000004                  TST24:  SCOPE
017272  012737  000024  001202          MOV     #24,$TSTNM              ; LOAD THE NO. OF THIS TEST
017300  012737  017356  001442          MOV     #TST25,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
017306  104410                          MSTCLR                  ;MASTER CLEAR KMC11
017310  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
017314  012702  000012                  MOV     #12,R2          ;SAVE LU REG FOR TYPEOUT
017320  004737  030276                  JSR     PC,SYNC         ;SINGLE CLOCK 5 SYNC CHARACTERS
017324  000005                          5
017326  104413  000054                  DATACLK,        54
017332  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017334  021244                          021244                  ;PORT4 LU12
017336  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
017342  042704  000277                  BIC     #277,R4         ;CLEAR UNWANTED BITS
017346  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
017350  120504                          CMPB    R5,R4           ;IS ACTIVE CLEAR?
017352  001401                          BEQ     1$              ;BR IF YES
017354  104040                          ERROR   40              ;ERROR ACTIVE IS NOT CLEAR
017356                          1$:
                                        ;***************************** TEST 25 ****************************
                                        ;*DDCMP IN ACTIVE TEST
                                        ;*SET LU LOOP, SINGLE STEP 5 SYNCS AND A NON-SYNC (301)
                                        ;*VERIFY THAT IN ACTIVE IS SET
                                        ;;*****************************************************************
                                        ;  TEST 25
                                        ;-------------
                                        ;;*****************************************************************
017356  000004                  TST25:  SCOPE
017360  012737  000025  001202          MOV     #25,$TSTNM              ; LOAD THE NO. OF THIS TEST
017366  012737  017446  001442          MOV     #TST26,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
017374  104410                          MSTCLR                  ;MASTER CLEAR KMC11
017376  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
017402  012702  000012                  MOV     #12,R2          ;SAVE LU REG FOR TYPEOUT
017406  004737  030276                  JSR     PC,SYNC         ;SINGLE CLOCK 5 SYNC CHARACTERS
017412  000005                          5
017414  104413  000064                  DATACLK,        64
017420  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017422  021244                          021244                  ;PORT4 LU12
017424  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
017430  042704  000277                  BIC     #277,R4         ;CLEAR UNWANTED BITS
017434  012705  000100                  MOV     #BIT6,R5                ;PUT "EXPECTED" IN R5
017440  120504                          CMPB    R5,R4           ;IS ACTIVE SET?
017442  001401                          BEQ     1$              ;BR IF YES
017444  104040                          ERROR   40              ;ERROR ACTIVE IS NOT SET
017446                          1$:
```

```
3800                                        ;*************************** TEST 26 ***************************
3801                                        ;*DDCMP IN ACTIVE TEST
3802                                        ;*SET LU LOOP, SINGLE STEP 1 SYNC AND A NON-SYNC (301)
3803                                        ;*VERIFY THAT IN ACTIVE DOES NOT SET
3804                                        ;;*************************************************************
3805
3806                                        ;  TEST 26
3807                                        ;-------------
3808                                    ;;*************************************************************
3809  017446  000004                TST26:  SCOPE
3810  017450  012737  000026  001202        MOV     #26,$TSTNM              ; LOAD THE NO. OF THIS TEST
3811  017456  012737  017534  001442        MOV     #TST27,NEXT             ; POINT TO THE START OF NEXT TEST.
3812                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3813  017464  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3814  017466  012711  004000                MOV     #BIT11,(R1)     ;SET LU LOOP
3815  017472  012702  000012                MOV     #12,R2          ;SAVE LU REG FOR TYPEOUT
3816  017476  004737  030276                JSR     PC,SYNC         ;SINGLE CLOCK 1 SYNC CHARACTERS
3817  017502  000001                        1
3818  017504  104413  000024                DATACLK,        24
3819  017510  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3820  017512  021244                        021244                  ;PORT4 LU12
3821  017514  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3822  017520  042704  000277                BIC     #277,R4         ;CLEAR UNWANTED BITS
3823  017524  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
3824  017526  120504                        CMPB    R5,R4           ;IS ACTIVE CLEAR?
3825  017530  001401                        BEQ     1$              ;BR IF YES
3826  017532  104040                        ERROR   40              ;ERROR ACTIVE IS NOT CLEAR
3827  017534                        1$:
3828
3829
3830                                        ;*************************** TEST 27 ***************************
3831                                        ;*DDCMP IN ACTIVE TEST
3832                                        ;*SET LU LOOP, SINGLE STEP 2 SYNCS AND A NON-SYNC (301)
3833                                        ;*VERIFY THAT IN ACTIVE IS SET
3834                                        ;;*************************************************************
3835
3836                                        ;  TEST 27
3837                                        ;-------------
3838                                    ;;*************************************************************
3839  017534  000004                TST27:  SCOPE
3840  017536  012737  000027  001202        MOV     #27,$TSTNM              ; LOAD THE NO. OF THIS TEST
3841  017544  012737  017624  001442        MOV     #TST30,NEXT             ; POINT TO THE START OF NEXT TEST.
3842                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3843  017552  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3844  017554  012711  004000                MOV     #BIT11,(R1)     ;SET LU LOOP
3845  017560  012702  000012                MOV     #12,R2          ;SAVE LU REG FOR TYPEOUT
3846  017564  004737  030276                JSR     PC,SYNC         ;SINGLE CLOCK 2 SYNC CHARACTERS
3847  017570  000002                        2
3848  017572  104413  000034                DATACLK,        34
3849  017576  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3850  017600  021244                        021244                  ;PORT4 LU12
3851  017602  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3852  017606  042704  000277                BIC     #277,R4         ;CLEAR UNWANTED BITS
3853  017612  012705  000100                MOV     #BIT6,R5                ;PUT "EXPECTED" IN R5
3854  017616  120504                        CMPB    R5,R4           ;IS ACTIVE SET?
3855  017620  001401                        BEQ     1$              ;BR IF YES
```

```
3856  017622  104040                          ERROR   40              ;ERROR ACTIVE IS NOT SET
3857  017624                          1$:
3858
3859
3860                                          ;*************************** TEST 30 ***************************
3861                                          ;*IN CLEAR TEST
3862                                          ;*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
3863                                          ;*WAIT FOR IN RDY, THEN SET IN CLEAR
3864                                          ;*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
3865                                          ;;*************************************************************
3866
3867                                          ;  TEST 30
3868                                          ;-------------
3869                                          ;;*************************************************************
3870  017624  000004                  TST30:  SCOPE
3871  017626  012737  000030  001202          MOV     #30,$TSTNM              ; LOAD THE NO. OF THIS TEST
3872  017634  012737  017776  001442          MOV     #TST31,NEXT             ; POINT TO THE START OF NEXT TEST.
3873                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3874  017642  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3875  017644  012702  000012                  MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3876  017650  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3877  017654  004737  030444                  JSR     PC,CHAR         ;LOAD SILO WITH 3 SYNCS
3878  017660  000301                          301                     ;AND A NON-SYNC  (301)
3879  017662  104413  000053                  DATACLK,        53      ;SINGLE CLOCK THE DATA
3880  017666  104414  000002                  TIMER,  2               ;WAIT FOR INRDY
3881  017672  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3882  017674  021244                          021244                  ;PORT4 LU 12
3883  017676  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3884  017702  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
3885  017706  012705  000020                  MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
3886  017712  120504                          CMPB    R5,R4           ;IS INRDY SET?
3887  017714  001401                          BEQ     1$
3888  017716  104040                          ERROR   40              ;ERROR, INRDY IS NOT SET
3889  017720                          1$.
3890  017720  012761  000200  000004          MOV     #BIT7,4(R1)     ;LOAD PORT4
3891  017726  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3892  017730  122112                          122112                  ;SET IN CLEAR
3893  017732  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3894  017734  021244                          021244                  ;PORT4 LU 12
3895  017736  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3896  017742  042704  000277                  BIC     #277,R4         ;CLEAR UNWANTED BITS
3897  017746  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
3898  017750  120504                          CMPB    R5,R4           ;IS IN ACTIVE CLEAR?
3899  017752  001401                          BEQ     2$
3900  017754  104040                          ERROR   40              ;ERROR, IN ACTIVE IS NOT CLEAR
3901  017756                          2$:
3902  017756  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3903  017762  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
3904  017766  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
3905  017770  120504                          CMPB    R5,R4           ;IS INRDY CLEARED?
3906  017772  001401                          BEQ     3$
3907  017774  104040                          ERROR   40              ;ERROR, INRDY IS NOT CLEARED
3908  017776                          3$:
3909
3910
3911                                          ;*************************** TEST 31 ***************************
```

```
                                        ;*DDCMP BASIC RECEICER TEST
                                        ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
                                        ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
                                        ;;*****************************************************************

                                        ;  TEST 31
                                        ;-------------
                                    ;;*****************************************************************
020000  012737  000031  001202  TST31: SCOPE
020006  012737  020112  001442        MOV     #31,$TSTNM              ; LOAD THE NO. OF THIS TEST
                                      MOV     #TST32,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
020014  104410                        MSTCLR                        ;MASTER CLEAR KMC11
020016  012702  000012                MOV     #12,R2                ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
020022  012711  004000                MOV     #BIT11,(R1)           ;SET LINE UNIT LOOP
020026  004737  030444                JSR     PC,CHAR               ;LOAD SILO WITH 3 SYNCS
020032  000000                        0                             ;AND THE CHARACTER 0
020034  104413  000053                DATACLK,        53            ;SINGLE CLOCK THE DATA
020040  104414  000002                TIMER,  2                     ;WAIT FOR INRDY
020044  104412                        ROMCLK                        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020046  021244                        021244                        ;PORT4 LU 12
020050  016104  000004                MOV     4(R1),R4              ;PUT "FOUND" IN R4
020054  042704  000357                BIC     #357,R4               ;CLEAR UNWANTED BITS
020060  012705  000020                MOV     #BIT4,R5                      ;PUT "EXPECTED" IN R5
020064  120504                        CMPB    R5,R4                 ;IS INRDY SET?
020066  001401                        BEQ     1$
020070  104040                        ERROR   40                    ;ERROR, INRDY IS NOT SET
020072                          1$:
020072  104412                        ROMCLK                        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020074  021204                        021204                        ;PORT4 IN DATA
020076  016104  000004                MOV     4(R1),R4              ;PUT "FOUND" IN R4
020102  005005                        CLR     R5                    ;PUT "EXPECTED" IN R5
020104  120504                        CMPB    R5,R4                 ;WAS A 0 RECEIVED?
020106  001401                        BEQ     2$
020110  104010                        ERROR   10                    ;ERROR, RECEIVED DATA IS WRONG
020112                          2$:


                                        ;************************** TEST 32 **************************
                                        ;*DDCMP BASIC RECEICER TEST
                                        ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
                                        ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
                                        ;;*****************************************************************

                                        ;  TEST 32
                                        ;-------------
                                    ;;*****************************************************************
020112  000004                  TST32: SCOPE
020114  012737  000032  001202        MOV     #32,$TSTNM              ; LOAD THE NO. OF THIS TEST
020122  012737  020230  001442        MOV     #TST33,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
020130  104410                        MSTCLR                        ;MASTER CLEAR KMC11
020132  012702  000012                MOV     #12,R2                ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
020136  012711  004000                MOV     #BIT11,(R1)           ;SET LINE UNIT LOOP
020142  004737  030444                JSR     PC,CHAR               ;LOAD SILO WITH 3 SYNCS
020146  000125                        125                           ;AND THE CHARACTER 125
```

```
3968  020150  104413  000053                DATACLK,        53      ;SINGLE CLOCK THE DATA
3969  020154  104417  000002                TIMER,  2               ;WAIT FOR INRDY
3970  020160  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3971  020162  021244                        021244                  ;PORT4 LU 12
3972  020164  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3973  020170  042704  000357                BIC     #357,R4         ;CLEAR UNWANTED BITS
3974  020174  012705  000020                MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
3975  020200  120504                        CMPB    R5,R4           ;IS INRDY SET?
3976  020202  001401                        BEQ     1$
3977  020204  104040                        ERROR   40              ;ERROR, INRDY IS NOT SET
3978  020206                          1$:
3979  020206  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3980  020210  021204                        021204                  ;PORT4 IN DATA
3981  020212  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3982  020216  012705  000125                MOV     #125,R5         ;PUT "EXPECTED" IN R5
3983  020222  120504                        CMPB    R5,R4           ;WAS A 125 RECEIVED?
3984  020224  001401                        BEQ     2$
3985  020226  104010                        ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
3986  020230                          2$:
3987
3988
3989                                        ;***************************** TEST 33 ****************************
3990                                        ;*DDCMP BASIC RECEICER TEST
3991                                        ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3992                                        ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3993                                        ;:**********************************************************************
3994
3995                                        ;  TEST 33
3996                                        ;-------------
3997                                    ;;**********************************************************************
3998  020230  000004                  TST33:  SCOPE
3999  020232  012737  000033  001202        MOV     #33,$TSTNM              ; LOAD THE NO. OF THIS TEST
4000  020240  012737  020346  001442        MOV     #TST34,NEXT             ; POINT TO THE START OF NEXT TEST.
4001                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
4002  020246  104410                        MSTCLR                  ;MASTER CLEAR KMC11
4003  020250  012702  000012                MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
4004  020254  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4005  020260  004737  030444                JSR     PC,CHAR         ;LOAD SILO WITH 3 SYNCS
4006  020264  000252                        252                     ;AND THE CHARACTER 252
4007  020266  104413  000053                DATACLK,        53      ;SINGLE CLOCK THE DATA
4008  020272  104414  000002                TIMER,  2               ;WAIT FOR INRDY
4009  020276  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4010  020300  021244                        021244                  ;PORT4 LU 12
4011  020302  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
4012  020306  042704  000357                BIC     #357,R4         ;CLEAR UNWANTED BITS
4013  020312  012705  000020                MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
4014  020316  120504                        CMPB    R5,R4           ;IS INRDY SET?
4015  020320  001401                        BEQ     1$
4016  020322  104040                        ERROR   40              ;ERROR, INRDY IS NOT SET
4017  020324                          1$:
4018  020324  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4019  020326  021204                        021204                  ;PORT4 IN DATA
4020  020330  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
4021  020334  012705  000252                MOV     #252,R5         ;PUT "EXPECTED" IN R5
4022  020340  120504                        CMPB    R5,R4           ;WAS A 252 RECEIVED?
4023  020342  001401                        BEQ     2$
```

```
 4024  020344  104010                          ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
 4025  020346                            2$:
 4026
 4027
 4028                                          ;*************************** TEST 34 ***************************
 4029                                          ;*DDCMP BASIC RECEICER TEST
 4030                                          ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
 4031                                          ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
 4032                                          ;;***********************************************************************
 4033
 4034                                          ;  TEST 34
 4035                                          ;-------------
 4036                                      ;;**********************************************************************
 4037  020346  000004                    TST34:  SCOPE
 4038  020350  012737  000034  001202          MOV     #34,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4039  020356  012737  020464  001442          MOV     #TST35,NEXT             ; POINT TO THE START OF NEXT TEST.
 4040                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4041  020364  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4042  020366  012702  000012                  MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
 4043  020372  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4044  020376  004737  030444                  JSR     PC,CHAR         ;LOAD SILO WITH 3 SYNCS
 4045  020402  000377                          377                     ;AND THE CHARACTER 377
 4046  020404  104413  000053                  DATACLK,        53      ;SINGLE CLOCK THE DATA
 4047  020410  104414  000002                  TIMER,  2               ;WAIT FOR INRDY
 4048  020414  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4049  020416  021244                          021244                  ;PORT4 LU 12
 4050  020420  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4051  020424  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
 4052  020430  012705  000020                  MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
 4053  020434  120504                          CMPB    R5,R4           ;IS INRDY SET?
 4054  020436  001401                          BEQ     1$
 4055  020440  104040                          ERROR   40              ;ERROR, INRDY IS NOT SET
 4056  020442                            1$:
 4057  020442  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4058  020444  021204                          021204                  ;PORT4 IN DATA
 4059  020446  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4060  020452  012705  000377                  MOV     #377,R5         ;PUT "EXPECTED" IN R5
 4061  020456  120504                          CMPB    R5,R4           ;WAS A 377 RECEIVED?
 4062  020460  001401                          BEQ     2$
 4063  020462  104010                          ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
 4064  020464                            2$:
 4065
 4066
 4067                                          ;*************************** TEST 35 ***************************
 4068                                          ;*DDCMP DATA TEST
 4069                                          ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
 4070                                          ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
 4071                                          ;;***********************************************************************
 4072
 4073                                          ;  TEST 35
 4074                                          ;-------------
 4075                                      ;;**********************************************************************
 4076  020464  000004                    TST35:  SCOPE
 4077  020466  012737  000035  001202          MOV     #35,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4078  020474  012737  020614  001442          MOV     #TST36,NEXT             ; POINT TO THE START OF NEXT TEST.
 4079                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
```

```
4080  020502  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4081  020504  005037  031062                  CLR     SCHAR           ;START BINARY COUNT AT ZERO
4082  020510  005037  031064                  CLR     STUFLG          ;CLEAR BITSTUFF FLAG
4083  020514  005002                          CLR     R2              ;R2 IS "EXPECTED" DATA
4084  020516  012703  000073                  MOV     #73,R3          ;R3 IS CHARACTER COUNT
4085  020522  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4086  020526  004737  030622                  JSR     PC,SILOLD       ;LOAD SILO WITH COUNT PATTERN
4087  020532  104413  000043                  DATACLK,        43      ;SYNC RECEIVER AND GET IT ACTIVE
4088  020536  104413  000730          1$:     DATACLK,        730     ;CLOCK IN 73 CHARACTERS
4089  020542  004737  031066          4$:     JSR     PC,INRDY        ;WAIT FOR INRDY
4090  020546  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4091  020550  021204                          021204                  ;PORT4 IN DATA
4092  020552  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4093  020556  010205                          MOV     R2,R5           ;PUT "EXPECTED" IN R5
4094  020560  120504                          CMPB    R5,R4           ;IS DATA CORRECT?
4095  020562  001401                          BEQ     2$              ;BR IF YES
4096  020564  104010                          ERROR   10              ;DATA ERROR
4097  020566  005202          2$:     INC     R2              ;NEXT CHARACTER
4098  020570  022702  000400                  CMP     #400,R2         ;ALL DONE?
4099  020574  001407                          BEQ     3$              ;BR IF YES
4100  020576  005303                          DEC     R3              ;DECREMENT CHARACTER COUNT
4101  020600  001360                          BNE     4$              ;BR IF SILO NOT EMPTY
4102  020602  004737  030622                  JSR     PC,SILOLD       ;LOAD SILO WITH MORE OF COUNT PATTERN
4103  020606  012703  000073                  MOV     #73,R3          ;RELOAD CHARACTER COUNT
4104  020612  000751                          BR      1$              ;CONTINUE
4105  020614                          3$:
4106
4107                                          ;****************************** TEST 36 ****************************
4108                                          ;*DDCMP DATA TEST
4109                                          ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
4110                                          ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
4111                                          ;*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
4112                                          ;*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12
4113                                          ;:**************************************************************
4114
4115                                          ;  TEST 36
4116                                          ;-------------
4117                                  ;;******************************************************************
4118  020614  000004                  TST36:  SCOPE
4119  020616  012737  000036  001202          MOV     #36,$TSTNM              ; LOAD THE NO. OF THIS TEST
4120  020624  012737  020754  001442          MOV     #TST37,NEXT             ; POINT TO THE START OF NEXT TEST.
4121                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4122  020632  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4123  020634  005037  031062                  CLR     SCHAR           ;START BINARY COUNT AT ZERO
4124  020640  005037  031064                  CLR     STUFLG          ;CLEAR BITSTUFF FLAG
4125  020644  005002                          CLR     R2              ;R2 IS "EXPECTED" DATA
4126  020646  012703  000073                  MOV     #73,R3          ;R3 IS CHARACTER COUNT
4127  020652  005011                          CLR     (R1)            ;CLEAR LU LOOP IN MAINT REG
4128  020654  012761  000040  000004          MOV     #BIT5,4(R1)     ;LOAD PORT4
4129  020662  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4130  020664  122112                          122112                  ;SET LU LOOP IN LU REG 12
4131  020666  004737  030622                  JSR     PC,SILOLD       ;LOAD SILO WITH COUNT PATTERN
4132  020672  104413  000043                  DATACLK,        43      ;SYNC RECEIVER AND GET IT ACTIVE
4133  020676  104413  000730          1$:     DATACLK,        730     ;CLOCK IN 73 CHARACTERS
4134  020702  004737  031066          4$:     JSR     PC,INRDY        ;WAIT FOR INRDY
```

```
 4136  020706  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4137  020710  021204                          021204                  ;PORT4 IN DATA
 4138  020712  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4139  020716  010205                          MOV     R2,R5           ;PUT "EXPECTED" IN R5
 4140  020720  120504                          CMPB    R5,R4           ;IS DATA CORRECT?
 4141  020722  001401                          BEQ     2$              ;BR IF YES
 4142  020724  104010                          ERROR   10              ;DATA ERROR
 4143  020726  005202                  2$:     INC     R2              ;NEXT CHARACTER
 4144  020730  022702  000400                  CMP     #400,R2         ;ALL DONE?
 4145  020734  001407                          BEQ     3$              ;BR IF YES
 4146  020736  005303                          DEC     R3              ;DECREMENT CHARACTER COUNT
 4147  020740  001360                          BNE     4$              ;BR IF SILO NOT EMPTY
 4148  020742  004737  030622                  JSR     PC,SILOLD       ;LOAD SILO WITH MORE OF COUNT PATTERN
 4149  020746  012703  000073                  MOV     #73,R3          ;RELOAD CHARACTER COUNT
 4150  020752  000751                          BR      1$              ;CONTINUE
 4151  020754                          3$:
 4152
 4153
 4154                                          ;***************************** TEST 37 ****************************
 4155                                          ;*TRANSMITTER MARK TEST
 4156                                          ;*SINGLE CLOCK 3 SYNCS AND A 301 AND 20 EXTRA
 4157                                          ;*CLOCK TICKS, VERIFY THAT A 301, A 377 AND A 377
 4158                                          ;*WERE RECEIVED INDICATING THAT THE TRANSMITTER WENT
 4159                                          ;*TO A MARK STATE FOR 16 BITS WHEN OUT SILO WAS EMPTY
 4160                                          ;;*************************************************************
 4161
 4162                                          ;  TEST 37
 4163                                          ;-------------
 4164                                  ;;*************************************************************
 4165  020754  000004                  TST37:  SCOPE
 4166  020756  012737  000037  001202          MOV     #37,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4167  020764  012737  021114  001442          MOV     #TST40,NEXT             ; POINT TO THE START OF NEXT TEST.
 4168                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4169  020772  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4170  020774  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4171  021000  004737  030444                  JSR     PC,CHAR         ;LOAD SILO WITH 3 SYNCS
 4172  021004  000301                          301                     ;AND A 301
 4173  021006  104413  000073                  DATACLK,        73      ;CLOCK THE 301 IN AND 20 EXTRA TICKS
 4174  021012  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
 4175  021016  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4176  021020  021204                          021204                  ;PORT4 IN DATA
 4177  021022  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4178  021026  012705  000301                  MOV     #301,R5         ;PUT "EXPECTED" IN R5
 4179  021032  120504                          CMPB    R5,R4           ;WAS A 301 RECEIVED?
 4180  021034  001401                          BEQ     1$
 4181  021036  104010                          ERROR   10              ;ERROR FIRST CHARACTER INCORRECT
 4182  021040  004737  031066          1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
 4183  021044  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4184  021046  021204                          021204                  ;PORT4 IN DATA
 4185  021050  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4186  021054  012705  000377                  MOV     #377,R5         ;PUT "EXPECTED" IN R5
 4187  021060  120504                          CMPB    R5,R4           ;WAS A 377 RECEIVED?
 4188  021062  001401                          BEQ     2$
 4189  021064  104010                          ERROR   10              ;ERROR, 377 WAS NOT RECEIVED
 4190  021066  004737  031066          2$:     JSR     PC,INRDY        ;WAIT FOR INRDY
 4191  021072  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
                                                021204                  ;PORT4 IN DATA
                                                MOV     4(R1),R4        ;PUT "FOUND" IN R4
                                                MOV     #377,R5         ;PUT "EXPECTED" IN R5
                                                CMPB    R5,R4           ;WAS A 377 RECEIVED?
                                                BEQ     3$
                                                ERROR   10              ;ERROR, 177 WAS NOT RECEIVED
                                        3$:

                                                ;***************************** TEST 40 ****************************
                                                ;*CABLE TURNAROUND TEST
                                                ;*CLEAR LINE UNIT LOOP, SET DTR
                                                ;*VERIFY THAT RING AND MODEM READY ARE SET
                                                ;*CLEAR DTR, VERIFY THAT RING AND MRDY ARE CLEARED
                                                ;:**************************************************************

                                                ;  TEST 40
                                                ;-------------
                                        ;;**************************************************************
                                        TST40:  SCOPE
                                                MOV     #40,$TSTNM              ; LOAD THE NO. OF THIS TEST
                                                MOV     #TST41,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                        ;R1 CONTAINS BASE KMC11 ADDRESS
                                                MSTCLR                  ;MASTER CLEAR KMC11
                                                BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
                                                BNE     .+12            ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
                                                BIT     #BIT14,STAT1    ;IS TURNAROUND CONNECTOR ON?
                                                BEQ     2$              ;SKIP TEST IF NO
                                                CLR     (R1)            ;CLEAR LINE UNIT LOOP
                                                MOV     #100,4(R1)      ;LOAD PORT4
                                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                                122113                  ;SET DTR
                                                TIMER,  2               ;WAIT
                                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                                021264                  ;PORT4 LU13
                                                MOV     4(R1),R4        ; PUT FOUND IN R4       ++++NEW
                                                BIC     #23,R4          ; CLEAR JUNK
                                                MOV     #310,R5         ; GET EXPECTED.
                                                BIT     #BIT13,STAT1    ; IS LINE UNIT M8202?
                                                BEQ     .+6
                                                BIC     #BIT7,R5        ; NO RING ON M8202

                                                BIT     #BIT2,STAT3     ; IS THIS V.35 MODEM?
                                                BEQ     3$              ; NO THEN GO AHEAD.

