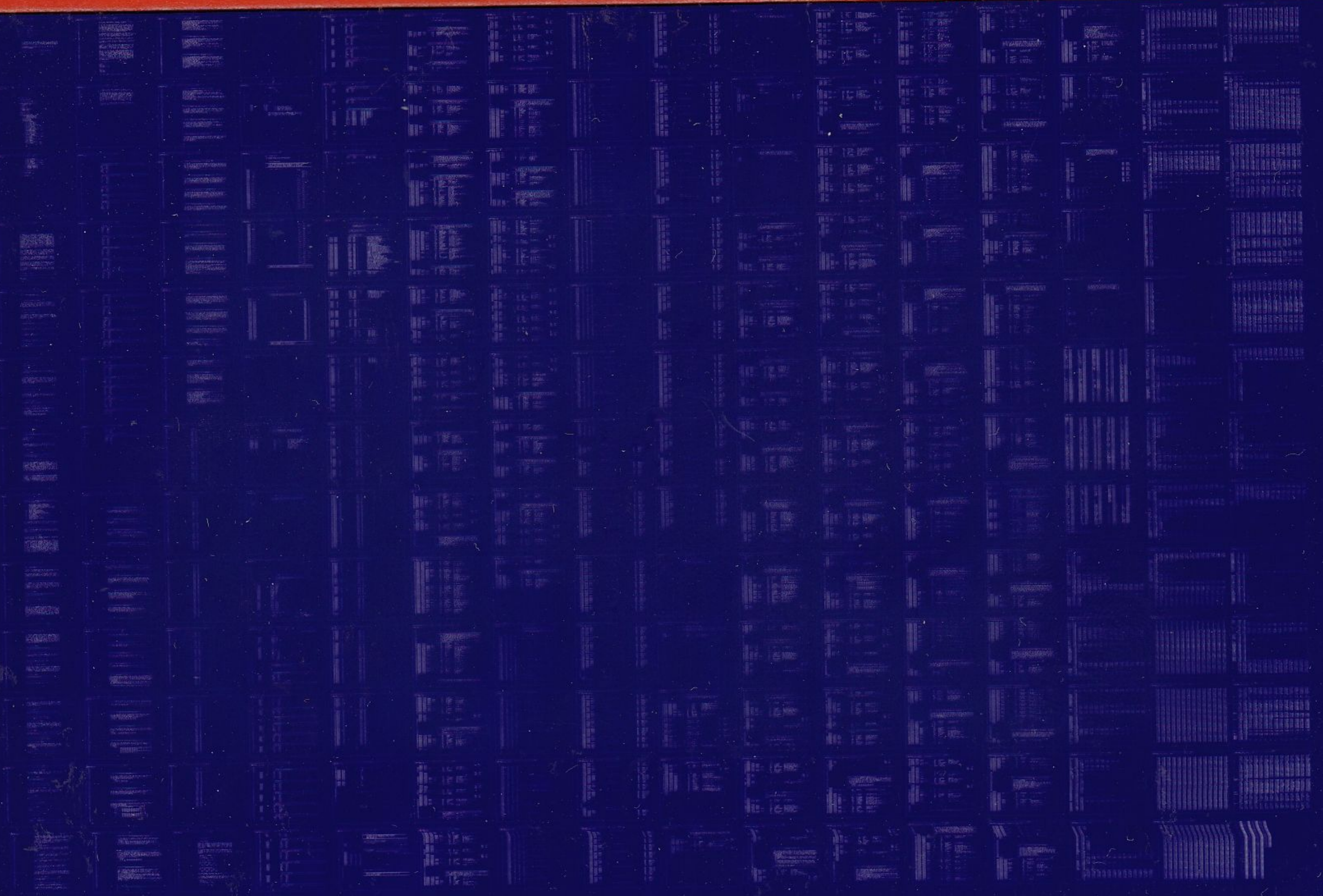


M8203

M8203 STATIC DIAG. # 1
CZDMRB0

AH-E232B-MC
COPYRIGHT 1980
FICHE 1 OF 2

JAN 1980
digital
MADE IN USA



M8203

M8203 STATIC DIAG. # 1
CZDMRB0

AH-E232B-MC

COPYRIGHT 1980
FICHE 2 OF 2

JAN 1980

digital

MADE IN USA

5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39

.REM @

IDENTIFICATION

PRODUCT CODE: AC-E231B-MC
PRODUCT NAME: CZDMRB0 M8203 STATIC DIAG #1
PRODUCT DATE: OCTOBER 1979
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: DAVID HOFFMAN

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

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

COPYRIGHT (C) 1979-BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

CONTENTS

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

- 1.0 INTRODUCTION
- 2.0 HARDWARE REQUIREMENTS
- 3.0 PRELIMINARY PROGRAM REQUIREMENTS
- 4.0 GENERAL PROGRAM CONSIDERATIONS
 - 4.1 DIAGNOSTIC SUPERVISOR
 - 4.2 EXECUTION TIME
 - 4.3 XXDP+
 - 4.4 ACT/SLIDE
 - 4.5 APT
 - 4.6 MEMORY MANAGEMENT
 - 4.7 MEMORY PARITY OPTION
 - 4.8 ERROR LOGGING
- 5.0 PROGRAM LOAD MEDIA
- 6.0 OPERATING INSTRUCTIONS
 - 6.1 LOADING AND STARTING PROCEDURES
 - 6.1.1 LOADING PROCEDURES
 - 6.1.2 STARTING PROCEDURES
 - 6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION
 - 6.2 INITIAL DIALOGUE
 - 6.3 PROGRAM OPTIONS
 - 6.3.1 START COMMAND
 - 6.3.1.1 TESTS SWITCH
 - 6.3.1.2 PASS SWITCH
 - 6.3.1.3 FLAGS SWITCH
 - 6.3.1.4 END OF PASS SWITCH
 - 6.3.1.5 EFFECT OF START COMMAND
 - 6.3.2 RESTART COMMAND
 - 6.3.2.1 TESTS, PASS, AND FLAG SWITCHES
 - 6.3.2.2 UNITS SWITCH
 - 6.3.2.3 EFFECT OF RESTART COMMAND
 - 6.3.3 CONTINUE COMMAND
 - 6.3.3.1 PASS SWITCH
 - 6.3.3.2 FLAGS SWITCH
 - 6.3.3.3 EFFECT OF CONTINUE COMMAND
 - 6.3.4 PROCEED COMMAND
 - 6.3.4.1 FLAGS SWITCH
 - 6.3.4.2 EFFECT OF PROCEED COMMAND
 - 6.3.5 ADD COMMAND
 - 6.3.5.1 UNITS SWITCH
 - 6.3.5.2 EFFECT OF ADD COMMAND
 - 6.3.6 DROP COMMAND
 - 6.3.6.1 UNITS SWITCH
 - 6.3.6.2 EFFECT OF DROP COMMAND
 - 6.3.7 PRINT COMMAND
 - 6.3.7.1 EFFECT OF PRINT COMMAND

58	6.3.8 DISPLAY COMMAND
59	6.3.8.1 UNITS SWITCH
60	6.3.8.2 EFFECT OF DISPLAY COMMAND
61	6.3.9 FLAGS COMMAND
62	6.3.9.1 EFFECT OF FLAGS COMMAND
63	6.3.10 ZFLAGS COMMAND
64	6.3.10.1 EFFECT OF ZFLAGS COMMAND
65	6.3.11 CONTROL CHARACTERS
66	6.3.12 HARDWARE PARAMETERS
67	6.3.13 SOFTWARE PARAMETERS
68	6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE
69	
70	7.0 DEVICE INFORMATION TABLES
71	
72	8.0 TEST DESCRIPTIONS
73	8.1 DATA PATTERNS USED
74	
75	9.0 ERROR INFORMATION
76	9.1 ERROR REPORTING

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

1.0 INTRODUCTION

THE M8203 IS A SINGLE-LINE SYNCHRONOUS LINE UNIT MODULE WHICH SUPPORTS BOTH CHARACTER -ORIENTED (DDCMP, BSC, ETC.) AND BIT-ORIENTED (SDLC, HDLC, ETC.) PROTOCOLS, AND WHICH IS CURRENTLY EMPLOYED IN THE DMP-11 DDCMP MULTIDROP PROJECT. THE PURPOSE OF THIS PROGRAM IS TO PERFORM DIAGNOSTIC TESTING OF ALL M8203 LOGIC IN A RELATIVELY STATIC MANNER. THE FOLLOWING FUNCTIONS WILL BE PERFORMED: LINE UNIT REGISTER ADDRESSING, USYRT ADDRESSING, STATIC BIT INTERACTION AND READ/WRITE LOGIC TESTS, BASIC TRANSMITTER AND RECEIVER SEQUENCING AND DATA BUFFERING AND STATIC OPERATIONS IN CHARACTER AND BIT-STUFFING MODES. IN ADDITION DATA MESSAGES WILL BE SENT AT SPEEDS OF 2400 BAUD TO 1 MEGABAUD, WITH LOOPBACK IN THE USYRT, ON THE LINE UNIT AT TTL LEVEL, OR THROUGH AN EXTERNAL TEST CONNECTOR WITH A SPECIFIC MODEM INTERFACE SELECTED.

THE STATIC LOGIC TESTS WILL PROVIDE EXTENSIVE TROUBLESHOOTING CAPABILITIES, SUCH AS TIGHT SCOPE LOOPS, SWITCH OPTIONS, AND ABILITY TO 'LOCK' ONTO INTERMITTENT ERRORS. IN ADDITION TESTS WILL BE DESIGNED AND STRUCTURED TO ACHIEVE MAXIMUM FAULT RESOLUTION AND FACILITATE REPLACEMENT OF THE SMALLEST FIELD REPLACEABLE UNIT.

THIS PROGRAM WILL BE IMPLEMENTED USING THE DIAGNOSTIC SUPERVISOR AND A STRUCTURED PROGRAMMING APPROACH. BECAUSE THE DESIGN WILL CONFORM TO THE SUPERVISOR (STANDALONE VERSION) THE PROGRAM WILL BE COMPATIBLE WITH ACT, APT, XXDP+, AND SLIDE.

THROUGH DIALOGUE WITH THE OPERATOR, THE PROGRAM WILL ALLOW MODIFICATION OF DEVICE PARAMETERS, SUCH AS UNIBUS ADDRESS, VECTOR ADDRESSES AND DEVICE PRIORITY. IN ADDITION, THE OPERATOR CAN SPECIFY PARTICULAR TESTS TO BE RUN AND A VARIETY OF LOOPING, RUNNING, AND REPORTING MODES.

DEVICE ERRORS WILL BE REPORTED AS THEY OCCUR. THE REPORT WILL INCLUDE A TEST NUMBER AND DESCRIPTION OF THE ERROR, GOOD AND BAD TEST DATA, AND APPLICABLE DEVICE REGISTER CONTENTS.

2.0 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS:

PDP-11/04,05,10,20,30,34,35,40,45,50,60, OR 70
16K MEMORY
CONSOLE TERMINAL

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114

DMC-11 OR KMC-11 MICROPROCESSOR
M8203 LINE UNIT AND BC08S-1 CABLE AND BERG CONNECTORS

3.0 PRELIMINARY PROGRAM REQUIREMENTS

THIS PROGRAM OPERATES THE MICROPROCESSOR EXTENSIVELY IN ORDER TO TEST THE LINE INIT. FOR THIS REASON, THE MICROPROCESSOR DIAGNOSTIC AND SUBSYSTEM FUNCTIONAL TESTS SHOULD BE RUN FIRST, AND ANY FAULTS FOUND IN THE MICROPROCESSOR MODULE SHOULD BE REPAIRED, PRIOR TO RUNNING THE M8203 STATIC LOGIC TESTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 DIAGNOSTIC SUPERVISOR

THIS PROGRAM IS COMPATIBLE WITH THE STANDALONE DIAGNOSTIC SUPERVISOR, AND MUST BE LOADED TO BE CO-RESIDENT WITH THE SUPERVISOR, OR BE PREVIOUSLY COMBINED WITH THE SUPERVISOR AND LOADED AS A SINGLE FILE. IN EITHER CASE, THE COMBINED PROGRAM WILL NOT EXCEED 16K OF MEMORY.

4.2 EXECUTION TIME

THE MAXIMUM TIME REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS IS ABOUT 45 SECONDS PER PASS FOR EACH UNIT.

4.3 XXDP+

THIS PROGRAM MAY BE LOADED UNDER XXDP+, AND MAY BE RUN IN DUMP MODE OR CHAIN MODE.

4.4 ACT/SLIDE

THIS PROGRAM MAY BE LOADED UNDER ACT OR SLIDE AND MAY BE RUN IN DUMP MODE OR CHAIN MODE.

4.5 APT

THIS PROGRAM MAY BE LOADED BY THE APT SYSTEM (INCLUDING APT-RD) AND RUN IN PROGRAM MODE OR SCRIPT MODE.

4.6 MEMORY MANAGEMENT

MEMORY MANAGEMENT IS NOT UTILIZED IN THIS PROGRAM. IF IT IS INSTALLED, IT IS DISABLED BY THE PROGRAM.

115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171

4.7 MEMORY PARITY OPTION

IF PARITY MEMORY IS INSTALLED, MEMORY PARITY TRAPS ARE DISABLED BY THE PROGRAM.

4.8 ERROR LOGGING

AT THE END OF EACH PASS ON ALL UNITS, THE PROGRAM PRINTS OUT THE CUMULATIVE TOTAL NUMBER OF ERRORS SINCE THE LAST START OR RESTART COMMAND.

5.0 PROGRAM LOAD MEDIA

THIS PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM ACT, SLIDE, OR APT SYSTEMS, OR FROM ANY MEDIA SUPPORTED BY XXDP+. WHEN USING THE PAPER TAPE ABSOLUTE LOADER, THE PROGRAM SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC SUPERVISOR. WHEN USING XXDP+, THE DIAGNOSTIC SUPERVISOR SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC PROGRAM.

6.0 OPERATING INSTRUCTIONS

6.1 LOADING AND STARTING PROCEDURES

6.1.1 LOADING PROCEDURES

THIS PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER. IT MAY ALSO BE LOADED FROM ANY XXDP+ LOAD MEDIA. WHEN LOADED UNDER XXDP+, THE DIAGNOSTIC SUPERVISOR WILL BE LOADED AUTOMATICALLY.

6.1.2 STARTING PROCEDURES

THE PROGRAM STARTS AT LOCATION 200. USE STANDARD DEC PROCEDURES TO START THE PROGRAM.

6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION

THE DIAGNOSTIC CAN BE EXECUTED STANDALONE UNDER XXDP+, WITHOUT READING THE REMAINDER OF THIS DOCUMENT, AS FOLLOWS:

- A) LOAD AND START DIAGNOSTIC USING RUN COMMAND
- B) RECEIVE DIAGNOSTIC SUPERVISOR IDENTIFICATION AND PROMPT (DRS-(>))
- C) ENTER STA<CR>
- D) ANSWER HARDWARE AND SOFTWARE QUESTIONS
- E) GET END OF PASS MESSAGES OR ERROR MESSAGES
- F) TO END EXECUTION, ENTER CONTROL/C

172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228

6.2 INITIAL DIALOGUE

AFTER THE PROGRAM AND THE SUPERVISOR ARE LOADED AND THE PROGRAM IS STARTED, THE FOLLOWING IDENTIFICATION IS TYPED :

```
DRS LOADED  
DIAG. RUN-TIME SERVICES  
CZDMR-B-0  
M8203 STATIC LOGIC TESTS - PART 1 OF 2  
UNIT IS M8203  
DR>
```

THE OPERATOR THEN PROCEEDS BY TYPING ONE OR MORE OF THE COMMANDS DESCRIBED IN THE FOLLOWING SECTION 6.3. (FOR MORE DETAILED INFORMATION, REFER TO THE DIAGNOSTIC SUPERVISOR FUNCTIONAL SPECIFICATION).

6.3 PROGRAM OPTIONS

6.3.1 START COMMAND

```
*****  
STA(RT)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:  
  <FLAG-LIST>/EOP:<INCR>  
*****
```

6.3.1.1 TESTS SWITCH (/TESTS:<TEST-LIST>)

<TEST-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (1:2 ETC.) OR RANGES OF DECIMAL NUMBERS (1-5:8-10 ETC.) THAT SPECIFY THE TESTS TO BE EXECUTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS RANGE FROM 1 TO THE LARGEST TEST NUMBER IN THE DIAGNOSTIC. THEY MAY BE SPECIFIED IN ANY ORDER. TESTS WILL BE EXECUTED IN NUMERICAL ORDER REGARDLESS OF THE ORDER OF SPECIFICATION. THE DEFAULT IS TO EXECUTE ALL TESTS. ON THIS AND ALL SWITCHES, THE ANGLE BRACKETS <> ARE PUNCTUATION USED IN THE DEFINITION ONLY, AND ARE NOT TO BE TYPED BY THE OPERATOR. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.2 PASS SWITCH (/PASS:<PASS-CNT>)

<PASS-CNT> IS A DECIMAL NUMBER INDICATING THE DESIRED NUMBER OF PASSES. A PASS IS DEFINED AS THE EXECUTION OF THE FULL DIAGNOSTIC (ALL SELECTED TESTS) AGAINST ALL UNITS SUBMITTED. THE DEFAULT IS NON-ENDING EXECUTION. IN THIS CASE EXIT FROM THE PROGRAM IS ACCOMPLISHED EITHER BY TYPING A CONTROL/C OR BY OCCURANCE OF AN ERROR WITH THE HALT ON ERROR FLAG BEING SET. THE EXIT IS A RETURN TO COMMAND MODE. SEE EXAMPLE AT END OF 6.3.1.5.

229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285

6.3.1.3 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS A SEQUENCE OF ELEMENTS OF THE FORM <FLAG>, <FLAG=1>, OR <FLAG=0>, SEPARATED BY COLONS, WHERE <FLAG> HAS ONE OF THE FOLLOWING VALUES:

- HOE HALT ON ERROR, CAUSING COMMAND MODE TO BE ENTERED WHEN AN ERROR IS ENCOUNTERED
- LOE LOOP ON ERROR, CAUSING THE DIAGNOSTIC TO LOOP CONTINUOUSLY WITHIN THE SMALLEST DEFINED BLOCK OF CODING (SEGMENT, SUBTEST, OR TEST) CONTAINING THE ERROR
- IER INHIBIT ERROR REPORTING
- IBE INHIBIT BASIC ERROR REPORTS
- IXE INHIBIT EXTENDED ERROR REPORTS
- PRI DIRECT ALL MESSAGES TO A LINE PRINTER
- PNT PRINT NUMBER OF TEST BEING EXECUTED
- BOE BELL ON ERROR
- UAM RUN IN UNATTENDED MODE, BYPASSING MANUAL INTERVENTION TESTS
- ISR INHIBIT STATISTICAL REPORTS
- IDU INHIBIT DROPPING OF UNITS BY DIAGNOSTIC
- LOT LOOP ON TEST

THE FLAGS NAMED OR EQUATED TO 1 ARE SET, THOSE EQUATED TO 0 ARE CLEARED. A FLAG NOT SPECIFIED IS CLEARED. IF THE FLAGS SWITCH IS NOT GIVEN ALL FLAGS ARE CLEARED. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.4 END OF PASS SWITCH (/EOP:<INCR>)

<INCR> IS A DECIMAL NUMBER INDICATING HOW OFTEN (IN TERMS OF PASSES) IT IS DESIRED THAT THE END OF PASS MESSAGE BE PRINTED. THE DEFAULT IS AT THE END OF EVERY PASS. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.5 EFFECT OF START COMMAND

THE EFFECT OF THE START COMMAND IS TO INITIATE THE HARDWARE PARAMETER DIALOGUE, THE SOFTWARE PARAMETER DIALOGUE, AND THEN THE DIAGNOSTIC TESTS THEMSELVES.

THE HARDWARE PARAMETER DIALOGUE COMMENCES WITH THE QUESTION '# UNITS?' TO WHICH THE OPERATOR REPLIES WITH A DECIMAL NUMBER N FROM 1 TO 16. THE TERM 'UNIT' REFERS TO THE DEVICE TO WHICH THIS SERIES OF DIAGNOSTICS IS DEDICATED. FOLLOWING THIS ARE THE QUESTIONS WHEREBY THE P-TABLES THEMSELVES WILL BE BUILT. EACH P-TABLE IS A CORE-RESIDENT TABLE CONTAINING ALL THE HARDWARE INFORMATION FOR ONE UNIT. THE OPERATOR MUST SUPPLY N (NUMBER OF UNITS) VALUES FOR EACH QUESTION. HE MAY DO THIS BY GIVING ONE ANSWER TO EACH QUESTION (IN WHICH CASE THE SERIES OF QUESTIONS WILL BE POSED N TIMES) OR BY GIVING N VALUES, SEPARATED BY COMMAS, TO EACH QUESTION (SERIES WILL BE POSED ONCE). EACH QUESTION IS FOLLOWED BY THE RESPONSE RADIX (D FOR DECIMAL, B FOR BINARY, O FOR

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342

OCTAL, L FOR YES/NO) IN PARENTHESES AND THE DEFAULT VALUE AFTER THE PARENTHESES.