                                                BIC     #BIT7,R5        ; YES-NO RING ON V.35 MODEM
                                        3$:
                                                CMP     R5,R4           ; ARE RING AND MODEM READY SET?
                                                BEQ     1$              ; WARNING! IF V.35 AND AUTO STARTED,
                                                                        ; YOU WILL GET THIS ERROR. YOU MUST
                                                                        ; MANUALL NASWER THE QUESTIONS FOR V.35.
                                                ERROR   11              ;ERROR, RING OR MRDY NOT SET
                                        1$:     CLR     4(R1)           ;CLEAR PORT4
                                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                                122113                  ;CLEAR DTR
                                                TIMER,  2
```

```
 4248  021266  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4249  021270  021264                          021264                  ;PORT4 LU13
 4250  021272  016104  000004                  MOV     4(R1),R4        ; PUR FOUND IN R4                  +++NEW
 4251  021276  042704  000023                  BIC     #23,R4          ; STRIP JUNK
 4252  021302  005005                          CLR     R5              ; SET EXPECTED.
 4253  021304  032737  020000  002050          BIT     #BIT13,STAT1    ; IS THIS A M8202?
 4254  021312  001402                          BEQ     .+6
 4255  021314  052705  000010                  BIS     #BIT3,R5        ; YES THEN EXPECT MRDY SET.
 4256  021320  032737  000004  002054          BIT     #BIT2,STAT3     ; IS THIS V.35?
 4257  021326  001402                          BEQ     4$
 4258  021330  042704  000200                  BIC     #BIT7,R4
 4259  021334                          4$:
 4260  021334  120405                          CMPB    R4,R5           ; ARE RING AND MRDY CLEAR?
 4261  021336  001401                          BEQ     2$
 4262                                          ; WARNING! YOU MAY GET THIS ERROR IF V.35
 4263                                          ; AND AUTOSTART. YOU MUST MANNUALLY ANSWER
 4264                                          ; ALL QUESTIONS IF V.35.
 4265  021340  104011                          ERROR   11              ;ERROR, RING OR MRDY NOT CLEAR
 4266  021342                          2$:
 4267
 4268
 4269                                          ;************************** TEST 41 ************************
 4270                                          ;*CABLE TURNAROUND TEST
 4271                                          ;*CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
 4272                                          ;*VERIFY THAT ALL MODEM SIGNALS ARE SET
 4273                                          ;;*********************************************************
 4274
 4275                                          ;  TEST 41
 4276                                          ;-------------
 4277                                  ;;*********************************************************
 4278  021342  000004                  TST41:  SCOPE
 4279  021344  012737  000041  001202          MOV     #41,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4280  021352  012737  021536  001442          MOV     #TST42,NEXT             ; POINT TO THE START OF NEXT TEST.
 4281                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4282  021360  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4283  021362  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
 4284  021370  001004                          BNE     .+12            ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
 4285  021372  032737  040000  002050          BIT     #BIT14,STAT1    ;IS TURNAROUND CONNECTOR ON?
 4286  021400  001456                          BEQ     1$              ;SKIP TEST IF NO
 4287  021402  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4288  021406  012761  000100  000004          MOV     #100,   4(R1)           ;LOAD PORT4
 4289  021414  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4290  021416  122113                          122113                  ;CLEAR ALL MODEM SIGNALS,EXCEPT DTR
 4291  021420  104414  000002                  TIMER,  2               ;WAIT
 4292  021424  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
 4293  021432  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4294  021434  122111                          122111                  ;SET SOM
 4295  021436  004537  031530                  JSR     R5,MESLD        ;FILL OUT DATA SILO
 4296  021442  032012                          MESDAT                  ;WITH 64 CHARACTERS
 4297  021444  000100                          64.
 4298  021446  012700  000050                  MOV     #50,R0          ;PREPARE FOR DELAY
 4299  021452  005011                          CLR     (R1)            ;CLEAR LINE UNIT LOOP
 4300  021454                          2$:
 4301  021454  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4302  021456  021264                          021264                  ;PORT4 LU13
 4303  021460  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
```

```
 4304  021464  042704  000023                  BIC     #23,R4          ;CLEAR UNWANTED BITS
 4305  021470  012705  000354                  MOV     #354,R5         ;PUT "EXPECTED" IN R5
 4306  021474  032737  000004  002054          BIT     #BIT2,STAT3
 4307  021502  001402                          BEQ     4$
 4308  021504  042705  000200                  BIC     #BIT7,R5        ; NO RING ON V.35
 4309  021510                          4$:
 4310  021510  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
 4311  021516  001402                          BEQ     .+6             ;BR IF NO
 4312  021520  042705  000200                  BIC     #BIT7,R5        ;NO RING ON M8202
 4313  021524  120504                          CMPB    R5,R4           ;COMPARE EXPECTED AND FOUND
 4314  021526  001403                          BEQ     1$              ;BR IF OK
 4315  021530  005300                          DEC     R0              ;DEC DELAY COUNT
 4316  021532  001350                          BNE     2$              ;BR IF NOT ZERO
 4317  021534  104011                          ERROR   11              ;ERROR, ALL SIGNALS ARE NOT SET
 4318                                                                  ; WARNING YOU MUST MANUALLY ANSWER QUESTIONS
 4319                                                                  ; IF YOU HAVE V.35.
 4320  021536                          1$:
 4321
 4322
 4323                                          ;************************** TEST 42 **************************
 4324                                          ;*TEST OF CRC OPERATION
 4325                                          ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 4326                                          ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 4327                                          ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
 4328                                          ;;*********************************************************************
 4329
 4330                                          ;  TEST 42
 4331                                          ;-------------
 4332                                  ;;*********************************************************************
 4333  021536  000004                  TST42:  SCOPE
 4334  021540  012737  000042  001202          MOV     #42,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4335  021546  012737  022052  001442          MOV     #TST43,NEXT             ; POINT TO THE START OF NEXT TEST.
 4336  021554  012737  021570  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
 4337                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4338  021562  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4339  021564  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
 4340  021570  004737  031572          64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
 4341  021574  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
 4342  021576  012737  120001  031226          MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
 4343  021604  012737  000000  021644          MOV     #0,66$          ;LOAD CHAR FOR SOFTWARE BCC
 4344  021612  005037  021646                  CLR     67$             ;CLEAR OLD SOFTWARE BCC
 4345  021616  004737  031232                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4346  021622  000000                          0                       ;AND THE CHARACTER 0
 4347  021624  104413  000021                  DATACLK,        21      ;GET TRANSMITTER ACTIVE
 4348  021630  104413  000001          65$:    DATACLK,        1       ;SHIFT BCC ONCE
 4349  021634  005200                          INC     R0              ;BUMP SHIFT COUNT
 4350  021636  004537  031122                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4351  021642  000001                          1                       ;ONE SHIFT
 4352  021644  000000          66$:    0                               ;DATA CHARACTER
 4353  021646  000000          67$:    0                               ;OLD BCC
 4354  021650  103405                          BCS     68$             ;BR IF SOFT BCC LSB IS SET
 4355  021652  004737  031344                  JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
 4356  021656  103006                          BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
 4357  021660  104012                          ERROR   12              ;ERROR, BCC LSB IS SET
 4358  021662  000404                          BR      69$             ;CONTINUE
 4359  021664  004737  031344          68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
```

```
 4360  021670  103401                        BCS     69$             ;BR IF HARD BCC LSB IS SET
 4361  021672  104016                        ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
 4362  021674                          69$:
 4363  021674  006037  021644                ROR     66$             ;SHIFT SOFT DATA
 4364  021700  013737  031230  021646        MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
 4365  021706  022700  000010                CMP     #10,R0          ;DONE YET?
 4366  021712  001346                        BNE     65$             ;BR IF NOT DONE
 4367  021714  104405                        SCOP1                   ;SCOPE SUBTEST (SW09=1)
 4368  021716  012737  021724  001444        MOV     #71$,LOCK       ;NEW SCOPE1
 4369  021724  004737  031572          71$:  JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
 4370  021730  005000                        CLR     R0              ;START SHIFT COUNTER AT ZERO
 4371  021732  012737  120001  031226        MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
 4372  021740  012737  000000  022000        MOV     #0,73$          ;LOAD CHAR FOR SOFTWARE BCC
 4373  021746  005037  022002                CLR     74$             ;CLEAR OLD SOFTWARE BCC
 4374  021752  004737  031232                JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4375  021756  000000                        0                       ;AND THE CHARACTER 0
 4376  021760  104413  000032                DATACLK,        32      ;GET RECEIVER ACTIVE
 4377  021764  104413  000001          72$:  DATACLK,        1       ;SHIFT BCC ONCE
 4378  021770  005200                        INC     R0              ;BUMP SHIFT COUNT
 4379  021772  004537  031122                JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4380  021776  000001                        1                       ;ONE SHIFT
 4381  022000  000000                  73$:  0                       ;DATA CHARACTER
 4382  022002  000000                  74$:  0                       ;OLD BCC
 4383  022004  103405                        BCS     75$             ;BR IF SOFT BCC LSB IS SET
 4384  022006  004737  031356                JSR     PC,GETQ1        ;GET HARDWARE RECEIVER BCC LSB
 4385  022012  103006                        BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
 4386  022014  104013                        ERROR   13              ;ERROR, BCC LSB IS SET
 4387  022016  000404                        BR      76$             ;CONTINUE
 4388  022020  004737  031356          75$:  JSR     PC,GETQ1        ;GET HARDWARE RECEIVER BCC LSB
 4389  022024  103401                        BCS     76$             ;BR IF HARD BCC LSB IS SET
 4390  022026  104017                        ERROR   17              ;ERROR, BCC LSB IS CLEAR
 4391  022030                          76$:
 4392  022030  006037  022000                ROR     73$             ;SHIFT SOFT DATA
 4393  022034  013737  031230  022002        MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
 4394  022042  022700  000010                CMP     #10,R0          ;DONE YET?
 4395  022046  001346                        BNE     72$             ;BR IF NOT DONE
 4396  022050  104405                        SCOP1                   ;SCOPE SUBTEST (SW09=1)
 4397  022052                          77$:
 4398
 4399
 4400                                        ;*************************** TEST 43 ***************************
 4401                                        ;*TEST OF CRC OPERATION
 4402                                        ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 4403                                        ;*377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 4404                                        ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
 4405                                        ;;*************************************************************
 4406
 4407                                        ;  TEST 43
 4408                                        ;-------------
 4409                                  ;;*************************************************************
 4410  022052  000004                TST43:  SCOPE
 4411  022054  012737  000043  001202        MOV     #43,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4412  022062  012737  022365  001442        MOV     #TST44,NEXT             ; POINT TO THE START OF NEXT TEST.
 4413  022070  012737  022104  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
 4414                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
 4415  022076  104410                        MSTCLR                  ;MASTER CLEAR KMC11
```

```
 4416  022100  012711  004000                 MOV     #BIT11,(R1)     ;SET LU LOOP
 4417  022104  004737  031572          64$:   JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
 4418  022110  005000                         CLR     R0              ;START SHIFT COUNTER AT ZERO
 4419  022112  012737  120001  031226         MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
 4420  022120  012737  000377  022160         MOV     #377,66$;       ;LOAD CHAR FOR SOFTWARE BCC
 4421  022126  005037  022162                 CLR     67$             ;CLEAR OLD SOFTWARE BCC
 4422  022132  004737  031232                 JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4423  022136  000377                         377                     ;AND THE CHARACTER 377
 4424  022140  104413  000021                 DATACLK,        21      ;GET TRANSMITTER ACTIVE
 4425  022144  104413  000001          65$:   DATACLK,        1       ;SHIFT BCC ONCE
 4426  022150  005200                         INC     R0              ;BUMP SHIFT COUNT
 4427  022152  004537  031122                 JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4428  022156  000001                         1                       ;ONE SHIFT
 4429  022160  000000          66$:   0                               ;DATA CHARACTER
 4430  022162  000000          67$:   0                               ;OLD BCC
 4431  022164  103405                         BCS     68$             ;BR IF SOFT BCC LSB IS SET
 4432  022166  004737  031344                 JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
 4433  022172  103006                         BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
 4434  022174  104012                         ERROR   12              ;ERROR, BCC LSB IS SET
 4435  022176  000404                         BR      69$             ;CONTINUE
 4436  022200  004737  031344          68$:   JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
 4437  022204  103401                         BCS     69$             ;BR IF HARD BCC LSB IS SET
 4438  022206  104016                         ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
 4439  022210                         69$:
 4440  022210  006037  022160                 ROR     66$             ;SHIFT SOFT DATA
 4441  022214  013737  031230  022162         MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
 4442  022222  022700  000010                 CMP     #10,R0          ;DONE YET?
 4443  022226  001346                         BNE     65$             ;BR IF NOT DONE
 4444  022230  104405                         SCOP1                   ;SCOPE SUBTEST (SW09=1)
 4445  022232  012737  022240  001444         MOV     #71$,LOCK       ;NEW SCOPE1
 4446  022240  004737  031572          71$:   JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
 4447  022244  005000                         CLR     R0              ;START SHIFT COUNTER AT ZERO
 4448  022246  012737  120001  031226         MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
 4449  022254  012737  000377  022314         MOV     #377,73$;       ;LOAD CHAR FOR SOFTWARE BCC
 4450  022262  005037  022316                 CLR     74$             ;CLEAR OLD SOFTWARE BCC
 4451  022266  004737  031232                 JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4452  022272  000377                         377                     ;AND THE CHARACTER 377
 4453  022274  104413  000032                 DATACLK,        32      ;GET RECEIVER ACTIVE
 4454  022300  104413  000001          72$:   DATACLK,        1       ;SHIFT BCC ONCE
 4455  022304  005200                         INC     R0              ;BUMP SHIFT COUNT
 4456  022306  004537  031122                 JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4457  022312  000001                         1                       ;ONE SHIFT
 4458  022314  000000          73$:   0                               ;DATA CHARACTER
 4459  022316  000000          74$:   0                               ;OLD BCC
 4460  022320  103405                         BCS     75$             ;BR IF SOFT BCC LSB IS SET
 4461  022322  004737  031356                 JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
 4462  022326  103006                         BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
 4463  022330  104013                         ERROR   13              ;ERROR, BCC LSB IS SET
 4464  022332  000404                         BR      76$             ;CONTINUE
 4465  022334  004737  031356          75$:   JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
 4466  022340  103401                         BCS     76$             ;BR IF HARD BCC LSB IS SET
 4467  022342  104017                         ERROR   17              ;ERROR, BCC LSB IS CLEAR
 4468  022344                         76$:
 4469  022344  006037  022314                 ROR     73$             ;SHIFT SOFT DATA
 4470  022350  013737  031230  022316         MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
 4471  022356  022700  000010                 CMP     #10,R0          ;DONE YET?
```

```
                                        BNE     72$             ;BR IF NOT DONE
                                        SCOP1                   ;SCOPE SUBTEST (SW09=1)
                                77$:
                                        ;*************************** TEST 44 ***************************
                                        ;*TEST OF CRC OPERATION
                                        ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
                                        ;*125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
                                        ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
                                        ;;*************************************************************
                                        ;  TEST 44
                                        ;-------------
                                ;;*************************************************************
022366  000004                TST44:  SCOPE
022370  012737  000044  001202        MOV     #44,$TSTNM              ; LOAD THE NO. OF THIS TEST
022376  012737  022702  001442        MOV     #TST45,NEXT             ; POINT TO THE START OF NEXT TEST.
022404  012737  022420  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
022412  104410                        MSTCLR                  ;MASTER CLEAR KMC11
022414  012711  004000                MOV     #BIT11,(R1)     ;SET LU LOOP
022420  004737  031572        64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
022424  005000                        CLR     R0              ;START SHIFT COUNTER AT ZERO
022426  012737  120001  031226        MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
022434  012737  000125  022474        MOV     #125,66$;       ;LOAD CHAR FOR SOFTWARE BCC
022442  005037  022476                CLR     67$             ;CLEAR OLD SOFTWARE BCC
022446  004737  031232                JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
022452  000125                        125                     ;AND THE CHARACTER 125
022454  104413  000021                DATACLK,        21      ;GET TRANSMITTER ACTIVE
022460  104413  000001        65$:    DATACLK,        1       ;SHIFT BCC ONCE
022464  005200                        INC     R0              ;BUMP SHIFT COUNT
022466  004537  031122                JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
022472  000001                        1                       ;ONE SHIFT
022474  000000                66$:    0                       ;DATA CHARACTER
022476  000000                67$:    0                       ;OLD BCC
022500  103405                        BCS     68$             ;BR IF SOFT BCC LSB IS SET
022502  004737  031344                JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
022506  103006                        BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
022510  104012                        ERROR   12              ;ERROR, BCC LSB IS SET
022512  000404                        BR      69$             ;CONTINUE
022514  004737  031344        68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
022520  103401                        BCS     69$             ;BR IF HARD BCC LSB IS SET
022522  104016                        ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
022524                        69$:
022524  006037  022474                ROR     66$             ;SHIFT SOFT DATA
022530  013737  031230  022476        MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
022536  022700  000010                CMP     #10,R0          ;DONE YET?
022542  001346                        BNE     65$             ;BR IF NOT DONE
022544  104405                        SCOP1                   ;SCOPE SUBTEST (SW09=1)
022546  012737  022554  001444        MOV     #71$,LOCK       ;NEW SCOPE1
022554  004737  031572        71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
022560  005000                        CLR     R0              ;START SHIFT COUNTER AT ZERO
022562  012737  120001  031226        MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
022570  012737  000125  022630        MOV     #125,73$;       ;LOAD CHAR FOR SOFTWARE BCC
022576  005037  022632                CLR     74$             ;CLEAR OLD SOFTWARE BCC
```

```
 4528  022602  004737  031232                 JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4529  022606  000125                         125                     ;AND THE CHARACTER 125
 4530  022610  104413  000032                 DATACLK,        32      ;GET RECEIVER ACTIVE
 4531  022614  104413  000001          72$:   DATACLK,        1       ;SHIFT BCC ONCE
 4532  022620  005200                         INC     R0              ;BUMP SHIFT COUNT
 4533  022622  004537  031122                 JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4534  022626  000001                         1                       ;ONE SHIFT
 4535  022630  000000                   73$:  0                       ;DATA CHARACTER
 4536  022632  000000                   74$:  0                       ;OLD BCC
 4537  022634  103405                         BCS     75$             ;BR IF SOFT BCC LSB IS SET
 4538  022636  004737  031356                 JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
 4539  022642  103006                         BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
 4540  022644  104013                         ERROR   13              ;ERROR, BCC LSB IS SET
 4541  022646  000404                         BR      76$             ;CONTINUE
 4542  022650  004737  031356          75$:   JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
 4543  022654  103401                         BCS     76$             ;BR IF HARD BCC LSB IS SET
 4544  022656  104017                         ERROR   17              ;ERROR, BCC LSB IS CLEAR
 4545  022660                           76$:
 4546  022660  006037  022630                 ROR     73$             ;SHIFT SOFT DATA
 4547  022664  013737  031230  022632         MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
 4548  022672  022700  000010                 CMP     #10,R0          ;DONE YET?
 4549  022676  001346                         BNE     72$             ;BR IF NOT DONE
 4550  022700  104405                         SCOP1                   ;SCOPE SUBTEST (SW09=1)
 4551  022702                           77$:
 4552
 4553
 4554                                         ;***************************** TEST 45 ****************************
 4555                                         ;*TEST OF CRC OPERATION
 4556                                         ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 4557                                         ;*252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 4558                                         ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
 4559                                         ;;**************************************************************
 4560
 4561                                         ;  TEST 45
 4562                                         ;-------------
 4563                                   ;;*****************************************************************
 4564  022702  000004                   TST45: SCOPE
 4565  022704  012737  000045  001202         MOV     #45,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4566  022712  012737  023216  001442         MOV     #TST46,NEXT             ; POINT TO THE START OF NEXT TEST.
 4567  022720  012737  022734  001444         MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
 4568                                                                 ;R1 CONTAINS BASE KMC11 ADDRESS
 4569  022726  104410                         MSTCLR                  ;MASTER CLEAR KMC11
 4570  022730  012711  004000                 MOV     #BIT11,(R1)     ;SET LU LOOP
 4571  022734  004737  031572          64$:   JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
 4572  022740  005000                         CLR     R0              ;START SHIFT COUNTER AT ZERO
 4573  022742  012737  120001  031226         MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
 4574  022750  012737  000252  023010         MOV     #252,66$;       ;LOAD CHAR FOR SOFTWARE BCC
 4575  022756  005037  023012                 CLR     67$             ;CLEAR OLD SOFTWARE BCC
 4576  022762  004737  031232                 JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
 4577  022766  000252                         252                     ;AND THE CHARACTER 252
 4578  022770  104413  000021                 DATACLK,        21      ;GET TRANSMITTER ACTIVE
 4579  022774  104413  000001          65$:   DATACLK,        1       ;SHIFT BCC ONCE
 4580  023000  005200                         INC     R0              ;BUMP SHIFT COUNT
 4581  023002  004537  031122                 JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
 4582  023006  000001                         1                       ;ONE SHIFT
 4583  023010  000000                   66$:  0                       ;DATA CHARACTER
```

```
4584  023012  000000                    67$:    0                       ;OLD BCC
4585  023014  103405                            BCS     68$             ;BR IF SOFT BCC LSB IS SET
4586  023016  004737  031344                    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4587  023022  103006                            BCC     69$             ;BR IF HARD BCC LS3 IS CLEAR
4588  023024  104012                            ERROR   12              ;ERROR, BCC LSB IS SET
4589  023026  000404                            BR      69$             ;CONTINUE
4590  023030  004737  031344            68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4591  023034  103401                            BCS     69$             ;BR IF HARD BCC LSB IS SET
4592  023036  104016                            ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
4593  023040                            69$:
4594  023040  006037  023010                    ROR     66$             ;SHIFT SOFT DATA
4595  023044  013737  031230  023012            MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
4596  023052  022700  000010                    CMP     #10,R0          ;DONE YET?
4597  023056  001346                            BNE     65$             ;BR IF NOT DONE
4598  023060  104405                            SCOP1                   ;SCOPE SUBTEST (SW09=1)
4599  023062  012737  023070  001444            MOV     #71$,LOCK       ;NEW SCOPE1
4600  023070  004737  031572            71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4601  023074  005000                            CLR     R0              ;START SHIFT COUNTER AT ZERO
4602  023076  012737  120001  031226            MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFTWARE BCC
4603  023104  012737  000252  023144            MOV     #252,73$;       ;LOAD CHAR FOR SOFTWARE BCC
4604  023112  005037  023146                    CLR     74$             ;CLEAR OLD SOFTWARE BCC
4605  023116  004737  031232                    JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4606  023122  000252                            252                     ;AND THE CHARACTER 252
4607  023124  104413  000032                    DATACLK,        32      ;GET RECEIVER ACTIVE
4608  023130  104413  000001            72$:    DATACLK,        1       ;SHIFT BCC ONCE
4609  023134  005200                            INC     R0              ;BUMP SHIFT COUNT
4610  023136  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4611  023142  000001                            1                       ;ONE SHIFT
4612  023144  000000                    73$:    0                       ;DATA CHARACTER
4613  023146  000000                    74$:    0                       ;OLD BCC
4614  023150  103405                            BCS     75$             ;BR IF SOFT BCC LSB IS SET
4615  023152  004737  031356                    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4616  023156  103006                            BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
4617  023160  104013                            ERROR   13              ;ERROR, BCC LSB IS SET
4618  023162  000404                            BR      76$             ;CONTINUE
4619  023164  004737  031356            75$:    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4620  023170  103401                            BCS     76$             ;BR IF HARD BCC LSB IS SET
4621  023172  104017                            ERROR   17              ;ERROR, BCC LSB IS CLEAR
4622  023174                            76$:
4623  023174  006037  023144                    ROR     73$             ;SHIFT SOFT DATA
4624  023200  013737  031230  023146            MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
4625  023206  022700  000010                    CMP     #10,R0          ;DONE YET?
4626  023212  001346                            BNE     72$             ;BR IF NOT DONE
4627  023214  104405                            SCOP1                   ;SCOPE SUBTEST (SW09=1)
4628  023216                            77$:
4629
4630
4631                                            ;************************** TEST 46 **************************
4632                                            ;*TRANSMITTER CRC TEST
4633                                            ;*USING THE CRC16 POLYNOMINAL, SINGLE CLOCK A BINARY
4634                                            ;*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
4635                                            ;;**********************************************************************
4636
4637                                            ;  TEST 46
4638                                            ;-------------
4639                                    ;;******************************************************************
```