FOLLOWING THE HARDWARE QUESTIONS ARE THE SOFTWARE QUESTIONS TO BUILD THE SOFTWARE TABLES, WHICH DEFINE THE MODE (QUICK VERIFY ETC.) THAT THE DIAGNOSTIC WILL EXECUTE IN.

WHEN THE QUESTION '# UNITS?' IS ANSWERED, MEMORY STORAGE IS ALLOCATED FOR THE P-TABLES, AND IF THERE IS NOT ENOUGH TO ACCOMMODATE THEM THE MESSAGE 'TOO MANY UNITS' IS ISSUED. IN THIS CASE THE DIAGNOSTIC MUST BE EXECUTED MORE THAN ONCE TO TEST ALL UNITS.

EXAMPLE:

STA/TESTS:1:2-4:6:8-10/PASS:3/FLAGS:IER:HOE=1:UAM:LOE

THIS COMMAND WILL CAUSE THREE PASSES TO BE MADE, EACH PASS CONSISTING OF TESTS 1,2,3,4,6,8,9, AND 10 EXECUTED AGAINST ALL UNITS. THERE IS NO DIFFERENCE BETWEEN SAYING <FLAG> AND SAYING <FLAG=1>. THE NOTATION <FLAG=0> IS MEANINGFUL ONLY ON A COMMAND OTHER THAN START TO CLEAR A FLAG THAT WAS PREVIOUSLY SET. NOTE THAT ON ALL COMMANDS ONLY THE FIRST THREE LETTERS ARE SCANNED.

6.3.2 RESTART COMMAND

```
*****  
RES(TART)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:  
  <FLAG-LIST>/UNITS:<UNIT-LIST>  
*****
```

6.3.2.1 TESTS, PASS, AND FLAGS SWITCHES

<TEST-LIST>, <PASS-CNT>, AND <FLAG-LIST> ARE AS IN THE START COMMAND.

6.3.2.2 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (0,1 ETC.) OR RANGES OF DECIMAL NUMBERS (0-5, 8-10 ETC.) THAT SPECIFY THE UNITS TO BE TESTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS MAY RANGE FROM 0 THRU N-1 (N IS THE NUMBER OF UNITS SPECIFIED IN THE PREVIOUS START COMMAND). THE NUMBER INDICATES THE POSITION OF THE P-TABLE AS THE DATA WAS ENTERED DURING THE HARDWARE DIAGLOGUE. THE UNITS WHICH ARE SELECTED MUST NOT HAVE BEEN DROPPED BY THE DROP COMMAND. SEE THE DISCUSSION OF ADD AND DROP COMMANDS BELOW. DEFAULT IS TO TEST ALL UNITS WHICH HAVE NOT BEEN DROPPED BY A DROP COMMAND.

6.3.2.3 EFFECT OF RESTART COMMAND

343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399

THE RESTART COMMAND DIFFERS FROM THE START COMMAND IN THAT THE P-TABLES FROM THE PREVIOUS START COMMAND (THERE MUST HAVE BEEN ONE) ARE USED, INSTEAD OF NEW ONES BEING BUILT. THE UNITS SWITCH GIVES THE ABILITY TO SELECT A SUBSET OF THESE. THE SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED (OPERATOR WILL BE ASKED). THE COMMAND CAN BE USED AFTER COMMAND MODE HAS BEEN REENTERED IN ANY OF THE THREE NORMAL WAYS: A) THE REQUESTED NUMBER OF PASSES HAVE BEEN MADE B) AN ERROR WAS ENCOUNTERED WITH THE HALT ON ERROR FLAG SET C) A CONTROL/C WAS ENTERED BY THE OPERATOR.

6.3.3 CONTINUE COMMAND

CON(TINUE)/PASS:<PASS-CNT/FLAGS:<FLAG-LIST>

6.3.3.1 PASS SWITCH (/PASS:<PASS-CNT>)

<PASS-CNT> IS SAME AS IN START COMMAND, BUT THE DEFAULT IS THE UNSATISFIED PASS-CNT FROM THE PREVIOUS START OR RESTART. IF NONE REMAINS, THE DEFAULT IS NON-ENDING EXECUTION.

6.3.3.2 FLAG SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS SAME AS IN START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

6.3.3.3 EFFECT OF CONTINUE COMMAND

CONTINUE MUST FOLLOW A START OR RESTART, AND COMMAND MODE MUST HAVE BEEN ENTERED DUE TO A HALT ON ERROR OR A CONTROL/C. THE EFFECT OF THE COMMAND IS TO GO TO THE BEGINNING OF THE TEST THAT WAS BEING EXECUTED WHEN THE HALT OR CONTROL/C TOOK PLACE. SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

6.3.4 PROCEED COMMAND

PRO(CEED)/FLAGS:<FLAG-LIST>

6.3.4.1 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS AS IN THE START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

6.3.4.2 EFFECT OF PROCEED COMMAND

400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456

PROCEED MUST FOLLOW A START, RESTART, OR CONTINUE. COMMAND MODE MUST HAVE BEEN ENTERED VIA A HALT ON ERROR. THE EFFECT OF THE COMMAND IS TO BEGIN EXECUTION AT THE LOCATION FOLLOWING THE ERROR CALL. NEITHER HARDWARE NOR SOFTWARE PARAMETERS MAY BE ALTERED.

6.3.5 ADD COMMAND

ADD/UNITS:<UNIT-LIST>

6.3.5.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.5.2 EFFECT OF ADD COMMAND

THE UNITS SPECIFIED ARE ADDED TO THE TEST SEQUENCE. EACH UNIT MUST HAVE A P-TABLE IN MEMORY DUE TO AN EARLIER HARDWARE DIALOGUE. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR CONTINUE. THE UNITS SWITCH MUST BE SPECIFIED. THE ADD COMMAND IS MEANINGFUL ONLY FOR UNITS THAT WERE PREVIOUSLY DROPPED.

6.3.6 DROP COMMAND

DRO(P)/UNITS:<UNIT-LIST>

6.3.6.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.6.2 EFFECT OF DROP COMMAND

THE UNITS SPECIFIED WILL BE DROPPED FROM TESTING. THE UNITS WILL BE RESELECTED ONLY BY THE EXECUTION OF AN ADD OR START COMMAND. THE UNITS SWITCH MUST BE ENTERED. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR A CONTINUE COMMAND.

6.3.7 PRINT COMMAND

PRI(NT)

457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513

6.3.7.1 EFFECT OF PRINT COMMAND

THE TOTAL NUMBER OF ERRORS FOR EACH UNIT SINCE THE LAST START OR RESTART COMMAND ARE PRINTED. THE ISR (INHIBIT STATISTICAL REPORTING) FLAG IS CLEARED.

6.3.8 DISPLAY COMMAND

DIS(PLAY)/UNITS:<UNIT-LIST>

6.3.8.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.8.2 EFFECT OF DISPLAY COMMAND

THE HARDWARE P-TABLES FOR ALL UNITS UNDER TEST ARE PRINTED OUT IN THE FORMAT IN WHICH THEY WERE ENTERED. ANY UNITS THAT WERE DROPPED BY THE OPERATOR "DROP" COMMAND ARE SO DESIGNATED.

6.3.9 FLAGS COMMAND

FLA(GS)

6.3.9.1 EFFECT OF FLAGS COMMAND

THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.

6.3.10 ZFLAGS COMMAND

ZFL(AGS)

6.3.10.1 EFFECT OF ZFLAGS COMMAND

ALL FLAGS ARE CLEARED.

6.3.11 CONTROL CHARACTERS

A CONTROL C (C) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES A RETURN TO COMMAND MODE.

A CONTROL Z (Z) ENTERED DURING ONE OF THE THREE OPERATOR

514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570

DIALOGUES- HARD CORE QUESTIONS (SEE 6.2), HARDWARE DIALOGUE (SEE 6.3.1.5), OR SOFTWARE DIALOGUE (SEE 6.3.1.5) CAUSES THE DEFAULTS TO BE TAKEN FOR THE REMAINDER OF THAT DIALOGUE.

A CONTROL 0 (O) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES ALL TELETYPE OUTPUT TO BE SUPPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR UNTIL ANOTHER 0 IS TYPED, WHICH RESTORES NORMAL TELETYPE OUTPUT.

6.3.12 HARDWARE PARAMETERS

THE FOLLOWING 4 QUESTIONS WILL BE ASKED ON A START COMMAND. THE VALUE LOCATED TO THE LEFT OF THE QUESTION MARK IS THE DEFAULT VALUE THAT WILL BE TAKEN ON A CARRIAGE RETURN RESPONSE.

1. DEVICE CSR ADDRESS : (O) 160170?

THIS IS THE ADDRESS AT WHICH THE CSR REGISTERS (SELO) RESIDE ON THE UNIBUS. THE ALLOWABLE RANGE IS 160000-177776 (OCTAL), AND THE DEFAULT VALUE IS 160170.

2. DEVICE VECTOR ADDRESS : (O) 300 ?

THIS IS THE ADDRESS OF THE INPUT INTERRUPT VECTOR FOR THIS DEVICE. THE ALLOWABLE RANGE IS 000-674 (OCTAL), AND THE DEFAULT VALUE IS 300.

3. DEVICE PRIORITY LEVEL : (O) 5 ?

THIS IS THE CPU PRIORITY AT WHICH THE INTERRUPT HANDLERS OF THIS DEVICE WILL BE EXECUTED. THE ALLOWABLE RANGE IS 0-7, AND THE DEFAULT VALUE IS 5.

4. MICROPROCESSOR RUN SWITCH - TYPE 0 IF OFF, 1 IF ON : (O) 1 ?

THIS TELLS THE PROGRAM IF THE RUN SWITCH ON THE MICROPROCESSOR IS SET OR NOT. IF IT IS SET, RUN IS NOT INHIBITED, AND TESTS REQUIRING THE RUN STATE CAN BE EXECUTED. THE ALLOWABLE VALUES ARE 0 AND 1, AND THE DEFAULT VALUE IS 1 (RUN IS NOT INHIBITED).

6.3.13 SOFTWARE PARAMETERS

NO SOFTWARE PARAMETER QUESTIONS ARE ASKED BY PART 1 OF THE STATIC LOGIC TESTS.

6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE

THE FULL CAPABILITY OF THE HARDWARE DIALOGUE IS REVEALED BY THE FOLLOWING DISCUSSION OF WHAT HAPPENS INTERNALLY.

AS SOON AS THE QUESTION "# UNITS?" IS ANSWERED (WITH THE NUMBER N, SAY) SPACE IN CORE IS ALLOCATED FOR N P-TABLES. ALL OF THE P-TABLES ARE OF THE SAME FORMAT, AND THERE IS A

571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627

ONE-TO ONE CORRESPONDENCE BETWEEN THE HARDWARE PARAMETER QUESTIONS AND THE SLOTS IN THE P-TABLE FORMAT.

ON THE FIRST TRIP THRU THE QUESTIONS, ALL OF THE SLOTS IN ALL OF THE P-TABLES ARE FILLED. IF THE OPERATOR TYPES IN LESS THAN N EXPLICIT VALUES IN RESPONSE TO A PARTICULAR QUESTION, THESE VALUES ARE PLACED IN THE P-TABLES (ONE VALUE GOING INTO THE PROPER SLOT OF EACH P-TABLE BEGINNING WITH THE FIRST P-TABLE) UNTIL THE STRING OF VALUES IS EXHAUSTED. THE LAST VALUE IN THE STRING BECOMES THE NEW DEFAULT AND IS USED TO FILL THAT SLOT IN THE REMAINING P-TABLES.

ON SUBSEQUENT TRIPS THRU THE QUESTIONS, THE SAME PROCESS IS CARRIED OUT, EXCEPT THAT THE EARLIEST P-TABLE NOT TO HAVE RECEIVED AN EXPLICIT VALUE IN ANY OF ITS SLOTS NOW ASSUMES THE ROLE THAT TABLE NUMBER ONE PLAYED IN THE FIRST TRIP.

THE SERIES OF QUESTIONS IS REISSUED UNTIL AT LEAST ONE QUESTION HAS RECEIVED N EXPLICIT VALUES FROM THE OPERATOR.

IN GIVING A STRING OF VALUES, COMMAS WITHOUT INTERVENING VALUES MAY BE USED TO INDICATE A REPETITION OF THE LAST NAMED VALUE.

A STRING OF VALUES MAY BE GIVEN AS A RANGE (6-10 FOR EXAMPLE). IF THE VALUES REPRESENT PURE NUMERICAL DATA, THIS SAMPLE RANGE TRANSLATES TO THE STRING 6,7,8,9,10 (AN INCREMENT OF 1). IF THE VALUES ARE ADDRESSES, THE SAMPLE RANGE TRANSLATES TO THE STRING 6,8,10 (AN INCREMENT OF 2).

NOW LET US SEE HOW WE COULD USE THESE CAPABILITIES TO CONSTRUCT A SET OF P-TABLES. ASSUME THAT WE HAVE 16 UNITS, AND THAT THERE ARE THREE HARDWARE PARAMETERS FOR EACH (THREE SLOTS IN THE P-TABLE, THREE HARDWARE QUESTIONS IN THE DIALOGUE). LET THE DESIRED VALUE FOR THE FIRST PARAMETER BE THE NUMBER 75 FOR ALL 16 TABLES. LET THE DESIRED VALUE FOR THE SECOND PARAMETER BE EQUAL TO THE UNIT NUMBER (0,1,2,...,15) EXCEPT FOR UNIT 12, WHICH SHOULD RECEIVE THE VALUE 11. LET THE DESIRED VALUE FOR THE THIRD PARAMETER BE THE NUMBER 76 FOR THE FIRST 7 UNITS AND THE NUMBER 77 FOR THE LAST 9 UNITS.

THE FOLLOWING DIALOGUE WOULD ACCOMPLISH THIS GOAL:

UNITS (D) ? 16
UNIT 0
<QUESTION 1> ? 75
<QUESTION 2> ? 0-6
<QUESTION 3> ? 76

UNIT 7
<QUESTION 1> ?
<QUESTION 2> ? 7-11,,13-15
<QUESTION 3> ? 77

THE FIRST TIME THE SERIES IS ASKED, SLOT ONE RECEIVES A 75 IN ALL 16 TABLES. SLOT TWO RECEIVES THE VALUES 0,1,2,...,6

628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645

IN TABLES 0 THRU 6 AND A CONSTANT 6 IN TABLES 7 THRU 15.
SLOT THREE RECEIVES A CONSTANT 76 IN ALL 16 TABLES.

THE SECOND TIME THRU THE SERIES, TABLES 7 THRU THE END ARE
GOING TO BE AFFECTED (NOTE THAT THIS PIECE OF INFORMATION IS
PRINTED OUT FOR THE OPERATOR IN THE FORM 'UNIT XX' AT
THE BEGINNING OF EACH SERIES). QUESTION 1 IS RESPONDED TO
BY A <CR>, SO SLOT ONE STAYS AT CONSTANT 75 IN TABLES 7
THRU 15, SINCE NO NEW EXPLICIT VALUES ARE TYPED IN. SLOT TWO
GETS THE VALUES 7,8,9,10,11 IN TABLES 7 THRU 11, AND
GETS AN 11 IN SLOT 12, AND GETS THE VALUES 13,14,15 IN
TABLES 13 THRU 15. SLOT THREE GETS THE VALUE 77 IN TABLES 7
THRU 15.

THE DIALOGUE IS TERMINATED WHEN THE SOFTWARE RECOGNIZES THAT
16 EXPLICIT VALUES HAVE BEEN GIVEN FOR AT LEAST ONE QUESTION
(NAMELY QUESTION 2).

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

7.0 DEVICE INFORMATION TABLES

```
*****  
* MAINTENANCE REGISTER - BSEL1  
*****  
RUN      = BIT7  
MCLR     = BIT6  
STEPLU   = BIT4  
LULOOK   = BIT3  
ROMO     = BIT2  
ROMI     = BIT1  
STEPMP   = BIT0  
  
*****  
* OBUS REG 10 - TRANSMITTER BUFFER  
*****  
TX7      = BIT7  
TX6      = BIT6  
TX5      = BIT5  
TX4      = BIT4  
TX3      = BIT3  
TX2      = BIT2  
TX1      = BIT1  
TX0      = BIT0  
  
*****  
* OBUS REG 11  
*****  
OC       = BIT7  
GOAH    = BIT3  
ABORT   = BIT2  
EOM     = BIT1  
SOM     = BIT0  
  
*****  
* OBUS REG 12  
*****  
IC       = BIT7  
BPOLL   = BIT6  
LULP    = BIT5  
  
*****  
* OBUS REG 13  
*****  
POLL    = BIT7  
DTR     = BIT6  
SELFR   = BIT5  
HDX     = BIT4  
MAINT1  = BIT3  
MAINT2  = BIT2  
SELSBY  = BIT1  
  
*****
```

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114

```
* OBUS REG 14
:*****
TXEN   = BIT6
DISSI  = BIT5
RDAX   = BIT4
WAX    = BIT3
ENAX   = BIT2
AX2    = BIT1
AX1    = BIT0
```

```
* OBUS REG 17
:*****
CRC2   = BIT7
CRC1   = BIT6
IDLE   = BIT5
SECA   = BIT4
STRIP  = BIT3
RDALL  = BIT2
IERR   = BIT1
DDCMP  = BIT0
```

```
* IBUS REG 10 - RECEIVER BUFFER
:*****
RX7    = BIT7
RX6    = BIT6
RX5    = BIT5
RX4    = BIT4
RX3    = BIT3
RX2    = BIT2
RX1    = BIT1
RX0    = BIT0
```

```
* IBUS REG 11
:*****
OC     = BIT7
OACT   = BIT6
SW3    = BIT5
ORDY   = BIT4
SW2    = BIT3
SW1    = BIT2
SW0    = BIT1
UNRR   = BIT0
```

```
* IBUS REG 12
:*****
IC     = BIT7
IACT   = BIT6
LULP   = BIT5
IRDY   = BIT4
OVR    = BIT3
RAB    = BIT2
EBLK   = BIT1
BCC    = BIT0
```

115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171

```
*****  
* IBUS REG 13  
*****  
RING      = BIT7  
DTR       = BIT6  
RTS       = BIT5  
HDX       = BIT4  
MODR      = BIT3  
CS        = BIT2  
STBY      = BIT1  
CARR      = BIT0  
  
*****  
* IBUS REG 14  
*****  
READY     = BIT7  
TXEN      = BIT6  
DISSI     = BIT5  
RDAX      = BIT4  
WAX       = BIT3  
ENAX      = BIT2  
AX2       = BIT1  
AX1       = BIT0  
  
*****  
* IBUS REG 17  
*****  
SIGR      = BIT7  
SIGQ      = BIT6  
TXDATA    = BIT5  
OCOR      = BIT4  
ICIR      = BIT3  
TESTMD    = BIT2  
MCLK      = BIT1  
DDCMP     = BIT0  
  
*****  
* AX0-15 - USYRT REG 0 (READ ONLY)  
*****  
RX7       = BIT7  
RX6       = BIT6  
RX5       = BIT5  
RX4       = BIT4  
RX3       = BIT3  
RX2       = BIT2  
RX1       = BIT1  
RX0       = BIT0  
  
*****  
* AX0-16 - USYRT REG 1 (READ ONLY)  
*****  
RERR      = BIT7  
ASBC2     = BIT6  
ASBC1     = BIT5  
ASBC0     = BIT4  
ROR       = BIT3
```

172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228

RABT = BIT2
REOM = BIT1
RSOM = BIT0

* AX1-15 - USYRT REG 2

TX7 = BIT7
TX6 = BIT6
TX5 = BIT5
TX4 = BIT4
TX3 = BIT3
TX2 = BIT2
TX1 = BIT1
TX0 = BIT0

* AX1-16 - USYRT REG 3

TERR = BIT7
TXGA = BIT3
TXAB = BIT2
TEOM = BIT1
TSOM = BIT0

* AX2-15 - USYRT REG 4

SYN7 = BIT7
SYN6 = BIT6
SYN5 = BIT5
SYN4 = BIT4
SYN3 = BIT3
SYN2 = BIT2
SYN1 = BIT1
SYN0 = BIT0
SYNCH = 226

* AX2-16 - USYRT REG 5

APA = BIT7
DDC = BIT6
STR = BIT5
SEC = BIT4
IDL = BIT3
CRCTY2 = BIT2
CRCTY1 = BIT1
CRCTY0 = BIT0

* AX3-15 - USYRT REG 6

I422 = BIT7
XYZ = BIT6
C32BCC = BIT5
V35 = BIT4

229 INTGRL = BIT3
230 C32ENB = BIT2
231 OP = BIT1
232 TEST = BIT0
233 AX315U = I422!XYZ!C32BCC!V35!INTGRL!OP

* AX3-16 - USYRT REG 7

237 *****
238 TXLEN2 = BIT7
239 TXLEN1 = BIT6
240 TXLEN0 = BIT5
241 RXLEN2 = BIT2
242 RXLEN1 = BIT1
243 RXLEN0 = BIT0
244
245
246
247
248