```
 4640  023216  000004                    TST46:  SCOPE
 4641  023220  012737  000046  001202            MOV     #46,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4642  023226  012737  023454  001442            MOV     #TST47,NEXT             ; POINT TO THE START OF NEXT TEST.
 4643                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
 4644  023234  104410                            MSTCLR                  ;MASTER CLEAR KMC11
 4645  023236  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4646  023242  005003                            CLR     R3              ;ZERO BIT COUNT
 4647  023244  005004                            CLR     R4              ;R4 CONTAINS CHAR TO BE LOADED IN SILO
 4648  023246  005005                            CLR     R5              ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
 4649  023250  005037  023352                    CLR     4$              ;CLEAR SOFT BCC
 4650  023254  012737  120001  031226            MOV     #CRC16,XPOLY    ;LOAD POLYNOMINAL
 4651  023262  004737  031374                    JSR     PC,SYNLD        ;LOAD SILO WITH 2 SYNCS, SOM SET
 4652  023266  010461  000004                    MOV     R4,4(R1)        ;PORT4 CHAR
 4653  023272  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4654  023274  122110                            122110                  ;LOAD OUT DATA
 4655  023276  005204                            INC     R4              ;INCREMENT TO NEXT CHARACTER
 4656  023300  010461  000004                    MOV     R4,4(R1)        ;PORT4 CHAR
 4657  023304  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4658  023306  122110                            122110                  ;LOAD OUT DATA
 4659  023310  005204                            INC     R4              ;INCREMENT TO NEXT CHARACTER
 4660  023312  010461  000004                    MOV     R4,4(R1)        ;PORT4 CHAR
 4661  023316  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4662  023320  122110                            122110                  ;LOAD OUT DATA
 4663  023322  004737  030260                    JSR     PC,OCOR         ;WAIT FOR OCOR
 4664  023326  104413  000021                    DATACLK,21              ;CLOCK DATA
 4665  023332  010537  023350            1$:     MOV     R5,3$           ;LOAD CHAR FOR SOFT CRC
 4666  023336  104413  000001            2$:     DATACLK,1               ;SHIFT BCC ONCE
 4667  023342  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
 4668  023346  000001                            1                       ;SOFT SHIFT COUNT
 4669  023350  000000            3$:     0                               ;SOFT CHARACTER
 4670  023352  000000            4$:     0                               ;OLD SOFT BCC
 4671  023354  103405                            BCS     5$              ;BR IF SOFT BCC LSB IS SET
 4672  023356  004737  031344                    JSR     PC,GETQO                ;GET HARDWARE TRANSMITTER BCC LSB
 4673  023362  103006                            BCC     6$              ;BR IF OK (CLEARED)
 4674  023364  104020                            ERROR   20              ;ERROR, BCC LSB WAS SET
 4675  023366  000404                            BR      6$              ;CONTINUE WITH TEST
 4676  023370  004737  031344            5$:     JSR     PC,GETQO                ;GET HARDWARE TRANSMITTER BCC LSB
 4677  023374  103401                            BCS     6$              ;BR IF OK (SET)
 4678  023376  104021                            ERROR   21              ;ERROR, BCC LSB WAS CLEAR
 4679
 4680  023400                            6$:
 4681  023400  006037  023350                    ROR     3$              ;SHIFT SOFT DATA
 4682  023404  013737  031230  023352            MOV     CALBCC,4$       ;LOAD OLD SOFT BCC
 4683  023412  005203                            INC     R3              ;INCREMENT BIT COUNTER
 4684  023414  022703  000010                    CMP     #10,R3          ;DONE A FULL CHARACTER YET?
 4685  023420  001346                            BNE     2$              ;BR IF NO
 4686  023422  005003                            CLR     R3              ;RESTART BIT COUNTER
 4687  023424  005204                            INC     R4              ;INCREMENT DATA FOR SILO
 4688  023426  022704  000400                    CMP     #400,R4         ;DONE BINARY COUNT YET?
 4689  023432  003404                            BLE     9$              ;BR IF YES
 4690  023434  010461  000004                    MOV     R4,4(R1)        ;PORT4 DATA
 4691  023440  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4692  023442  122110                            122110                  ;LOAD OUT DATA
 4693  023444  005205                    9$:     INC     R5              ;INCREMENT DATA
 4694  023446  022705  000400                    CMP     #400,R5         ;DONE BINARY PATTERN YET?
 4695  023452  001327                            BNE     1$              ;BR IF NO
```

```
 4696  023454                          7$:
 4697
 4698
 4699                                          ;*************************** TEST 47 ***************************
 4700                                          ;*RECEIVER CRC TEST
 4701                                          ;*USING THE CRC16 POLYNOMINAL, SINGLE CLOCK A BINARY
 4702                                          ;*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
 4703                                          ;;*************************************************************************
 4704
 4705                                          ;  TEST 47
 4706                                          ;------------
 4707                                    ;;*************************************************************************
 4708  023454  000004                   TST47:  SCOPE
 4709  023456  012737  000047  001202          MOV     #47,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4710  023464  012737  023712  001442          MOV     #TST50,NEXT             ; POINT TO THE START OF NEXT TEST.
 4711                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4712  023472  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4713  023474  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4714  023500  005003                          CLR     R3              ;ZERO BIT COUNT
 4715  023502  005004                          CLR     R4              ;R4 CONTAINS CHAR TO BE LOADED IN SILO
 4716  023504  005005                          CLR     R5              ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
 4717  023506  005037  023610                  CLR     4$              ;CLEAR SOFT BCC
 4718  023512  012737  120001  031226          MOV     #CRC16,XPOLY    ;LOAD POLYNOMINAL
 4719  023520  004737  031374                  JSR     PC,SYNLD        ;LOAD SILO WITH 2 SYNCS, SOM SET
 4720  023524  010461  000004                  MOV     R4,4(R1)        ;PORT4 CHAR
 4721  023530  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4722  023532  122110                          122110                  ;LOAD OUT DATA
 4723  023534  005204                          INC     R4              ;INCREMENT TO NEXT CHARACTER
 4724  023536  010461  000004                  MOV     R4,4(R1)        ;PORT4 CHAR
 4725  023542  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4726  023544  122110                          122110                  ;LOAD OUT DATA
 4727  023546  005204                          INC     R4              ;INCREMENT TO NEXT CHARACTER
 4728  023550  010461  000004                  MOV     R4,4(R1)        ;PORT4 CHAR
 4729  023554  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4730  023556  122110                          122110                  ;LOAD OUT DATA
 4731  023560  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
 4732  023564  104413  000032                  DATACLK,32              ;CLOCK DATA
 4733  023570  010537  023606          1$:     MOV     R5,3$           ;LOAD CHAR FOR SOFT CRC
 4734  023574  104413  000001          2$:     DATACLK,1               ;SHIFT BCC ONCE
 4735  023600  004537  031122                  JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
 4736  023604  000001                          1                       ;SOFT SHIFT COUNT
 4737  023606  000000          3$:     0                               ;SOFT CHARACTER
 4738  023610  000000          4$:     0                               ;OLD SOFT BCC
 4739  023612  103405                          BCS     5$              ;BR IF SOFT BCC LSB IS SET
 4740  023614  004737  031356                  JSR     PC,GETQI                ;GET HARDWARE RECEIVER BCC LSB
 4741  023620  103006                          BCC     6$              ;BR IF OK (CLEARED)
 4742  023622  104022                          ERROR   22              ;ERROR, BCC LSB WAS SET
 4743  023624  000404                          BR      6$              ;CONTINUE WITH TEST
 4744  023626  004737  031356          5$:     JSR     PC,GETQI                ;GET HARDWARE RECEIVER BCC LSB
 4745  023632  103401                          BCS     6$              ;BR IF OK (SET)
 4746  023634  104023                          ERROR   23              ;ERROR, BCC LSB WAS CLEAR
 4747
 4748  023636                          6$:
 4749  023636  006037  023606                  ROR     3$              ;SHIFT SOFT DATA
 4750  023642  013737  031230  023610          MOV     CALBCC,4$       ;LOAD OLD SOFT BCC
 4751  023650  005203                          INC     R3              ;INCREMENT BIT COUNTER
```

```
 4752  023652  022703  000010                    CMP     #10,R3          ;DONE A FULL CHARACTER YET?
 4753  023656  001346                            BNE     2$              ;BR IF NO
 4754  023660  005003                            CLR     R3              ;RESTART BIT COUNTER
 4755  023662  005204                            INC     R4              ;INCREMENT DATA FOR SILO
 4756  023664  022704  000400                    CMP     #400,R4         ;DONE BINARY COUNT YET?
 4757  023670  003404                            BLE     9$              ;BR IF YES
 4758  023672  010461  000004                    MOV     R4,4(R1)        ;PORT4 DATA
 4759  023676  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4760  023700  122110                            122110                  ;LOAD OUT DATA
 4761  023702  005205                  9$:       INC     R5              ;INCREMENT DATA
 4762  023704  022705  000400                    CMP     #400,R5         ;DONE BINARY PATTERN YET?
 4763  023710  001327                            BNE     1$              ;BR IF NO
 4764  023712                           7$:
 4765
 4766
 4767                                            ;***************************** TEST 50 ****************************
 4768                                            ;*TRANSMITTER DDCMP CRC TEST
 4769                                            ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
 4770                                            ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
 4771                                            ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
 4772                                            ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
 4773                                            ;;*************************************************************************
 4774
 4775                                            ;  TEST 50
 4776                                            ;-------------
 4777                                    ;;*****************************************************************
 4778  023712  000004                   TST50:  SCOPE
 4779  023714  012737  000050  001202            MOV     #50,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4780  023722  012737  024244  001442            MOV     #TST51,NEXT             ; POINT TO THE START OF NEXT TEST.
 4781                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
 4782  023730  104410                            MSTCLR                  ;MASTER CLEAR KMC11
 4783
 4784                                            ;LOAD OUT DATA SILO
 4785
 4786  023732  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4787  023736  012704  032012                    MOV     #MESDAT,R4      ;LOAD POINTER TO DATA
 4788  023742  005037  024036                    CLR     10$             ;CLEAR SOFT BCC
 4789  023746  012700  000004                    MOV     #4,R0           ;LOAD CHARACTER COUNT
 4790  023752  004737  031374                    JSR     PC,SYNLD        ;LOAD 2 SYNCS IN OUT SILO
 4791  023756  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
 4792  023762  004537  031530                    JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
 4793  023766  032012                            MESDAT                  ;ADDRESS OF MESSAGE
 4794  023770  000004                            4                       ;NUMBER OF CHARACTERS
 4795  023772  004737  031504                    JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
 4796  023776  004737  030260                    JSR     PC,OCOR         ;WAIT FOR OCOR
 4797  024002  005003                            CLR     R3              ;CLEAR BIT COUNTER
 4798  024004  104413  000022                    DATACLK,22              ;CLOCK DATA
 4799  024010  112405                  12$:      MOVB    (R4)+,R5        ;LOAD R5 WITH CHAR
 4800  024012  010502                            MOV     R5,R2           ;LOAD R2 WITH CHAR
 4801
 4802                                            ;CHECK FIRST FOUR CHARACTER MESSAGE
 4803                                            ;IN THE BIT WINDOW (0,125,252,377)
 4804
 4805  024014  012737  120001  031226            MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL
 4806  024022  010537  024034                    MOV     R5,67$          ;LOAD SOFT CHAR FOR BCC
 4807  024026  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
```

```
 4808  024032  000010                          10                      ;SHIFT COUNT
 4809  024034  000000                  67$:    0                       ;CHARACTER
 4810  024036  000000                  10$:    0                       ;OLD BCC
 4811  024040  013737  031230  024036          MOV     CALBCC,10$      ;LOAD SOFT BCC FOR NEXT SHIFT
 4812  024046  104413  000001          64$:    DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
 4813  024052  106002                          RORB    R2              ;SHIFT SOFT DATA
 4814  024054  103005                          BCC     65$             ;BR IF A SPACE
 4815  024056  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4816  024062  103406                          BCS     66$             ;BR IF OK (MARK)
 4817  024064  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
 4818  024066  000404                          BR      66$             ;CONTINUE
 4819  024070  004737  030226          65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4820  024074  103001                          BCC     66$             ;BR IF OK (SPACE)
 4821  024076  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
 4822  024100                          66$:
 4823  024100  005203                          INC     R3              ;BUMP BIT COUNTER
 4824  024102  022703  000010                  CMP     #10,R3          ;DONE FULL 8 BITS YET
 4825  024106  001357                          BNE     64$             ;BR IF NO
 4826  024110  005003                          CLR     R3              ;CLEAR BIT COUNTER
 4827  024112  005300                          DEC     R0              ;DEC CHARACTER COUNT
 4828  024114  001335                          BNE     12$             ;BR IF NOT DONE YET
 4829
 4830                                          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
 4831
 4832  024116  013700  031230                  MOV     CALBCC,R0       ;PUT BCC IN R0
 4833  024122  104413  000001          68$:    DATACLK,1               ;SHIFT HARDWARE BCC
 4834  024126  006000                          ROR     R0              ;SHIFT SOFT BCC
 4835  024130  103005                          BCC     69$             ;BR IF CARRY CLEAR
 4836  024132  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4837  024136  103406                          BCS     70$             ;BR IF OK (MARK)
 4838  024140  104014                          ERROR   14              ;ERROR, CRC WRONG (SPACE)
 4839  024142  000404                          BR      70$             ;CONTINUE
 4840  024144  004737  030226          69$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4841  024150  103001                          BCC     70$             ;BR IF OK (SPACE)
 4842  024152  104014                          ERROR   14              ;ERROR, CRC WRONG (MARK)
 4843  024154                          70$:
 4844  024154  005203                          INC     R3              ;BUMP BIT COUNTER
 4845  024156  022703  000020                  CMP     #20,R3          ;FINISHED BCC YET?
 4846  024162  001357                          BNE     68$             ;BR IF NO
 4847  024164  005003                          CLR     R3              ;CLEAR BIT COUNTER
 4848
 4849                                          ;CHECK TO SEE IF TRANSMITTER IS MARKING
 4850
 4851  024166  104413  000001          2$:     DATACLK,        1       ;CLOCK TRANSMITTER
 4852  024172  004737  030226                  JSR     PC,GETSI        ;LOOK AT WINDOW
 4853  024176  103401                          BCS     3$              ;IT SHOULD BE MARKING
 4854  024200  104024                          ERROR   24              ;ERROR, BIT WAS A SPACE
 4855  024202  005203                  3$:     INC     R3              ;BUMP BIT COUNTER
 4856  024204  022703  000007                  CMP     #7,R3           ;DONE YET
 4857  024210  001366                          BNE     2$              ;BR IF NO
 4858  024212  104413  000010                  DATACLK,        10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
 4859  024216  005003                          CLR     R3              ;CLEAR BIT COUNTER
 4860  024220  104413  000001          4$:     DATACLK,        1       ;SHIFT OUT NEXT BIT
 4861  024224  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4862  024230  103401                          BCS     .+4             ;BR IF IT IS A MARK
 4863  024232  104024                          ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
```

```
 4864  024234  005203                          INC     R3              ;INC BIT COUNT
 4865  024236  022703  000020                  CMP     #20,R3          ;DONE YET?
 4866  024242  001366                          BNE     4$              ;BR IF NO
 4867  024244                          5$:
 4868
 4869
 4870                                          ;***************************** TEST 51 ****************************
 4871                                          ;*RECEIVER DDCMP CRC TEST
 4872                                          ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
 4873                                          ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
 4874                                          ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
 4875                                          ;;*************************************************************
 4876
 4877                                          ;  TEST 51
 4878                                          ;-------------
 4879                                  ;;*************************************************************
 4880  024244  000004                  TST51:  SCOPE
 4881  024246  012737  000051  001202          MOV     #51,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4882  024254  012737  024446  001442          MOV     #TST52,NEXT             ; POINT TO THE START OF NEXT TEST.
 4883                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4884  024262  104410                          MSTCLR                  ;MASTER CLEAR KMC11
 4885  024264  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
 4886  024270  012702  032012                  MOV     #MESDAT,R2      ;LOAD POINTER TO DATA
 4887  024274  012700  000004                  MOV     #4,R0           ;LOAD CHARACTER COUNT
 4888  024300  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 SYNCS IN OUT SILO
 4889  024304  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
 4890  024310  004537  031530                  JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
 4891  024314  032012                          MESDAT                  ;ADDRESS OF MESSAGE
 4892  024316  000004                          4                       ;NUMBER OF CHARACTERS
 4893  024320  004737  031504                  JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
 4894  024324  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
 4895  024330  104413  000114                  DATACLK,114             ;CLOCK DATA
 4896  024334  004737  031066          3$:     JSR     PC,INRDY        ;WAIT FOR INRDY
 4897  024340  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4898  024342  021204                          021204                  ;GET IN DATA
 4899  024344  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4900  024350  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
 4901  024352  120504                          CMPB    R5,R4           ;COMPARE RECEIVED DATA
 4902  024354  001401                          BEQ     1$              ;BR IF OK
 4903  024356  104010                          ERROR   10              ;DATA ERROR
 4904  024360  005300                  1$:     DEC     R0              ;DEC CHARACTER COUNT
 4905  024362  001364                          BNE     3$              ;BR IF NOT DONE YET
 4906
 4907                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
 4908
 4909  024364  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
 4910  024370  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4911  024372  021204                          021204                  ;GET FIRST HALF OF CRC
 4912  024374  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
 4913  024402  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
 4914  024410  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
 4915  024414  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4916  024416  021244                          021244
 4917  024420  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 4918  024424  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
 4919  024430  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
```

```
4920  024434  120504                  CMPB    R5,R4           ;IS IN BCC MATCH SET?
4921  024436  001401                  BEQ     25$
4922  024440  104015                  ERROR   15              ;IN BCC MATCH ERROR
4923  024442                  25$:
4924  024442  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4925  024444  021204                  021204                  ;GET LAST HALF
4926  024446                  2$:
4927
4928
4929                                  ;*************************** TEST 52 ***************************
4930                                  ;*DDCMP EOM FUNCTION TEST
4931                                  ;*THIS TEST LOADS OUT SILO WITH: 2 SYNCS,4 CHAR MESSAGE,EOM
4932                                  ;*4 CHARACTER MESS,EOM. THE DATA STREAM IS CHECKED TO BE
4933                                  ;*4 CHAR,BCC,4 CHAR,BCC,MARKS. THIS TEST VERIFYS THAT
4934                                  ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
4935                                  ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4936                                  ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4937                                  ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4938                                  ;;*************************************************************
4939
4940                                  ;  TEST 52
4941                                  ;-------------
4942                                  ;;*************************************************************
4943  024446  000004          TST52:  SCOPE
4944  024450  012737  000052  001202  MOV     #52,$TSTNM              ; LOAD THE NO. OF THIS TEST
4945  024456  012737  025546  001442  MOV     #TST53,NEXT             ; POINT TO THE START OF NEXT TEST.
4946                                                          ;R1 CONTAINS BASE KMC11 ADDRESS
4947  024464  104410                  MSTCLR                  ;MASTER CLEAR KMC11
4948
4949                                  ;LOAD OUT DATA SILO
4950
4951  024466  012711  004000          MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4952  024472  012704  032012          MOV     #MESDAT,R4      ;LOAD POINTER TO DATA
4953  024476  005037  024606          CLR     10$             ;CLEAR SOFT BCC
4954  024502  012700  000004          MOV     #4,R0           ;LOAD CHARACTER COUNT
4955  024506  004737  031374          JSR     PC,SYNLD        ;LOAD 2 SYNCS IN OUT SILO
4956  024512  004737  030412          JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
4957  024516  004537  031530          JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
4958  024522  032012                  MESDAT                  ;ADDRESS OF MESSAGE
4959  024524  000004                  4                       ;NUMBER OF CHARACTERS
4960  024526  004737  031504          JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
4961  024532  004537  031530          JSR     R5,MESLD        ;LOAD FOUR MORE CHARACTERS
4962  024536  032012                  MESDAT                  ;ADDRESS OF MESSAGE
4963  024540  000004                  4                       ;NUMBER OF CHACTERS
4964  024542  004737  031504          JSR     PC,EOM          ;SET EOM
4965  024546  004737  030260          JSR     PC,OCOR         ;WAIT FOR OCOR
4966  024552  005003                  CLR     R3              ;CLEAR BIT COUNTER
4967  024554  104413  000022          DATACLK,22              ;CLOCK DATA
4968  024560  112405          12$:    MOVB    (R4)+,R5        ;LOAD R5 WITH CHAR
4969  024562  010502                  MOV     R5,R2           ;LOAD R2 WITH CHAR
4970
4971                                  ;CHECK FIRST FOUR CHARACTER MESSAGE
4972                                  ;IN THE BIT WINDOW (0,125,252,377)
4973
4974  024564  012737  120001  031226  MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL
4975  024572  010537  024604          MOV     R5,67$          ;LOAD SOFT CHAR FOR BCC
```

```
 4976  024576  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
 4977  024602  000010                            10                      ;SHIFT COUNT
 4978  024604  000000                    67$:    0                       ;CHARACTER
 4979  024606  000000                    10$:    0                       ;OLD BCC
 4980  024610  013737  031230  024606            MOV     CALBCC,10$      ;LOAD SOFT BCC FOR NEXT SHIFT
 4981  024616  104413  000001            64$:    DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
 4982  024622  106002                            RORB    R2              ;SHIFT SOFT DATA
 4983  024624  103005                            BCC     65$             ;BR IF A SPACE
 4984  024626  004737  030226                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4985  024632  103406                            BCS     66$             ;BR IF OK (MARK)
 4986  024634  104006                            ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
 4987  024636  000404                            BR      66$             ;CONTINUE
 4988  024640  004737  030226            65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4989  024644  103001                            BCC     66$             ;BR IF OK (SPACE)
 4990  024646  104006                            ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
 4991  024650                            66$:
 4992  024650  005203                            INC     R3              ;BUMP BIT COUNTER
 4993  024652  022703  000010                    CMP     #10,R3          ;DONE FULL 8 BITS YET
 4994  024656  001357                            BNE     64$             ;BR IF NO
 4995  024660  005003                            CLR     R3              ;CLEAR BIT COUNTER
 4996  024662  005300                            DEC     R0              ;DEC CHARACTER COUNT
 4997  024664  001335                            BNE     12$             ;BR IF NOT DONE YET
 4998
 4999                                            ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
 5000
 5001  024666  013700  031230                    MOV     CALBCC,R0       ;PUT BCC IN R0
 5002  024672  104413  000001            68$:    DATACLK,1               ;SHIFT HARDWARE BCC
 5003  024676  006000                            ROR     R0              ;SHIFT SOFT BCC
 5004  024700  103005                            BCC     69$             ;BR IF CARRY CLEAR
 5005  024702  004737  030226                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5006  024706  103406                            BCS     70$             ;BR IF OK (MARK)
 5007  024710  104014                            ERROR   14              ;ERROR, CRC WRONG (SPACE)
 5008  024712  000404                            BR      70$             ;CONTINUE
 5009  024714  004737  030226            69$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5010  024720  103001                            BCC     70$             ;BR IF OK (SPACE)
 5011  024722  104014                            ERROR   14              ;ERROR, CRC WRONG (MARK)
 5012  024724                            70$:
 5013  024724  005203                            INC     R3              ;BUMP BIT COUNTER
 5014  024726  022703  000020                    CMP     #20,R3          ;FINISHED BCC YET?
 5015  024732  001357                            BNE     68$             ;BR IF NO
 5016  024734  005003                            CLR     R3              ;CLEAR BIT COUNTER
 5017  024736  012700  000004                    MOV     #4,R0           ;RESET CHARACTER COUNTER
 5018  024742  012704  032012                    MOV     #MESDAT,R4      ;LOAD MESSAGE POINTER
 5019  024746  005037  025000                    CLR     11$             ;CLR SOFT BCC
 5020  024752  112405                    13$:    MOVB    (R4)+,R5        ;LOAD CHAR IN R5
 5021  024754  010502                            MOV     R5,R2           ;LOAD CHAR IN R2
 5022
 5023                                            ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
 5024
 5025  024756  012737  120001  031226            MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL
 5026  024764  010537  024776                    MOV     R5,76$          ;LOAD SOFT CHAR FOR BCC
 5027  024770  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
 5028  024774  000010                            10                      ;SHIFT COUNT
 5029  024776  000000                    76$:    0                       ;CHARACTER
 5030  025000  000000                    11$:    0                       ;OLD BCC
 5031  025002  013737  031230  025000            MOV     CALBCC,11$      ;LOAD SOFT BCC FOR NEXT SHIFT
```