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

8.0 TEST DESCRIPTIONS

```
*****  
; TEST 1 - MICROPROCESSOR CSR ADDRESSING TEST (SELO)  
*  
* THIS TEST ADDRESSES THE FIRST MICROPROCESSOR CSR (SELO), TO MAKE SURE  
* THAT A NON-EXISTENT MEMORY TIME-OUT TRAP DOES NOT OCCUR WHILE  
* ATTEMPTING TO ADDRESS THE MICROPROCESSOR.  
*****
```

```
*****  
; TEST 2 - INBUS/OUTBUS REG 14 INITIALIZATION TEST  
*  
* MASTER CLEAR (MCLR) IS SET IN THE MICROPROCESSOR, IBUS REG 14 IS READ  
* AND COMPARED TO 200.  
*****
```

```
*****  
; TEST 3 - INBUS/OUTBUS REG 14 READ/WRITE BIT TEST  
*  
* WRITE, READ, AND COMPARE ALL WORDS OF DATA PATTERN A INTO REG 14,  
* A BYTE AT A TIME. NON-R/W BITS ARE MASKED OFF TO 0 BEFORE WRITING AND  
* READING.  
* DATA PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177.  
*****
```

```
*****  
; TEST 4 - REG 14 MASTER CLEAR TEST  
* WRITE 377 INTO REG 14, ISSUE MASTER CLEAR, READ REG 14 AND COMPARE  
* TO 200.  
*****
```

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114

```
*****  
TEST 5 - REG 14 UNIBUS RESET (INIT) TEST  
* WRITE 377 INTO REG 14, ISSUE UNIBUS RESET (INIT), READ REG 14 AND COMPARE  
* TO 200.  
*****
```

```
*****  
TEST 6 - LINE UNIT FALSE SELECTION TEST  
*  
* FIRST, A MASTER CLEAR IS PERFORMED. THEN, THE PROGRAM SINGLE-STEPS THE  
* MICROPROCESSOR THROUGH AN INSTRUCTION WHICH LOADS 041 (OCT) INTO THE MAR  
* REGISTER (OBUS* ADRS 14). THEN, THE LINE UNIT REGISTER 14 IS READ AND CHECKED  
* TO BE UNAFFECTED (STILL = 0). THIS TEST IS INTENDED TO DETECT A FALSE  
* SELECTION OF THE LINE UNIT REGISTERS, WHEN THE LINE UNIT IS NOT BEING  
* ACCESSED.  
*****
```

```
*****  
TEST 7 - INBUS REG MASTER CLEAR TEST  
*  
* FIRST, ALL READ/WRITE BITS OF REGS 10-17 ARE SET BY LOADING A  
* DIFFERENT WORD OF PATTERN G INTO EACH REG. THEN,  
* A MASTER CLEAR IS ISSUED AND EACH REG IS READ AND COMPARED TO A WORD OF  
* PATTERN M, WHICH CONTAINS THE INITIALIZED STATES OF THE REGS. (UNPREDICTABLE  
* BITS ARE MASKED OFF TO 0 BEFORE COMPARISON).  
* PATTERN G = 000,000,240,120,177,000,000,001  
* PATTERN M = 000,020,000,000,200,000,000,051  
*****
```

```
*****  
TEST 8 - REGISTER 10-17 ADDRESSING TEST  
*  
* FIRST, A MASTER CLEAR IS ISSUED. THEN,  
* WRITE A DIFFERENT WORD OF DATA PATTERN B INTO EACH OF REGS 10-17,  
* AND AFTER EACH WRITE, READ AND COMPARE ALL REGS TO EXPECTED VALUES.  
* UNPREDICTABLE BITS ARE MASKED OFF TO 0 WHEN READ FOR COMPARISON.  
* PATTERN B = 000,000,040,100,220,000,000,051  
*****
```

```
*****  
TEST 9 - REG 11 READ/WRITE BIT TEST  
*****
```


115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171

```
*  
* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN C INTO REG 11 :  
* DATA PATTERN C = 020,020,020.  
;*****  
  
;*****  
TEST 10 - REG 12 READ/WRITE BIT TEST  
*  
* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN D INTO REG 12 :  
* DATA PATTERN D = 000,040,000.  
;*****  
  
;*****  
TEST 11 - REG 13 READ/WRITE BIT TEST  
*  
* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN E INTO REG 13 :  
* DATA PATTERN E = 000,120,020,100,120,000.  
;*****  
  
;*****  
TEST 12 - REG 17 READ/WRITE BIT TEST  
*  
* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN F INTO REG 17 :  
* DATA PATTERN F = 050,051,050.  
;*****  
  
;*****  
TEST 13 - MAINTENANCE CLOCK BIT TEST  
*  
* FIRST, A MASTER CLEAR IS ISSUED TO INIT ALL REGS. THEN, THE MICROPROCESSOR  
* IS PLACED IN A LOOP ON AN INSTRUCTION, BY SETTING THE INSTRUCTION IN SEL6  
* AND SETTING ROMI AND RUN IN BSEL1. THE INSTRUCTION IS ONE WHICH REPETITIVELY  
* READS LINE UNIT REG 17 INTO BSEL2. THE PDP-11 CAN THEN SCAN BSEL2 TO MONITOR  
* THE MAINTENANCE CLOCK BIT, MCLK. THE FOLLOWING SEQUENCE IS THEN PERFORMED  
* TO MONITOR MCLK :  
* - THE PROGRAM REPEATEDLY CHECKS THE MCLK BIT FOR THE 1 STATE, AND IF IT IS  
* NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC (DEPENDING ON THE PROCESSOR)  
* AN ERROR IS REPORTED. (THE MAINTENANCE CLOCK HAS A PERIOD OF 41.6 MICRO-  
* SEC).  
* - THE PROGRAM NEXT REPEATEDLY CHECKS THE MCLK BIT FOR THE 0 STATE, AND IF  
* IT IS NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC AN ERROR IS REPORTED.
```

172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228

* - THE PROGRAM NEXT REPEATEDLY CHECKS MCLK BIT FOR THE 1 STATE AGAIN, AND
* IF IT IS NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC, AN ERROR IS REPORTED.
*
* IF THE P-TABLE FOR THIS UNIT INDICATES THAT THE MICROPROCESSOR RUN SWITCH
* IS OFF, THE TEST WILL BE SKIPPED.
:*****

:*****
* TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST
*
* FIRST, ALL READ/WRITE BITS OF EXTENDED REGS AX0-AX3 ARE SET BY LOADING
* A DIFFERENT WORD OF PATTERN H INTO EACH REG. THEN, A MASTER CLEAR IS
* ISSUED AND EACH REG IS READ AND COMPARED TO A WORD OF PATTERN I, WHICH
* CONTAINS THE INITIALIZED STATES OF ALL THE EXTENDED REGS.
* PATTERN H = 000,000,377,017,377,377,375,377
* PATTERN I = 000,000,000,000,000,000,103,000,000
:*****

:*****
* TEST 15 - EXTENDED REGISTER ADDRESSING TEST
*
* FIRST, ISSUE A MASTER CLEAR TO PUT REGS INTO INITIALIZED STATES SHOWN IN
* PATTERN I. THEN, WRITE A DIFFERENT WORD OF PATTERN J INTO EACH EXTENDED (AX)
* REG, AND AFTER EACH WRITE, READ AND COMPARE ALL EXTENDED REGS TO EXPECTED
* VALUES.
* PATTERN I = 000,000,000,000,000,000,103,000,000
* PATTERN J = 000,000,001,002,004,103,040,100
:*****

:*****
* TEST 16 - REGS 15,16 / AX2-15,AX2-16 READ/WRITE BIT TEST
*
* USING REGS 15,16, THE INDIRECT REGS AX2-15,AX2-16 (USYRT REGS 4,5) ARE
* WRITTEN AND READ USING EACH WORD OF PATTERN K. AX2-15 IS COMPARED
* TO THE WORD WRITTEN, AND AX2-16 IS ALWAYS COMPARED TO 103. (AX2-16 IS NOT
* WRITEABLE).
* PATTERN K =
* FOR REG 15: 000,377,125,252,001,002,004,010,020,040,100,200,000,000,
* 000,000,000,000,000,000,000,376,375,373,367,357,337,277,177,
* 377,377,377,377,377,377,377,377
* FOR REG 16: 000,377,125,252,000,000,000,000,000,000,000,000,001,002,
* 004,010,020,040,100,200,377,377,377,377,377,377,377,377,
* 376,375,373,367,357,337,277,177.
:*****

229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285

```
*****  
TEST 17 - AX0-15,AX0-16 READ/WRITE BIT TEST  
*  
* IN THIS TEST, A MASTER CLEAR IS DONE, AND THEN A WRITE, READ, AND COMPARE  
* ARE PERFORMED IN REGS AX0-15,AX0-16 USING EACH WORD OF PATTERN L.  
* ANY BITS IN AX0-15,AX0-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)  
* IN THE EXPECTED VALUE BEFORE COMPARISON.  
* PATTERN L =  
*   FOR REG 15: 000,377,000  
*   FOR REG 16: 000,377,000.  
*****
```

```
*****  
TEST 18 - AX1-15,AX1-16 READ/WRITE BIT TEST  
*  
* IN THIS TEST, A MASTER CLEAR IS DONE, AND THEN A WRITE, READ, AND COMPARE  
* ARE PERFORMED IN REGS AX1-15,AX1-16 USING EACH WORD OF PATTERN K.  
* ANY BITS IN AX1-15,AX1-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)  
* IN THE EXPECTED VALUE BEFORE COMPARISON.  
*****
```

```
*****  
TEST 19 - AX3-15,AX3-16 READ/WRITE BIT TEST  
*  
* IN THIS TEST A MASTER CLEAR IS DONE AND THEN A WRITE, READ, AND COMPARE ARE  
* PERFORMED IN REGS AX3-15,AX3-16 USING EACH WORD OF PATTERN V FOR WRITING,  
* AND PATTERN U FOR COMPARING.  
* ANY BITS IN AX3-15,AX3-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)  
* IN THE EXPECTED VALUE BEFORE COMPARISON.  
* PATTERN V =  
*   FOR REG 15 : 000,333,331,323,313,233,133,000,000,000,000,  
*               000,000,000,000,000,000,000,000  
*   FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,  
*               100,200,346,345,343,307,247,147  
* PATTERN U =  
*   FOR REG 15 : 000,001,013,011,021,101,301,000,000,000,000,  
*               000,000,000,000,000,000,000,000  
*   FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,  
*               100,200,346,345,343,307,247,147  
*****
```

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342

```
*****  
; TEST 20 - REG 17 - AX2-16 READ/WRITE, MASTER CLEAR TEST  
*  
* THIS TEST CONSISTS OF 2 SUBTESTS. IN THE FIRST SUBTEST, EACH BYTE OF PAT O  
* IS WRITTEN INTO REG 17 AND AFTER EACH WRITE, AX2-16 IS READ AND COMPARED  
* TO A BYTE OF PAT P.  
* PATTERN O = 000,041,004,010,020,040,100,101,200,201,300,111,301,375  
* PATTERN P = 000,113,200,040,020,010,001,104,007,105,007,144,107,157  
* IN THE SECOND SUBTEST, REG 17 IS LOADED WITH 375, A MASTER CLEAR IS ISSUED,  
* AND AX2-16 IS COMPARED TO ITS INITIALIZED STATE (103).  
*****
```

```
*****  
; TEST 21 - TRANSMITTER BUFFER DATA TEST  
* A MASTER CLEAR IS DONE FIRST, AND THEN A BYTE OF PATTERN N IS LOADED INTO  
* REG 11 AND THE NEXT BYTE IS LOADED TWICE INTO REG 10. THE PROGRAM THEN WAITS  
* AT LEAST 50 MICRO-SEC, AND THEN IT READS AND COMPARES AX1-15 TO THE BYTE WHICH  
* WAS LOADED INTO REG 10, AND IT READS AND COMPARES AX1-16 TO THE BYTE WHICH WAS  
* LOADED INTO REG 11. THIS PROCESS IS REPEATED (INCLUDING THE MASTER CLEAR)  
* FOR EACH PAIR OF BYTES IN PATTERN N.  
* PATTERN N =  
* FOR REG 10: 000,125,252,377,000,000,000  
* FOR REG 11: 000,000,000,000,005,012,017  
*****
```

```
*****  
; TEST 22 - TRANSMITTER BUFFER SEQUENCING TEST  
*  
* FIRST, A MASTER CLEAR IS DONE, AND THE PROGRAM CHECKS FOR ORDY=1, OCOR=0.  
* THEN, 2 TSOM CHARS ARE LOADED INTO THE TX SILO, AND ALLOWED TO RIPPLE  
* DOWN TO THE OUTPUT. THE PROGRAM CHECKS FOR ORDY=1, OCOR=1.  
* NEXT, THE PROGRAM CYCLES THE STEPLU BIT UNTIL OCOR=0 AGAIN, AND CHECKS FOR  
* THIS TO OCCUR WITHIN 3 CYCLES.  
* THE SILO IS THEN FILLED WITH 64 BYTES OF A 256-BYTE BINARY COUNT PATTERN  
* (000-377) AND THE PROGRAM CHECKS FOR ORDY=0 AFTER THE 64TH CHAR IS LOADED.  
* THE PROGRAM CYCLES STEPLU FOR 8 CYCLES AND CHECKS THAT AFTER THE 8TH, ORDY=1.  
* AX1-15 IS READ AND COMPARED TO EXPECTED DATA.  
* THE REST OF THE BINARY COUNT DATA BYTES ARE LOADED, CYCLED 8 CLOCKS, READ AND  
* COMPARED, A BYTE AT A TIME. UPON COMPLETION, THE SILO IS CHECKED TO BE EMPTY  
* WITH ORDY=1, OCOR=0.  
*****
```

```
*****  
; TEST 23 - TX MSG TIMING TEST, CHAR MODE, WITH CRC  
*
```

343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399

* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)
* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS
* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN
* CHARACTER ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND CRC-16.
* THE FOLLOWING STEPS ARE DONE:
* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN CHAR MODE.
* SOM IS SET TWICE TO SEND 2 SYNCH CHARS. THEN, 2 000 CHARS ARE SENT, AND
* THEN 2 TERMINATING SYNCHS ARE SENT.
* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE
* CLEARED STATE AFTER THE 3RD SYNCH COMPLETES.
;*****

;*****
TEST 24 - TX MSG TIMING TEST, BIT MODE, WITH CRC
*
* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)
* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS
* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN
* BIT ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND CRC-CCITT-1.
* THE FOLLOWING STEPS ARE DONE:
* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN BIT MODE.
* SOM IS SET TWICE TO SEND 2 FLAG CHARS. THEN, 2 000 CHARS ARE SENT, AND
* THEN 2 TERMINATING FLAGS ARE SENT.
* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE
* CLEARED STATE.
;*****

;*****
TEST 25 - TX MSG TIMING TEST, CHAR MODE, WITH NO CRC
*
* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)
* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS
* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN
* CHARACTER ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND NO ERROR CHECKING.
* THE FOLLOWING STEPS ARE DONE:
* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN CHAR MODE.
* SOM IS SET TWICE TO SEND 2 SYNCH CHARS. THEN, 2 000 CHARS ARE SENT, AND
* THEN 2 TERMINATING SYNCHS ARE SENT.
* THE TEST IS PERFORMED WITH TXEN (REG 14, BIT6) SET, AND THE PROGRAM CHECKS
* THAT THIS HOLDS RTS HIGH PAST THE END OF THE MESSAGE.
* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE
* CLEARED STATE.
;*****

;*****

400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456

TEST 26 - TX UNDERRUN SET AND CLEAR TEST - CHAR MODE

*
* IN THIS TEST, A TX UNDERRUN ERROR IS FORCED IN EACH OF 2 SITUATIONS,
* AND THEN CLEARED DIFFERENTLY IN EACH.
* IN THE FIRST, A MESSAGE IS INITIATED, A 000 CHAR IS SENT, AND THE TX
* BUFFER IS NOT SERVICED IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG,
* WHICH CAUSES UNRR TO SET IN REG 11. THEN, SOM IS SET TO CLEAR THE ERROR,
* AND THIS IS VERIFIED.
* IN THE SECOND SITUATION, A MSG IS INITIATED, A 000 CHAR IS SENT, AND THE TX
* BUFFER IS NOT SERVICED IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG, WHICH
* AGAIN CAUSES UNRR TO SET. THEN, A MASTER CLEAR IS DONE, AND THE UNRR BIT
* IS CHECKED TO BE CLEARED.
;*****

TEST 27 - TRANSMIT CHAR LENGTH TIMING TEST - CHAR MODE, CRC

*
* THE LINE UNIT IS PLACED IN CHAR MODE (DDCMP) AND A MESSAGE IS INITIATED
* WITH AN 8-BIT SYNCH AND A 5-BIT SYNCH CHAR. NEXT, A 000 CHAR IS SENT WITH
* EACH OF THE FOLLOWING TX CHAR LENGTHS : 5 BITS, 6 BITS, 7 BITS, 8 BITS.
* (FOR EXAMPLE, A 5-BIT CHAR REQUIRES 5 CLOCK CYCLES TO BE TRANSMITTED). TWO
* TERMINATING SYNCHS ARE SENT AFTER THE DATA.
;*****

TEST 28 - TRANSMIT CHAR LENGTH TIMING TEST - BIT MODE, CRC

*
* THE LINE UNIT IS PLACED IN BIT MODE AND A MESSAGE IS INITIATED
* WITH 2 FLAG CHARS. NEXT, 2 8-BIT 000 CHARS ARE SENT, FOLLOWED BY 000 CHARS
* WITH EACH OF THE FOLLOWING TRANSMITTER CHAR LENGTHS:
* 1 BIT, 2 BITS, 3 BITS, 4 BITS, 5 BITS, 6 BITS, 7 BITS, AND 8 BITS.
* (FOR EXAMPLE, A 5-BIT CHAR REQUIRES 5 CLOCK CYCLES TO BE TRANSMITTED).
* TWO TERMINATING FLAGS ARE SENT AFTER THE DATA.
;*****

TEST 29 - TXDATA BIT TEST - CHAR MODE, NO CRC

*
* THE LINE UNIT IS INITIALIZED AND A MSG IS INITIATED (USING STEPLU) WITH CRC-
* 16 SELECTED IN CHAR MODE. TWO SYNCHS, 000,125,252,377,000, AND 2 TERMINATING
* SYNCHS ARE THEN SENT. THE PROGRAM CHECKS EACH BIT OF THE TRANSMITTED
* DATA CHARS, BY MONITORING TXDATA (REG 17) AS THE DATA IS CLOCKED OUT OF
* THE USYRT TRANSMITTER.
;*****

457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513

```
*****  
TEST 30 - USYRT RECEIVER MSG TEST - CHAR MODE, CRC  
*  
* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH  
* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH CRC-16  
* SELECTED. TWO SYNCHS, 000,125,252,377,000, AND FOUR TERMINATING SYNCHS ARE  
* SENT. THE PROGRAM MONITORS IACT, AND THE RCV'D CHARS AND CRC BYTES ARE READ  
* FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR IACT  
* STILL SET AFTER THE MESSAGE.  
*****
```

```
*****  
TEST 31 - USYRT RECEIVER MSG TEST - BIT MODE, CRC  
*  
* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH  
* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH CRC-  
* CCITT-1. TWO FLAGS, 000,125,252,377,000, AND TWO TERMINATING FLAGS ARE THEN  
* SENT. THE PROGRAM MONITORS IACT, RSOM, AND THE RCV'D CHARS ARE READ  
* FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR  
* IACT = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT STILL = 0.  
*****
```

```
*****  
TEST 32 - USYRT RECEIVER MSG TEST - CHAR MODE, NO CRC  
*  
* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH  
* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH NO  
* ERROR DETECTION. TWO SYNCHS, 000,125,252,377,000, AND TWO SYNCHS ARE  
* THEN SENT. THE PROGRAM MONITORS IACT, AND THE RECEIVED CHARS ARE READ FROM  
* AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR IACT  
* STILL = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT = 0.  
*****
```

```
*****  
TEST 33 - USYRT RECEIVER MSG TEST - BIT MODE, NO CRC  
*  
* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH  
* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH ERROR  
* DETECTION INHIBITED. TWO FLAGS, 000,125,252,377,000, AND TWO TERMINATING FLAGS  
* ARE THEN SENT. THE PROGRAM MONITORS IACT, RSOM, AND THE RCV'D CHARS ARE  
* READ FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR
```