```
5032  025010  104413  000001          73$:    DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
5033  025014  106002                          RORB    R2              ;SHIFT SOFT DATA
5034  025016  103005                          BCC     74$             ;BR IF A SPACE
5035  025020  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5036  025024  103406                          BCS     75$             ;BR IF OK (MARK)
5037  025026  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
5038  025030  000404                          BR      75$             ;CONTINUE
5039  025032  004737  030226          74$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5040  025036  103001                          BCC     75$             ;BR IF OK (SPACE)
5041  025040  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
5042  025042                          75$:
5043  025042  005203                          INC     R3              ;BUMP BIT COUNTER
5044  025044  022703  000010                  CMP     #10,R3          ;DONE FULL 8 BITS YET
5045  025050  001357                          BNE     73$             ;BR IF NO
5046  025052  005003                          CLR     R3              ;CLEAR BIT COUNTER
5047  025054  005300                          DEC     R0              ;DEC CHARACTER COUNT
5048  025056  001335                          BNE     13$             ;BR IF NOT DONE YET
5049
5050                                          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5051
5052  025060  013700  031230                  MOV     CALBCC,R0       ;PUT BCC IN R0
5053  025064  104413  000001          77$:    DATACLK,1               ;SHIFT HARDWARE BCC
5054  025070  006000                          ROR     R0              ;SHIFT SOFT BCC
5055  025072  103005                          BCC     78$             ;BR IF CARRY CLEAR
5056  025074  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5057  025100  103406                          BCS     79$             ;BR IF OK (MARK)
5058  025102  104014                          ERROR   14              ;ERROR, CRC WRONG (SPACE)
5059  025104  000404                          BR      79$             ;CONTINUE
5060  025106  004737  030226          78$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5061  025112  103001                          BCC     79$             ;BR IF OK (SPACE)
5062  025114  104014                          ERROR   14              ;ERROR, CRC WRONG (MARK)
5063  025116                          79$:
5064  025116  005203                          INC     R3              ;BUMP BIT COUNTER
5065  025120  022703  000020                  CMP     #20,R3          ;FINISHED BCC YET?
5066  025124  001357                          BNE     77$             ;BR IF NO
5067  025126  005003                          CLR     R3              ;CLEAR BIT COUNTER
5068
5069                                          ;CHECK TO SEE IF TRANSMITTER IS MARKING
5070
5071  025130  104413  000001          2$:     DATACLK,        1       ;CLOCK TRANSMITTER
5072  025134  004737  030226                  JSR     PC,GETSI        ;LOOK AT WINDOW
5073  025140  103401                          BCS     3$              ;IT SHOULD BE MARKING
5074  025142  104024                          ERROR   24              ;ERROR, BIT WAS A SPACE
5075  025144  005203                  3$:     INC     R3              ;BUMP BIT COUNTER
5076  025146  022703  000007                  CMP     #7,R3           ;DONE YET
5077  025152  001366                          BNE     2$              ;BR IF NO
5078  025154  104413  000010                  DATACLK,        10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
5079  025160  005003                          CLR     R3              ;CLEAR BIT COUNTER
5080  025162  104413  000001          4$:     DATACLK,        1       ;SHIFT OUT NEXT BIT
5081  025166  004737  030226                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5082  025172  103401                          BCS     .+4             ;BR IF IT IS A MARK
5083  025174  104024                          ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
5084  025176  005203                          INC     R3              ;INC BIT COUNT
5085  025200  022703  000020                  CMP     #20,R3          ;DONE YET?
5086  025204  001366                          BNE     4$              ;BR IF NO
5087
```

```
5088                                                ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
5089                                                ;WAS RECEIVED CORRECTLY (0,125,252,377)
5090
5091  025206  104413  000001                        DATACLK,        1       ;GET LAST BIT IN RECEIVER
5092  025212  012703  000004                        MOV     #4,R3           ;R3=CHARACTER COUNT
5093  025216  012702  032012                        MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5094  025222  004737  031066              40$:      JSR     PC,INRDY        ;WAIT FOR INRDY
5095  025226  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5096  025230  021204                              021204
5097  025232  016104  000004                        MOV     4(R1),R4        ;PUT "FOUND" IN R4
5098  025236  112205                              MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5099  025240  120504                              CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5100  025242  001401                              BEQ     41$             ;BR IF YES
5101  025244  104010                              ERROR   10              ;RECEIVE DATA ERROR
5102  025246  005303              41$:      DEC     R3              ;DEC CHARACTER COUNT
5103  025250  001364                              BNE     40$             ;BR IF NOT DONE YET
5104
5105                                                ;CHECK TO SEE THAT IN BCC MATCH IS SET
5106                                                ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5107
5108  025252  004737  031066                        JSR     PC,INRDY        ;WAIT FOR INRDY
5109  025256  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5110  025260  021204                              021204                  ;GET FIRST HALF OF CRC
5111  025262  116137  000004  001302                MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5112  025270  042737  177400  001302                BIC     #177400,$TMP2   ;CLEAR HI BYTE
5113  025276  004737  031066                        JSR     PC,INRDY        ;WAIT FOR INRDY
5114  025302  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5115  025304  021244                              021244
5116  025306  016104  000004                        MOV     4(R1),R4        ;PUT "FOUND" IN R4
5117  025312  042704  000376                        BIC     #376,R4         ;CLEAR UNWANTED BITS
5118  025316  012705  000001                        MOV     #1,R5           ;PUT "EXPECTED" IN R5
5119  025322  120504                              CMPB    R5,R4           ;IS IN BCC MATCH SET?
5120  025324  001401                              BEQ     50$
5121  025326  104015                              ERROR   15              ;IN BCC MATCH ERROR
5122  025330                              50$:
5123  025330  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5124  025332  021204                              021204                  ;GET LAST HALF
5125  025334  116137  000004  001301                MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5126  025342  042737  000377  001300                BIC     #377,$TMP1      ;CLEAR LO BYTE
5127  025350  053737  001300  001302                BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5128  025356  023737  031230  001302                CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5129  025364  001401                              BEQ     42$             ;BR IF OK
5130  025366  104027                              ERROR   27
5131
5132                                                ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
5133                                                ;WAS RECEIVED CORRECTLY (0,125,252,377)
5134
5135  025370  012703  000004              42$:      MOV     #4,R3           ;R3=CHARACTER COUNT
5136  025374  012702  032012                        MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5137  025400  004737  031066              43$:      JSR     PC,INRDY        ;WAIT FOR INRDY
5138  025404  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5139  025406  021204                              021204
5140  025410  016104  000004                        MOV     4(R1),R4        ;PUT "FOUND" IN R4
5141  025414  112205                              MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5142  025416  120504                              CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5143  025420  001401                              BEQ     44$             ;BR IF YES
```

```
025422  104010                          ERROR   10              ;RECEIVE DATA ERROR
025424  005303                  44$:    DEC     R3              ;DEC CHARACTER COUNT
025426  001364                          BNE     43$             ;BR IF NOT DONE YET

                                        ;CHECK TO SEE THAT IN BCC MATCH IS SET
                                        ;AND THAT THE BCC WAS RECEIVED CORRECTLY

025430  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
025434  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025436  021204                          021204                  ;GET FIRST HALF OF CRC
025440  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
025446  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
025454  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
025460  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025462  021244                          021244
025464  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
025470  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
025474  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
025500  120504                          CMPB    R5,R4           ;IS IN BCC MATCH SET?
025502  001401                          BEQ     51$
025504  104015                          ERROR   15              ;IN BCC MATCH ERROR
025506                          51$:
025506  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025510  021204                          021204                  ;GET LAST HALF
025512  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
025520  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
025526  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
025534  023737  031230  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
025542  001401                          BEQ     5$              ;BR IF OK
025544  104027                          ERROR   27
025546                          5$:


                                        ;************************ TEST 53 **************************
                                        ;*DDCMP EOM FUNCTION TEST
                                        ;*THIS TEST LOADS OUT SILO WITH: 2 SYNCS,4 CHAR MESSAGE,EOM
                                        ;*SOM,4 CHAR MESS,EOM. THE DATA STREAM IS CHECKED TO BE
                                        ;*4 CHAR,BCC,4 CHAR,BCC,MARKS. THIS TEST VERIFYS THAT
                                        ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
                                        ;*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
                                        ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
                                        ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
                                        ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
                                        ;***********************************************************

                                        ;       TEST 53
                                        ;-------------
                                ;**********************************************************
025546  000004                  TST53:  SCOPE
025550  012737  000053  001202          MOV     #53,$TSTNM              ; LOAD THE NO. OF THIS TEST
025556  012737  026746  001442          MOV     #TST54,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
025564  104410                          MSTCLR                  ;MASTER CLEAR KMC11

                                        ;LOAD OUT DATA SILO

```

```
5200  025566  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5201  025572  012704  032012                    MOV     #MESDAT,R4      ;LOAD POINTER TO DATA
5202  025576  005037  025712                    CLR     10$             ;CLEAR SOFT BCC
5203  025602  012700  000004                    MOV     #4,R0           ;LOAD CHARACTER COUNT
5204  025606  004737  031374                    JSR     PC,SYNLD        ;LOAD 2 SYNCS IN OUT SILO
5205  025612  004737  030412                    JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5206  025616  004537  031530                    JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
5207  025622  032012                            MESDAT                  ;ADDRESS OF MESSAGE
5208  025624  000004                            4                       ;NUMBER OF CHARACTERS
5209  025626  004737  031504                    JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
5210  025632  004737  031454                    JSR     PC,SOM          ;LOAD GARBAGE CHAR WITH SOM SET
5211  025636  004537  031530                    JSR     R5,MESLD        ;LOAD FOUR MORE CHARACTERS
5212  025642  032012                            MESDAT                  ;ADDRESS OF MESSAGE
5213  025644  000004                            4                       ;NUMBER OF CHACTERS
5214  025646  004737  031504                    JSR     PC,EOM          ;SET EOM
5215  025652  004737  030260                    JSR     PC,OCOR         ;WAIT FOR OCOR
5216  025656  005003                            CLR     R3              ;CLEAR BIT COUNTER
5217  025660  104413  000022                    DATACLK,22              ;CLOCK DATA
5218  025664  112405                  12$:      MOVB    (R4)+,R5        ;LOAD R5 WITH CHAR
5219  025666  010502                            MOV     R5,R2           ;LOAD R2 WITH CHAR
5220
5221                                            ;CHECK FIRST FOUR CHARACTER MESSAGE
5222                                            ;IN THE BIT WINDOW (0,125,252,377)
5223
5224  025670  012737  120001  031226            MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL
5225  025676  010537  025710                    MOV     R5,67$          ;LOAD SOFT CHAR FOR BCC
5226  025702  004537  031122                    JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
5227  025706  000010                            10                      ;SHIFT COUNT
5228  025710  000000                  67$:      0                       ;CHARACTER
5229  025712  000000                  10$:      0                       ;OLD BCC
5230  025714  013737  031230  025712            MOV     CALBCC,10$      ;LOAD SOFT BCC FOR NEXT SHIFT
5231  025722  104413  000001          64$:      DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
5232  025726  106002                            RORB    R2              ;SHIFT SOFT DATA
5233  025730  103005                            BCC     65$             ;BR IF A SPACE
5234  025732  004737  030226                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5235  025736  103406                            BCS     66$             ;BR IF OK (MARK)
5236  025740  104006                            ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
5237  025742  000404                            BR      66$             ;CONTINUE
5238  025744  004737  030226          65$:      JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5239  025750  103001                            BCC     66$             ;BR IF OK (SPACE)
5240  025752  104006                            ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
5241  025754                           66$:
5242  025754  005203                            INC     R3              ;BUMP BIT COUNTER
5243  025756  022703  000010                    CMP     #10,R3          ;DONE FULL 8 BITS YET
5244  025762  001357                            BNE     64$             ;BR IF NO
5245  025764  005003                            CLR     R3              ;CLEAR BIT COUNTER
5246  025766  005300                            DEC     R0              ;DEC CHARACTER COUNT
5247  025770  001335                            BNE     12$             ;BR IF NOT DONE YET
5248
5249                                            ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5250
5251  025772  013700  031230                    MOV     CALBCC,R0       ;PUT BCC IN R0
5252  025776  104413  000001          68$:      DATACLK,1               ;SHIFT HARDWARE BCC
5253  026002  006000                            ROR     R0              ;SHIFT SOFT BCC
5254  026004  103005                            BCC     69$             ;BR IF CARRY CLEAR
5255  026006  004737  030226                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
```

```
5256  026012  103406                        BCS     70$             ;BR IF OK (MARK)
5257  026014  104014                        ERROR   14              ;ERROR, CRC WRONG (SPACE)
5258  026016  000404                        BR      70$             ;CONTINUE
5259  026020  004737  030226        69$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5260  026024  103001                        BCC     70$             ;BR IF OK (SPACE)
5261  026026  104014                        ERROR   14              ;ERROR, CRC WRONG (MARK)
5262  026030                        70$:
5263  026030  005203                        INC     R3              ;BUMP BIT COUNTER
5264  026032  022703  000020                CMP     #20,R3          ;FINISHED BCC YET?
5265  026036  001357                        BNE     68$             ;BR IF NO
5266  026040  005003                        CLR     R3              ;CLEAR BIT COUNTER
5267
5268                                        ;CHECK CHARACTER LOADED WITH SOM (000), IN THE BIT WINDOW
5269
5270  026042  005005                        CLR     R5              ;CHARACTER LOADED WITH SOM
5271  026044  010502                        MOV     R5,R2           ;LOAD R2 WITH CHAR
5272  026046  104413  000001        32$:    DATACLK,        1       ;CLOCK TRANSMITTER
5273  026052  106002                        RORB    R2              ;SHIFT SOFT DATA
5274  026054  103005                        BCC     30$             ;BR IF SPACE
5275  026056  004737  030226                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5276  026062  103406                        BCS     31$             ;BR IF OK (MARK)
5277  026064  104006                        ERROR   6               ;ERROR,BIT WINDOW WAS A SPACE
5278  026066  000404                        BR      31$             ;CONTINUE
5279  026070  004737  030226        30$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5280  026074  103001                        BCC     31$             ;BR IF OK (SPACE)
5281  026076  104006                        ERROR   6               ;ERROR,BIT WINDOW WAS A MARK
5282  026100  005203                31$:    INC     R3              ;BUMP BIT COUNTER
5283  026102  022703  000010                CMP     #10,R3          ;DONE CHARACTER YET?
5284  026106  001357                        BNE     32$             ;BR IF NO
5285  026110  005003                        CLR     R3              ;RESET BIT COUNTER
5286  026112  012700  000004                MOV     #4,R0           ;RESET CHARACTER COUNTER
5287  026116  012704  032012                MOV     #MESDAT,R4      ;LOAD MESSAGE POINTER
5288  026122  005037  026154                CLR     11$             ;CLR SOFT BCC
5289  026126  112405                13$:    MOVB    (R4)+,R5        ;LOAD CHAR IN R5
5290  026130  010502                        MOV     R5,R2           ;LOAD CHAR IN R2
5291
5292                                        ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
5293
5294  026132  012737  120001  031226        MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL
5295  026140  010537  026152                MOV     R5,76$          ;LOAD SOFT CHAR FOR BCC
5296  026144  004537  031122                JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
5297  026150  000010                        10                      ;SHIFT COUNT
5298  026152  000000                76$:    0                       ;CHARACTER
5299  026154  000000                11$:    0                       ;OLD BCC
5300  026156  013737  031230  026154        MOV     CALBCC,11$      ;LOAD SOFT BCC FOR NEXT SHIFT
5301  026164  104413  000001        73$:    DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
5302  026170  106002                        RORB    R2              ;SHIFT SOFT DATA
5303  026172  103005                        BCC     74$             ;BR IF A SPACE
5304  026174  004737  030226                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5305  026200  103406                        BCS     75$             ;BR IF OK (MARK)
5306  026202  104006                        ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
5307  026204  000404                        BR      75$             ;CONTINUE
5308  026206  004737  030226        74$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5309  026212  103001                        BCC     75$             ;BR IF OK (SPACE)
5310  026214  104006                        ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
5311  026216                        75$:
```

```
 5312  026216  005203                        INC     R3              ;BUMP BIT COUNTER
 5313  026220  022703  000010                CMP     #10,R3          ;DONE FULL 8 BITS YET
 5314  026224  001357                        BNE     73$             ;BR IF NO
 5315  026226  005003                        CLR     R3              ;CLEAR BIT COUNTER
 5316  026230  005300                        DEC     R0              ;DEC CHARACTER COUNT
 5317  026232  001335                        BNE     13$             ;BR IF NOT DONE YET
 5318
 5319                                        ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
 5320
 5321  026234  013700  031230                MOV     CALBCC,R0       ;PUT BCC IN R0
 5322  026240  104413  000001        77$:    DATACLK,1               ;SHIFT HARDWARE BCC
 5323  026244  006000                        ROR     R0              ;SHIFT SOFT BCC
 5324  026246  103005                        BCC     78$             ;BR IF CARRY CLEAR
 5325  026250  004737  030226                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5326  026254  103406                        BCS     79$             ;BR IF OK (MARK)
 5327  026256  104014                        ERROR   14              ;ERROR, CRC WRONG (SPACE)
 5328  026260  000404                        BR      79$             ;CONTINUE
 5329  026262  004737  030226        78$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5330  026266  103001                        BCC     79$             ;BR IF OK (SPACE)
 5331  026270  104014                        ERROR   14              ;ERROR, CRC WRONG (MARK)
 5332  026272                        79$.
 5333  026272  005203                        INC     R3              ;BUMP BIT COUNTER
 5334  026274  022703  000020                CMP     #20,R3          ;FINISHED BCC YET?
 5335  026300  001357                        BNE     77$             ;BR IF NO
 5336  026302  005003                        CLR     R3              ;CLEAR BIT COUNTER
 5337
 5338                                        ;CHECK TO SEE IF TRANSMITTER IS MARKING
 5339
 5340  026304  104413  000001        2$:     DATACLK,        1       ;CLOCK TRANSMITTER
 5341  026310  004737  030226                JSR     PC,GETSI        ;LOOK AT WINDOW
 5342  026314  103401                        BCS     3$              ;IT SHOULD BE MARKING
 5343  026316  104024                        ERROR   24              ;ERROR, BIT WAS A SPACE
 5344  026320  005203                3$:     INC     R3              ;BUMP BIT COUNTER
 5345  026322  022703  000007                CMP     #7,R3           ;DONE YET
 5346  026326  001366                        BNE     2$              ;BR IF NO
 5347  026330  104413  000010                DATACLK,        10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
 5348  026334  005003                        CLR     R3              ;CLEAR BIT COUNTER
 5349  026336  104413  000001        4$:     DATACLK,        1       ;SHIFT OUT NEXT BIT
 5350  026342  004737  030226                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5351  026346  103401                        BCS     .+4             ;BR IF IT IS A MARK
 5352  026350  104024                        ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
 5353  026352  005203                        INC     R3              ;INC BIT COUNT
 5354  026354  022703  000020                CMP     #20,R3          ;DONE YET?
 5355  026360  001366                        BNE     4$              ;BR IF NO
 5356
 5357                                        ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
 5358                                        ;WAS RECEIVED CORRECTLY (0,125,252,377)
 5359
 5360  026362  104413  000001                DATACLK,        1       ;GET LAST BIT IN RECEIVER
 5361  026366  012703  000004                MOV     #4,R3           ;R3=CHARACTER COUNT
 5362  026372  012702  032012                MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
 5363  026376  004737  031066        40$:    JSR     PC,INRDY        ;WAIT FOR INRDY
 5364  026402  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5365  026404  021204                        021204
 5366  026406  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
 5367  026412  112205                        MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
```

```
5368  026414  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5369  026416  001401                          BEQ     41$             ;BR IF YES
5370  026420  104010                          ERROR   10              ;RECEIVE DATA ERROR
5371  026422  005303                  41$:    DEC     R3              ;DEC CHARACTER COUNT
5372  026424  001364                          BNE     40$             ;BR IF NOT DONE YET
5373
5374                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5375                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5376
5377  026426  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
5378  026432  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5379  026434  021204                          021204                  ;GET FIRST HALF OF CRC
5380  026436  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5381  026444  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
5382  026452  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
5383  026456  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5384  026460  021244                          021244
5385  026462  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5386  026466  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
5387  026472  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
5388  026476  120504                          CMPB    R5,R4           ;IS IN BCC MATCH SET?
5389  026500  001401                          BEQ     50$
5390  026502  104015                          ERROR   15              ;IN BCC MATCH ERROR
5391  026504                          50$:
5392  026504  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5393  026506  021204                          021204                  ;GET LAST HALF
5394  026510  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5395  026516  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
5396  026524  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5397  026532  023737  031230  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5398  026540  001401                          BEQ     45$             ;BR IF OK
5399  026542  104027                          ERROR   27
5400
5401                                          ;CHECK THAT CHARACTER LOADED WITH SOM WAS RECEIVED (000)
5402
5403  026544  004737  031066          45$:    JSR     PC,INRDY        ;WAIT FOR INRDY
5404  026550  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5405  026552  021204                          021204                  ;GET RECEIVE DATA
5406  026554  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5407  026560  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
5408  026562  120504                          CMPB    R5,R4           ;IS RECEVIED DATA CORRECT?
5409  026564  001401                          BEQ     42$             ;BR IF YES
5410  026566  104010                          ERROR   10              ;RECEIVE DATA ERROR
5411
5412                                          ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
5413                                          ;WAS RECEIVED CORRECTLY (0,125,252,377)
5414
5415  026570  012703  000004          42$:    MOV     #4,R3           ;R3=CHARACTER COUNT
5416  026574  012702  032012                  MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5417  026600  004737  031066          43$:    JSR     PC,INRDY        ;WAIT FOR INRDY
5418  026604  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5419  026606  021204                          021204
5420  026610  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5421  026614  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5422  026616  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5423  026620  001401                          BEQ     44$             ;BR IF YES
```

```
5424  026622  104010                    ERROR   10              ;RECEIVE DATA ERROR
5425  026624  005303            44$:    DEC     R3              ;DEC CHARACTER COUNT
5426  026626  001364                    BNE     43$             ;BR IF NOT DONE YET
5427
5428                                    ;CHECK TO SEE THAT IN BCC MATCH IS SET
5429                                    ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5430
5431  026630  004737  031066            JSR     PC,INRDY        ;WAIT FOR INRDY
5432  026634  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5433  026636  021204                    021204                  ;GET FIRST HALF OF CRC
5434  026640  116137  000004  001302    MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5435  026646  042737  177400  001302    BIC     #177400,$TMP2   ;CLEAR HI BYTE
5436  026654  004737  031066            JSR     PC,INRDY        ;WAIT FOR INRDY
5437  026660  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5438  026662  021244                    021244
5439  026664  016104  000004            MOV     4(R1),R4        ;PUT "FOUND" IN R4
5440  026670  042704  000376            BIC     #376,R4         ;CLEAR UNWANTED BITS
5441  026674  012705  000001            MOV     #1,R5           ;PUT "EXPECTED" IN R5
5442  026700  120504                    CMPB    R5,R4           ;IS IN BCC MATCH SET?
5443  026702  001401                    BEQ     51$
5444  026704  104015                    ERROR   15              ;IN BCC MATCH ERROR
5445  026706                    51$:
5446  026706  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5447  026710  021204                    021204                  ;GET LAST HALF
5448  026712  116137  000004  001301    MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5449  026720  042737  000377  001300    BIC     #377,$TMP1      ;CLEAR LO BYTE
5450  026726  053737  001300  001302    BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5451  026734  023737  031230  001302    CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5452  026742  001401                    BEQ     5$              ;BR IF OK
5453  026744  104027                    ERROR   27
5454  026746                    5$:
5455
5456
5457                                    ;***************************** TEST 54 ****************************
5458                                    ;*EMPTY SILO TEST
5459                                    ;*LOAD SILO WITH 2 SYNCS, 4 CHAR MESSAGE, SINGLE CLOCK
5460                                    ;*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
5461                                    ;*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
5462                                    ;*4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
5463                                    ;;**********************************************************************
5464
5465                                    ;  TEST 54
5466                                    ;-------------
5467                              ;;**********************************************************************
5468  026746  000004            TST54:  SCOPE
5469  026750  012737  000054  001202    MOV     #54,$TSTNM              ; LOAD THE NO. OF THIS TEST
5470  026756  012737  027200  001442    MOV     #TST55,NEXT             ; POINT TO THE START OF NEXT TEST.
5471                                                            ;R1 CONTAINS BASE KMC11 ADDRESS
5472  026764  104410                    MSTCLR                  ;MASTER CLEAR KMC11
5473  026766  012711  004000            MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5474  026772  012702  032012            MOV     #MESDAT,R2      ;R2 POINTS TO MESSAGE
5475  026776  012700  000004            MOV     #4,R0           ;R0 = CHAR COUNT
5476  027002  004737  031374            JSR     PC,SYNLD        ;LOAD SILO WITH TWO SYNCS
5477  027006  004737  030412            JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5478  027012  004537  031530            JSR     R5,MESLD        ;LOAD MESSAGE IN SILO
5479  027016  032012                    MESDAT                  ;START OF MESSAGE
```

```
5480  027020  000004                          4                       ;CHARACTER COUNT
5481  027022  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5482  027026  104413  000065                  DATACLK,        65      ;CLOCK DATA (EMPTY SILO)
5483  027032  004537  031530                  JSR     R5,MESLD        ;PUT MORE CHARACTERS IN SILO
5484  027036  032012                          MESDAT
5485  027040  000004                          4
5486  027042  004737  030260                  JSR     PC,OCOR
5487  027046  104413  000005                  DATACLK,        5       ;CLOCK UNTIL RTS IS CLEARED
5488  027052  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5489  027054  021264                          021264                  ;GET RTS
5490  027056  032761  000040  000004          BIT     #BIT5,4(R1)     ;IS IT CLEAR?
5491  027064  001401                          BEQ     5$              ;BR IF YES
5492  027066  104034                          ERROR   34              ;ERROR, RTS NOT CLEAR
5493  027070  104413  000041          5$:     DATACLK,        41      ;CLOCK XMITTER SOME MORE
5494  027074  004737  031066          1$:     JSR     PC,INRDY        ;OK LETS CHECK WHAT WAS RECEIVED
5495  027100  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5496  027102  021204                          021204                  ;GET RECEIVE DATA
5497  027104  016104  000004                  MOV     4(R1),R4        ;PUT IT IN R4
5498  027110  112205                          MOVB    (R2)+,R5        ;R5 = "EXPECTED"
5499  027112  120504                          CMPB    R5,R4           ;IS DATA CORRECT?
5500  027114  001401                          BEQ     2$              ;BR IF OK
5501  027116  104010                          ERROR   10              ;DATA ERROR
5502  027120  005300          2$:             DEC     R0              ;DEC CHAR COUNT
5503  027122  001364                          BNE     1$              ;BR IF NOT DONE YET
5504  027124  004737  031066          3$:     JSR     PC,INRDY        ;WAIT FOR INRDY
5505  027130  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5506  027132  021204                          021204                  ;GET RECEIVE DATA
5507  027134  016104  000004                  MOV     4(R1),R4        ;PUT IT IN "FOUND"
5508  027140  012705  000377                  MOV     #377,R5         ;R5 = "EXPECTED"
5509  027144  120504                          CMPB    R5,R4           ;SHOULD SEE 377
5510  027146  001401                          BEQ     4$              ;BR IF OK
5511  027150  104010                          ERROR   10              ;ERROR, TRANSMITTER DID NOT ABORT
5512  027152  004737  031066          4$:     JSR     PC,INRDY        ;WAIT FOR INRDY
5513  027156  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5514  027160  021204                          021204                  ;GET RECEIVE DATA
5515  027162  016104  000004                  MOV     4(R1),R4        ;PUT IT IN "FOUND"
5516  027166  012705  000377                  MOV     #377,R5         ;R5 = "EXPECTED"
5517  027172  120504                          CMPB    R5,R4           ;SHOULD SEE 377
5518  027174  001401                          BEQ     10$             ;BR IF OK
5519  027176  104010                          ERROR   10              ;ERROR, TRANSMITTER DID NOT ABORT
5520  027200                          10$:
5521
5522
5523                                          ;***************************** TEST 55 ****************************
5524                                          ;*HALF DUPLEX TEST
5525                                          ;*SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCS AND A
5526                                          ;*MESSAGE. VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR
5527                                          ;;*************************************************************
5528
5529                                          ;  TEST 55
5530                                          ;-------------
5531                                  ;;*************************************************************
5532  027200  000004                  TST55:  SCOPE
5533  027202  012737  000055  001202          MOV     #55,$TSTNM              ; LOAD THE NO. OF THIS TEST
5534  027210  012737  027316  001442          MOV     #TST56,NEXT             ; POINT TO THE START OF NEXT TEST.
5535                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
```

```
5536  027216  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5537  027220  012702  000012                  MOV     #12,R2          ;SAVE R2 FOR TYPEOUT
5538  027224  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5539  027230  012761  000020  000004          MOV     #BIT4,4(R1)     ;LOAD PORT4
5540  027236  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5541  027240  122113                          122113                  ;SET H/D BIT
5542  027242  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 SYNCS
5543  027246  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5544  027252  004537  031530                  JSR     R5,MESLD        ;LOAD 4 CHAR MESSAGE
5545  027256  032012                          MESDAT                  ;ADDRESS OF MESSAGE
5546  027260  000004                          4                       ;CHARACTER COUNT
5547  027262  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5548  027266  104413  000073                  DATACLK,        73      ;SEND MESSAGE
5549  027272  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5550  027274  021244                          021244                  ;READ LU-12
5551  027276  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5552  027302  042704  000257                  BIC     #257,R4         ;CLEAR UNWANTED BITS
5553  027306  005005                          CLR     R5              ;R5 = "EXPECTED"
5554  027310  120504                          CMPB    R5,R4           ;IN-ACTIVE AND IN-RDY SHOULD BE CLEAR
5555  027312  001401                          BEQ     1$              ;BR IF OK
5556  027314  104035                          ERROR   35              ;ERROR BOTH ARE NOT CLEAR
5557  027316                          1$:
5558
5559
5560                                          ;****************************** TEST 56 ****************************
5561                                          ;*DDCMP CABLE DATA TEST
5562                                          ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
5563                                          ;*4 SYNCS,16 CHAR,EOM,16 CHAR,EOM,16 CHAR,EOM
5564                                          ;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
5565                                          ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
5566                                          ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
5567                                          ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
5568                                          ;:**************************************************************
5569
5570                                          ;  TEST 56
5571                                          ;-------------
5572                                  ;;**************************************************************
5573  027316  000004                  TST56:  SCOPE
5574  027320  012737  000056  001202          MOV     #56,$TSTNM              ; LOAD THE NO. OF THIS TEST
5575  027326  012737  027706  001442          MOV     #TST57,NEXT             ; POINT TO THE START OF NEXT TEST.
5576                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5577  027334  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5578  027336  032737  040000  002050          BIT     #BIT14,STAT1    ;SKIP TEST IF NO
5579  027344  001557                          BEQ     3$              ;LOOPBACK CONNECTOR ON
5580  027346  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5581  027352  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 SYNCS
5582  027356  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 MORE SYNCS
5583  027362  012737  120001  031226          MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFT CRC CALC
5584  027370  005037  027420                  CLR     6$              ;CLEAR OLD BCC
5585  027374  012703  000020                  MOV     #16.,R3         ;CHARACTER COUNT
5586  027400  012702  032016                  MOV     #FLTDAT,R2      ;R2= POINTER
5587  027404  112237  027416          7$:     MOVB    (R2)+,5$        ;LOAD CHAR FOR SOFT BCC CALC.
5588  027410  004537  031122                  JSR     R5,SIMBCC       ;CALC SOFT BCC
5589  027414  000010                          10                      ;SHIFT COUNT
5590  027416  000000                  5$:     0                       ;CHARACTER
5591  027420  000000                  6$:     0                       ;OLD BCC
```

```
5592  027422  013737  031230  027420          MOV     CALBCC,6$       ;LOAD OLD BCC
5593  027430  005303                          DEC     R3              ;DEC COUNT
5594  027432  001364                          BNE     7$              ;BR IF NOT DONE YET
5595  027434  004537  031530                  JSR     R5,MESLD        ;LOAD SILO
5596  027440  032016                          FLTDAT                  ;MESSAGE ADDRESS
5597  027442  000020                          16.                     ;CHARACTER COUNT
5598  027444  004737  031504                  JSR     PC,EOM          ;LOAD AN EOM
5599  027450  004537  031530                  JSR     R5,MESLD        ;LOAD SILO
5600  027454  032016                          FLTDAT                  ;MESSAGE ADDRESS
5601  027456  000020                          16.                     ;CHARACTER COUNT
5602  027460  004737  031504                  JSR     PC,EOM          ;LOAD AN EOM
5603  027464  004537  031530                  JSR     R5,MESLD        ;LOAD SILO
5604  027470  032016                          FLTDAT                  ;MESSAGE ADDRESS
5605  027472  000020                          16.                     ;CHARACTER COUNT
5606  027474  004737  031504                  JSR     PC,EOM          ;LOAD AN EOM
5607  027500  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5608  027504  005011                          CLR     (R1)            ;CLEAR LINE UNIT LOOP
5609  027506  012700  000003                  MOV     #3,R0           ;R0 = MESSAGE COUNT
5610  027512  012703  000020                  MOV     #16.,R3         ;R3= CHARACTER COUNT
5611  027516  012702  032016                  MOV     #FLTDAT,R2      ;LOAD MESSAGE POINTER IN R2
5612  027522  004737  031066          1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
5613  027526  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5614  027530  021204                          021204                  ;GET DATA FROM IN SILO
5615  027532  016104  000004                  MOV     4(R1),R4        ;PUT CHARACTER IN "FOUND"
5616  027536  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5617  027540  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT
5618  027542  001401                          BEQ     2$              ;BR IF OK
5619  027544  104025                          ERROR   25              ;DATA ERROR
5620  027546                          2$:
5621  027546  005303                          DEC     R3              ;DEC CHARACTER COUNT
5622  027550  001364                          BNE     1$              ;BR IF NOT DONE THIS MESSAGE
5623  027552  012703  000020                  MOV     #16.,R3         ;RESET CHARACTER COUNT
5624
5625                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5626                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5627
5628  027556  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
5629  027562  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5630  027564  021204                          021204                  ;GET FIRST HALF OF CRC
5631  027566  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5632  027574  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
5633  027602  004737  031066                  JSR     PC,INRDY        ;WAIT FOR INRDY
5634  027606  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5635  027610  021244                          021244
5636  027612  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5637  027616  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
5638  027622  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
5639  027626  120504                          CMPB    R5,R4           ;IS IN BCC MATCH SET?
5640  027630  001401                          BEQ     25$
5641  027632  104015                          ERROR   15              ;IN BCC MATCH ERROR
5642  027634                          25$:
5643  027634  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5644  027636  021204                          021204                  ;GET LAST HALF
5645  027640  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5646  027646  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
5647  027654  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
```

```
5648  027662  023737  031230  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5649  027670  001401                          BEQ     4$              ;BR IF OK
5650  027672  104027                          ERROR   27
5651  027674  012702  032016          4$:     MOV     #FLTDAT,R2      ;RESET MESSAGE POINTER
5652  027700  005300                          DEC     R0              ;DECREMENT COUNTER
5653  027702  001307                          BNE     1$              ;BR IF NOT DONE
5654  027704  104420                  3$:     ADVANCE         ; ADVANCE TO NEXT TEST
5655
5656
5657                                          ;***************************** TEST 57 ****************************
5658                                          ;*DDCMP CABLE DATA TEST
5659                                          ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
5660                                          ;*4 SYNCS,59 DATA CHARACTERS,EOM WITH GARBAGE CHARACTER
5661                                          ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
5662                                          ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
5663                                          ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
5664                                          ;;*************************************************************
5665
5666                                          ;  TEST 57
5667                                          ;-------------
5668                                  ;;*************************************************************
5669  027706  000004                  TST57:  SCOPE
5670  027710  012737  000057  001202          MOV     #57,$TSTNM              ; LOAD THE NO. OF THIS TEST
5671  027716  012737  003662  001442          MOV     #$EOP,NEXT              ; POINT TO THE END OF PASS HANDLER.
5672                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5673  027724  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5674  027726  032737  040000  002050          BIT     #BIT14,STAT1    ;SKIP TEST IF NO
5675  027734  001533                          BEQ     3$              ;LOOPBACK CONNECTOR ON
5676  027736  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5677  027742  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 SYNCS
5678  027746  004737  031374                  JSR     PC,SYNLD        ;LOAD 2 MORE SYNCS
5679  027752  012737  120001  031226          MOV     #CRC16,XPOLY    ;LOAD POLYNOMIAL FOR SOFT CRC CALC
5680  027760  005037  030010                  CLR     6$              ;CLEAR OLD BCC
5681  027764  012703  000073                  MOV     #59.,R3         ;CHARACTER COUNT
5682  027770  012702  032012                  MOV     #MESDAT,R2      ;R2= POINTER
5683  027774  112237  030006          7$:     MOVB    (R2)+,5$        ;LOAD CHAR FOR SOFT BCC CALC.
5684  030000  004537  031122                  JSR     R5,SIMBCC       ;CALC SOFT BCC
5685  030004  000010                          10                      ;SHIFT COUNT
5686  030006  000000                  5$:     0                       ;CHARACTER
5687  030010  000000                  6$:     0                       ;OLD BCC
5688  030012  013737  031230  030010          MOV     CALBCC,6$       ;LOAD OLD BCC
5689  030020  005303                          DEC     R3              ;DEC COUNT
5690  030022  001364                          BNE     7$              ;BR IF NOT DONE YET
5691  030024  004537  031530                  JSR     R5,MESLD        ;LOAD SILO
5692  030030  032012                          MESDAT                  ;MESSAGE ADDRESS
5693  030032  000073                          59.                     ;CHARACTER COUNT
5694  030034  004737  031504                  JSR     PC,EOM          ;LOAD AN EOM
5695  030040  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5696  030044  005011                          CLR     (R1)            ;CLEAR LINE UNIT LOOP
5697  030046  012700  000073                  MOV     #59.,R0         ;R0= CHARACTER COUNT
5698  030052  012702  032012                  MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5699  030056  004737  031066          1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
5700  030062  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5701  030064  021204                          021204                  ;GET DATA FROM IN SILO
5702  030066  016104  000004                  MOV     4(R1),R4        ;PUT CHARACTER IN "FOUND"
5703  030072  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
```

```
5704  030074  120504                        CMPB    R5,R4           ;IS RECEIVED DATA CORRECT
5705  030076  001401                        BEQ     2$              ;BR IF OK
5706  030100  104025                        ERROR   25              ;DATA ERROR
5707  030102                          2$:
5708  030102  005300                        DEC     R0              ;DECREMENT COUNTER
5709  030104  001364                        BNE     1$              ;BR IF NOT DONE
5710
5711                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5712                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5713
5714  030106  004737  031066                 JSR     PC,INRDY        ;WAIT FOR INRDY
5715  030112  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5716  030114  021204                        021204                  ;GET FIRST HALF OF CRC
5717  030116  116137  000004  001302        MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5718  030124  042737  177400  001302        BIC     #177400,$TMP2   ;CLEAR HI BYTE
5719  030132  004737  031066                 JSR     PC,INRDY        ;WAIT FOR INRDY
5720  030136  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5721  030140  021244                        021244
5722  030142  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
5723  030146  042704  000376                BIC     #376,R4         ;CLEAR UNWANTED BITS
5724  030152  012705  000001                MOV     #1,R5           ;PUT "EXPECTED" IN R5
5725  030156  120504                        CMPB    R5,R4           ;IS IN BCC MATCH SET?
5726  030160  001401                        BEQ     25$
5727  030162  104015                        ERROR   15              ;IN BCC MATCH ERROR
5728  030164                          25$:
5729  030164  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5730  030166  021204                        021204                  ;GET LAST HALF
5731  030170  116137  000004  001301        MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5732  030176  042737  000377  001300        BIC     #377,$TMP1      ;CLEAR LO BYTE
5733  030204  053737  001300  001302        BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5734  030212  023737  031230  001302        CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5735  030220  001401                        BEQ     3$              ;BR IF OK
5736  030222  104027                        ERROR   27
5737  030224  104420                  3$:   ADVANCE         ; ADVANCE TO NEXT TEST
5738
5739
5740                                    ;SUBROUTINES
5741                                    ;-----------
5742
5743  030226                            GETSI:
5744                                          ;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
5745                                          ;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
5746
5747  030226  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5748  030230  021364                        021364                  ;PORT4_LU 17
5749  030232  017737  151636  030256        MOV     @KMP04,NITCH    ;STORE LU 17
5750  030240  106137  030256                ROLB    NITCH
5751  030244  106137  030256                ROLB    NITCH
5752  030250  106137  030256                ROLB    NITCH           ;PUT SI IN THE CARRY BIT
5753  030254  000207                        RTS     PC
5754  030256  000000                  NITCH:  0
5755
5756
5757  030260                            OCOR:
5758                                          ;THIS SUBROUTINE SPINS ON OCOR
5759
```

```
5760  030260  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5761  030262  021364                          021364                  ;PORT4 LU 17
5762  030264  032777  000020  151602          BIT     #BIT4,@KMP04    ;IS OCOR SET?
5763  030272  001772                          BEQ     OCOR            ;BR IF NO
5764  030274  000207                          RTS     PC              ;OK OCOR IS SET, GO BACK
5765
5766
5767  030276                          SYNC:
5768                                          ;THIS SUBROUNTINE LOADS THE SILO WITH THE NUMBER OF SYNC
5769                                          ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
5770                                          ;AND A NON-SYNC CHARACTER (301)
5771
5772  030276  013637  001276                 MOV     @(SP)+,$TMP0    ;GET COUNT
5773  030302  062746  000002                 ADD     #2,-(SP)        ;ADJUST STACK
5774  030306  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5775  030314  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5776  030316  122114                          122114                  ;LOAD SYNC REGISTER
5777  030320  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5778  030324  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
5779  030332  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5780  030334  122111                          122111                  ;SET SOM
5781  030336  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5782  030344  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5783  030346  122110                          122110                  ;LOAD OUT DATA
5784  030350  005337  001276                 DEC     $TMP0           ;ALL DONE?
5785  030354  001361                          BNE     1$              ;BR IF NOT
5786  030356  004737  030412                 JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5787  030362  005061  000004                 CLR     4(R1)           ;LOAD PORT4
5788  030366  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5789  030370  122111                          122111                  ;SET SOM
5790  030372  012761  000301  000004          MOV     #301,4(R1)      ;LOAD PORT4
5791  030400  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5792  030402  122110                          122110                  ;LOAD OUT DATA
5793  030404  004737  030260                 JSR     PC,OCOR         ;WAIT FOR OCOR
5794  030410  000207                          RTS     PC
5795
5796
5797  030412                          OUTRDY:
5798                                          ;THIS SUBROUTINE SPINS ON OUT READY
5799
5800  030412  005037  001306                  CLR     $TMP4           ;CLEAR TIMER
5801  030416                          1$:
5802  030416  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5803  030420  021224                          021224                  ;PORT4 LU11
5804  030422  032777  000020  151444          BIT     #BIT4,@KMP04    ;IS OUT RDY SET?
5805  030430  001004                          BNE     2$              ;BR IF YES
5806  030432  005237  001306                 INC     $TMP4           ;INC TIMER
5807  030436  001367                          BNE     1$              ;KEEP CHECKING IF NOT DONE
5808  030440  104036                          ERROR   36              ;ERROR, OUT READY NOT SET
5809  030442  000207                  2$:     RTS     PC
5810
5811
5812  030444                          CHAR:
5813                                          ;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNCS
5814                                          ;AND THE CHARACTER PASSED TO IT.
5815
```

```
5816  030444  013637  001300                  MOV     @(SP)+,$TMP1    ;GET CHARACTER
5817  030450  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
5818  030454  012737  000003  001276          MOV     #3,$TMP0        ;SET FOR 3 SYNCS
5819  030462  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5820  030470  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5821  030472  122114                          122114                  ;LOAD SYNC REGISTER
5822  030474  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5823  030500  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
5824  030506  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5825  030510  122111                          122111                  ;SET SOM
5826  030512  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5827  030520  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION  ROMCLK PC=5304
5828  030522  122110                          122110                  ;LOAD OUT DATA
5829  030524  005337  001276                  DEC     $TMP0           ;ALL DONE?
5830  030530  001361                          BNE     1$              ;BR IF NOT
5831  030532  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5832  030536  013761  001300  000004          MOV     $TMP1,4(R1)     ;LOAD PORT4
5833  030544  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5834  030546  122110                          122110                  ;LOAD OUT DATA
5835  030550  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5836  030554  000207                          RTS     PC
5837
5838
5839  030556                          CHARSD:
5840                                          ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.
5841
5842  030556  013637  001300                  MOV     @(SP)+,$TMP1    ;GET CHARACTER
5843  030562  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
5844  030566  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5845  030572  013761  001300  000004          MOV     $TMP1,4(R1)     ;LOAD PORT4
5846  030600  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5847  030602  122110                          122110                  ;LOAD OUT DATA
5848  030604  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5849  030610  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5850  030612  122110                          122110                  ;LOAD GARBAGE CHAR
5851  030614  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5852  030620  000207                          RTS     PC
5853
5854
5855  030622                          S!LOLD:
5856                                          ;THIS SUBROUTINE FILLS THE OUT SILO
5857                                          ; WITH A BINARY COUNT PATTERN
5858
5859  030622  012737  000073  001300          MOV     #73,$TMP1       ;LOAD COUNT
5860  030630  005737  031062                  TST     SCHAR           ;FIRST TIME HERE?
5861  030634  100470                          BMI     4$              ;BR IF BITSTUFF
5862  030636  001032                          BNE     2$              ;BR IF NO
5863  030640  062737  000002  001300          ADD     #2,$TMP1        ;ADD 2 TO CHARACTER COUNT
5864  030646  012737  000003  001276          MOV     #3,$TMP0        ;SET FOR 3 SYNCS
5865  030654  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5866  030662  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5867  030664  122114                          122114                  ;LOAD SYNC REGISTER
5868  030666  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5869  030672  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
5870  030700  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5871  030702  122111                          122111                  ;SET SOM
```

```
5872  030704  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
5873  030712  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5874  030714  122110                          122110                  ;LOAD OUT DATA
5875  030716  005337  001276                  DEC     $TMP0           ;ALL DONE?
5876  030722  001361                          BNE     1$              ;BR IF NOT
5877  030724  004737  030412          2$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5878  030730  013761  031062  000004          MOV     SCHAR,4(R1)     ;LOAD PORT4
5879  030736  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5880  030740  122110                          122110                  ;LOAD OUT DATA
5881  030742  005737  031064                  TST     STUFLG          ;BITSTUFF???
5882  030746  001407                          BEQ     6$              ;BR IF NO
5883  030750  013737  031062  030762          MOV     SCHAR,5$        ;IT IS SDLD SO CHECK BITSTUFFING
5884  030756  004537  031612                  JSR     R5,STFFCL       ;ADD ANY BIT STUFF CLOCK TICKS
5885  030762  000000                  5$:     0                       ;CHARACTER
5886  030764  000010                          10                      ;CHIFT COUNT
5887  030766  005237  031062          6$:     INC     SCHAR           ;NEXT CHARACTER
5888  030772  022737  000400  031062          CMP     #400,SCHAR      ;ALL DONE?
5889  031000  001403                          BEQ     3$
5890  031002  005337  001300                  DEC     $TMP1           ;DECREMENT COUNT
5891  031006  001346                          BNE     2$              ;BR IF NOT DONE
5892  031010  004737  030260          3$:     JSR     PC,OCOR         ;WAIT FOR OCOR
5893  031014  000207                          RTS     PC
5894  031016  005037  031062          4$:     CLR     SCHAR           ;START PATTERN AT ZERO
5895  031022  012737  177777  031064          MOV     #-1,STUFLG      ;SET BITSTUFF FLAG
5896  031030  005037  032010                  CLR     BITCON          ;CLEAR STUFF COUNT
5897  031034  062737  000002  001300          ADD     #2,$TMP1        ;ADD 2 TO CHARACTER COUNT
5898  031042  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
5899  031050  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5900  031052  122111                          122111                  ;SET SOM!
5901  031054  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5902  031056  122110                          122110                  ;LOAD GARBAGE CHAR
5903  031060  000721                          BR      2$              ;GO LOAD SILO
5904  031062  000000                  SCHAR:  0
5905  031064  000000                  STUFLG: 0
5906
5907
5908  031066                          INRDY:
5909                                          ;THIS SUBROUTINE SPINS ON INRDY
5910                                          ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5911                                          ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5912                                          ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5913                                          ;INITIALLY LOADED INTO $TMP0, THE SMALLER THE NUMBER
5914                                          ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5915
5916  031066  012737  000000  001276          MOV     #0,$TMP0        ;SET UP DELAY COUNTER
5917  031074                          1$:
5918  031074  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5919  031076  021244                          021244                  ;PORT4 LU12
5920  031100  032777  000020  150766          BIT     #BIT4,@KMP04    ;IS INRDY SET?
5921  031106  001004                          BNE     2$              ;BR IF YES
5922  031110  005237  001276                  INC     $TMP0           ;INC DELAY
5923  031114  001367                          BNE     1$              ;TRY AGAIN
5924  031116  104037                          ERROR   37              ;ERROR,NO INRDY
5925  031120  000207                  2$:     RTS     PC              ;RETURN
5926
5927
```