514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570

* IACT = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT STILL = 0.
;*****

;*****
TEST 34 - SILO-DISABLED TRANSMITTER LOAD TEST
*
* THIS TEST DISABLES THE SILOS, LOADS A 125 CHARACTER INTO THE TX SILO, AND
* READS AX1-15 AND CHECKS THAT THE DATA DID NOT GET LOADED INTO THE USYRT TX
* BUFFER.
;*****

;*****
TEST 35 - SILO-DISABLED MESSAGE TEST - BIT MODE, NO CRC
*
* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU)
* WITH LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, WITH SILO
* DISABLE SET, AND WITH NO ERROR DETECTION. TWO FLAGS, 000,125,252, AND
* TERMINATING FLAGS ARE THEN SENT BY LOADING THE TRANSMITTED CHARS INTO
* REG AX1. THE PROGRAM MONITORS OACT, IACT, RSOM, REOM, ORDY, OCOR, ICIR,
* IRDY, AND THE RECEIVED CHARS ARE READ FROM AX0 AND COMPARED TO EXPECTED
* VALUES.
;*****

;*****
TEST 36 - RECEIVER BUFFER TEST - CHAR MODE, CRC
*
* FIRST, A MASTER CLEAR IS DONE AND THE PROGRAM CHECKS FOR ICIR = 1 AND IRDY
* = 0. THEN, 2 SOM CHARS ARE LOADED AND CLOCKED INTO THE USYRT, AND 64
* BYTES OF A 256-BYTE BINARY COUNT DATA PATTERN (000-377) ARE LOADED INTO
* THE TX SILO.
* THE LINE UNIT IS THEN CLOCKED UNTIL IRDY = 1, AND THE PROGRAM CHECKS FOR
* THIS TO OCCUR WITHIN 40-43 CYCLES. THE PROGRAM READS THE RCV SILO, CHECKS THE
* CHAR FOR 000, AND CHECKS FOR IRDY = 0 AGAIN.
* THE LINE UNIT IS THEN CLOCKED IN GROUPS OF 8 CYCLES, AND AFTER EACH, THE
* PROGRAM CHECKS FOR ICIR = 1, IRDY = 1, UNTIL THE 64TH GROUP, AFTER WHICH
* IT CHECKS FOR ICIR = 0, IRDY = 1. THE SECOND DATA CHAR IS READ FROM THE
* RECEIVER SILO AND COMPARED TO 001. THEN, THE PROGRAM CHECKS FOR ICIR = 1,
* IRDY = 1 AGAIN.
* THE REST OF THE BINARY COUNT DATA BYTES ARE CYCLED 8 CLOCKS AND READ AND
* COMPARED A BYTE AT A TIME.
;*****

571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627

```
*****  
; TEST 37 - RECEIVER CHAR LENGTH TIMING TEST - CHAR MODE, NO CRC  
*  
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH NO ERROR DETECTION, AND A MSG IS  
* INITIATED WITH 2 SYNCH CHARS. NEXT, FIFTEEN 000 CHARS ARE LOADED INTO THE  
* TRANSMITTER SILO. THE LINE UNIT IS THEN CLOCKED USING STEPLU WITH LULOOP  
* SET, WHILE THE RECEIVER CHAR LENGTH IS SET TO THE FOLLOWING VALUES : 5,6,7,8.  
* FOR EACH RCV CHAR LENGTH, THE PROGRAM CHECKS TO MAKE SURE THET USYRT RECEIVER  
* FLAGS OCCUR THE PROPER NO. OF CYCLES APART, FOR EACH RCV CHAR LENGTH.  
* (FOR EXAMPLE A 5-BIT CHAR TAKES 5 CLOCK CYCLES TO BE RECEIVED). A MASTER  
* CLEAR IS THEN DONE TO TERMINATE THE OPERATION.  
*****
```

```
*****  
; TEST 38 - RECEIVER CHAR LENGTH TIMING TEST - BIT MODE, NO CRC  
*  
* THE LINE UNIT IS PLACED IN BIT MODE WITH NO ERROR DETECTION, AND A MESSAGE IS  
* INITIATED WITH 2 FLAG CHARS. NEXT, FIFTEEN 000 CHARS ARE LOADED INTO THE  
* TRANSMITTER SILO. THE LINE UNIT IS THEN CLOCKED USING STEPLU WITH LULOOP  
* SET, WHILE THE RCV CHAR LENGTH IS SET TO THE FOLLOWING VALUES : 8,8,8,7,6,5,  
* 4,3,2,1. FOR EACH RCV CHAR LENGTH, THE PROGRAM CHECKS TO MAKE SURE THAT THE  
* USYRT RECEIVER FLAGS OCCUR THE PROPER NO. OF CYCLES APART, FOR EACH RCV  
* CHAR LENGTH. (FOR EXAMPLE, A 5 BIT CHAR TAKES 5 CLOCK CYCLES TO BE RECEIVED).  
* A MASTER CLEAR IS THEN DONE TO TERMINATE THE OPERATION.  
*****
```

```
*****  
; TEST 39 - TRANSMITTER UNDERRUN ERROR, IDLE MARKING, CHAR MODE, NO CRC  
*  
* THE LINE UNIT IS PLACED IN CHAR MODE, AND THE IDLE BIT IS SET. THEN, A  
* MSG IS INITIATED, A 000 CHAR IS SENT, AND THE TX BUFFER IS NOT SERVICED  
* IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG, WHICH CAUSES A TX UNDERRUN  
* ERROR. THEN, THE RECEIVER IS CLOCKED AND CHECKED FOR TWO 377 CHARS TO BE  
* RECEIVED (LINE MARKING) BEFORE SHUTTING DOWN WITH A MASTER CLEAR.  
*****
```

```
*****  
; TEST 40 - MSG TERMINATION WITH GA CHARS - BIT MODE, NO CRC  
*  
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS  
* INITIATED IN BIT MODE.  
* 2 FLAG CHARACTERS ARE SENT, FOLLOWED BY  
* THE FOLLOWING DATA CHARACTERS : 000, 125, 252, 377, 000. THEN THE LOOP  
* MODE BIT (STRIP) IS SET AND 2 TERMINATING GO-AHEAD CHARACTERS ARE
```

628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684

* SENT. EACH USYRT RCV FLAG IS TIMED AS IT IS RECEIVED, AND THE 5 DATA
* WORDS ARE READ AND COMPARED TO EXPECTED VALUES.
* ALSO, THE FIRST GA CHAR IS CHECKED BY SCANNING THE TXDATA BIT AS THE GA
* IS BEING TRANSMITTED (GA = 376 OCTAL).
* THE TEST ALSO CHECKS FOR SETTING OF RAB AND EBLK
* IN LOOP MODE.

TEST 41 - IDLE SYNCHS TEST - CHAR MODE

*
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
* INITIATED IN CHAR MODE. 24(DEC) SYNCHS ARE SENT.
* EACH SYNCH IS TIMED AS IT IS RECEIVED, AND THE BITS ARE CHECKED
* FOR A VALID SYNCH CHAR FOR EACH OF THE 22 SYNCHS WHICH FOLLOW
* THE FIRST TWO (THESE PERFORM SYNCHRONIZATION, AND ARE NOT READ).
* WHILE THE LAST SYNCH IS BEING TRANSMITTED, OC IS SET, AND THE
* NEXT CHAR RCV'D AFTER THE SYNCH IS CHECKED TO BE 377 (LINE MARKING).
* THEN, A MASTER CLEAR IS ISSUED.
* THE SYNCH CHAR USED IS 226 (OCTAL).

TEST 42 - STRIP SYNCH TEST

*
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
* INITIATED IN CHAR MODE AND WITH THE STRIP SYNCH
* BIT SET. THEN 24 (DEC) SYNCHS ARE SENT
* FOLLOWED BY THE FOLLOWING DATA CHARACTERS : 377, 000, 125, 252,
* AND THEN 2 TERMINATING SYNCHS.
* EACH OF THE 23 SYNCHS AFTER THE FIRST ARE CHECKED AT THE TRANSMITTER OUTPUT,
* BY SCANNING THE TXDATA BIT.
* EACH USYRT RCV FLAG IS TIMED AS IT IS RECEIVED, AND THE 4 DATA WORDS
* ARE READ AND COMPARED TO EXPECTED VALUES.
* FINALLY, THE LINE UNIT IS CLOCKED FOR SEVERAL CHAR TIMES, AND A CHECK
* IS MADE FOR OACT = 0 (TEQM SHOULD CAUSE TX ENABLE TO DROP).
* THE ABOVE TEST IS REPEATED FOR EACH OF THE FOLLOWING SYNCH CHAR DATA
* PATTERNS : 226,000,125,252,376,177.

8.1 DATA PATTERNS USED

***** DATA PATTERN A *****

685	
686	
687	
688	
689	
690	
691	
692	
693	
694	
695	
696	
697	
698	
699	
700	
701	
702	
703	
704	
705	
706	
707	
708	
709	
710	
711	
712	
713	
714	
715	
716	
717	
718	
719	
720	
721	
722	
723	
724	
725	
726	
727	
728	
729	
730	
731	
732	
733	
734	
735	
736	
737	
738	
739	
740	
741	

PATA:

.BYTE	125
.BYTE	252
.BYTE	000
.BYTE	377
.BYTE	001
.BYTE	002
.BYTE	004
.BYTE	010
.BYTE	020
.BYTE	040
.BYTE	100
.BYTE	200
.BYTE	376
.BYTE	375
.BYTE	373
.BYTE	367
.BYTE	357
.BYTE	337
.BYTE	277
.BYTE	177

***** DATA PATTERN B *****

PATB:

.BYTE	000
.BYTE	000
.BYTE	040
.BYTE	100
.BYTE	220
.BYTE	000
.BYTE	000
.BYTE	051

***** DATA PATTERN C *****

PATC:

.BYTE	020
.BYTE	020
.BYTE	020

***** DATA PATTERN D *****

PATD:

.BYTE	000
.BYTE	040
.BYTE	000

***** DATA PATTERN E *****

PATE:

.BYTE	000
.BYTE	120
.BYTE	020
.BYTE	100
.BYTE	120
.BYTE	000

***** DATA PATTERN F *****

PATF:

.BYTE	050
-------	-----

```
742          .BYTE 051
743          .BYTE 050
744
745          ***** DATA PATTERN G *****
746          PATG:
747              .BYTE 000
748              .BYTE 000
749              .BYTE 240
750              .BYTE 120
751              .BYTE 177
752              .BYTE 000
753              .BYTE 000
754              .BYTE 001
755
756          ***** DATA PATTERN H *****
757          PATH:
758              .BYTE 000
759              .BYTE 000
760              .BYTE 377
761              .BYTE 017
762              .BYTE 377
763              .BYTE 377
764              .BYTE 375
765              .BYTE 377
766
767          ***** DATA PATTERN I *****
768          PATI:
769              .BYTE 000
770              .BYTE 000
771              .BYTE 000
772              .BYTE 000
773              .BYTE 000
774              .BYTE 103
775              .BYTE 000
776              .BYTE 000
777
778          ***** DATA PATTERN J *****
779          PATJ:
780              .BYTE 000
781              .BYTE 000
782              .BYTE 010
783              .BYTE 002
784              .BYTE 004
785              .BYTE 103
786              .BYTE 001
787              .BYTE 100
788
789          ***** DATA PATTERN K *****
790          PATK:
791              .BYTE 000
792              .BYTE 000
793              .BYTE 377
794              .BYTE 377
795              .BYTE 125
796              .BYTE 125
797              .BYTE 252
798              .BYTE 252
799              .BYTE 001
```

799	.BYTE	000
800	.BYTE	002
801	.BYTE	000
802	.BYTE	004
803	.BYTE	000
804	.BYTE	010
805	.BYTE	000
806	.BYTE	020
807	.BYTE	000
808	.BYTE	040
809	.BYTE	000
810	.BYTE	100
811	.BYTE	000
812	.BYTE	200
813	.BYTE	000
814	.BYTE	000
815	.BYTE	001
816	.BYTE	000
817	.BYTE	002
818	.BYTE	000
819	.BYTE	004
820	.BYTE	000
821	.BYTE	010
822	.BYTE	000
823	.BYTE	020
824	.BYTE	000
825	.BYTE	040
826	.BYTE	000
827	.BYTE	100
828	.BYTE	000
829	.BYTE	200
830	.BYTE	376
831	.BYTE	377
832	.BYTE	375
833	.BYTE	377
834	.BYTE	373
835	.BYTE	377
836	.BYTE	367
837	.BYTE	377
838	.BYTE	357
839	.BYTE	377
840	.BYTE	337
841	.BYTE	377
842	.BYTE	277
843	.BYTE	377
844	.BYTE	177
845	.BYTE	377
846	.BYTE	377
847	.BYTE	376
848	.BYTE	377
849	.BYTE	375
850	.BYTE	377
851	.BYTE	373
852	.BYTE	377
853	.BYTE	367
854	.BYTE	377
855	.BYTE	357

856	.BYTE	377
857	.BYTE	337
858	.BYTE	377
859	.BYTE	277
860	.BYTE	377
861	.BYTE	177
862		
863	***** DATA PATTERN L *****	
864	PATL:	
865	.BYTE	000
866	.BYTE	000
867	.BYTE	377
868	.BYTE	377
869	.BYTE	000
870	.BYTE	000
871		
872	***** DATA PATTERN M *****	
873	PATM:	
874	.BYTE	000
875	.BYTE	020
876	.BYTE	000
877	.BYTE	000
878	.BYTE	200
879	.BYTE	000
880	.BYTE	000
881	.BYTE	051
882		
883	***** DATA PATTERN N *****	
884	PATN:	
885	.BYTE	000
886	.BYTE	000
887	.BYTE	000
888	.BYTE	125
889	.BYTE	000
890	.BYTE	252
891	.BYTE	000
892	.BYTE	377
893	.BYTE	005
894	.BYTE	000
895	.BYTE	012
896	.BYTE	000
897	.BYTE	017
898	.BYTE	000
899		
900	***** DATA PATTERN O *****	
901	PATO:	
902	.BYTE	000
903	.BYTE	041
904	.BYTE	004
905	.BYTE	010
906	.BYTE	020
907	.BYTE	040
908	.BYTE	100
909	.BYTE	101
910	.BYTE	200
911	.BYTE	201
912	.BYTE	300

913	.BYTE	111
914	.BYTE	301
915	.BYTE	375

***** DATA PATTERN P *****

917	PATP:	
918		
919	.BYTE	000
920	.BYTE	113
921	.BYTE	200
922	.BYTE	040
923	.BYTE	020
924	.BYTE	010
925	.BYTE	001
926	.BYTE	104
927	.BYTE	007
928	.BYTE	105
929	.BYTE	007
930	.BYTE	144
931	.BYTE	107
932	.BYTE	157

***** DATA PATTERN U *****

933	PATU:	
934		
935	.BYTE	000
936	.BYTE	000
937	.BYTE	001
938	.BYTE	000
939	.BYTE	013
940	.BYTE	000
941	.BYTE	011
942	.BYTE	000
943	.BYTE	021
944	.BYTE	000
945	.BYTE	101
946	.BYTE	000
947	.BYTE	301
948	.BYTE	000
949	.BYTE	000
950	.BYTE	001
951	.BYTE	000
952	.BYTE	002
953	.BYTE	000
954	.BYTE	004
955	.BYTE	000
956	.BYTE	040
957	.BYTE	000
958	.BYTE	100
959	.BYTE	000
960	.BYTE	200
961	.BYTE	000
962	.BYTE	346
963	.BYTE	000
964	.BYTE	345
965	.BYTE	000
966	.BYTE	343
967	.BYTE	000
968	.BYTE	307
969	.BYTE	000

970	.BYTE	247
971	.BYTE	000
972	.BYTE	147

***** DATA PATTERN V *****

973		
974	PATV:	.BYTE 000
975		.BYTE 000
976		.BYTE 333
977		.BYTE 000
978		.BYTE 331
979		.BYTE 000
980		.BYTE 323
981		.BYTE 000
982		.BYTE 313
983		.BYTE 000
984		.BYTE 233
985		.BYTE 000
986		.BYTE 133
987		.BYTE 000
988		.BYTE 000
989		.BYTE 001
990		.BYTE 000
991		.BYTE 002
992		.BYTE 000
993		.BYTE 004
994		.BYTE 000
995		.BYTE 040
996		.BYTE 000
997		.BYTE 100
998		.BYTE 000
999		.BYTE 200
1000		.BYTE 000
1001		.BYTE 346
1002		.BYTE 000
1003		.BYTE 345
1004		.BYTE 000
1005		.BYTE 343
1006		.BYTE 000
1007		.BYTE 307
1008		.BYTE 000
1009		.BYTE 247
1010		.BYTE 000
1011		.BYTE 147
1012		
1013		
1014		
1015		
1016		
1017		
1018		

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

9.0 ERROR INFORMATION

9.1 ERROR REPORTING

ERRORS ARE REPORTED BY THE PROGRAM AS THEY OCCUR (IF NOT INHIBITED). THE REPORT CONFORMS TO THE DIAGNOSTIC SUPERVISOR ERROR REPORT FORMAT, AND CONSISTS OF A DESCRIPTION OF THE ERROR, THE TEST NUMBER, SUBTEST NUMBER, PC OF THE ERROR CALL, DEVICE ADDRESS, AND BASIC AND EXTENDED ERROR INFORMATION.

THE FOLLOWING EXAMPLE PROVIDES A TYPICAL ERROR REPORT, WHICH DESCRIBES AN 'IRDY NOT SET' ERROR, AND PROVIDES THE PC OF THE ERROR CALL AND THE PC OF THE CALL TO THE SUBROUTINE REPORTING IT, THE FAILING REGISTER NAME, AND DEVICE REGISTER CONTENTS :

CZDMR DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC: 006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

FAILING REG: INBUS/OUTBUS REG 12

LINE UNIT INBUS REGS:
REG10 REG11 REG12 REG13
000 120 000 257
REG14 REG15 REG16 REG17
024 377 377 035

LINE UNIT EXTENDED REGS:
AX0-15 AX0-16 AX1-15 AX1-16
000 000 000 000
AX2-15 AX2-16 AX3-15 AX3-16
000 000 000 000

FOR OTHER ERRORS, THE REPORT MAY BE MORE EXTENSIVE, AND REQUIRE ADDITIONAL DATA TO BE REPORTED.

IF EXTENDED ERROR INFORMATION HAD BEEN INHIBITED USING THE IXE FLAG PRIOR TO RUNNING THE TEST, THE ABOVE ERROR WOULD HAVE BEEN REPORTED IN THE FOLLOWING SHORTENED FORM :

CZDMR DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC:006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

FAILING REG: INBUS/OUTBUS REG 12

58
59
60
61
62
63
64
65

@

1
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

002000 .TITLE CZDMRB M8203 STATIC DIAG #1
 .=2000

002000 .MCALL SVC ; INITIALIZE SUPERVISOR MACROS
 SVC

002000 BGNMOD LU1MOD

000001 \$LSTIN= 1
000001 \$LSTTAG= 1
000001 SVCINS= 1 ; LIST INSTRUCTIONS, SHIFTED RIGHT
000001 SVCTST= 1 ; LIST TEST TAGS, SHIFTED RIGHT
000001 SVCSUB= 1 ; LIST SUBTEST TAGS, SHIFTED RIGHT
000001 SVCGBL= 1 ; LIST GLOBAL TAGS, SHIFTED RIGHT
000001 SVCTAG= 1 ; LIST OTHER TAGS, SHIFTED RIGHT

; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.

1
2
3
4
5
6
7 002000
8
10
11
12
13
14
16
17 002000
002000
002000 103
002001 132
002002 104
002003 115
002004 122
002005 000
002006 000
002007 000
002010
002010 102
002011
002011 060
002012
002012 000000
002014
002014 000055
002016
002016 035530
002020
002020 000000
002022
002022 002252
002024
002024 000000
002026
002026 036206
002030
002030 000000
002032
002032 000000
002034
002034 000000
002036
002036 000000
002040
002040 002124
002042
002042 000000
002044
002044 000000
002046

.SBTTL PROGRAM HEADER

:++
: THE PROGRAM HEADER IS THE INTERFACE BETWEEN
: THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
:--

POINTER BGNAU,BGNDU

:XX
: IF ANY OPTIONAL POINTERS ARE TO BE USED IN THE 'HEADER', CHANGE
: 'POINTER' TO CONTAIN THE CORRECT ARGUMENTS. IF ALL OPTIONAL
: POINTERS ARE TO BE USED, CHANGE 'POINTER' TO BE 'POINTER ALL'.
:XX

HEADER CZDMR,B,0,45.,0

L\$NAME::
.ASCII /C/
.ASCII /Z/
.ASCII /D/
.ASCII /M/
.ASCII /R/
.BYTE 0
.BYTE 0
.BYTE 0
L\$REV::
.ASCII /B/
L\$DEPO::
.ASCII /0/
L\$UNIT::
.WORD 0
L\$TIML::
.WORD 45.
L\$HPCP::
.WORD L\$HARD
L\$SPCP::
.WORD 0
L\$HPTP::
.WORD L\$HW
L\$SPTP::
.WORD 0
L\$LADP::
.WORD L\$LAST
L\$STA::
.WORD 0
L\$CO::
.WORD 0
L\$DTYP::
.WORD 0
L\$APT::
.WORD 0
L\$DTP::
.WORD L\$DISPATCH
L\$PRIO::
.WORD 0
L\$ENVI::
.WORD 0
L\$EXP1::

002046 000000
002050
002050 003
002051 003
002052
002052 000000
002054 000000
002056
002056 000000
002060
002060 003462
002062
002062 000000
002064
002064 000000
002066
002066 000000
002070
002070 021606
002072
002072 021524
002074
002074 000000
002076
002076 003470
002100
002100 104035
002102
002102 000000
002104
002104 021072
002106
002106 021522
002110
002110 021442
002112
002112 021064
002114
002114 000000
002116
002116 000000
002120
002120 000000

LSMRE:: .WORD 0
 .BYTE C\$REVISION
 .BYTE C\$EDIT
L\$EF:: .WORD 0
 .WORD 0
L\$SPC:: .WORD 0
L\$DEVP:: .WORD L\$DVTYP
L\$REPP:: .WORD 0
L\$EXP4:: .WORD 0
L\$EXP5:: .WORD 0
L\$AUT:: .WORD L\$AU
L\$DUT:: .WORD L\$DU
L\$LUN:: .WORD 0
L\$DESP:: .WORD L\$DESC
L\$LOAD:: EMT E\$LOAD
L\$ETP:: .WORD 0
L\$I\$CP:: .WORD L\$I\$INIT
L\$I\$CCP:: .WORD L\$I\$CLEAN
L\$I\$ACP:: .WORD L\$I\$AUTO
L\$I\$PRT:: .WORD L\$I\$PROT
L\$I\$TEST:: .WORD 0
L\$I\$DLY:: .WORD 0
L\$I\$HIME:: .WORD 0

18
20
21
22
24
25
26
27
28
29
30
31

;
: CHANGE THE 'HEADER' TO CONTAIN THE PROPER ARGUMENTS.
;
.EVEN

16
17
18
19
20

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20 002302

.SBTTL GLOBAL EQUATES SECTION

:/
:/ THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
:/ ARE USED IN MORE THAN ONE TEST.
:/

EQUALS

:
: BIT DIFINITIONS

100000	BIT15== 100000
040000	BIT14== 40000
020000	BIT13== 20000
010000	BIT12== 10000
004000	BIT11== 4000
002000	BIT10== 2000
001000	BIT09== 1000
000400	BIT08== 400
000200	BIT07== 200
000100	BIT06== 100
000040	BIT05== 40
000020	BIT04== 20
000010	BIT03== 10
000004	BIT02== 4
000002	BIT01== 2
000001	BIT00== 1

001000	BIT9== BIT09
000400	BIT8== BIT08
000200	BIT7== BIT07
000100	BIT6== BIT06
000040	BIT5== BIT05
000020	BIT4== BIT04
000010	BIT3== BIT03
000004	BIT2== BIT02
000002	BIT1== BIT01
000001	BIT0== BIT00

:
: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

000040	EF.START== 32.	: START COMMAND WAS ISSUED
000037	EF.RESTART== 31.	: RESTART COMMAND WAS ISSUED
000036	EF.CONTINUE== 30.	: CONTINUE COMMAND WAS ISSUED

000035
000034

EF.NEW== 29.
EF.PWR== 28.

; A NEW PASS HAS BEEN STARTED
; A POWER-FAIL/POWER-UP OCCURRED

.....
; PRIORITY LEVEL DEFINITIONS
.....

000340
000300
000240
000200
000140
000100
000040
000000

PRI07== 340
PRI06== 300
PRI05== 240
PRI04== 200
PRI03== 140
PRI02== 100
PRI01== 40
PRI00== 0

.....
; OPERATOR FLAG BITS
.....

000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

EVL== 4
LOT== 10
ADR== 20
IDU== 40
ISR== 100
UAM== 200
BOE== 400
PNT== 1000
PRI== 2000
IXE== 4000
IBE== 10000
IER== 20000
LOE== 40000
HOE== 100000

21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46

.....
; * PROGRAM EVENT FLAG DEFINITIONS
; *

.....
; * MAINTENANCE REGISTER - BSEL1
; *

000200
000100
000020
000010
000004
000002
000001

RUN = BIT7
MCLR = BIT6
STEPLU = BIT4
LULOOP = BIT3
ROMO = BIT2
ROMI = BIT1
STEPMP = BIT0

.....
; * OBUS REG 10 - TRANSMITTER BUFFER
; *

47	000200	TX7	=	BIT7
48	000100	TX6	=	BIT6
49	000040	TX5	=	BIT5
50	000020	TX4	=	BIT4
51	000010	TX3	=	BIT3
52	000004	TX2	=	BIT2
53	000002	TX1	=	BIT1
54	000001	TX0	=	BIT0

;* OBUS REG 11

58	000200	OC	=	BIT7
60	000010	GOAH	=	BIT3
61	000004	ABORT	=	BIT2
62	000002	EOM	=	BIT1
63	000001	SOM	=	BIT0

;* OBUS REG 12

68	000200	IC	=	BIT7
69	000100	BPOLL	=	BIT6
70	000040	LULP	=	BIT5

;* OBUS REG 13

75	000200	POLL	=	BIT7
76	000100	DTR	=	BIT6
77	000040	SELFR	=	BIT5
78	000020	HDX	=	BIT4
79	000010	MAINT1	=	BIT3
80	000004	MAINT2	=	BIT2
81	000002	SELSBY	=	BIT1

;* OBUS REG 14

86	000100	TXEN	=	BIT6
87	000040	DISSI	=	BIT5
88	000020	RDAX	=	BIT4
89	000010	WAX	=	BIT3
90	000004	ENAX	=	BIT2
91	000002	AX2	=	BIT1
92	000001	AX1	=	BIT0

;* OBUS REG 17

97	000200	CRC2	=	BIT7
98	000100	CRC1	=	BIT6
99	000040	IDLE	=	BIT5
100	000020	SECA	=	BIT4
101	000010	STRIP	=	BIT3
102	000004	RDALL	=	BIT2
103	000002	IERR	=	BIT1

104 000001 DDCMP = BIT0
105
106
107
108
109 000200
110 000100
111 000040
112 000020
113 000010
114 000004
115 000002
116 000001

;* IBUS REG 10 - RECEIVER BUFFER

RX7 = BIT7
RX6 = BIT6
RX5 = BIT5
RX4 = BIT4
RX3 = BIT3
RX2 = BIT2
RX1 = BIT1
RX0 = BIT0

117
118
119
120
121 000200
122 000100
123 000040
124 000020
125 000010
126 000004
127 000002
128 000001

;* IBUS REG 11

OC = BIT7
OACT = BIT6
SW3 = BIT5
ORDY = BIT4
SW2 = BIT3
SW1 = BIT2
SW0 = BIT1
UNRR = BIT0

129
130
131
132
133 000200
134 000100
135 000040
136 000020
137 000010
138 000004
139 000002
140 000001

;* IBUS REG 12

IC = BIT7
IACT = BIT6
LULP = BIT5
IRDY = BIT4
OVRR = BIT3
RAB = BIT2
EBLK = BIT1
BCC = BIT0

141
142
143
144
145 000200
146 000100
147 000040
148 000020
149 000010
150 000004
151 000002
152 000001

;* IBUS REG 13

RING = BIT7
DTR = BIT6
RTS = BIT5
HDX = BIT4
MODR = BIT3
CS = BIT2
STBY = BIT1
CARR = BIT0

153
154
155
156
157 000200
158 000100
159 000040
160 000020

;* IBUS REG 14

READY = BIT7
TXEN = BIT6
DISSI = BIT5
RDAX = BIT4

161	000010	WAX	=	BIT3
162	000004	ENAX	=	BIT2
163	000002	AX2	=	BIT1
164	000001	AX1	=	BIT0

165
166
167
:*****
:* IBUS REG 17
:*****

168		SIGR	=	BIT7
169	000200	SIGQ	=	BIT6
170	000100	TXDATA	=	BIT5
171	000040	OCOR	=	BIT4
172	000020	ICIR	=	BIT3
173	000010	TESTMD	=	BIT2
174	000004	MCLK	=	BIT1
175	000002	DDCMP	=	BIT0
176	000001			

177
178
179
:*****
:* AX0-15 - USYRT REG 0 (READ ONLY)
:*****

180		RX7	=	BIT7
181	000200	RX6	=	BIT6
182	000100	RX5	=	BIT5
183	000040	RX4	=	BIT4
184	000020	RX3	=	BIT3
185	000010	RX2	=	BIT2
186	000004	RX1	=	BIT1
187	000002	RX0	=	BIT0
188	000001			

189
190
191
:*****
:* AX0-16 - USYRT REG 1 (READ ONLY)
:*****

192		RERR	=	BIT7
193	000200	ASBC2	=	BIT6
194	000100	ASBC1	=	BIT5
195	000040	ASBC0	=	BIT4
196	000020	ROR	=	BIT3
197	000010	RAET	=	BIT2
198	000004	REOM	=	BIT1
199	000002	RSOM	=	BIT0
200	000001			

201
202
203
:*****
:* AX1-15 - USYRT REG 2
:*****

204		TX7	=	BIT7
205	000200	TX6	=	BIT6
206	000100	TX5	=	BIT5
207	000040	TX4	=	BIT4
208	000020	TX3	=	BIT3
209	000010	TX2	=	BIT2
210	000004	TX1	=	BIT1
211	000002	TX0	=	BIT0
212	000001			

213
214
215
:*****
:* AX1-16 - USYRT REG 3
:*****

216		TERR	=	BIT7
217	000200			

```
218      000010      TXGA      = BIT3
219      000004      TXAB      = BIT2
220      000002      TEOM      = BIT1
221      000001      TSOM      = BIT0
222
223      :*****
224      :* AX2-15 - USYRT REG 4
225      :*****
226      000200      SYN7      = BIT7
227      000100      SYN6      = BIT6
228      000040      SYN5      = BIT5
229      000020      SYN4      = BIT4
230      000010      SYN3      = BIT3
231      000004      SYN2      = BIT2
232      000002      SYN1      = BIT1
233      000001      SYN0      = BIT0
234      000226      SYNCH     = 226
235
236      :*****
237      :* AX2-16 - USYRT REG 5
238      :*****
239      000200      APA       = BIT7
240      000100      DDC       = BIT6
241      000040      STR       = BIT5
242      000020      SEC       = BIT4
243      000010      IDL       = BIT3
244      000004      CRCTY2    = BIT2
245      000002      CRCTY1    = BIT1
246      000001      CRCTY0    = BIT0
247
248      :*****
249      :* AX3-15 - USYRT REG 6
250      :*****
251      000200      I422      = BIT7
252      000100      XYZ       = BIT6
253      000040      C32BCC    = BIT5
254      000020      V35       = BIT4
255      000010      INTGRL    = BIT3
256      000004      C32ENB    = BIT2
257      000002      OP        = BIT1
258      000001      TEST      = BIT0
259      000372      AX315U    = I422!XYZ!C32BCC!V35!INTGRL!OP
260
261      :*****
262      :* AX3-16 - USYRT REG 7
263      :*****
264      000200      TXLEN2    = BIT7
265      000100      TXLEN1    = BIT6
266      000040      TXLEN0    = BIT5
267      000004      RXLEN2    = BIT2
268      000002      RXLEN1    = BIT1
269      000001      RXLEN0    = BIT0
270
271
272
273
274
```

```
275 :*****
276 :* TX CONTROL BITS DEFINED ON WORD BASIS
277 :*****
278 004000 TXGOA = BIT11
279 002000 TXABT = BIT10
280 001000 TXEOM = BIT9
281 000400 TXSOM = BIT8
282
283
284
285
286
287 :*****
288 :* RCV CONTROL BITS DEFINED ON WORD BASIS
289 :*****
290 004000 RXOVR = BIT11
291 002000 RXABT = BIT10
292 001000 RXEBL = BIT9
293 000400 RXBCC = BIT8
294
295
296
297
298 :*****
299 :* ADDRESS EQUATES FOR REGISTER STORAGE TABLE (LUREG:)
300 :*****
301 002302 LUR10 = LUREG+0 ;LINE UNIT IBUS REG 10
302 002304 LUR11 = LUREG+2 ;LINE UNIT IBUS REG 11
303 002306 LUR12 = LUREG+4 ;LINE UNIT IBUS REG 12
304 002310 LUR13 = LUREG+6 ;LINE UNIT IBUS REG 13
305 002312 LUR14 = LUREG+10 ;LINE UNIT IBUS REG 14
306 002314 LUR15 = LUREG+12 ;LINE UNIT IBUS REG 15
307 002316 LUR16 = LUREG+14 ;LINE UNIT IBUS REG 16
308 002320 LUR17 = LUREG+16 ;LINE UNIT IBUS REG 17
309 002322 AX0.15 = LUREG+20 ;USYRT REG 0
310 002324 AX0.16 = LUREG+22 ;USYRT REG 1
311 002326 AX1.15 = LUREG+24 ;USYRT REG 2
312 002330 AX1.16 = LUREG+26 ;USYRT REG 3
313 002332 AX2.15 = LUREG+30 ;USYRT REG 4
314 002334 AX2.16 = LUREG+32 ;USYRT REG 5
315 002336 AX3.15 = LUREG+34 ;USYRT REG 6
316 002340 AX3.16 = LUREG+36 ;USYRT REG 7
317
318
319
320
321
322 100000 CHPCHK = BIT15
323
324 100000 BCCCHK = BIT15
325 100000 CRCCHK = BIT15
326
327
328
329
330
331 :*****
```


332
333
334 021000
335 122000
336 121000
337
338
339
340
341 000001
342 000002
343
344
345
346
347

;* MICROINSTRUCTION DEFINITIONS

MVIOX = 021000 :MOVE IBUS TO OBUS*
MVIXO = 122000 :MOVE IBUS* TO OBUS
MVIXOX = 121000 :MOVE IBUS* TO OBUS*

***** ERROR1 BIT FLAG DEFINITIONS *****

RRDYTO = BIT0
WRDYTO = BIT1

```
1      .SBTTL  GLOBAL DATA SECTION
2
3      :///////////////////////////////////////////////////
4      :/      THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
5      :/      IN MORE THAN ONE TEST.
6      :///////////////////////////////////////////////////
7
8      :*****
9      :* STORAGE FOR DEVICE REGISTERS
10     :*****
11     002302  LUREG: .BLKW 16.
12
13     :*****
14     :* MISCELLANEOUS STORAGE
15     :*****
16     002342  000000  SCRACH: .WORD 0          ;GEN'L PURPOSE SCRATCH WORD
17     002344  000000  LOGDEV: .WORD 0         ;LOGICAL DEVICE NUMBER
18     002346  000000  PSTACK: .WORD 0        ;CONTAINS BASE LEVEL PROGRAM STACK POINTER
19     002350  000000  PRIOR: .WORD 0        ;CPU PRIORITY FOR PRINTOUT
20     002352  000000  SUBRPC: .WORD 0       ;PC OF SUBR CALL FOR ERROR REPORTS
21     002354  000000  INTFLG: .WORD 0      ;INTERRUPT RECEIVED FLAGS
22                                     ; BIT 0 FOR TX, BIT 1 FOR RCV
23     002356  000000  ERRFLG: .WORD 0      ;SUBROUTINE ERROR FLAG
24     002360  000000  TIMFLG: .WORD 0      ;EVENT TIME-OUT FLAG
25     002362  000000  RETADR: .WORD 0      ;SUBR ERROR RETURN ADDRESS
26     002364  000000  REDBYT: .WORD 0      ;LO BYTE CONTAINS BYTE READ FROM LU REG
27     002366  000000  WRIBYT: .WORD 0      ;LO BYTE CONTAINS BYTE TO LOAD INTO LU REG
28     002370  000000  RAX15: .WORD 0       ;LO BYTE CONTAINS BYTE READ FROM REG 15
29     002372  000000  RAX16: .WORD 0       ;LO BYTE CONTAINS BYTE READ FROM REG 16
30     002374  000000  WAX15: .WORD 0       ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 15
31     002376  000000  WAX16: .WORD 0       ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 16
32     002400  000000  REGNUM: .WORD 0      ;NUMBER (10-17) OF LINE UNIT REG BEING TESTED
33     002402  000000  AXNUM: .WORD 0       ;NUMBER (0-7) OF EXTENDED REG BYTE BEING TESTED
34     002404  000000  GOODAT: .WORD 0      ;STORAGE FOR EXPECTED DATA
35     002406  000000  BADDAT: .WORD 0      ;STORAGE FOR ACTUAL DATA
36     002410  000000  LOADAT: .WORD 0      ;CONTAINS TEST DATA LOADED INTO REG
37     002412  000000  FRSTIM: .WORD 0      ;FLAG=0 IF PROGRAM JUST LOADED
38     002414  000000  SAVE4: .WORD 0       ;SAVE LOC 4 HERE (ERROR TRAP VECTOR)
39     002416  000000  SAVE6: .WORD 0       ;SAVE LOC 6 HERE (ERROR TRAP VECTOR)
40     002420  000000  ERROR1: .WORD 0      ;SUBR ERROR BIT FLAGS (DEF'D IN GLOBAL EQUATES)
41     002422  000000  TXWORD: .WORD 0      ;BITS 0-11 CONTAIN DATA TO LOAD INTO TX SILO
42     002424  000000  RXWORD: .WORD 0      ;BITS 0-11 CONTAIN DATA READ FROM RCV SILO
43     002426  000000  DISILO: .WORD 0      ;CONTAINS CURRENT STATE OF DISSI IN BIT 5
44     002430  000000  CHPTYP: .WORD 0      ;USYRT CHIP TYPE, =0 FOR SIG, ELSE =1
45     002432  000000  SAVLEN: .WORD 0      ;SAVED TX AND RCV CHAR LENGTHS
46     002434  000000  DEVMAP: .WORD 0      ;BIT MAP OF ACTIVE DEVICES
47     002436  000000  DEVPTR: .WORD 0      ;DEVICE MAP BIT POINTER
48     002440  000000  UNIT: .WORD 0        ;CONTAINS UNIT NO. (1 TO N)
49     002442  000000  TSTNUM: .WORD 0      ;CONTAINS TEST NUMBER FOR SOME TESTS
50     002444  000000  STARES: .WORD 0      ;FLAG=0 IF FIRST PASS AFTER STA OR RES
51
52     :***** CURRENT DEVICE PARAMETERS *****
53     002446  160170  MPCSR: .WORD 160170   ;POINTER TO MICROPROCESSOR CSR'S
54     002450  160171  BSEL1: .WORD 160171  ;POINTER TO BSEL1
55     002452  160172  BSEL2: .WORD 160172  ;POINTER TO BSEL2
56     002454  160174  SEL4: .WORD 160174   ;POINTER TO SEL4
```

```
58 002456 160176 SEL6: .WORD 160176 ; POINTER TO SEL6
59 002460 000300 MPIVEC: .WORD 300 ; MICROPROCESSOR INPUT INTERRUPT VECTOR
60 002462 000304 MPOVEC: .WORD 304 ; MICROPROCESSOR OUTPUT INTERRUPT VECTOR
61 002464 000240 MPRIOR: .WORD 240 ; MICROPROCESSOR DEVICE PRIORITY
62 002466 000000 LUSWI1: .WORD 0 ; LINE UNIT SWITCH PACK #1
63 002470 000000 LUSWI2: .WORD 0 ; LINE UNIT SWITCH PACK #2
64 002472 000000 LUSWI3: .WORD 0 ; LINE UNIT SWITCH PACK #3
65 002474 000000 TSTCON: .WORD 0 ; TEST CONNECTOR INDICATOR
66 002476 000000 RUNINH: .WORD 0 ; RUN SWITCH INDICATOR
67
68 ;***** STORAGE FOR DATA READ IN ADDRESS TESTS *****
69 002500 000 REDDAT: .BYTE 0
70 002501 000 .BYTE 0
71 002502 000 .BYTE 0
72 002503 000 .BYTE 0
73 002504 000 .BYTE 0
74 002505 000 .BYTE 0
75 002506 000 .BYTE 0
76 002507 000 .BYTE 0
77
78 ;:***** GEN'L PURPOSE SCRATCH STORAGE *****
79 002510 000000 REG0: .WORD 0
80 002512 000000 REG1: .WORD 0
81 002514 000000 REG2: .WORD 0
82 002516 000000 REG3: .WORD 0
83 002520 000000 REG4: .WORD 0
84 002522 000000 REG5: .WORD 0
85 002524 000000 REG6: .WORD 0
86 002526 000000 REG7: .WORD 0
87
88 ;:***** SCRATCH STORAGE FOR MESSAGE REPORTING *****
89 002530 000000 TMP0: .WORD 0
90 002532 000000 TMP1: .WORD 0
91 002534 000000 TMP2: .WORD 0
92 002536 000000 TMP3: .WORD 0
93 002540 000000 TMP4: .WORD 0
94 002542 000000 TMP5: .WORD 0
95 002544 000000 TMP6: .WORD 0
96 002546 000000 TMP7: .WORD 0
97
98 ;***** INBUS LU REG BIT MASKS FOR UNPREDICTABLE BITS *****
99 002550 UPBITS:
100 002550 000 .BYTE 000 ; MASK FOR REG 10
101 002551 056 .BYTE 056 ; MASK FOR REG 11
102 002552 000 .BYTE 000 ; MASK FOR REG 12
103 002553 257 .BYTE 257 ; MASK FOR REG 13
104 002554 100 .BYTE 100 ; MASK FOR REG 14
105 002555 377 .BYTE 377 ; MASK FOR REG 15
106 002556 377 .BYTE 377 ; MASK FOR REG 16
107 002557 306 .BYTE 306 ; MASK FOR REG 17
108
109 002560 200 R14NRW: .BYTE 200 ; REG 14 NON-R/W BITS
110
111 ;***** MASKS FOR EXTENDED REGISTER NON-READ/WRITE BITS *****
112 002561 ANBITS:
113 002561 377 .BYTE 377 ; MASK FOR AX0-15
114 002562 377 .BYTE 377 ; MASK FOR AX0-16
```