```
 5928  031122                          SIMBCC:
 5929                                          ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
 5930                                          ;IN XPOLY. THE CORRECT CRC IS $LPADRED IN CALBCC, AND THE
 5931                                          ;STATE OF THE LSB OF THE BCC IS $LPADRED IN THE C BIT.
 5932
 5933  031122  010046                          MOV     R0,-(SP)        ;SAVE R0 ON STACK
 5934  031124  012537  001276                  MOV     (R5)+,$TMP0     ;$TMP0 = SHIFT COUNT
 5935  031130  012537  001300                  MOV     (R5)+,$TMP1     ;$TMP1 = CHARACTER
 5936  031134  012537  031230                  MOV     (R5)+,CALBCC    ;CALBCC = OLD BCC
 5937  031140  013700  031230          1$:     MOV     CALBCC,R0       ;PUT OLD BCC IN R0
 5938  031144  000241                          CLC
 5939  031146  006037  031230                  ROR     CALBCC          ;SHIFT OLD BCC
 5940  031152  006037  001300                  ROR     $TMP1           ;SHIFT CHARACTER
 5941  031156  005500                          ADC     R0              ;ADD CHAR CARRY TO OLD BCC
 5942  031160  006000                          ROR     R0              ;PUT BIT0 TO CARRY BIT
 5943  031162  103011                          BCC     2$              ;CARRY IS FEEDBACK BIT
 5944  031164  013700  031226                  MOV     XPOLY,R0        ;IF FEEDBACK = 1
 5945  031170  043700  031230                  BIC     CALBCC,R0       ;EXCLUSIVLY OR XPOLY TO CALBCC
 5946  031174  043737  031226  031230          BIC     XPOLY,CALBCC
 5947  031202  050037  031230                  BIS     R0,CALBCC
 5948  031206  005337  001276          2$:     DEC     $TMP0           ;DEC SHIFT COUNT
 5949  031212  001352                          BNE     1$              ;BR IF NOT DONE
 5950  031214  013700  031230                  MOV     CALBCC,R0       ;PUT RESULT IN R0
 5951  031220  006000                          ROR     R0              ;SHIFT BIT0 TO CARRY
 5952  031222  012600                          MOV     (SP)+,R0        ;RESTORE R0
 5953  031224  000205                          RTS     R5              ;$LPADR
 5954  031226  000000                  XPOLY:  0
 5955  031230  000000                  CALBCC: 0
 5956          000200                  LRC8=200
 5957          120001                  CRC16=120001
 5958          102010                  CRC.CCITT=102010
 5959
 5960
 5961  031232                          BCCLD:
 5962                                          ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCS
 5963                                          ;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
 5964                                          ;WITH THE SOM BIT CLEAR (ENABLE CRC)
 5965
 5966  031232  013637  001300                  MOV     @(SP)+,$TMP1    ;GET CHARACTER
 5967  031236  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
 5968  031242  012737  000002  001276          MOV     #2,$TMP0        ;SET FOR 2 SYNCS
 5969  031250  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
 5970  031256  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5971  031260  122114                          122114                  ;LOAD SYNC REGISTER
 5972  031262  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
 5973  031266  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
 5974  031274  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5975  031276  122111                          122111                  ;SET SOM
 5976  031300  012761  000026  000004          MOV     #26,4(R1)       ;LOAD PORT4
 5977  031306  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5978  031310  122110                          122110                  ;LOAD OUT DATA
 5979  031312  005337  001276                  DEC     $TMP0           ;ALL DONE?
 5980  031316  001361                          BNE     1$              ;BR IF NOT
 5981  031320  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
 5982  031324  013761  001300  000004          MOV     $TMP1,4(R1)     ;LOAD PORT4
 5983  031332  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
5984  031334  122110                          122110                  ;LOAD OUT DATA
5985  031336  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
5986  031342  000207                          RTS     PC
5987
5988
5989  031344                          GETQO:
5990                                          ;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5991                                          ;BCC LSB AND PUTS IT IN THE CARRY BIT
5992
5993  031344  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5994  031346  021364                          021364                  ;PORT4_LU-17
5995  031350  106177  150520                  ROLB    @KMP04          ;PUT QO IN CARRY
5996  031354  000207                          RTS     PC              ;RETURN
5997
5998
5999  031356                          GETQI:
6000                                          ;THIS SUBROUTINE READS THE STATE OF THE RECEIVE
6001                                          ;BCC LSB AND PUTS IT IN THE CARRY BIT
6002
6003  031356  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6004  031360  021364                          021364                  ;PORT4_LU-17
6005  031362  106177  150506                  ROLB    @KMP04          ;PUT QO IN CARRY
6006  031366  106177  150502                  ROLB    @KMP04          ;PUT QI IN CARRY
6007  031372  000207                          RTS     PC              ;RETURN
6008
6009
6010  031374                          SYNLD:
6011                                          ;THIS SUBROUTINE LOADS OUT SILO WITH
6012                                          ;2 SYNC CHARACTERS WITH SOM SET
6013
6014  031374  012737  000002  001276          MOV     #2,$TMP0        ;LOAD COUNTER FOR 2 SYNCS
6015  031402  012761  000026  000004          MOV     #26,4(R1)       ;PORT4_26
6016  031410  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6017  031412  122114                          122114                  ;LOAD SYNC REG
6018  031414  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
6019  031420  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
6020  031426  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6021  031430  122111                          122111                  ;SET SOM
6022  031432  012761  000026  000004          MOV     #26,4(R1)       ;PORT_26
6023  031440  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6024  031442  122110                          122110                  ;LOAD OUT DATA WITH SYNC
6025  031444  005337  001276                  DEC     $TMP0           ;DECREMENT COUNTER
6026  031450  001361                          BNE     1$              ;BR IF NOT DONE
6027  031452  000207                          RTS     PC              ;RETURN
6028
6029
6030  031454                          SOM:
6031                                          ;THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
6032                                          ;GARBAGE CHARACTER (0)
6033
6034  031454  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
6035  031460  012761  000001  000004          MOV     #1,4(R1)        ;PORT4_1
6036  031466  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6037  031470  122111                          122111                  ;SET SOM
6038  031472  005061  000004                  CLR     4(R1)           ;CLEAR DATA CHAR
6039  031476  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
 6040  031500  122110                          122110                  ;LOAD GARBAGE CHARACTER
 6041  031502  000207                          RTS     PC              ;RETURN
 6042
 6043
 6044  031504                          EOM:
 6045                                          ;THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
 6046                                          ;GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
 6047
 6048  031504  004737  030412                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
 6049  031510  012761  000002  000004          MOV     #2,4(R1)        ;PORT4 2
 6050  031516  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6051  031520  122111                          122111                  ;SET EOM
 6052  031522  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6053  031524  122110                          122110                  ;LOAD GARBAGE CHARACTER
 6054  031526  000207                          RTS     PC              ;RETURN
 6055
 6056
 6057  031530                          MESLD:
 6058                                          ;THIS SUBROUTINE LOADS SILO WITH MESSAGE
 6059                                          ;THE FIRST ARUGUMENT IS THE ADDRESS OF THE MESSAGE
 6060                                          ;THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
 6061
 6062  031530  010046                          MOV     R0,-(SP)        ;SAVE R0
 6063  031532  012500                          MOV     (R5)+,R0        ;R0=MESSAGE POINTER
 6064  031534  012537  001276                  MOV     (R5)+,$TMP0     ;$TMP0=CHARACTER COUNT
 6065  031540  004737  030412          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUT RDY
 6066  031544  112061  000004                  MOVB    (R0)+,4(R1)     ;LOAD PORT4 WITH CHARACTER
 6067  031550  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6068  031552  122110                          122110                  ;LOAD OUT DATA SILO
 6069  031554  005337  001276                  DEC     $TMP0           ;DEC CHAR COUNT
 6070  031560  001367                          BNE     1$              ;BR IF NOT DONE
 6071  031562  004737  030260                  JSR     PC,OCOR         ;WAIT FOR OCOR
 6072  031566  012600                          MOV     (SP)+,R0        ;RESTORE R0
 6073  031570  000205                          RTS     R5              ;RETURN
 6074
 6075
 6076  031572                          CLRIO:
 6077                                          ;THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
 6078                                          ;CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
 6079
 6080  031572  012761  000200  000004          MOV     #BIT7,4(R1)     ;LOAD PORT4
 6081  031600  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6082  031602  122112                          122112                  ;SET IN CLR!
 6083  031604  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6084  031606  122111                          122111                  ;SET OUT CLR!
 6085  031610  000207                          RTS     PC              ;RETURN
 6086
 6087
 6088  031612                          STFFCL:
 6089                                          ;THIS SUBROUTINE ADDS ANY NECESSSARY BIT STUFF CLOCK TICKS
 6090                                          ;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
 6091
 6092  031612  010046                          MOV     R0,-(SP)        ;SAVE R0
 6093  031614  012500                          MOV     (R5)+,R0        ;PUT CHAR IN R0
 6094  031616  012537  001302                  MOV     (R5)+,$TMP2     ;PUT SHIFT COUNT IN $TMP2
 6095  031622  106000                  1$:     RORB    R0              ;LOOK AT NEXT BIT
```

```
 6096  031624  103403                          BCS     2$              ;BR IF A MARK
 6097  031626  005037  032010                  CLR     BITCON          ;IT WAS A SPACE, CLEAR 1'S COUNTER
 6098  031632  000412                          BR      3$              ;CONTINUE
 6099  031634  005237  032010          2$:     INC     BITCON          ;INC CONSECUTIVE 1'S COUNTER
 6100  031640  022737  000005  032010          CMP     #5,BITCON       ;IS IT 5 YET?
 6101  031646  001004                          BNE     3$              ;BR IF NO
 6102  031650  005037  032010                  CLR     BITCON          ;YES!  SO START AGAIN
 6103  031654  104413  000001                  DATACLK,        1       ;GIVE EXTRA TICK TO STUFF ZERO
 6104  031660  005337  001302          3$:     DEC     $TMP2           ;DEC SHIFT COUNT
 6105  031664  001356                          BNE     1$              ;BR IF NOT DONE
 6106  031666  012600                          MOV     (SP)+,R0        ;RESTORE R0
 6107  031670  000205                          RTS     R5              ;RETURN
 6108
 6109
 6110  031672                          STFFCK:
 6111                                          ;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
 6112                                          ;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
 6113                                          ;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
 6114
 6115  031672  010046                          MOV     R0,-(SP)        ;SAVE R0
 6116  031674  012500                          MOV     (R5)+,R0        ;PUT CHAR IN R0
 6117  031676  012537  001302                  MOV     (R5)+,$TMP2     ;PUT SHIFT COUNT IN $TMP2
 6118  031702  106000                  1$:     RORB    R0              ;SHIFT OUT NEXT BIT
 6119  031704  103403                          BCS     2$              ;BR IF IT IS A MARK
 6120  031706  005037  032010                  CLR     BITCON          ;IT WAS A SPACE, CLEAR 1'S COUNTER
 6121  031712  000416                          BR      3$              ;CONTINUE
 6122  031714  005237  032010          2$:     INC     BITCON          ;INC CONSECUTIVE I'S COUNTER
 6123  031720  022737  000005  032010          CMP     #5,BITCON       ;5 IN A ROW YET?
 6124  031726  001010                          BNE     3$              ;BR IF NO
 6125  031730  005037  032010                  CLR     BITCON          ;YES, SO START OVER
 6126  031734  104413  000001                  DATACLK,        1       ;EXTRA TICK TO STUFF ZERO
 6127  031740  004737  030226                  JSR     PC,GETSI        ;LOOK AT WINDOW
 6128  031744  103001                          BCC     3$              ;IS IT A ZERO, BR IF YES
 6129  031746  104030                          ERROR   30              ;NO, ERROR ZERO WAS NOT STUFFED
 6130  031750  005337  001302          3$:     DEC     $TMP2           ;DEC SHIFT COUNT
 6131  031754  001352                          BNE     1$              ;BR IF NOT DONE
 6132  031756  012600                          MOV     (SP)+,R0        ;RESTORE R0
 6133  031760  000205                          RTS     R5              ;RETURN
 6134
 6135
 6136  031762                          CTSDLY:
 6137                                          ;THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
 6138                                          ;BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
 6139
 6140  031762  010046                          MOV     R0,-(SP)        ;SAVE R0
 6141  031764  012700  000032                  MOV     #32,R0          ;LOAD R0 WITH COUNT
 6142  031770  027777  147250  147246  1$:     CMP     @$TKS,@$TKS     ;WASTE TIME
 6143  031776  005300                          DEC     R0              ;DECREMENT COUNTER
 6144  032000  001373                          BNE     1$              ;DO IT AGAIN IF NOT = 0
 6145  032002  012600                          MOV     (SP)+,R0        ;RESTORE R0
 6146  032004  000207                          RTS     PC              ;RETURN
 6147
 6148
 6149  032006  000176                  FLAG    ^B<01111110>            ;FLAG CHARACTER
 6150  032010  000000                  BITCON: 0
 6151  032012     000      125      252 MESDAT: .BYTE   0,125,252,377
```

```
 6152  032015      377
 6153  032016      001       002       004  FLTDAT: .BYTE   1,2,4,10,20,40,100,200,376,375,373,367,357,337,277,177
 6154  032021      010       020       040
 6155  032024      100       200       376
 6156  032027      375       373       367
 6157  032032      357       337       277
 6158  032035      177
 6159  032036      100       140       160  STUFDT: .BYTE   100,140,160,170,3,300,174,176,177,1
 6160  032041      170       003       300
 6161  032044      174       176       177
 6162  032047      001
 6163  032050      363       347       317          .BYTE   363,347,317,200,0,377,377,377,200,37
 6164  032053      200       000       377
 6165  032056      377       377       200
 6166  032061      037
 6167                                       .EVEN
 6168  032062  046200  047111  020105  EM1:    .ASCIZ  <200>/LINE UNIT INITALIZATION TEST/
       032120  046200  047111  020105  EM2:    .ASCIZ  <200>^LINE UNIT REGISTER READ/ONLY TEST^
       032163     200  044514  042516  EM3:    .ASCIZ  <200>^LINE UNIT REGISTER WRITE/READ TEST^
       032227     200  044514  042516  EM4:    .ASCIZ  <200>/LINE UNIT INTERNAL CLOCK FAILURE/
       032271     200  051124  047101  EM5:    .ASCIZ  <200>/TRANSMITTER DATA ERROR/
       032321     200  042522  042503  EM6:    .ASCIZ  <200>/RECEIVER TEST/
       032340  051200  041505  044505  EM7:    .ASCIZ  <200>/RECEIVER DATA ERROR/
       032365     200  047515  042504  EM10:   .ASCII  <200>/MODEM SIGNAL ERROR/
       032410  052200  044510  020123          .ASCII  <200>/THIS ERROR COULD BE CAUSED IF YOU HAVE V.35 AND/
       032470  040600  052125  051517          .ASCII  <200>/AUTOSIZED. YOU MUST MANUALLY ANSWER QUESTIONS IF/
       032551     200  047531  020125          .ASCIZ  <200>/YOU HAVE V.35/
       032570  052200  040522  051516  EM11:   .ASCIZ  <200>/TRANSMITTER CRC ERROR/
       032617     200  042522  042503  EM12:   .ASCIZ  <200>/RECEIVER CRC ERROR/
       032643     200  047111  041040  EM13:   .ASCIZ  <200>/IN BCC MATCH ERROR (LU REG 12)/
       032703     200  051124  047101  EM14:   .ASCIZ  <200>/TRANSMITTER FAILED TO GO TO MARK STATE/
       032753     200  040503  046102  EM15:   .ASCIZ  <200>/CABLE DATA TEST/
       032774  043200  040514  020107  EM16:   .ASCIZ  <200>/FLAG ERROR/
       033010  052200  040522  051516  EM17:   .ASCIZ  <200>/TRANSMITTER FAILED TO STUFF A ZERO/
       033054  051600  044527  041524  EM20:   .ASCIZ  <200>/SWITCH PAC TEST/
       033075     200  041101  051117  EM21:   .ASCIZ  <200>/ABORT ERROR/
       033112  052200  040522  051516  EM22:   .ASCIZ  <200>/TRANSMITTER ERROR/
       033135     200  040510  043114  EM23:   .ASCIZ  <200>/HALF DUPLEX TEST/
       033157     200  052517  020124  EM24:   .ASCIZ  <200>/OUT READY NOT SET/
       033202  044600  020116  042522  EM25:   .ASCIZ  <200>/IN READY NOT SET/

       033224  042600  050130  041505  DH1:    .ASCIZ  <200>/EXPECTED  FOUND/
       033245     200  054105  042520  DH2:    .ASCIZ  <200>/EXPECTED  FOUND  LU-REGISTER/
       033303     200  044103  051101  DH3:    .ASCIZ  <200>/CHARACTER    BIT THAT FAILED/
       033341     200  047503  051122  DH4:    .ASCIZ  <200>/CORRECT CRC    BIT THAT FAILED/
       033401     200  054105  042520  DH5:    .ASCIZ  <200>/EXPECTED   FOUND   SHIFT/
       033433     200  054105  042520  DH6:    .ASCIZ  <200>/EXPECTED   FOUND   CHARACTER   SHIFT/
       033501     200  046102  041517  DH7:    .ASCIZ  <200>/BLOCK END NOT SET/
       033524  051200  051524  042040  DH10:   .ASCIZ  <200>/RTS DID NOT CLEAR/
               033550                   .EVEN

       033550  000002                   DT1:    2
       033552     003       007                .BYTE   3,7
       033554  001274                         $REG5
       033556     003       002                .BYTE   3,2
       033560  001272                         $REG4
```

```
033562  000003                  DT2:    3
033564     003      007                 .BYTE   3,7
033566  001274                         $REG5
033570     003      010                 .BYTE   3,10
033572  001272                         $REG4
033574     003      002                 .BYTE   3,2
033576  001266                         $REG2
033600  000002                  DT3:    2
033602     003      017                 .BYTE   3,17
033604  001274                         $REG5
033606     002      002                 .BYTE   2,2
033610  001270                         $REG3
033612  000002                  DT4:    2
033614     006      021                 .BYTE   6,21
033616  031230                         CALBCC
033620     002      002                 .BYTE   2,2
033622  001270                         $REG3
033624  000003                  DT5:    3
033626     001      011                 .BYTE   1,11
033630  001462                         ZERO
033632     001      011                 .BYTE   1,11
033634  001464                         ONE
033636     002      002                 .BYTE   2,2
033640  001262                         $REG0
033642  000003                  DT6:    3
033644     001      011                 .BYTE   1,11
033646  001464                         ONE
033650     001      011                 .BYTE   1,11
033652  001462                         ZERO
033654     002      002                 .BYTE   2,2
033656  001262                         $REG0
033660  000004                  DT7:    4
033662     001      011                 .BYTE   1,11
033664  001462                         ZERO
033666     001      011                 .BYTE   1,11
033670  001464                         ONE
033672     003      007                 .BYTE   3,7
033674  001274                         $REG5
033676     002      001                 .BYTE   2,1
033700  001270                         $REG3
033702  000004                  DT10:   4
033704     001      011                 .BYTE   1,11
033706  001464                         ONE
033710     001      011                 .BYTE   1,11
033712  001462                         ZERO
033714     003      007                 .BYTE   3,7
033716  001274                         $REG5
033720     002      001                 .BYTE   2,1
033722  001270                         $REG3
033724  000002                  DT11:   2
033726     003      007                 .BYTE   3,7
033730  032006                         FLAG
033732     002      002                 .BYTE   2,2
033734  001270                         $REG3
033736  000002                  DT12:   2
033740     006      004                 .BYTE   6,4
```

```
033742  031230                          CALBCC
033744     006      002                 .BYTE   6,2
033746  001302                          $TMP2


033750                                  CORMAX:
        000001                          .END
```

```
ABASE = 000000              952     993
ACDW1 = 000000              952     995
ACDW2 = 000000              952     996
ACPUOP= 000000              952     967
ADDW0 = 000000              952     997
ADDW1 = 000000              952     998
ADDW10= 000000              952    1007
ADDW11= 000000              952    1008
ADDW12= 000000              952    1009
ADDW13= 000000              952    1010
ADDW14= 000000              952    1011
ADDW15= 000000              952    1012
ADDW2 = 000000              952     999
ADDW3 = 000000              952    1000
ADDW4 = 000000              952    1001
ADDW5 = 000000              952    1002
ADDW6 = 000000              952    1003
ADDW7 = 000000              952    1004
ADDW8 = 000000              952    1005
ADDW9 = 000000              952    1006
ADEVCT= 000000              952     958
ADEVM = 000000              952     994
ADRCNT  006057             2060*   2075*   2084#
ADVANC= 104420             2229#   5654    5737
AENV  = 000002                1#    952     963
AENVM = 000000              952     964
AFATAL= 000000              952     955
AMADR1= 000000              952     980
AMADR2= 000000              952     984
AMADR3= 000000              952     987
AMADR4= 000000              952     990
AMAMS1= 000000              952     974
AMAMS2= 000000              952     982
AMAMS3= 000000              952     985
AMAMS4= 000000              952     988
AMSGAD= 000000              952     960
AMSGLG= 000000              952     961
AMSGTY= 000000              952     954
AMTYP1= 000000              952     975
AMTYP2= 000000              952     983
AMTYP3= 000000              952     986
AMTYP4= 000000              952     989
APASS = 000000              952     957
APRIOR= 000000              952
APTCSU= 000040             1785    1890#
APTENV= 000001             1778    1846    1888#   2290
APTSIZ= 000200             1887#
APTSPO= 000100             1780    1848    1889#
APT.SI  013716             1453    2889#
ASWREG= 000000              952     965
ATESTN= 000000              952     956
AUDONE  003354             1490    1511    1550#
AUNIT = 000000              952     959
AUSTRT  003126             1489#
AUSWR = 000000              952     966
AU'O.S  012236             1451    2608#
```

```
AVECT1= 000000   952    991
AVECT2= 000000   952    992
BASE    014134  2900*  2911   2925*  2935#
BCCLD   031232  4345   4374   4422   4451   4499   4528   4576   4605   5961#
BINWRD  006412  2132*  2135*  2136   2173#
BITCON  032010  5896*  6097*  6099*  6100   6102*  6120*  6122*  6123   6125*  6150#
BIT0  = 000001   815#  2397   2398
BIT00 = 000001   805#  815
BIT01 = 000002   804#  814
BIT02 = 000004   803#  813
BIT03 = 000010   802#  812
BIT04 = 000020   801#  811
BIT05 = 000040   800#  810
BIT06 = 000100   799#  809
BIT07 = 000200   798#  808
BIT08 = 000400   797#  807
BIT09 = 001000   796#  806
BIT1  = 000002   814#  1458   2391   2397   2398   2770
BIT10 = 002000   795#  2751
BIT11 = 004000   794#  1721   3343   3377   3422   3481   3534   3587   3640   3694   3754   3784   3814
                 3844   3876   3926   3965   4004   4043   4085   4170   4287   4339   4416   4493   4570
                 4645   4713   4786   4885   4951   5200   5473   5538   5580   5676
BIT12 = 010000   793#  2682   2768
BIT13 = 020000   792#  2685   2697   2754   2772   3037   4216   4230   4253   4283   4310
BIT14 = 040000   791#  1703   2696   2721   2772   2783   4218   4285   5578   5674
BIT15 = 100000   790#  1398
BIT2  = 000004   813#  1458   1599   2398   2707   2717   4234   4256   4306
BIT3  = 000010   812#  2774   2781   4255
BIT4  = 000020   811#  2381   2404   2406   3458   3885   3935   3974   4013   4052   5539   5762   5804
                 5920
BIT5  = 000040   810#  2886   3040   3400   3512   3565   3618   3671   4129   5490
BIT6  = 000100   809#  2386   2387   2776   3793   3853
BIT7  = 000200   808#  2387   2506   2886   3430   3890   4232   4237   4258   4308   4312   6080
BIT8  = 000400   807#  2763   2765   2778   2780   2786   2789   2846
BIT9  = 001000   806#  2760   2786   2789   2844   2846
BM      010631  2429#  2732
BPTVEC= 000014   822#
BRLVL   013650  2841   2851   2859   2872#
BRW     004360  1604   1738#
CALBCC  031230  4364   4393   4441   4470   4518   4547   4595   4624   4682   4750   4811   4832   4980
                5001   5031   5052   5128   5171   5230   5251   5300   5321   5397   5451   5592   5648
                5688   5734   5936*  5937   5939*  5945   5946*  5947*  5950   5955#  6168
CHAR    030444  3877   3927   3966   4005   4044   4171   5812#
CHARSD  030556  5839#
CHRCNT  006410  2130*  2133   2137   2153*  2171#  2172
CKSWR   011340  1429   1744   2236   2454#
CKSWR1  011414  2466#  2478
CKSWR2  011426  2469#
CKSWR3  011432  2471#
CKSWR4  011436  2472#  2480   2487
CKSWR5  011542  2455   2457   2461   2496#
CLKX    001452  1027#
CLRIO   031572  4340   4369   4417   4446   4494   4523   4571   4600   6076#
CNERR   011154  1523   2429#
CNT.MA  002302  1041   1275#  1397   1399   1401   2534
CNVRT = 104417  1520   1629   1631   1633   1635   2228#  2271   2273   2348   2469
```