115	002563	000	.BYTE	000	:MASK FOR AX1-15
116	002564	360	.BYTE	360	:MASK FOR AX1-16
117	002565	000	.BYTE	000	:MASK FOR AX2-15
118	002566	000	.BYTE	000	:MASK FOR AX2-16
119	002567	004	.BYTE	004	:MASK FOR AX3-15
120	002570	030	.BYTE	030	:MASK FOR AX3-16

121
122 :***** DATA PATTERN A *****

123	002571		PATA:		
124	002571	125	.BYTE	125	
125	002572	252	.BYTE	252	
126	002573	000	.BYTE	000	
127	002574	377	.BYTE	377	
128	002575	001	.BYTE	001	
129	002576	002	.BYTE	002	
130	002577	004	.BYTE	004	
131	002600	010	.BYTE	010	
132	002601	020	.BYTE	020	
133	002602	040	.BYTE	040	
134	002603	100	.BYTE	100	
135	002604	200	.BYTE	200	
136	002605	376	.BYTE	376	
137	002606	375	.BYTE	375	
138	002607	373	.BYTE	373	
139	002610	367	.BYTE	367	
140	002611	357	.BYTE	357	
141	002612	337	.BYTE	337	
142	002613	277	.BYTE	277	
143	002614	177	.BYTE	177	

144
145 :***** DATA PATTERN B *****

146	002615		PATB:		
147	002615	000	.BYTE	000	
148	002616	000	.BYTE	000	
149	002617	040	.BYTE	040	
150	002620	100	.BYTE	100	
151	002621	220	.BYTE	220	
152	002622	000	.BYTE	000	
153	002623	000	.BYTE	000	
154	002624	051	.BYTE	051	

155
156 :***** DATA PATTERN C *****

157	002625		PATC:		
158	002625	020	.BYTE	020	
159	002626	020	.BYTE	020	
160	002627	020	.BYTE	020	

161
162 :***** DATA PATTERN D *****

163	002630		PATD:		
164	002630	000	.BYTE	000	
165	002631	040	.BYTE	040	
166	002632	000	.BYTE	000	

167
168 :***** DATA PATTERN E *****

169	002633		PATE:		
170	002633	000	.BYTE	000	
171	002634	120	.BYTE	120	

172	002635	020	.BYTE	020
173	002636	100	.BYTE	100
174	002637	120	.BYTE	120
175	002640	000	.BYTE	000

***** DATA PATTERN F *****

177			PATF:	
178	002641		.BYTE	050
179	002641	050	.BYTE	051
180	002642	051	.BYTE	050
181	002643	050		

***** DATA PATTERN G *****

182			PATG:	
183			.BYTE	000
184	002644		.BYTE	000
185	002644	000	.BYTE	240
186	002645	000	.BYTE	120
187	002646	240	.BYTE	177
188	002647	120	.BYTE	000
189	002650	177	.BYTE	000
190	002651	000	.BYTE	000
191	002652	000	.BYTE	001
192	002653	001		

***** DATA PATTERN H *****

193			PATH:	
194			.BYTE	000
195	002654		.BYTE	000
196	002654	000	.BYTE	377
197	002655	000	.BYTE	017
198	002656	377	.BYTE	377
199	002657	017	.BYTE	377
200	002660	377	.BYTE	375
201	002661	377	.BYTE	377
202	002662	375		
203	002663	377		

***** DATA PATTERN I *****

204			PATI:	
205			.BYTE	000
206	002664		.BYTE	000
207	002664	000	.BYTE	000
208	002665	000	.BYTE	000
209	002666	000	.BYTE	000
210	002667	000	.BYTE	000
211	002670	000	.BYTE	103
212	002671	103	.BYTE	000
213	002672	000	.BYTE	000
214	002673	000		

***** DATA PATTERN J *****

215			PATJ:	
216			.BYTE	000
217	002674		.BYTE	000
218	002674	000	.BYTE	010
219	002675	000	.BYTE	002
220	002676	010	.BYTE	004
221	002677	002	.BYTE	103
222	002700	004	.BYTE	001
223	002701	103	.BYTE	100
224	002702	001		
225	002703	100		

***** DATA PATTERN K *****

226			PATK:	
227			.BYTE	000
228	002704	000		

229	002705	000	.BYTE	000
230	002706	377	.BYTE	377
231	002707	377	.BYTE	377
232	002710	125	.BYTE	125
233	002711	125	.BYTE	125
234	002712	252	.BYTE	252
235	002713	252	.BYTE	252
236	002714	001	.BYTE	001
237	002715	000	.BYTE	000
238	002716	002	.BYTE	002
239	002717	000	.BYTE	000
240	002720	004	.BYTE	004
241	002721	000	.BYTE	000
242	002722	010	.BYTE	010
243	002723	000	.BYTE	000
244	002724	020	.BYTE	020
245	002725	000	.BYTE	000
246	002726	040	.BYTE	040
247	002727	000	.BYTE	000
248	002730	100	.BYTE	100
249	002731	000	.BYTE	000
250	002732	200	.BYTE	200
251	002733	000	.BYTE	000
252	002734	000	.BYTE	000
253	002735	001	.BYTE	001
254	002736	000	.BYTE	000
255	002737	002	.BYTE	002
256	002740	000	.BYTE	000
257	002741	004	.BYTE	004
258	002742	000	.BYTE	000
259	002743	010	.BYTE	010
260	002744	000	.BYTE	000
261	002745	020	.BYTE	020
262	002746	000	.BYTE	000
263	002747	040	.BYTE	040
264	002750	000	.BYTE	000
265	002751	100	.BYTE	100
266	002752	000	.BYTE	000
267	002753	200	.BYTE	200
268	002754	376	.BYTE	376
269	002755	377	.BYTE	377
270	002756	375	.BYTE	375
271	002757	377	.BYTE	377
272	002760	373	.BYTE	373
273	002761	377	.BYTE	377
274	002762	367	.BYTE	367
275	002763	377	.BYTE	377
276	002764	357	.BYTE	357
277	002765	377	.BYTE	377
278	002766	337	.BYTE	337
279	002767	377	.BYTE	377
280	002770	277	.BYTE	277
281	002771	377	.BYTE	377
282	002772	177	.BYTE	177
283	002773	377	.BYTE	377
284	002774	377	.BYTE	377
285	002775	376	.BYTE	376

286	002776	377	.BYTE	377
287	002777	375	.BYTE	375
288	003000	377	.BYTE	377
289	003001	373	.BYTE	373
290	003002	377	.BYTE	377
291	003003	367	.BYTE	367
292	003004	377	.BYTE	377
293	003005	357	.BYTE	357
294	003006	377	.BYTE	377
295	003007	337	.BYTE	337
296	003010	377	.BYTE	377
297	003011	277	.BYTE	277
298	003012	377	.BYTE	377
299	003013	177	.BYTE	177

***** DATA PATTERN L *****

301				
302	003014			
303	003014	000	.BYTE	000
304	003015	000	.BYTE	000
305	003016	377	.BYTE	377
306	003017	377	.BYTE	377
307	003020	000	.BYTE	000
308	003021	000	.BYTE	000

***** DATA PATTERN M *****

309				
310				
311	003022			
312	003022	000	.BYTE	000
313	003023	020	.BYTE	020
314	003024	000	.BYTE	000
315	003025	000	.BYTE	000
316	003026	200	.BYTE	200
317	003027	000	.BYTE	000
318	003030	000	.BYTE	000
319	003031	051	.BYTE	051

***** DATA PATTERN N *****

320				
321				
322	003032			
323	003032	000	.BYTE	000
324	003033	000	.BYTE	000
325	003034	000	.BYTE	000
326	003035	125	.BYTE	125
327	003036	000	.BYTE	000
328	003037	252	.BYTE	252
329	003040	000	.BYTE	000
330	003041	377	.BYTE	377
331	003042	005	.BYTE	005
332	003043	000	.BYTE	000
333	003044	012	.BYTE	012
334	003045	000	.BYTE	000
335	003046	017	.BYTE	017
336	003047	000	.BYTE	000

***** DATA PATTERN O *****

337				
338				
339	003050			
340	003050	000	.BYTE	000
341	003051	041	.BYTE	041
342	003052	004	.BYTE	004

343	003053	010	.BYTE	010
344	003054	020	.BYTE	020
345	003055	040	.BYTE	040
346	003056	100	.BYTE	100
347	003057	101	.BYTE	101
348	003060	200	.BYTE	200
349	003061	201	.BYTE	201
350	003062	300	.BYTE	300
351	003063	111	.BYTE	111
352	003064	301	.BYTE	301
353	003065	375	.BYTE	375

***** DATA PATTERN P *****
PATP:

356	003066	000	.BYTE	000
357	003066	113	.BYTE	113
358	003067	200	.BYTE	200
359	003070	040	.BYTE	040
360	003071	020	.BYTE	020
361	003072	010	.BYTE	010
362	003073	001	.BYTE	001
363	003074	104	.BYTE	104
364	003075	007	.BYTE	007
365	003076	105	.BYTE	105
366	003077	007	.BYTE	007
367	003100	144	.BYTE	144
368	003101	107	.BYTE	107
369	003102	157	.BYTE	157
370	003103			

***** DATA PATTERN U *****
PATU:

371				
372				
373	003104	000	.BYTE	000
374	003105	000	.BYTE	000
375	003106	001	.BYTE	001
376	003107	000	.BYTE	000
377	003110	013	.BYTE	013
378	003111	000	.BYTE	000
379	003112	011	.BYTE	011
380	003113	000	.BYTE	000
381	003114	021	.BYTE	021
382	003115	000	.BYTE	000
383	003116	101	.BYTE	101
384	003117	000	.BYTE	000
385	003120	301	.BYTE	301
386	003121	000	.BYTE	000
387	003122	000	.BYTE	000
388	003123	001	.BYTE	001
389	003124	000	.BYTE	000
390	003125	002	.BYTE	002
391	003126	000	.BYTE	000
392	003127	004	.BYTE	004
393	003130	000	.BYTE	000
394	003131	040	.BYTE	040
395	003132	000	.BYTE	000
396	003133	100	.BYTE	100
397	003134	000	.BYTE	000
398	003135	200	.BYTE	200
399	003136	000	.BYTE	000

400	003137	346	.BYTE	346
401	003140	000	.BYTE	000
402	003141	345	.BYTE	345
403	003142	000	.BYTE	000
404	003143	343	.BYTE	343
405	003144	000	.BYTE	000
406	003145	307	.BYTE	307
407	003146	000	.BYTE	000
408	003147	247	.BYTE	247
409	003150	000	.BYTE	000
410	003151	147	.BYTE	147

411				
412				
413	003152	000	.***** PATTERN V *****	
414	003153	000	PATV: .BYTE	000
415	003154	333	.BYTE	000
416	003155	000	.BYTE	333
417	003156	331	.BYTE	000
418	003157	000	.BYTE	331
419	003160	323	.BYTE	000
420	003161	000	.BYTE	323
421	003162	313	.BYTE	000
422	003163	000	.BYTE	313
423	003164	233	.BYTE	000
424	003165	000	.BYTE	233
425	003166	133	.BYTE	000
426	003167	000	.BYTE	133
427	003170	000	.BYTE	000
428	003171	001	.BYTE	000
429	003172	000	.BYTE	001
430	003173	002	.BYTE	000
431	003174	000	.BYTE	002
432	003175	004	.BYTE	000
433	003176	000	.BYTE	004
434	003177	040	.BYTE	000
435	003200	000	.BYTE	040
436	003201	100	.BYTE	000
437	003202	000	.BYTE	100
438	003203	200	.BYTE	000
439	003204	000	.BYTE	200
440	003205	346	.BYTE	000
441	003206	000	.BYTE	346
442	003207	345	.BYTE	000
443	003210	000	.BYTE	345
444	003211	343	.BYTE	000
445	003212	000	.BYTE	343
446	003213	307	.BYTE	000
447	003214	000	.BYTE	307
448	003215	247	.BYTE	000
449	003216	000	.BYTE	247
450	003217	147	.BYTE	000
451			.BYTE	147

452	003220		ENDPAT:	
453			.EVEN	
454				
455				
456				

457
458
459
460
461 003220 000400
462 003222 000400
463 003224 000000
464 003226 000125
465 003230 000252
466 003232 000377
467 003234 000000
468 003236 001000
469 003240 001000
470 003242 001000
471 003244 001000
472
473 003246 000377
474 003250 000000
475 003252 000125
476 003254 000252
477 003256 001000
478 003260 001000
479
480
481
482
483
484
485 003262
486
487
488
489
490
491
492
493
494
495

;*** TEST MESSAGES TO BE TRANSMITTED ***

MSG1: TXSOM
TXSOM
000
125
252
377
000
TXEOM
TXEOM
TXEOM
TXEOM

MSG4: 377
000
125
252
TXEOM
TXEOM

;*** RECEIVED DATA BUFFER (64. WORDS) ***
RCVBUF: .BLKW 64.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

.SBTTL GLOBAL SUBROUTINES

:/ THE GLOBAL SUBROUTINES ARE CALLED BY MORE THAN ONE TEST

* STPCLK - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO EXECUTE AN INSTRUCTION WHICH IS PASSED IN THE WORD FOLLOWING THE CALL.

```
STPCLK:
BISB #ROMO!ROMI,@BSEL1 ;SET ROMO, ROMI BITS IN BSEL1
MOV @(SP),@SEL6 ;PUT INSTRUCTION INTO SEL6
BISB #ROMO!ROMI!STEPMP,@BSEL1 ;SET ROMO, ROMI, STEPMP IN BSEL1
BICB #ROMO!ROMI!STEPMP,@BSEL1 ;CLEAR ROMO, ROMI, STEPMP IN BSEL1
ADD #2,(SP) ;FIX UP RETURN PC
RTS PC ;RETURN
```

* MSTCLR - THIS SUBROUTINE ISSUES A MASTER CLEAR AND SETS LULOOP

```
MSTCLR:
MOV R1,-(SP) ;SAVE R1
MOVB #MCLR,@BSEL1 ;SET MASTER CLEAR BIT
BICB #RUN!MCLR,@BSEL1 ;CLEAR RUN AND MCLR BITS
MOV #20.,R1 ;INITIALIZE STALL COUNTER
2$:
NOP ;STALL IN LOOP FOR SEVERAL MICRO-SEC
DEC R1
BNE 2$
BISB #LULOOP,@BSEL1 ;SET LU LOOP
MOV (SP)+,R1 ;RESTORE R1
CLR SAVLEN ;CLEAR CHAR LENGTH FROM SETUP
RTS PC ;RETURN
```

* READLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO EXECUTE AN INSTRUCTION WHICH READS THE LINE UNIT REG WHOSE NUMBER IS PASSED IN REGNUM, INTO REDBYT.

```
READLU:
MOV R1,-(SP) ;SAVE R1
MOV REGNUM,R1 ;GET LINE UNIT REG NUMBER
ASL R1 ;SHIFT INTO SOURCE BITS 4-7
ASL R1
ASL R1
```

003540				
003540	152777	000006	176702	
003546	017677	000000	176702	
003554	152777	000007	176666	
003562	142777	000007	176660	
003570	062716	000002		
003574	000207			
003576				
003576	010146			
003600	112777	000100	176642	
003606	142777	000300	176634	
003614	012701	000024		
003620	000240			
003622	005301			
003624	001375			
003626	152777	000010	176614	
003634	012601			
003636	005037	002432		
003642	000207			
003644				
003644	010146			
003646	013701	002400		
003652	006301			
003654	006301			
003656	006301			

```

58 003660 006301          ASL      R1
59 003662 052701 000004   BIS      #4,R1          ;SET DESTINATION = BSEL4
60 003666 052701 021000   BIS      #MVIOX,R1     ;SET REST OF MOVE INSTRUCTION
61 003672 010137 003702   MOV      R1,2$         ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
62 003676 004737 003540   JSR      PC,STPCLK     ;EXECUTE MOVE INSTRUCTION
63 003702 000000          .WORD    0             ;INSTRUCTION GOES HERE
64 003704 117737 176544 002364 2$:   MOVB    @BSEL4,REDBYT  ;GET LU REG CONTENTS INTO REDBYT
65 003712 105037 002365   CLRB    REDBYT+1       ;CLR HI BYTE OF STORAGE
66 003716 012601          MOV      (SP)+,R1      ;RESTORE R1
67 003720 000207          RTS      PC            ;RETURN
68
69
70
71
72
73
74
75
76
77
78 003722          ;*****
79 003722 010146          ;* WRITLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO
80 003724 013701 002400   ;*   EXECUTE AN INSTRUCTION WHICH LOADS THE BYTE CONTAINED IN WRIBYT
81 003730 052701 000100   ;*   INTO THE LU REG WHOSE NUMBER IS PASSED IN REGNUM,
82 003734 052701 122000   ;*****
83 003740 010137 003762   WRITLU:
84 003744 105037 002367   MOV      R1,-(SP)      ;SAVE R1
85 003750 113777 002366 176476 87:   MOV      REGNUM,R1     ;GET LINE UNIT REG NUMBER
86 003756 004737 003540   BIS      #100,R1      ;SET SOURCE = BSEL4
87 003762 000000          BIS      #MVIXO,R1    ;SET REST OF MOVE INSTRUCTION
88 003764 012601          MOV      R1,2$        ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
89 003766 000207          CLRB    WRIBYT+1      ;CLR HI BYTE OF STORAGE
90
91
92
93
94
95
96
97
98
99 003770 010146          ;*****
100 003772 013746 002400   ;* GETREG - THIS SUBROUTINE READS THE LINE UNIT REGISTERS 10-17 INTO THE
101 003776 012701 002302   ;*   REGISTER STORAGE TABLE (LUREG:).
102 004002 012737 000010 002400 3$:   ;*****
103 004010 004737 003644   GETREG: MOV      R1,-(SP)      ;SAVE R1
104 004014 113721 002364   MOV      REGNUM,-(SP)  ;SAVE CURRENT REG NO.
105 004020 105021          MOV      #LUR10,R1    ;INIT POINTER TO REG STORAGE TABLE
106 004022 005237 002400   MOV      #10,REGNUM    ;INIT LU REG NO. TO 10
107 004026 023727 002400 000020 JSR      PC,READLU     ;READ A LINE UNIT REG
108 004034 002765          MOVB    REDBYT,(R1)+  ;PUT BYTE READ INTO TABLE
109 004036 012637 002400   CLRB    (R1)+         ;CLEAR UPPER BYTE OF TABLE ENTRY
110 004042 012601          INC     REGNUM         ;INCREMENT REG NO.
111 004044 000207          CMP     REGNUM,#20    ;SEE IF ALL REGS READ YET
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

115
116
117
118
119
120
121
122
123
124
125 004046
126 004046 152777 000006 176374
127 004054 017677 000000 176374
128 004062 152777 000206 176360
129 004070 062716 000002
130 004074 000207
131
132
133
134
135
136
137
138
139
140
141
142
143 004076 010146
144 004100 013746 002400
145 004104 042737 000001 002420
146 004112 012737 000014 002400
147 004120 113737 002402 002366
148 004126 006237 002366
149 004132 152737 000024 002366
150 004140 053737 002426 002366
151 004146 004737 003722
152 004152 005001
153 004154 004737 003644 6\$:
154 004160 132737 000200 002364
155 004166 001006
156 004170 005201
157 004172 001370
158 004174 052737 000001 002420
159 004202 000424
160 004204 012737 000015 002400 9\$:
161 004212 004737 003644
162 004216 113737 002364 002370
163 004224 105037 002371
164 004230 012737 000016 002400
165 004236 004737 003644
166 004242 113737 002364 002372
167 004250 105037 002373
168 004254 012637 002400 12\$:
169 004260 012601
170 004262 000207
171