```
CONERR  011107          1518    2429#
CONN    010671          2429#   2687
CONTAB  003330          1527    1535#
CONVRT= 104416          1470    1526    2227#   2287    2630
CORMAX  033750          6168#
CR    = 000015           730#   1824    1834
CRC.CC= 102010          5958#
CRC16 = 120001          4342    4371    4419    4448    4496    4525    4573    4602    4650    4718    4805    4974    5025
                        5224    5294    5583    5679    5957#
CREAM   001502          1040#   1396*   2530*   2531    2533*   2537
CRLF  = 000200           731#   1795    1834
CSR     010363          2429#   2634
CSRMAP  012240          2610#
CTSDLY  031762          6136#
CYCLE   011606          1605    1666    2521#
DATABP  006764          2260*   2263    2285    2288#
DATACL= 104413          2224#   3384    3429    3433    3494    3495    3509    3547    3548    3562    3600    3601    3615
                        3653    3654    3668    3712    3715    3758    3788    3818    3848    3879    3929    3968    4007
                        4046    4087    4088    4133    4134    4173    4347    4348    4376    4377    4424    4425    4453
                        4454    4501    4502    4530    4531    4578    4579    4607    4608    4664    4666    4732    4734
                        4798    4812    4833    4851    4858    4860    4895    4967    4981    5002    5032    5053    5071
                        5078    5080    5091    5217    5231    5252    5272    5301    5322    5340    5347    5349    5360
                        5482    5487    5493    5548    6103    6126
DATAHD  006752          2259*   2281    2284#
DDISP = 177570           737#    893#    922
DELAY = 104411          2222#
DEVTAB  003342          1493    1540#
DH1     033224          1072    6168#
DH10    033524          6168#
DH2     033245          1075    6168#
DH3     033303          1078    1087    1099    1108    1123    1126    1132    6168#
DH4     033341          1081    1111    6168#
DH5     033401          1084    6168#
DH6     033433          1090    1102    1105    1120    6168#
DH7     033501          6168#
DISPLA  001242           922#   1415*   1421*   1733*
DISPRE  000174           883#   1421
DSWR  = 177570           736#    892#    921
DT1     033550          1073    6168#
DT10    033702          6168#
DT11    033724          6168#
DT12    033736          6168#
DT2     033562          1076    6168#
DT3     033600          1079    1100    1127    1133    6168#
DT4     033612          1082    1112    6168#
DT5     033624          1085    1091    1103    1106    1121    6168#
DT6     033642          1088    1109    6168#
DT7     033660          1124    6168#
DZDME = ****** U        1068
DZDMG = ****** U        1068
EMTVEC= 000030           825#
EM1     032062          1071    6168#
EM10    032365          1095    6168#
EM11    032570          1098    1119    6168#
EM12    032617          1101    6168#
EM13    032643          1104    6168#
```

```
EM14    032703          1107    6168#
EM15    032753          1110    6168#
EM16    032774          1113    6168#
EM17    033010          11'6    6168#
EM2     032120          1074    6168#
EM20    033054          1122    1128    6168#
EM21    033075          1125    6168#
EM22    033112          1131    6168#
EM23    033135          6168#
EM24    033157          6168#
EM25    033202          6168#
EM3     032163          1077    6168#
EM4     032227          1080    1083    6168#
EM5     032271          1086    6168#
EM6     032321          1089    6168#
EM7     032340          1092    6168#
EOM     031504          4795    4893    4960    4964    5209    5214    5598    5602    5606    5694    6044#
ERCT00  002304          1277#
ERCT01  002310          1280#
ERCT02  002314          1283#
ERCT03  002320          1286#
ERCT04  002324          1289#
ERCT05  002330          1292#
ERCT06  002334          1295#
ERCT07  002340          1298#
ERCT10  002344          1301#
ERCT11  002350          1304#
ERCT12  002354          1307#
ERCT13  002360          1310#
ERCT14  002364          1313#
ERCT15  002370          1316#
ERCT16  002374          1319#
ERCT17  002400          1322#
ERR     003244          1502    1515#   1519
ERRMSG  006740          2258*   2276    2279#
ERRPC   003322          1521    1532#
ERRVEC= 000004           818#   1708    1709*   1711*   1714*
ERTABO  007112          2273    2313#
EXIT  = 000205           845#
EXITER  007046          2299    2304#
FLAG    032006          6149#   6168
FLOAT   003156          1495#   1501
FLTDAT  032016          5586    5596    5600    5604    5611    5651    6153#
FY      003202          1503#   1507    1512
GETQI   031356          4384    4388    4461    4465    4538    4542    4615    4619    4740    4744    5999#
GETQO   031344          4355    4359    4432    4436    4509    4513    4586    4590    4672    4676    5989#
GETSI   030226          3498    3502    3551    3555    3604    3608    3657    3661    3718    3722    4815    4819    4836
                        4840    4852    4861    4984    4988    5005    5009    5035    5039    5056    5060    5072    5081
                        5234    5238    5255    5259    5275    5279    5304    5308    5325    5329    5341    5350    5743#
                        6127
GNS   = ****** U        2212    2215    2216    2217    2218    2219    2220    2221    2222    2223    2224    2225    2226
                        2227    2228    2229
HALTS   006770          2243    2290#
HILIM   006052          2057*   2065    2081#
HT    = 000011           728#   1793    1834
INCHAR  011546          2472    2500#
```

```
INIFLG  001506  1045#  1424  1454  1461*
INLP1   005766  2062#  2071
INPUT = 104415  2226#  2570  2621  2633  2641  2723  2731
INRDY   031066  4089  4135  4174  4182  4190  4896  4909  4914  5094  5108  5113  5137  5151
                5156  5363  5377  5382  5403  5417  5431  5436  5494  5504  5512  5612  5628
                5633  5699  5714  5719  5908#
INTTY   013664  2651  2672  2688  2703  2880#
IOTVEC= 000020   823#
KMAPTV  001470  1034#  1438  1563*  1564  2521  2535  2616*  2801*  2807*  2808*  2812  2837  2903*
                2904
KMCM    011175  1529  2429#
KMCR00  002100  1190#
KMCR01  002110  1195#
KMCR02  002120  1200#
KMCR03  002130  1205#
KMCR04  002140  1210#
KMCR05  002150  1215#
KMCR06  002160  1220#
KMCR07  002170  1225#
KMCR10  002200  1230#
KMCR11  002210  1235#
KMCR12  002220  1240#
KMCR13  002230  1245#
KMCR14  002240  1250#
KMCR15  002250  1255#
KMCR16  002260  1260#
KMCR17  002270  1265#
KMCSR   002066  1173#  1489*  1504  1510*  1539  1669  1736  2351  2405  2539*  2548  2596  2953*
KMCSRH  002070  1174#  2386*  2387*  2391*  2397*  2398*  2404*  2406*  2548*  2549*  2550
KMCTL   002072  1175#  2550*  2551*  2552
KMNUM   001472  1035#  1392  1646  2614*  2628*  2793*  2794  2802  2804
KMP04   002074  1176#  2375*  2381  2418  2423  2552*  2553*  2554  5749  5762  5804  5920  5995*
                6005*  6006*
KMP06   002076  1177#  2392*  2554*  2555*
KMRLVL  002060  1170#  1639  1640*  2557*  2558*  2559
KMRVEC  002056  1169#  1639*  1672  2540*  2541*  2557
KMS100  002102  1191#
KMS101  002112  1196#
KMS102  002122  1201#
KMS103  002132  1206#
KMS104  002142  1211#
KMS105  002152  1216#
KMS106  002162  1221#
KMS107  002172  1226#
KMS110  002202  1231#
KMS111  002212  1236#
KMS112  002222  1241#
KMS113  002232  1246#
KMS114  002242  1251#
KMS115  002252  1256#
KMS116  002262  1261#
KMS117  002272  1266#
KMS200  002104  1192#
KMS201  002114  1197#
KMS202  002124  1202#
KMS203  002134  1207#
```

```
KMS204  002144            1212#
KMS205  002154            1217#
KMS206  002164            1222#
KMS207  002174            1227#
KMS210  002204            1232#
KMS211  002214            1237#
KMS212  002224            1242#
KMS213  002234            1247#
KMS214  002244            1252#
KMS215  002254            1257#
KMS216  002264            1262#
KMS217  002274            1267#
KMS300  002106            1193#
KMS301  002116            1198#
KMS302  002126            1203#
KMS303  002136            1208#
KMS304  002146            1213#
KMS305  002156            1218#
KMS306  002166            1223#
KMS307  002176            1228#
KMS310  002206            1233#
KMS311  002216            1238#
KMS312  002226            1243#
KMS313  002236            1248#
KMS314  002246            1253#
KMS315  002256            1258#
KMS316  002266            1263#
KMS317  002276            1268#
KMTLVL  002064            1172#   1641    1642*   2561*   2562*
KMTVEC  002062            1171#   1641*   2559*   2560*   2561
KM.END  002300            1270#   2612    2908
KM.MAP  002100            1040    1189#   1396    1463    1473    2531    2533    2610    2615    2829    2906    2910
LF    = 000012             729#   1828    1834
LINE    010573            2429#   2724
LOBITS  006056            2059*   2083#
LOCK    001444            1021#   1735*   1747    1749    2090*   2267    3059*   3075*   3102*   3118*   3145*   3168*   3204*
                          3224*   4336*   4368*   4413*   4445*   4490*   4522*   4567*   4599*
LOKFLG  001510            1047#
LOLIM   006050            2056*   2067    2080#
LRCB  = 000200            5956#
LUTYPE= ****** U           687    5928
MASKX   001454            1028#
MASTEK  010015            2269    2429#
MCSRX   007745            1628    2429#
MDATA   011276            2151    2161    2446#
MEMLIM  001466            1033#   1587*
MEPASS  007620            1627    2429#
MERRPC  010072            2272    2429#
MERRX   007772            1634    2429#
MERR2   007645            2429#   2814
MERR3   007672            1560    2429#
MESDAT  032012            4296    4787    4793    4886    4891    4952    4958    4962    5018    5093    5136    5201    5207
                          5212    5287    5362    5416    5474    5479    5484    5545    5682    5692    5698    6151#
MESLD   031530            4295    4792    4890    4957    4961    5206    5211    5478    5483    5544    5595    5599    5603
                          5691    6057#
MILK    001504            1041#   1397*   1636    2529*   2534*   2538
```

```
MLOCK   007716          1601    2429#
MNEW    010017          1555    2429#
MODU    010461          2429#   2671
MPASSX  007761          1632    2429#
MPFAIL  007562          2347    2364    2429#
MR      007642          1608    2429#   2586
MRESET= 004000           845#
MSTCLR= 104410          2221#   2353    3004    3030    3061    3104    3147    3206    3258    3282    3306    3342    3376
                        3421    3480    3533    3586    3639    3693    3753    3783    3813    3843    3874    3924    3963
                        4002    4041    4080    4123    4169    4215    4282    4338    4415    4492    4569    4644    4712
                        4782    4884    4947    5196    5472    5536    5577    5673
MTITLE  001000           891#   1428
MTSTN   010003          2270    2429#   2571
MVECX   007753          1630    2429#
MV35    010762          2429#   2702
NEXT    001442          1020#   2089    2309    2951*   2977*   3002*   3028*   3058*   3101*   3144*   3203*   3256*   3280*
                        3304*   3340*   3374*   3419*   3478*   3531*   3584*   3637*   3691*   3751*   3781*   3811*   3841*
                        3872*   3922*   3961*   4000*   4039*   4078*   4121*   4167*   4213*   4280*   4335*   4412*   4489*
                        4566*   4642*   4710*   4780*   4882*   4945*   5194*   5470*   5534*   5575*   5671*
NITCH   030256          5749*   5750*   5751*   5752*   5754#
NOACT   010731          1440    2429#   2523
NODEV   003240          1491    1513#
NUM     010323          2429#   2622
OCOR    030260          3383    3428    3491    3544    3597    3650    3711    4663    4731    4796    4894    4965    5215
                        5481    5486    5547    5607    5695    5757#   5763    5793    5835    5851    5892    5985    6071
OK      003220          1505    1508#   1531
ONE     001464          1032#   6168
OUTRDY  030412          3482    3487    3535    3540    3588    3593    3641    3646    3698    3702    3707    3730    4791
                        4889    4956    5205    5477    5543    5777    5786    5797#   5822    5831    5844    5848    5868
                        5877    5972    5981    6018    6034    6048    6065
PACT00  002302          1276#
PACT01  002306          1279#
PACT02  002312          1282#
PACT03  002316          1285#
PACT04  002322          1288#
PACT05  002326          1291#
PACT06  002332          1294#
PACT07  002336          1297#
PACT10  002342          1300#
PACT11  002346          1303#
PACT12  002352          1306#
PACT13  002356          1309#
PACT14  002362          1312#
PACT15  002366          1315#
PACT16  002372          1318#
PACT17  002376          1321#
PARBIT= 040000           845#
PERFOR= 004537           845#
PFTAB   007324          2348    2370#
PIRQ  = 177772           735#
PIRQVE= 000240           829#
POPRO = 012600           840#   2303
POP1SP= 005726           838#
POP2SP= 022626           842#
PRIO    010422          2429#   2650
PRIRTY  014136          2896*   2902*   2915    2936#
```

```
PR0    = 000000   752#  2872  2873
PR1    = 000040   753#
PR2    = 000100   754#
PR3    = 000140   755#
PR4    = 000200   756#  2874
PR5    = 000240   757#  2875
PR6    = 000300   758#  2876
PR7    = 000340   759#   864   866   868   870  2877
PS     = 177776   732#   733  1389* 1598* 2841* 2851*
PSW    = 177776   733#
PUSHR0= 010046   839#  2300
PUSH1S= 005746   837#
PUSH2S= 024646   841#
PWRVEC= 000024   824#  2326* 2327* 2336* 2342* 2361* 2362*
QV.FLG  001511  1048#  1395* 1645*
RDCHR  = 104402  1942  2215#
RDLIN  = 104403  2015  2216#
RDOCT  = 104404  2064  2217#
RESREG  006766  2286  2289#
RESVEC= 000010   819#
RESOS  = 104407  2220#  2289
ROMCLK= 104412  2223#  2376  2379  2416  2421  2955  2980  3006  3032  3064  3066  3077  3079
                3107  3109  3120  3122  3153  3155  3174  3176  3211  3213  3229  3231  3259
                3283  3309  3318  3345  3347  3351  3379  3381  3386  3396  3424  3426  3431
                3435  3445  3455  3484  3489  3510  3537  3542  3563  3590  3595  3616  3643
                3648  3669  3700  3704  3709  3732  3759  3789  3819  3849  3881  3891  3893
                3931  3940  3970  3979  4009  4018  4048  4057  4090  4130  4136  4175  4183
                4191  4222  4225  4245  4248  4289  4293  4301  4653  4657  4661  4691  4721
                4725  4729  4759  4897  4910  4915  4924  5095  5109  5114  5123  5138  5152
                5157  5166  5364  5378  5383  5392  5404  5418  5432  5437  5446  5488  5495
                5505  5513  5540  5549  5613  5629  5634  5643  5700  5715  5720  5729  5747
                5760  5775  5779  5782  5788  5791  5802  5820  5824  5827  5833  5846  5849
                5866  5870  5873  5879  5899  5901  5918  5970  5974  5977  5983  5993  6003
                6016  6020  6023  6036  6039  6050  6052  6067  6081  6083
RUN      001500  1038#  1398* 2527* 2528* 2535
SAVACT   001474  1036#  1558  2812* 2904*
SAVNUM   001476  1037#  1392* 1643* 1646* 2805* 2898*
SAVPC    001460  1030#  1519* 1534  2096* 2315
SAVSP    001456  1029#
SAVOS  = 104406  2219#  2248
SCHAR    031062  4081*  4124* 5860  5878  5883  5887* 5888  5894* 5904#
SCOP1  = 104405  2218#  3074  3087  3117  3130  3164  3185  3220  3238  4367  4396  4444  4473
                4521  4550  4598  4627
SILOLD   030622  4086  4102  4132  4148  5855#
SIMBCC   031122  4350  4379  4427  4456  4504  4533  4581  4610  4667  4735  4807  4976  5027
                5226  5296  5588  5684  5928#
SOFTSW   011600  2470  2509#
SOM      031454  5210  6030#
SPACNT= 006411  2131* 2155  2158* 2172#
STACK  = 001200   723#  1390  1577  2310
STAT     001450  1026#
STAT1    002050  1162#  2542* 3037  4216  4218  4230  4253  4283  4285  4310  5578  5674
STAT2    002052  1163#  2543* 3262  3286
STAT3    002054  1164#  2544* 4234  4256  4306
STFFCK   031672  6110#
STFFCL   031612  5884  6088#
```

```
STKLMT= 177774            734#
STRTSW  001446           1025#   1430*   1433*   1434    1436    1450*   1456    1458    1553    1599    1606    2564    2587*
                         2617    2825
STUFDT  032036           6159#
STUFLG  031064           4082*   4125*   5881    5895*   5905#
SV05    006100           2100#
SWFLG   011544           1393*   2465*   2492*   2498#
SWR     001240            921#   1414*   1416    1420*   1430    1558    1563    1703    1721    1745    2237    2242    2298
                         2305    2307    2334    2354*   2394    2454    2491*
SWREG   000176            884#   1409*   1420    2454    2511
SW0    = 000001            787#
SW00   = 000001            777#    787    1434    1450    2617    2825
SW01   = 000002            776#    786    1606    2564    2587
SW02   = 000004            775#    785
SW03   = 000010            774#    784    1553
SW04   = 000020            773#    783
SW05   = 000040            772#    782
SW06   = 000100            771#    781    2394
SW07   = 000200            770#    780
SW08   = 000400            769#    779    2305
SW09   = 001000            768#    778    1745
SW1    = 000002            786#
SW10   = 002000            767#   2307
SW11   = 004000            766#
SW12   = 010000            765#   2237
SW13   = 020000            764#   2242
SW14   = 040000            763#
SW15   = 100000            762#
SW2    = 000004            785#
SW3    = 000010            784#
SW4    = 000020            783#
SW5    = 000040            782#
SW6    = 000100            781#
SW7    = 000200            780#
SW8    = 000400            779#
SW9    = 001000            778#
SYNC    030276           3756    3786    3816    3846    5767#
SYNLD   031374           4651    4719    4790    4888    4955    5204    5476    5542    5581    5582    5677    5678    6010#
TBITVE= 000014            820#
TEMP    011234           2138    2402*   2407*   2413*   2425*   2444#   3307*   3313*   3316*   3322*
TIMER  = 104414           2225#   3349    3880    3930    3969    4008    4047    4224    4247    4291
TKVEC  = 000060            827#
TLAST  = 027706           2590    5738#
TPVEC  = 000064            828#
TRAPVE= 000034             826#
TRTVEC= 000014             821#
TST1    014140           1697    2577    2595    2949#
TST10   015136           3144    3201#
TST11   015312           3203    3254#
TST12   015354           3256    3278#
TST13   015416           3280    3302#
TST14   015516           3304    3338#
TST15   015616           3340    3372#
TST16   015754           3374    3417#
TST17   016152           3419    3476#
TST2    014214           2951    2975#
```

```
TS:20   016334            3478    3529#
TST21   016516            3531    3582#
TST22   016700            3584    3635#
TST23   017062            3637    3689#
TST24   017270            3691    3749#
TST25   017356            3751    3779#
TST26   017446            3781    3809#
TST27   017534            3811    3839#
TST3    014262            2977    3000#
TST30   017624            3841    3870#
TST31   017776            3872    3920#
TST32   020112            3922    3959#
TST33   020230            3961    3998#
TST34   020346            4000    4037#
TST35   020464            4039    4076#
TST36   020614            4078    4119#
TST37   020754            4121    4165#
TST4    014334            3002    3026#
TST40   021114            4167    4211#
TST41   021342            4213    4278#
TST42   021536            4280    4333#
TST43   022052            4335    4410#
TST44   022366            4412    4487#
TST45   022702            4489    4564#
TST46   023216            4566    4640#
TST47   023454            4642    4708#
TST5    014426            3028    3056#
TST50   023712            4710    4778#
TST51   024244            4780    4880#
TST52   024446            4882    4943#
TST53   025546            4945    5192#
TST54   026746            5194    5468#
TST55   027200            5470    5532#
TST56   027316            5534    5573#
TST57   027706            5575    5669#   5738
TST6    014566            3058    3099#
TST60 = ****** U          5671
TST7    014726            3101    3142#
TTST    004146            1602*   1604*   1699#
TWOSYN= 010000             845#
TYPDAT  006754            2264    2282    2285#
TYPE   = 104401           1428    1440    1462    1517    1522    1528    1555    1560    1601    1608    1627    1628    1630
                          1632    1634    1798    1949    1955    1960    1964    1969    1970    1972    1975    1979    2044
                          2046    2062    2069    2070    2121    2161    2212#   2265    2266    2269    2270    2272    2274
                          2278    2283    2347    2363    2468    2471    2523    2569    2586    2592    2629    2649    2663
                          2670    2679    2686    2693    2701    2713    2814
TYPMSG  006654            2262    2265#
VEC     010401            2429#   2642
VECMAP  013400            2813    2825#
VECTR   014132            2895*   2901*   2920    2926*   2934#
WHAT    005770            2055*   2063#
WHERE   006054            2058*   2072    2082#
WHICH   013372            2631    2821#
WRDCNT  006406            2129*   2162*   2170#
WRKO.F  006742            2277    2280#
XBX     006542            2238    2240    2242#
```

```
XCSR    004104            1629    1667#
XERR    004126            1635    1676#
XHEAD   010077            1462    2429#
XPASS   004120            1633    1673#
XPOLY   031226            4342*   4371*   4419*   4448*   4496*   4525*   4573*   4602*   4650*   4718*   4805*   4974*   5025*
                          5224*   5294*   5583*   5679*   5944    5946    5954#
XSTATQ  011206            1471    2429#
XTSTN   007120            2271    2316#
XVEC    004112            1631    1670#
ZERO    001462            1031#   6168
$APTHD  002034            1144    1150#
$ASTAT= ****** U          1868    1883
$ATYC   004722            1839    1841#
$ATY1   004676            1837#
$ATY3   004704            1783    1838#
$ATY4   004714            1840#   2293
$AUTOB  001234             918#
$BASE   001372             993#   2900
$BDADR  001222             913#
$BDDAT  001226             915#
$CDW1   001376             995#   2897
$CDW2   001400             996#
$CHARC  004672            1800*   1810*   1817    1826*   1831#
$CKSWR= ****** U          2215
$CMTAG  001200             901#
$CM1  = 000006             933#    934#    935#    936#    937#    938#    939#
$CM2  = 000014             933#    934#    935#    936#    937#    938#    939#
$CM3  = 000006             931#    933
$CM4  = 000005             939#    940#    941#    942#    943#    944#
$CNTLG  005534            1990#
$CNTLU  005527            1964    1989#
$COD  = ****** U           687
$CPUOP  001344             967#
$CRAP = 177777               1#    2938#   2941    2944#   2964#   2967    2970#   2989#   2992    2995#   3015#   3018    3021#
                          3045#   3048    3051#   3088#   3091    3094#   3131#   3134    3137#   3190#   3193    3196#   3243#
                          3246    3249#   3267#   3270    3273#   3291#   3294    3297#   3326#   3329    3333#   3360#   3363
                          3367#   3405#   3408    3412#   3463#   3466    3471#   3516#   3519    3524#   3569#   3572    3577#
                          3622#   3625    3630#   3675#   3678    3684#   3738#   3741    3744#   3768#   3771    3774#   3798#
                          3801    3804#   3828#   3831    3834#   3858#   3861    3965#   3909#   3912    3915#   3948#   3951
                          3954#   3987#   3990    3993#   4026#   4029    4032#   4065#   4068    4071#   4106#   4109    4114#
                          4152#   4155    4160#   4199#   4202    4206#   4267#   4270    4273#   4321#   4324    4328#   4398#
                          4401    4405#   4475#   4478    4482#   4552#   4555    4559#   4629#   4632    4635#   4697#   4700
                          4703#   4765#   4768    4773#   4868#   4871    4875#   4927#   4930    4938#   5175#   5178    5187#
                          5455#   5458    5463#   5521#   5524    5527#   5558#   5561    5568#   5655#   5658    5664#
$CRLF   001313             946#   1799    1834    1969    1989    2049    2070    2121    2265    2266    2274    2569    2629
$DDW0   001402             997#   2905
$DDW1   001404             998#
$DDW10  001426            1007#
$DDW11  001430            1008#
$DDW12  001432            1009#
$DDW13  001434            1010#
$DDW14  001436            1011#
$DDW15  001440            1012#
$DDW2   001406             999#
$DDW3   001410            1000#
$DDW4   001412            1001#
```