```

:*****
:* LOOPIN - THIS SUBROUTINE PLACES THE MICROPROCESSOR IN A LOOP ON AN
:* INSTRUCTION, BY MOVING THE INSTRUCTION FROM THE WORD FOLLOWING THE CALL
:* INTO SEL6, AND SETTING RUN AND ROMI IN BSEL1. THE SUBROUTINE RETURNS
:* WITH THE MICROPROCESSOR STUCK IN THE LOOP, AND IF IT IS DESIRED TO
:* TERMINATE THE LOOP, THE PDP-11 PROGRAM MUST CLEAR THE RUN BIT IN
:* BSEL1, OR CALL SUBROUTINE MSTCLR TO DO THIS.
:*****

```

```

LOOPIN:
BISB #ROMO!ROMI,@BSEL1 ;SET ROMO, ROMI BITS IN BSEL1
MOV @ (SP),@SEL6 ;PUT MICROINSTRUCTION INTO SEL6
BISB #RUN!ROMO!ROMI,@BSEL1 ;SET RUN, ROMO, ROMI IN BSEL1
ADD #2,(SP) ;FIX UP RETURN PC
RTS PC ;RETURN WITH MICROPROCESSOR STUCK IN SINGLE
; INSTRUCTION LOOP

```

```

:*****
:* READAX - THIS SUBROUTINE READS THE USYRT REG PAIR WHOSE NUMBER (0-3)
:* IS PASSED IN BITS 1,2 OF AXNUM ON ENTRY, AND RETURNS THE BYTES READ IN
:* RAX15 AND RAX16. IF THE LINE UNIT DOES NOT RESPOND WITH READY IN REG 14,
:* RRDYTO BIT IS SET IN ERROR1 ON RETURN.
:*****

```

```

READAX: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;STORE CURRENT REG NO.
BIC #RRDYTO,ERROR1 ;CLEAR ERROR BIT
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB AXNUM,WRIBYT ;SET UP AX REG NO. BITS
ASR WRIBYT
BISB #RDAX!ENAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14
BIS DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
JSR PC,WRITLU ;SET RDAX AND ENAX IN REG 14
CLR R1 ;INIT TIMER
6$: JSR PC,READLU ;READ REG 14
BITB #READY,REDBYT ;SEE IF READY BIT SET IN REG 14 YET
BNE 9$ ;BR IF READY SET
INC R1 ;INCR TIMER
BNE 6$ ;BR IF TIMER DIDN'T TIME OUT YET
BIS #RRDYTO,ERROR1 ;SET ERROR FLAG FOR TIME OUT ON READ RDY
BR 12$ ;BR TO RETURN
9$: MOV #15,REGNUM ;SET REG NO. = 15
JSR PC,READLU ;READ REG 15
MOVB REDBYT,RAX15 ;STORE REG AX-15
CLRB RAX15+1 ;CLR HI BYTE OF STORAGE
MOV #16,REGNUM ;SET REG NO. = 16
JSR PC,READLU ;READ REG 16
MOVB REDBYT,RAX16 ;STORE REG AX-16
CLRB RAX16+1 ;CLR HI BYTE OF STORAGE
12$: MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

```

172
173
174
175
176
177
178
179
180
181 004264 010146
182 004266 013746 002400
183 004272 042737 000002 002420
184 004300 012737 000014 002400
185 004306 113737 002402 002366
186 004314 006237 002366
187 004320 053737 002426 002366
188 004326 004737 003722
189 004332 012737 000015 002400
190 004340 105037 002375
191 004344 113737 002374 002366
192 004352 004737 003722
193 004356 005237 002400
194 004362 105037 002377
195 004366 113737 002376 002366
196 004374 004737 003722
197 004400 012737 000014 002400
198 004406 113737 002402 002366
199 004414 006237 002366
200 004420 152737 000014 002366
201 004426 053737 002426 002366
202 004434 004737 003722
203 004440 005001
204 004442 004737 003644 6\$:
205 004446 132737 000200 002364
206 004454 001005
207 004456 005201
208 004460 001370
209 004462 052737 000002 002420
210 004470 012637 002400 9\$:
211 004474 012601
212 004476 000207
213
214
215
216
217
218
219
220
221
222 004500 010146
223 004502 013746 002402
224 004506 012737 014411 002530
225 004514 032737 000001 002402
226 004522 001403
227 004524 012737 014414 002530
228 004532 004737 003770 1\$:

```

:*****
:* WRITAX - THIS SUBROUTINE WRITES THE USYRT REG PAIR WHOSE NUMBER (0-3) IS
:* PASSED IN BITS 1,2 OF AXNUM ON ENTRY, WITH THE DATA FROM WAX15 AND
:* WAX16. IF LINE UNIT DOES NOT RESPOND WITH READY IN REG 14, WRDYTO BIT
:* IS SET IN ERROR1 ON RETURN.
:*****

```

```

WRITAX: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE CURRENT REG NO.
BIC #WRDYTO,ERROR1 ;CLEAR ERROR BIT
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB AXNUM,WRIBYT ;SET AX REG NO. BITS
ASR WRIBYT
BIS DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
JSR PC,WRITLU ;SET AX NO. BITS IN REG 14
MOV #15,REGNUM ;SET REG NO. = 15
CLRB WAX15+1 ;CLR HI BYTE OF STORAGE
MOVB WAX15,WRIBYT ;SET UP BYTE TO WRITE INTO REG 15
JSR PC,WRITLU ;WRITE BYTE INTO REG 15
INC REGNUM ;SET REG NO. = 16
CLRB WAX16+1 ;CLR HI BYTE OF STORAGE
MOVB WAX16,WRIBYT ;SET UP BYTE TO WRITE INTO REG 16
JSR PC,WRITLU ;WRITE BYTE INTO REG 16
MOV #14,REGNUM ;SET REG NO. = 14
MOVB AXNUM,WRIBYT ;SET AX REG NO. BITS
ASR WRIBYT
BISB #ENAX!WAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14
BIS DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
JSR PC,WRITLU ;SET ENAX AND WAX IN REG 14
CLR R1 ;INIT PROGRAM TIMER
6$: JSR PC,READLU ;READ REG 14
BITB #READY,REDBYT ;SEE IF READY BIT SET IN REG 14 YET
BNE 9$ ;BR IF READY SET
INC R1 ;INCR TIMER
BNE 6$ ;BR IF TIMER DIDN'T TIME OUT YET
BIS #WRDYTO,ERROR1 ;SET ERROR FLAG BIT FOR TIME OUT ON WRITE RDY
9$: MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

```

```

:*****
:* GETALL - THIS SUBROUTINE READS THE LINE UNIT REGS 10-17 AND THE EXTENDED
:* REGISTERS AX0-AX3 INTO REGISTER STORAGE TABLE (LUREG:).
:*****

```

```

GETALL: MOV R1,-(SP) ;SAVE R1
MOV AXNUM,-(SP) ;SAVE CURRENT AX REG BYTE NO.
MOV #DH5,TMP0 ;SET AX LO BYTE NO.
BIT #BIT0,AXNUM ;SEE IF LO OR HI BYTE
BEQ 1$ ;BR IF LO BYTE
MOV #DH6,TMP0 ;SET AX HI BYTE NO.
1$: JSR PC,GETREG ;READ AND STORE REGS 10-17

```

```
229 004536 142777 000010 175704      BICB  #LULoop,@BSEL1 ;CLEAR LULoop
230 004544 012701 002322      MOV   #AX0.15,R1    ;INIT POINTER TO REG STORAGE TABLE
231 004550 005037 002402      CLR   AXNUM          ;INIT AX REG BYTE NO. TO 0
232 004554 004737 004076      JSR   PC,READAX     ;READ 2 AX REG BYTES
233 004560 113721 002370      MOVb  RAX15,(R1)+   ;PUT LO BYTE READ INTO TABLE
234 004564 105021      CLRb  (R1)+         ;CLEAR UPPER BYTE OF TABLE ENTRY
235 004566 113721 002372      MOVb  RAX16,(R1)+   ;PUT HI BYTE READ INTO TABLE
236 004572 105021      CLRb  (R1)+         ;CLEAR UPPER BYTE OF TABLE ENTRY
237 004574 062737 000002 002402      ADD   #2,AXNUM      ;INCR AX REG BYTE NO.
238 004602 023727 002402 000010      CMP   AXNUM,#10    ;SEE IF ALL REGS READ YET
239 004610 002761      BLT   3$           ;BR IF NOT
240 004612 012637 002402      MOV   (SP)+,AXNUM  ;RESTORE CURRENT AX REG BYTE NO.
241 004616 012601      MOV   (SP)+,R1     ;RESTORE R1
242 004620 013737 002402 002532      MOV   AXNUM,TMP1
243 004626 006237 002532      ASR   TMP1          ;GET EXTENDED REG NO. FOR PRINTOUT
244 004632 000207      RTS   PC            ;RETURN
```

245
246
247
248
249
250
251
252
253
254
255
256
257

```
*****  
* OSIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ORDY (REG 11)  
* AND OCOR (REG 17) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET  
* AS PASSED IN BIT 0 (ORDY) AND BIT 1 (OCOR) OF THE WORD FOLLOWING THE  
* CALL.  
* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS IN  
* RETADR.  
*****
```

```
258 004634 013746 002400      OSIRDY: MOV  REGNUM,-(SP) ;SAVE LU REG NO.
259 004640 013746 002352      MOV  SUBRPC,-(SP)
260 004644 005737 002352      TST  SUBRPC        ;SEE IF THIS IS A NESTED CALL
261 004650 001006      BNE  1$            ;BR IF YES
262 004652 016637 000004 002352      MOV  4(SP),SUBRPC
263 004660 162737 000004 002352      SUB  #4,SUBRPC     ;GET PC OF SUBROUTINE CALL
264 004666 012737 000011 002400 1$: MOV  #11,REGNUM    ;SET REG NO. TO 11
265 004674 004737 003644      JSR  PC,READLU     ;READ REG 11
266 004700 032776 000001 000004      BIT  #BIT0,@4(SP) ;GET EXPECTED STATE OF ORDY
267 004706 001413      BEQ  3$            ;BR IF EXPECTED ORDY = 0
268 004710 132737 000020 002364      BITb #ORDY,REDBYT ;SEE IF ORDY = 1
269 004716 001022      BNE  9$            ;BR IF ORDY = 1
270 004720 004737 004500      JSR  PC,GETALL     ;GET REGS FOR PRINTOUT
271      ;REPORT ORDY NOT SET
272      ERRDF 7,EM7,ERR4
```

TRAP C\$ERDF
.WORD 7
.WORD EM7
.WORD ERR4

```
273 004724 104455  
274 004726 000007  
275 004730 012350  
276 004732 015576  
277 004734 000451  
278 004736 132737 000020 002364 3$: BR 16$ ;TAKE ERROR RETURN
279 004744 001407      BITb #ORDY,REDBYT ;SEE IF ORDY = 0
280 004746 004737 004500      BEQ  9$            ;BR IF ORDY = 0
281      JSR  PC,GETALL ;GET REGS FOR PRINTOUT
282      ;REPORT ORDY NOT CLEARED
283      ERRDF 8,EM8,ERR4
```

TRAP C\$ERDF
.WORD 8
.WORD EM8

273 004734 000451
274 004736 132737 000020 002364
275 004744 001407
276 004746 004737 004500
277
278 004752 104455
279 004754 000010
280 004756 012365


```

004760 015576                                .WORD  ERR4
279 004762 000436                                BR      16$                ;TAKE ERROR RETURN
280 004764 012737 000017 002400 9$:  MOV     #17,REGNUM        ;SET REG NO. = 17
281 004772 004737 003644                                JSR     PC,READLU         ;READ LU REG 17
282 004776 132776 000002 000004        BITB   #BIT1,@4(SP)       ;GET EXPECTED STATE OF OCOR
283 005004 001413                                BEQ     12$                ;BR IF EXPECTED OCOR = 0
284 005006 132737 000020 002364        BITB   #OCOR,REDBYT       ;SEE IF OCOR = 1
285 005014 001031                                BNE     20$                ;BR IF OCOR = 1
286 005016 004737 004500                                JSR     PC,GETALL        ;GET REGS FOR PRINTOUT
287                                ;REPORT OCOR NOT SET
288 005022                                ERRDF  9,EM9,ERR4
                                TRAP  C$ERDF
                                .WORD  9
                                .WORD  EM9
                                .WORD  ERR4
289 005032 000412                                BR      16$                ;TAKE ERROR RETURN
290 005034 132737 000020 002364 12$:  BITB   #OCOR,REDBYT       ;SEE IF OCOR = 0
291 005042 001416                                BEQ     20$                ;BR IF OCOR = 0
292 005044 004737 004500                                JSR     PC,GETALL        ;GET REGS FOR PRINTOUT
293                                ;REPORT OCOR NOT CLEARED
294 005050                                ERRDF  10,EM10,ERR4
                                TRAP  C$ERDF
                                .WORD  10
                                .WORD  EM10
                                .WORD  ERR4
295 005060 016637 000002 002400 16$:  MOV     2(SP),REGNUM      ;RESTORE LU REG NO.
296 005066 013706 002346                                MOV     PSTACK,SP        ;RESTORE STACK POINTER TO BASE LEVEL
297 005072 013746 002362                                MOV     RETADR,-(SP)     ;FIX ERROR RETURN PC
298 005076 000407                                BR      23$
299 005100 062766 000002 000004 20$:  ADD     #2,4(SP)         ;FIX UP ERROR-FREE RETURN PC
300 005106 012637 002352                                MCV    (SP)+,SUBRPC
301 005112 012637 002400                                MOV     (SP)+,REGNUM     ;RESTORE LU REG NO.
302 005116 000207 23$:  RTS     PC                ;RETURN
303
304
305
306
307
308
309
310
311 005120 010146                                ;*****
312 005122 012701 000310  WAIT50: MOV     R1,-(SP)        ;SAVE R1
313 005126 005301                                MOV     #200.,R1         ;INIT COUNTER
314 005130 001376 3$:  DEC     R1                ;DECREMENT COUNTER
315 005132 012601                                BNE     3$                ;BR IF NOT DONE YET
316 005134 000207                                MOV     (SP)+,R1         ;RESTORE R1
317                                RTS     PC                ;RETURN
318
319
320
321
322
323
324
325 005136 000240                                ;*****
326 005140 000240  STALL:  NOP
                                NOP

```

327 005142 000240
 328 005144 000207
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338 005146 013746 002400
 339 005152 042737 170000 002422
 340 005160 012737 000011 002400
 341 005166 113737 002423 002366
 342 005174 004737 003722
 343 005200 012737 000010 002400
 344 005206 113737 002422 002366
 345 005214 004737 003722
 346 005220 012637 002400
 347 005224 000207
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359 005226 010146
 360 005230 017601 000002
 361 005234 001426
 362 005236 100006
 363 005240 042701 100000
 364 005244 005737 002430
 365 005250 001401
 366 005252 005301
 367 005254 152777 000010 175166 2\$:
 368 005262 152777 000020 175160 3\$:
 369 005270 004737 005136
 370 005274 142777 000020 175146
 371 005302 004737 005136
 372 005306 005301
 373 005310 001364
 374 005312 062766 000002 000002 6\$:
 375 005320 012601
 376 005322 000207
 377
 378
 379
 380
 381
 382
 383

NOP
RTS PC

 ;* LDTXSI - THIS SUBROUTINE LOADS THE TX SILO (REGS 10,11) WITH THE DATA PASSED
 ;* IN BITS 0-11 OF TXWORD.

```
LDTXSI: MOV    REGNUM, -(SP)      ;SAVE LU REG NO.
        BIC    #170000, TXWORD ;CLEAR UNUSED BITS
        MOV    #11, REGNUM     ;SET REG NO. = 11
        MOVB  TXWORD+1, WRIBYT ;SET DATA TO BE WRITTEN INTO REG 11
        JSR   PC, WRITLU      ;LOAD DATA INTO REG 11
        MOV    #10, REGNUM     ;SET REG NO. = 10
        MOVB  TXWORD, WRIBYT  ;SET DATA TO BE WRITTEN INTO REG 10
        JSR   PC, WRITLU      ;LOAD DATA INTO REG 10
        MOV    (SP)+, REGNUM   ;RESTORE LU REG NO.
        RTS   PC              ;RETURN
```

 ;* STPLU - THIS SUBROUTINE CLOCKS THE LINE UNIT FOR THE NO. OF CYCLES PASSED
 ;* IN BITS 0-14 OF THE WORD FOLLOWING THE CALL.
 ;* IF BIT 15 = 1, A CHECK IS MADE TO DETERMINE IF THE USYRT CHIP TYPE
 ;* REQUIRES DECREMENTING THE NO. OF CYCLES BY 1.

```
STPLU: MOV    R1, -(SP)        ;SAVE R1
        MOV    @2(SP), R1      ;GET DESIRED NO. OF CYCLES
        BEQ   6$,              ;IF DESIRED CYCLES = 0, RETURN
        BPL   2$,              ;BR IF CHIP TYPE CHECK NOT NECESSARY
        BIC   #BIT15, R1      ;CLEAR FLAG BIT
        TST   CHPTYP          ;SEE IF SIG USYRT
        BEQ   2$,              ;BR IF YES
        DEC   R1               ;DECREMENT CYCLE COUNT
        BISB  #LULOOP, @BSEL1  ;SET LU LOOP BIT
        BISB  #STEPLU, @BSEL1 ;SET THE STEPLU BIT (CLOCK THE TRANSMITTER)
        JSR   PC, STALL        ;STALL
        BICB  #STEPLU, @BSEL1 ;CLEAR THE STEPLU BIT (CLOCK THE RECEIVER)
        JSR   PC, STALL        ;STALL
        DEC   R1               ;DECREMENT CYCLE COUNTER
        BNE   3$,              ;BR IF NOT DONE YET
        ADD   #2, 2(SP)        ;FIX UP RETURN PC
        MOV   (SP)+, R1        ;RESTORE R1
        RTS   PC              ;RETURN
```

 ;* OACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF OACT (REG 11) AND

```

384      ;*      REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
385      ;*      WORD FOLLOWING THE CALL.
386      ;*****
387 005324 013746 002400  OACTIV: MOV     REGNUM,-(SP)    ;SAVE LU REG NO.
388 005330 013746 002352      MOV     SUBRPC,-(SP)
389 005334 005737 002352      TST     SUBRPC
390 005340 001006          BNE     1$
391 005342 016637 000004 002352  MOV     4(SP),SUBRPC
392 005350 162737 000004 002352  SUB     #4,SUBRPC
393 005356 012737 000011 002400  1$: MOV     #11,REGNUM
394 005364 004737 003644          JSR     PC,READLU
395 005370 032776 000001 000004  BIT     #BIT0,@4(SP)
396 005376 001413          BEQ     3$
397 005400 132737 000100 002364  BITB   #OACT,REDBYT
398 005406 001031          BNE     9$
399 005410 004737 004500          JSR     PC,GETALL
400      ;REPORT OACT NOT SET
401 005414          ERRDF  11,EM11,ERR4
402 005414 104455          TRAP   C$ERDF
403 005416 000013          .WORD  11
404 005420 012444          .WORD  EM11
405 005422 015576          .WORD  ERR4
406 005424 000412
407 005426 132737 000100 002364  3$: BR     6$
408 005434 001416          BITB   #OACT,REDBYT
409 005436 004737 004500          BEQ     9$
410 005442          JSR     PC,GETALL
411 005442          ;REPORT OACT NOT CLEARED
412 005442          ERRDF  12,EM12,ERR4
413 005442 104455          TRAP   C$ERDF
414 005444 000014          .WORD  12
415 005446 012461          .WORD  EM12
416 005450 015576          .WORD  ERR4
417 005452 016637 000002 002400  6$: MOV     2(SP),REGNUM
418 005460 013706 002346          MOV     PSTACK,SP
419 005464 013746 002362          MOV     RETADR,-(SP)
420 005470 000407          BR     12$
421 005472 062766 000002 000004  9$: ADD     #2,4(SP)
422 005500 012637 002352          MOV     (SP)+,SUBRPC
423 005504 012637 002400          MOV     (SP)+,REGNUM
424 005510 000207 12$: RTS     PC
425      ;*****
426      ;* INITRN - THIS SUBROUTINE INITIATES TRANSMISSION OF A MESSAGE, BY DOING A
427      ;* MASTER CLEAR, LOADING AX2-15 AND REG 17 WITH THE DATA PASSED IN THE 2
428      ;* WORDS FOLLOWING THE CALL, LOADING 2 SOM CHARS INTO THE TX SILO, AND
429      ;* CLOCKING THE LINE UNIT UNTIL THE FIRST SYNCH OR FLAG HAS BEEN SERIALIZED
430      ;* IN THE USYRT. THE PROGRAM MONITORS ORDY,OCOR, AND OACT FOR VALID STATES,
431      ;* THROUGHOUT THE PROCESS.
432      ;* IF THE SUBROUTINE DETECTS AN ERROR, A RETURN IS MADE TO THE TEST, AT THE
433      ;* ADDRESS CONTAINED IN RETADR.
434      ;*****
435 005512 010146  INITRN: MOV     R1,-(SP)
436 005514 013746 002400      MOV     REGNUM,-(SP)

```