```
$DDW5   001414            1002#
$DDW6   001416            1003#
$DDW7   001420            1004#
$DDW8   001422            1005#
$DDW9   001424            1006#
$DEVCT  001326             958#
$DEVM   001374             994#   2903
$DOAGN  004100            1644    1653    1658    1664#
$ENDAD  004070             877    1426    1660#   2296
$ENDCT  004054            1655#
$ENV    001336             963#   1407    1443    1445    1778    1846    1870    2290    2456
$ENVM   001337             964#   1448    1780    1785    1848
$EOP    003662            1623#   5671
$EOPCT  004046            1652#   1656
$ERFLG  001203             904#   1394*   1626*   1688    1717    1719*   1740    2247*   2261    2275*   2349*
$ERMAX  001215             910#   1740
$ERROR  006512             867    2236#
$ERRPC  001216             911#   1403*   1647*   1696*   2244    2246*   2350*
$ERRTB  001512            1065#   2257
$ERTTL  001212             908#   1638    1678    2304*   2546*
$ETABL  001336             962#
$ETEND  001442            1015#   1156
$FATAL  001320             955#   1874*
$FFLG   005142            1837*   1840*   1868    1877*   1885#
$FILLC  001256             929#   1803    1834
$FILLS  001255             928#   1834
$GDADR  001220             912#
$GDDAT  001224             914#
$GET42  004060            1657#
$GTSWR= ****** U          2214
$HD   = 000000             697
$HIBTS  002034            1151#
$HIOCT  005722            2037*   2048#
$ICNT   001204             905#   1725*   1726    1728*   1739
$ILLUP  007316            2326    2342    2366#
$INPUT  005724            2053#   2226
$INTAG  001235             919#
$ITEMB  001214             909#   2252*   2292
$LF     001314             947#   1834    1979    1989    2049
$LFLG   005141            1878*   1884#
$LPADR  001206             906#   1405*   1605*   1609    1732*   1734    1739    2309*   2311    2585*   2595*   2597
$LPERR  001210             907#
$MADR1  001350             980#
$MADR2  001354             984#
$MADR3  001360             987#
$MADR4  001364             990#
$MAIL   001316             953#   1152    1156    1731    1778
$MAMS1  001346             974#   2899
$MAMS2  001352             982#
$MAMS3  001356             985#
$MAMS4  001362             988#
$MBADR  002036            1152#
$MFLG   005140            1838*   1844    1879*   1883#
$MNEW   005552            1993#   2471
$MSGAD  001332             960#   1854*   1857
$MSGLG  001334             961#   1859*
```

```
$MSGTY  001316          954#  1852  1860*  1872  1876*
$MSWR   005541         1991#  2468
$MTYP1  001347          975#
$MTYP2  001353          983#
$MTYP3  001357          986#
$MTYP4  001363          989#
$MXCNT  004362         1729   1739#
$N    = 000057            1#  2938  2944  2946  2953#  2964  2970  2972  2979#  2989  2995  2997  3004
                       3005#  3015  3021  3023  3030  3031#  3045  3051  3053  3061  3062#  3088  3094
                       3096   3104  3105#  3131  3137  3139  3147  3148#  3190  3196  3198  3206  3207#
                       3243   3249  3251  3258  3259#  3267  3273  3275  3282  3283#  3291  3297  3299
                       3306   3307#  3326  3333  3335  3342  3343#  3360  3367  3369  3376  3377#  3405
                       3412   3414  3421  3422#  3463  3471  3473  3480  3481#  3516  3524  3526  3533
                       3534#  3569  3577  3579  3586  3587#  3622  3630  3632  3639  3640#  3675  3684
                       3686   3693  3694#  3738  3744  3746  3753  3754#  3768  3774  3776  3783  3784#
                       3798   3804  3806  3813  3814#  3828  3834  3836  3843  3844#  3858  3865  3867
                       3874   3875#  3909  3915  3917  3924  3925#  3948  3954  3956  3963  3964#  3987
                       3993   3995  4002  4003#  4026  4032  4034  4041  4042#  4065  4071  4073  4080
                       4081#  4106  4114  4116  4123  4124#  4152  4160  4162  4169  4170#  4199  4206
                       4208   4215  4216#  4267  4273  4275  4282  4283#  4321  4328  4330  4338  4339#
                       4398   4405  4407  4415  4416#  4475  4482  4484  4492  4493#  4552  4559  4561
                       4569   4570#  4629  4635  4637  4644  4645#  4697  4703  4705  4712  4713#  4765
                       4773   4775  4782  4783#  4868  4875  4877  4884  4885#  4927  4938  4940  4947
                       4948#  5175  5187  5189  5196  5197#  5455  5463  5465  5472  5473#  5521  5527
                       5529   5536  5537#  5558  5568  5570  5577  5578#  5655  5664  5666  5673  5674#
                       5738#
$NULL   001254          927#  1805  1834
$NWTST= 000000         2948#  2974#  2999#  3025#  3055#  3098#  3141#  3200#  3253#  3277#  3301#  3337#  3371#
                       3416#  3475#  3528#  3581#  3634#  3688#  3748#  3778#  3808#  3838#  3869#  3919#  3958#
                       3997#  4036#  4075#  4118#  4164#  4210#  4277#  4332#  4409#  4486#  4563#  4639#  4707#
                       4777#  4879#  4942#  5191#  5467#  5531#  5572#  5668#
$OVER   004334         1701   1704  1715  1727  1733#
$PASS   001324          957#  1625*  1637  1649*  1650*  1667  1675  1723  1740  2545*
$PASTM  002042         1154#
$PWRDN  007126          865   1391  2326#  2361
$PWRMG  007312         2364#
$PWRUP  007200         2336   2342#
$QUES   001312          945#  1834  1972  1989  2046  2049  2069  2592  2664  2680  2694  2714
$RDCHR  005144         1909#  2215
$RDDEC= ****** U       2218
$RDLIN  005264         1937#  2216
$RDOCT  005564         2010#  2217
$RDSZ = 000007         1930#
$REGAD  001260          931#
$REG0   001262          933#  2105*  2110  6168
$REG1   001264          934#  1525*  1537  2104*  2111
$REG2   001266          935#  2103*  2112  6168
$REG3   001270          936#  2102*  2113  6168
$REG4   001272          937#  2101*  2114  6168
$REG5   001274          938#  2100*  2115  6168
$RTNAD  004102         1666#
$R2A  = ****** U       2218
$S    = 000061            1#  2951  2953#  2977  2979#  3002  3005#  3028  3031#  3058  3062#  3101  3105#
                       3144   3148#  3203  3207#  3256  3259#  3280  3283#  3304  3307#  3340  3343#  3374
                       3377#  3419  3422#  3478  3481#  3531  3534#  3584  3587#  3637  3640#  3691  3694#
                       3751   3754#  3781  3784#  3811  3814#  3841  3844#  3872  3875#  3922  3925#  3961
```

```
                    3964#   4000    4003#   4039    4042#   4078    4081#   4121    4124#   4167    4170#   4213    4216#
                    4280    4283#   4335    4339#   4412    4416#   4489    4493#   4566    4570#   4642    4645#   4710
                    4713#   4780    4783#   4882    4885#   4945    4948#   5194    5197#   5470    5473#   5534    5537#
                    5575    5578#   5671    5674#
$SAVRE= ****** U    2218
$SAVR6  007322      2335*   2343    2344*   2345*   2368#
$SCOPE  004134       863    1695#   2869
$SETUP= 0C0000      1648    1696    1898    1995
$SVLAD  004316      1712    1730#
$SVPC = 000040       875#    880
$SWR  = 164000         1#    697     944     945    1620    1648    1659    1665    1667    1689    1690    1691    1692
                    1703    1715    1717    1718    1719    1720    1721    1733    1739    2365    2950    2976    3001
                    3027    3057    3100    3143    3202    3255    3279    3303    3339    3373    3418    3477    3530
                    3583    3636    3690    3750    3780    3810    3840    3871    3921    3960    3999    4038    4077
                    4120    4166    4212    4279    4334    4411    4488    4565    4641    4709    4779    4881    4944
                    5193    5469    5533    5574    5670
$SWREG  001340       965#   1409
$SWRMK= 000000      1692
$TESTN  001322       956#   1731*
$TIMES  001310       944#   1648*   1720*   1726    1729*   1739
$TKB    001246       924#   1702    1896    1913    1919    2458    2460    2502    2882
$TKS    001244       923#   1700    1896    1911    1917    2500    2880    6142
$TMP0   001276       939#   1464*   2431    2837*   2838*   5772*   5784*   5818*   5829*   5864*   5875*   5916*   5922*
                    5934*   5948*   5968*   5979*   6014*   6025*   6064*   6069*
$TMP1   001300       940#   1465*   2433    5125*   5126*   5127    5168*   5169*   5170    5394*   5395*   5396    5448*
                    5449*   5450    5645*   5646*   5647    5731*   5732*   5733    5816*   5832    5842*   5845    5859*
                    5863*   5890*   5897*   5935*   5940*   5966*   5982
$TMP2   001302       941#   1467*   1488*   1515    1524*   2435    2625    2628    2740*   4912*   4913*   5111*   5112*
                    5127*   5128    5154*   5155*   5170*   5171    5380*   5381*   5396*   5397    5434*   5435*   5450*
                    5451    5631*   5632*   5647*   5648    5717*   5718*   5733*   5734    6094*   6104*   6117*   6130*
                    6168
$TMP3   001304       942#   1468*   2437    2637    2640    2645    2648    2727    2730    2735    2738
$TMP4   001306       943#   1469*   2439    2620*   2632*   2823    5800*   5806*
$TN   = 000060         1#    697    2948    2950#   2974    2976#   2999    3001#   3025    3027#   3055    3057#   3098
                    3100#   3141    3143#   3200    3202#   3253    3255#   3277    3279#   3301    3303#   3337    3339#
                    3371    3373#   3416    3418#   3475    3477#   3528    3530#   3581    3583#   3634    3636#   3688
                    3690#   3748    3750#   3778    3780#   3808    3810#   3838    3840#   3869    3871#   3919    3921#
                    3958    3960#   3997    3999#   4036    4038#   4075    4077#   4118    4120#   4164    4166#   4210
                    4212#   4277    4279#   4332    4334#   4409    4411#   4486    4488#   4563    4565#   4639    4641#
                    4707    4709#   4777    4779#   4879    4881#   4942    4944#   5191    5193#   5467    5469#   5531
                    5533#   5572    5574#   5668    5670#
$TPB    001252       926#   1823*   1834    2241*   2505*   2885*
$TPFLG  001257       930#   1772    1834
$TPS    001250       925#   1821    1834    2239    2503    2883
$TRAP   006414       869    2189#
$TRAP2  006436      2200#   2211
$TRP  = 000021      2204#   2213#   2215    2216#   2217#   2218#   2219#   2220#   2221#   2222#   2223#   2224#   2225#
                    2226#   2227#   2228#   2229#   2230#
$TRPAD  006450      2194    2211#
$TSTM   002040      1153#
$TSTNM  001202       903#   1404*   1688    1730*   1731    1733    1740    2318    2372    2574    2581    2583    2950*
                    2976*   3001*   3027*   3057*   3100*   3143*   3202*   3255*   3279*   3303*   3339*   3373*   3418*
                    3477*   3530*   3583*   3636*   3690*   3750*   3780*   3810*   3840*   3871*   3921*   3960*   3999*
                    4038*   4077*   4120*   4166*   4212*   4279*   4334*   4411*   4488*   4565*   4641*   4709*   4779*
                    4881*   4944*   5193*   5469*   5533*   5574*   5670*
$TTYIN  005520      1939    1940    1952    1970    1984    1988#
```

```
$TYPBN= ****** U        2213
$TYPDS= ****** U        2213
$TYPE   004414          1772#   1865    2204    2212
$TYPEC  004626          1802    1809    1816    1821#   1822
$TYPEX  004674          1827    1829    1832#
$TYPOC= ****** U        2213
$UNIT   001330           959#
$UNITM  002044          1155#
$USWR   001342           966#
$VECT1  001366           991#   2901    2902
$VECT2  001370           992#
$XTSTR  004174          1698    1706#
$Y    = 000000             1#   1157#
$$GET4= 000000          1659#
$40CAT= ****** U        1703
.     = 033750           857#    859    862#    875    876#    878#    880#    882#    886#    890#    900#    948    1034#
                        1035#   1036#   1037#   1046#   1134#   1140    1141#   1143#   1145#   1188#   1190#   1191#   1192#
                        1193#   1195#   1196#   1197#   1198#   1200#   1201#   1202#   1203#   1205#   1206#   1207#   1208#
                        1210#   1211#   1212#   1213#   1215#   1216#   1217#   1218#   1220#   1221#   1222#   1223#   1225#
                        1226#   1227#   1228#   1230#   1231#   1232#   1233#   1235#   1236#   1237#   1238#   1240#   1241#
                        1242#   1243#   1245#   1246#   1247#   1248#   1250#   1251#   1252#   1253#   1255#   1256#   1257#
                        1258#   1260#   1261#   1262#   1263#   1265#   1266#   1267#   1268#   1432    1442    1549#   1562
                        1667    1739    1740    1834    1886#   1896    1988#   1989    1995#   2049    2338    2367    2410#
                        2429#   2445#   2447#   2501    2504    2525    2618    2659    2767    2771    2817    2848    2881
                        2884    3038    4217    4231    4254    4284    4311    4862    5082    5351    6168#
.ADVAN  006060          2089#   2229
.BEGIN  003464          1577#
.CNVRT  006170          2122#   2228
.CONVR  006164          2121#   2227
.DATAC  007446          2224    2401#
.DELAY  007332          2222    2374#
.MSTCL  007362          2221    2385#
.RES05  006132          2110#   2220
.ROMCL  007400          2223    2390#
.SAV05  006072          2096#   2219
.SCOP1  004364          1744#   2218
.START  002402           887    1389#   1405    2488
.TIMER  007512          2225    2412#
.$ASTA= ****** U        1838    1841
.$X   = 002034          1140#   1145
```

```
COMMEN   830#
ENDCOM   830#
ERROR    724#  2962  2987  3013  3043  3073  3086  3116  3129  3163  3184  3219  3237  3265  3289
         3315  3324  3358  3393  3403  3442  3452  3461  3500  3504  3514  3553  3557  3567  3606
         3610  3620  3659  3663  3673  3720  3724  3766  3796  3826  3856  3888  3900  3907  3938
         3946  3977  3985  4016  4024  4055  4063  4096  4142  4181  4189  4197  4243  4265  4317
         4357  4361  4386  4390  4434  4438  4463  4467  4511  4515  4540  4544  4588  4592  4617
         4621  4674  4678  4742  4746  4817  4821  4838  4842  4854  4863  4903  4922  4986  4990
         5007  5011  5037  5041  5058  5062  5074  5083  5101  5121  5130  5144  5164  5173  5236
         5240  5257  5261  5277  5281  5306  5310  5327  5331  5343  5352  5370  5390  5399  5410
         5424  5444  5453  5492  5501  5511  5519  5556  5619  5641  5650  5706  5727  5736  5808
         5924  6129
ESCAPE   830#
GETPRI   830#
GETSWR   830#
HLT      843#
KMEND      1#  1610
KMFRNT     1#   687
MULT     830#
NEWTST   830#  2948  2974  2999  3025  3055  3098  3141  3200  3253  3277  3301  3337  3371  3416
         3475  3528  3581  3634  3688  3748  3778  3808  3838  3869  3919  3958  3997  4036  4075
         4118  4164  4210  4277  4332  4409  4486  4563  4639  4707  4777  4879  4942  5191  5467
         5531  5572  5668
POP      830#  1880  1881  2038  2354  2355  2928
PUSH     830#  1841  1843  1864  2012  2328  2334  2891
REPORT     1#   830#
SCOPE    725#  2949  2975  3000  3026  3056  3099  3142  3201  3254  3278  3302  3338  3372  3417
         3476  3529  3582  3635  3689  3749  3779  3809  3839  3870  3920  3959  3998  4037  4076
         4119  4165  4211  4278  4333  4410  4487  4564  4640  4708  4778  4880  4943  5192  5468
         5532  5573  5669
SETPRI   830#
SETTRA  2204#  2215  2216  2217  2218  2219  2220  2221  2222  2223  2224  2225  2226  2227  2228
         2229
SETUP    830#
SKIP     830#
SLASH    830#
SPACE    830#
STARS    830#   873   896   948   951  1137  1139  1146  1618  1685  1757  1836  1895  1901  1930
         1998  2183  2324  2340  2948  2974  2999  3025  3055  3098  3141  3200  3253  3277  3301
         3337  3371  3416  3475  3528  3581  3634  3688  3748  3778  3808  3838  3869  3919  3958
         3997  4036  4075  4118  4164  4210  4277  4332  4409  4486  4563  4639  4707  4777  4879
         4942  5191  5467  5531  5572  5668
SWRSU    830#
TRMTRP  2204#
TYPBIN   830#
TYPDEC   830#
TYPNAM   830#
TYPNUM   830#
TYPOCS   830#
TYPOCT   830#
TYPTXT   830#
$ABORT   687#
$AUTO      1#  1474
$BCC     687#  4321  4398  4475  4552
$BINCR   687#  4629  4697
$BINWI   687#  3675
```

```
$BUFFE      1#   2441
$CDATA    687#   5558   5655
$CLOCK    687#   3291
$COMP       1#    687#  3883  3895  3902  3933  3942  3972  3981  4011  4020  4050  4059  4177  4185
         4193   4917   5116  5159  5385  5439  5636  5722
$CRC      687#   4340   4369  4417  4446  4494  4523  4571  4600
$CRCSH    687#   4645   4713
$CYCLE      1#   2512
$EMPTY    687#   5455
$EOP        1#   1610
$ERTBL      1#   1067
$FINI       1#   5738
$FLAG     687#
$FLOAT    687#   3149   3169  3208  3225
$GETPA      1#
$HALF     687#   5521
$HEADE      1#    697
$INACT    687#   3738   3768  3798  3828
$INIT     687#
$LINE1    687#   3131   3190
$LU1      687#   2938   2964  2989  3015
$LU12     687#   3045
$LU17     687#   3088
$MARHI      1#
$MARK     687#   4152
$MATCH    687#   4906   5104  5147  5373  5427  5624  5710
$MOCK       1#
$MODEM    687#   4199
$MSG        1#   2429
$MULT     687#
$PASEN      1#   1624
$PATTE    687#   4065   4106
$PFAIL      1#   2319
$QOQI     687#   5989   5999
$QUEST      1#   2621   2633  2641  2723  2731
$RAMCL      1#   2373
$RCLK       1#   2376   2379  2416  2421  2955  2980  3006  3032  3064  3066  3077  3079  3107  3109
         3120   3122   3153  3155  3174  3176  3211  3213  3229  3231  3259  3283  3309  3318  3345
         3347   3351   3379  3381  3386  3396  3424  3426  3431  3435  3445  3455  3484  3489  3510
         3537   3542   3563  3590  3595  3616  3643  3648  3669  3700  3704  3709  3732  3759  3789
         3819   3849   3881  3891  3893  3931  3940  3970  3979  4009  4018  4048  4057  4090  4130
         4136   4175   4183  4191  4222  4225  4245  4248  4289  4293  4300  4653  4657  4661  4691
         4721   4725   4729  4759  4897  4910  4915  4924  5095  5109  5114  5123  5138  5152  5157
         5166   5364   5378  5383  5392  5404  5418  5432  5437  5446  5488  5495  5505  5513  5540
         5549   5613   5629  5634  5643  5700  5715  5720  5729  5747  5760  5775  5779  5782  5788
         5791   5802   5820  5824  5827  5833  5846  5849  5866  5870  5873  5879  5899  5901  5918
         5970   5974   5977  5983  5993  6003  6016  6020  6023  6036  6039  6050  6052  6067  6081
         6083
$RCRC     687#   4868
$REC      687#   3858   3909  3948  3987  4026
$ROVAR      1#   1017
$SCADD      1#   1696
$SCAD1      1#   1735
$SIMBC      1#   5928
$SINAC    687#
$SOFTC      1#   2449
```

```
$STUFF   687#
$SWPAC   687#  3243  3267
$TCMAR   687#  4786  4885  4951  5200
$TCRC    687#  4765  4927  5175
$TRANW   687#  4805  4974  5025  5224  5294
$TRANT   687#  3326  3360  3405
$TSTN      1#  2946  2972  2997  3023  3053  3096  3139  3198  3251  3275  3299  3335  3369  3414
           3473  3526  3579  3632  3686  3746  3776  3806  3836  3867  3917  3956  3995  4034  4073
           4116  4162  4208  4275  4330  4407  4484  4561  4637  4705  4775  4877  4940  5189  5465
           5529  5570  5666
$UPADD     1#  2347
$VARIA     1#   889
$WINDO   687#  3463  3516  3569  3622
$XZ        1#  2938  2944  2964  2970  2989  2995  3015  3021  3045  3051  3088  3094  3131  3137
           3190  3196  3243  3249  3267  3273  3291  3297  3326  3333  3360  3367  3405  3412  3463
           3471  3516  3524  3569  3577  3622  3630  3675  3684  3738  3744  3768  3774  3798  3804
           3828  3834  3858  3865  3909  3915  3948  3954  3987  3993  4026  4032  4065  4071  4106
           4114  4152  4160  4199  4206  4267  4273  4321  4328  4398  4405  4475  4482  4552  4559
           4629  4635  4697  4703  4765  4773  4868  4875  4927  4938  5175  5187  5455  5463  5521
           5527  5558  5568  5655  5664
$ZEROS   687#
$$CMRE   894#   933   934   935   936   937   938
$$CMTM   894#   939   940   941   942   943
$$ESCA   830#
$$NEWT   830#  2948  2974  2999  3025  3055  3098  3141  3200  3253  3277  3301  3337  3371  3416
           3475  3528  3581  3634  3688  3748  3778  3808  3838  3869  3919  3958  3997  4036  4075
           4118  4164  4210  4277  4332  4409  4486  4563  4639  4707  4777  4879  4942  5191  5467
           5531  5572  5668
$$SCOP     1#  1679
$$SET    2204#  2215  2216  2217  2218  2219  2220  2221  2222  2223  2224  2225  2226  2227  2228
           2229
$$SKIP    830#
.EQUAT     1#   720
.HEADE     1#   687
.SETUP     1#
.$ACT1     1#   871
.$APTB     1#   949#
.$APTH     1#  1135
.$APTY     1#  1834
.$CATC     1#
.$CMTA     1#   894
.$EOP      1#  1616
.$ERRO     1#
.$ERRT     1#
.$POWE     1#  2322
.$RDOC     1#  1996
.$READ     1#  1893
.$SCOP     1#  1683
.$TRAP     1#  2181
.$TYPE     1#  1755
.$TYPO     1#

. ABS.  033750    000
```

```
CZKCF    MACY11 30A(1052)  08-JUL-80  08:27  PAGE 142
CZKCF.P11    08-JUL-80 08:24          CROSS REFERENCE TABLE -- MACRO NAMES                    SEQ 0139

 ERRORS DETECTED:  0

 CZKCF,CZKCF/NL:TOC/SOL/CRF_CZKCF.MAC,CZKCF.P11
 RUN-TIME: 29 25 1 SECONDS
 RUN-TIME RATIO: 73/56=1.2
 CORE USED:  53K  (105 PAGES)
```