```

433 005520 013746 002402      MOV      AXNUM,-(SP)      ;SAVE AX BYTE NO.
434 005524 016637 000006 002352  MOV      6(SP),SUBRPC
435 005532 162737 000004 002352  SUB      #4,SUBRPC      ;GET PC OF SUBR CALL
436 005540 004737 003576      JSR      PC,MSTCLR      ;ISSUE A MASTER CLEAR
437 005544 004737 004634      JSR      PC,OSIRDY      ;CHECK ORDY=1, OCOR=0
438 005550 000001 1
439 005552 004737 005324      JSR      PC,OACTIV      ;CHK OACT=0
440 005556 000000 0
441 005560 012737 000004 002402  MOV      #4,AXNUM      ;SET AX BYTE NO. = 4 FOR AX2
442 005566 117637 000006 002374  MOV      @6(SP),WAX15    ;SET DATA BYTE TO LOAD INTO AX2-15
443 005574 012737 000400 002422  MOV      #TXSOM,TXWORD  ;SET TSOM BIT
444 005602 113737 002374 002422  MOV      WAX15,TXWORD   ;SET SYNCH CHAR
445 005610 005037 002376      CLR      WAX16
446 005614 004737 004264      JSR      PC,WRITAX      ;LOAD AX2
447 005620 012737 000017 002400  MOV      #17,REGNUM     ;SET REG NO. = 17
448 005626 062766 000002 000006  ADD      #2,6(SP)       ;INCR POINTER TO NEXT DATA BYTE
449 005634 117637 000006 002366  MOV      @6(SP),WRIBYT  ;SET DATA BYTE TO LOAD INTO REG 17
450 005642 004737 003722      JSR      PC,WRITLU      ;LOAD REG 17
451 005646 004737 005146      JSR      PC,LDTXSI      ;LOAD THE SILO WITH SOM CHAR
452 005652 004737 005146      JSR      PC,LDTXSI      ;LOAD ANOTHER SOM INTO SILO
453 005656 004737 005120      JSR      PC,WAIT50      ;WAIT FOR DATA TO RIPPLE
454 005662 004737 004634      JSR      PC,OSIRDY      ;CHK ORDY=1, OCOR=1
455 005666 000003 3
456 005670 004737 005324      JSR      PC,OACTIV      ;CHK FOR OACT = 0
457 005674 000000 0
458 005676 005001 1
459 005700 012737 000011 002400  MOV      #11,REGNUM     ;INIT CYCLE COUNTER
460 005706 152777 000010 174534 6$:  BISB     #LULOOP,@BSEL1 ;SET LU REG NO. = 11
461 005714 152777 000020 174526  BISB     #STEPLU,@BSEL1 ;SET LINE UNIT LOOP BIT
462 005722 004737 005136      JSR      PC,STALL      ;SET CLOCK BIT
463 005726 004737 003644      JSR      PC,STALL      ;STALL FOR MICRO-SEC
464 005732 132737 000100 002364  BITB     #OACT,REDBYT   ;READ REG 11
465 005740 001014 1
466 005742 142777 000020 174500  BNE      9$            ;SEE IF OACT = 1 YET
467 005750 004737 005136      BICB     #STEPLU,@BSEL1 ;BR IF OACT = 1
468 005754 005201 1
469 005756 020127 000003      JSR      PC,STALL      ;CLEAR CLOCK BIT
470 005762 002751 1
471 005764 004737 005324      INC      R1            ;STALL FOR A MICRO-SEC
472 005770 000001 1
473 005772 012737 000017 002400 9$:  CMP      R1,#3         ;INCR CYCLE COUNT
474 006000 005037 002430      BLT      6$            ;SEE IF 3 CYCLES DONE YET
475 006004 004737 003644      JSR      PC,OACTIV      ;BR IF NOT
476 006010 132737 000020 002364  JSR      PC,OACTIV      ;CHK FOR OACT = 1
477 006016 001403 1
478 006020 012737 000001 002430  MOV      #17,REGNUM     ;SET REG NO. = 17
479 006026 142777 000020 174414 12$:  CLR      CHPTYP        ;CLEAR USYRT CHIP INDICATOR
480 006034 004737 005136      JSR      PC,READLU     ;READ REG 17
481 006040 004737 004634      BITB     #OCOR,REDBYT   ;CHK FOR OCOR CLEARED YET
482 006044 000001 1
483 006046 062766 000002 000006  BEQ      12$          ;BR IF YES - IT IS SIG CHIP
484 006054 012637 002402      MOV      #1,CHPTYP     ;SET INDICATOR FOR OTHER CHIP TYPE
485 006060 012637 002400      BICB     #STEPLU,@BSEL1 ;CLEAR CLOCK BIT
486 006064 012601 1
487 006066 005037 002352      JSR      PC,STALL      ;STALL FOR MICRO-SEC
488 006072 000207 1
489      JSR      PC,OSIRDY      ;CHK FOR ORDY = 1, OCOR = 0
      1
      ADD      #2,6(SP)     ;FIX UP RETURN PC
      MOV      (SP)+,AXNUM  ;RESTORE AX BYTE NO.
      MOV      (SP)+,REGNUM ;RESTORE LU REG NO.
      MOV      (SP)+,R1    ;RESTORE R1
      CLR      SUBRPC      ;CLEAR SUBR CALL PC
      RTS      PC          ;RETURN

```

490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546

```

*****
* TXCHAR - THIS SUBROUTINE INITIATES TRANSMISSION OF A CHARACTER, BY LOADING
* THE TX SILO WITH DATA PASSED IN BITS 0-11 OF THE WORD FOLLOWING THE CALL
* AND CLOCKS THE LINE UNIT WITH THE NUMBER OF CYCLES PASSED IN BITS 0-14
* OF THE SECOND WORD FOLLOWING THE CALL. IF BIT 15 = 1, A CHK IS MADE TO
* DETERMINE IF THE USYRT CHIP TYPE REQUIRES DECREMENTING THE NO. OF CYCLES
* BY 1. THE PROGRAM CHECKS FOR VALID STATES OF ORDY,
* OCOR, AND OACT THROUGHOUT THE PROCESS.
* IF AN ERROR IS DETECTED, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
* CONTAINED IN RETADR.
*****

```

```

TXCHAR: MOV     R1,-(SP)      ;SAVE R1
        MOV     R2,-(SP)      ;SAVE R2
        MOV     4(SP),SUBRPC   ;GET PC OF SUBR CALL
        SUB     #4,SUBRPC     ;GET DATA TO BE TRANSMITTED
        MOV     @4(SP),TXWORD ;LOAD THE TX SILO WITH THE DATA
        JSR    PC,LDTXSI     ;WAIT FOR DATA TO RIPPLE DOWN SILO
        JSR    PC,WAIT50
        ADD     #2,4(SP)      ;INCR POINTER
        CLR     R1           ;INIT CYCLE COUNT
        MOV     @4(SP),R2     ;GET DESIRED NO. OF CYCLES
        TST    R2           ;SEE IF CHIP TYPE CHK SHOULD BE MADE
        BPL    9$           ;BR IF NOT
        BIC    #BIT15,R2     ;CLEAR FLAG BIT
        TST    CHPTYP        ;SEE IF SIG USYRT
        BEQ    9$           ;BR IF YES
        DEC    R2           ;DECREMENT NO. OF CYCLES
9$:     JSR    PC,OACTIV     ;CHK OACT = 1
        1
        CMP    R1,R2        ;SEE IF REQUIRED CYCLES DONE YET
        BEQ    12$         ;BR IF YES
        JSR    PC,OSIRDY    ;CHK ORDY=1, OCOR=1
        3
        JSR    PC,STPLU     ;STEP LU ONE CYCLE
        1
        INC    R1           ;INCR CYCLE COUNT
        BR    9$
12$:   JSR    PC,OSIRDY    ;CHK ORDY=1, OCOR=0
        1
        ADD     #2,4(SP)     ;FIX UP RETURN PC
        CLR    SUBRPC       ;CLEAR SUBR CALL PC
        MOV    (SP)+,R2     ;RESTORE R2
        MOV    (SP)+,R1     ;RESTORE R1
        RTS    PC          ;RETURN

```

```

*****
* ENDTRN - THIS SUBROUTINE CLEARS THE TRANSMITTER BY SETTING OC. THE PROGRAM
* WAITS FOR 50 US, AND CHECKS FOR ORDY=1, OCOR=0, OACT=0, RTS=0.
*****

```

```

547 006246 010146          ENDTRN: MOV      R1,-(SP)          ;SAVE R1
548 006250 013746 002400   MOV      REGNUM,-(SP)       ;SAVE LU REG NO.
549 006254 016637 000004 002352  MOV      4(SP),SUBRPC
550 006262 162737 000004 002352  SUB      #4,SUBRPC          ;GET PC OF SUBROUTINE CALL
551 006270 012737 000011 002400  MOV      #11,REGNUM        ;SET LU REG NO. = 11
552 006276 012737 000200 002366  MOV      #OC,WRIBYT        ;SET OC IN DATA
553 006304 004737 003722     JSR      PC,WRITLU         ;SET OC IN REG 11
554 006310 004737 005120     JSR      PC,WAIT50        ;STALL FOR >50 U.S.
555 006314 004737 004634     JSR      PC,OSIRDY        ;CHK ORDY=1, OCOR=0
556 006320 000001           1
557 006322 004737 005324     JSR      PC,OACTIV        ;CHK OACT = 0
558 006326 000000           0
559 006330 012737 000013 002400  MOV      #13,REGNUM        ;SET REG NO. = 13
560 006336 004737 003644     JSR      PC,READLU        ;READ REG 13
561 006342 032737 000040 002364  BIT      #RTS,REDBYT       ;CHK FOR RTS = 0
562 006350 001406           BEQ      3$               ;BR IF RTS = 0
563 006352 004737 004500     JSR      PC,GETALL        ;GET REGS FOR PRINTOUT
564
565 006356          ;REPORT RTS NOT CLEARED
006356 104455          ERRDF 65,EM65,ERR4
006360 000101          TRAP  C$ERDF
006362 014220          .WORD 65
006364 015576          .WORD EM65
566 006366 005037 002352     3$:  CLR      SUBRPC          ;CLEAR SUBR CALL PC
567 006372 012637 002400     MOV      (SP)+,REGNUM      ;RESTORE LU REG NO.
568 006376 012601         MOV      (SP)+,R1          ;RESTORE R1
569 006400 000207         RTS      PC                ;RETURN
570
571
572
573
574
575
576
577
578
579
580
581
582
583 006402 013746 002400   ;*****
584 006406 013746 002352   ;* ISIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ICIR (REG 17)
585 006412 005737 002352   ;* AND IRDY (REG 12) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET
586 006416 001006         ;* AS PASSED IN BIT 0 (ICIR) AND BIT 1 (IRDY) OF THE WORD FOLLOWING THE
587 006420 016637 000004 002352  ;* CALL.
588 006426 162737 000004 002352  ;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS
589 006434 012737 000012 002400  ;* IN RETADR.
590 006442 004737 003644   ;*****
591 006446 032776 000002 000004  ISIRDY: MOV      REGNUM,-(SP)  ;SAVE LU REG NO.
592 006454 001413         MOV      SUBRPC,-(SP)
593 006456 132737 000020 002364  TST      SUBRPC           ;SEE IF THIS IS A NESTED CALL
594 006464 001022         BNE     1$               ;BR IF YES
595 006466 004737 004500     MOV      4(SP),SUBRPC     ;GET PC OF SUBR CALL
596
597 006472          1$:  MOV      #12,REGNUM        ;SET REG NO. TO 12
006472 104455          JSR      PC,READLU        ;READ REG 12
006474 000021         BIT      #BIT1,@4(SP)     ;GET EXPECTED STATE OF IRDY
                                BEQ      3$               ;BR IF EXPECTED IRDY = 0
                                BITB   #IRDY,REDBYT       ;SEE IF IRDY = 1
                                BNE     9$               ;BR IF IRDY = 1
                                JSR      PC,GETALL        ;GET REGS FOR PRINTOUT
                                ;REPORT IRDY NOT SET
                                ERRDF 17,EM17,ERR4
                                TRAP  C$ERDF
                                .WORD 17

```

```

006476 012617
006500 015576
598 006502 000451
599 006504 132737 000020 002364 3$: BR 16$ :TAKE ERROR EXIT
600 006512 001407 BEQ 9$ :SEE IF IRDY = 0
601 006514 004737 004500 JSR PC,GETALL :BR IF IRDY = 0
602 :REPORT IRDY NOT CLEARED :GET REGS FOR PRINTOUT
603 ERRDF 18,EM18,ERR4
006520 104455 TRAP C$ERDF
006522 000022 .WORD 18
006524 012634 .WORD EM18
006526 015576 .WORD ERR4
604 006530 000436 BR 16$ :TAKE ERROR RETURN
605 006532 012737 000017 002400 9$: MOV #17,REGNUM :SET REG NO. = 17
606 006540 004737 003644 JSR PC,READLU :READ REG 17
607 006544 132776 000001 000004 BITB #BIT0,@4(SP) :GET EXPECTED STATE OF ICIR
608 006552 001413 BEQ 12$ :BR IF EXPECTED ICIR = 0
609 006554 132737 000010 002364 BITB #ICIR,REDBYT :SEE IF ICIR = 1
610 006562 001031 BNE 20$ :BR IF ICIR = 1
611 006564 004737 004500 JSR PC,GETALL :GET REGS FOR PRINTOUT
612 :REPORT ICIR NOT SET
613 ERRDF 19,EM19,ERR4
006570 104455 TRAP C$ERDF
006572 000023 .WORD 19
006574 012655 .WORD EM19
006576 015576 .WORD ERR4
614 006600 000412 BR 16$ :TAKE ERROR RETURN
615 006602 132737 000010 002364 12$: BITB #ICIR,REDBYT :SEE IF ICIR = 0
616 006610 001416 BEQ 20$ :BR IF ICIR = 0
617 006612 004737 004500 JSR PC,GETALL :GET REGS FOR PRINTOUT
618 :REPORT ICIR NOT CLEARED
619 ERRDF 20,EM20,ERR4
006616 104455 TRAP C$ERDF
006620 000024 .WORD 20
006622 012672 .WORD EM20
006624 015576 .WORD ERR4
620 006626 016637 000002 002400 16$: MOV 2(SP),REGNUM :RESTORE LU REG NO.
621 006634 013706 002346 MOV PSTACK,SP :RESTORE STACK POINTER TO BASE LEVEL
622 006640 013746 002362 MOV RETADR,-(SP) :FIX ERROR RETURN PC
623 006644 000407 BR 23$
624 006646 062766 000002 000004 20$: ADD #2,4(SP) :FIX UP ERROR-FREE RETURN PC
625 006654 012637 002352 MOV (SP)+,SUBRPC
626 006660 012637 002400 MOV (SP)+,REGNUM :RESTORE LU REG NO.
627 006664 000207 23$: RTS PC :RETURN
628
629
630
631
632
633
634
635
636
637
638
639
640 006666 013746 002400
:*****
:* IACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF IACT (REG 12) AND
:* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
:* WORD FOLLOWING THE CALL.
:* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN
:* RETADR.
:*****
IACTIV: MOV REGNUM,-(SP) ;SAVE LU REG NO.

```

```
641 006572 013746 002352      MOV      SUBRPC,-(SP)
642 006676 005737 002352      TST      SUBRPC          ;SEE IF THIS IS A NESTED CALL
643 006702 001006                BNE      1$              ;BR IF YES
644 006704 016637 000004 002352  MOV      4(SP),SUBRPC
645 006712 162737 000004 002352  SUB      #4,SUBRPC      ;GET PC OF SUBR CALL
646 006720 012737 000012 002400 1$:  MOV      #12,REGNUM     ;SET REG NO. = 12
647 006726 004737 003644      JSR      PC,READLU      ;READ REG 12
648 006732 032776 000001 000004  BIT      #BIT0,@4(SP)   ;GET EXPECTED STATE OF IACT
649 006740 001413                BEQ      3$              ;BR IF EXPECTED IACT = 0
650 006742 132737 000100 002364  BITB     #IACT,REDBYT   ;SEE IF IACT = 1
651 006750 001031                BNE      9$              ;BR IF IACT = 1
652 006752 004737 004500      JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
653                                     ;REPORT IACT NOT SET
654 006756                                     ERRDF    21,EM21,ERR4
                                     TRAP    C$ERDF
006756 104455                                     .WORD  21
006760 000025                                     .WORD  EM21
006762 012713                                     .WORD  ERR4
006764 015576
655 006766 000412                BR      6$              ;TAKE ERROR EXIT
656 006770 132737 000100 002364 3$:  BITB     #IACT,REDBYT   ;SEE IF IACT = 0
657 006776 001416                BEQ      9$              ;BR IF IACT = 0
658 007000 004737 004500      JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
659                                     ;REPORT IACT NOT CLEARED
660 007004                                     ERRDF    22,EM22,ERR4
                                     TRAP    C$ERDF
007004 104455                                     .WORD  22
007006 000026                                     .WORD  EM22
007010 012730                                     .WORD  ERR4
007012 015576
661 007014 016637 000002 002400 6$:  MOV      2(SP),REGNUM   ;RESTORE LU REG NO.
662 007022 013706 002346      MOV      PSTACK,SP     ;RESTORE PROGRAM STACK TO BASE LEVEL
663 007026 013746 002362      MOV      RETADR,-(SP)   ;FIX UP ERROR RETURN PC
664 007032 000407                BR      12$
665 007034 062766 000002 000004 9$:  ADD      #2,4(SP)       ;FIX UP ERROR-FREE RETURN PC
666 007042 012637 002352      MOV      (SP)+,SUBRPC
667 007046 012637 002400      MOV      (SP)+,REGNUM   ;RESTORE LU REG NO.
668 007052 000207                RTS      PC              ;RETURN
669
670
671
672
673
674
675                                     ;*****
676                                     ;* RSEOM - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF RSOM AND REOM IN
677                                     ;* AX0-16, AND REPORTS AN ERROR IF EITHER IS NOT SET TO THE STATE PASSED IN BITS
678                                     ;* 0,1, RESPECTIVELY, OF THE WORD FOLLOWING THE CALL.
679                                     ;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN RETADR.
680                                     ;*****
680 007054 013746 002402  RSEOM:  MOV      AXNUM,-(SP) ;SAVE AX BYTE NO.
681 007060 013746 002352      MOV      SUBRPC,-(SP)
682 007064 005737 002352      TST      SUBRPC          ;SEE IF THIS IS A NESTED CALL
683 007070 001006                BNE      1$              ;BR IF YES
684 007072 016637 000004 002352  MOV      4(SP),SUBRPC
685 007100 162737 000004 002352  SUB      #4,SUBRPC      ;GET PC OF SUBR CALL
686 007106 012737 000001 002402 1$:  MOV      #1,AXNUM       ;SET AX BYTE NO. FOR AX0-16
687 007114 004737 004076      JSR      PC,READAX      ;READ AX0
688 007120 032776 000001 000004  BIT      #BIT0,@4(SP)   ;GET EXPECTED STATE OF RSOM
689 007126 001413                BEQ      3$              ;BR IF EXPECTED RSOM = 0
```



```

690 007130 132737 000001 002372      BITB   #RSOM,RAX16      ;SEE IF RSOM = 1
691 007136 001022                BNE    9$              ;BR IF RSOM = 1
692 007140 004737 004500                JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
693                                ;REPORT RSOM NOT SET
694 007144                ERRDF   29,EM29,ERR6
                                TRAP   C$ERDF
                                .WORD  29
                                .WORD  EM29
                                .WORD  ERR6
695 007154 000444                BR     16$            ;TAKE ERROR EXIT
696 007156 132737 000001 002372 3$:   BITB   #RSOM,RAX16      ;SEE IF RSOM = 0
697 007164 001407                BEQ    9$              ;BR IF RSOM = 0
698 007166 004737 004500                JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
699                                ;REPORT RSOM NOT CLEARED
700 007172                ERRDF   28,EM28,ERR6
                                TRAP   C$ERDF
                                .WORD  28
                                .WORD  EM28
                                .WORD  ERR6
701 007202 000431                BR     16$            ;TAKE ERROR RETURN
702 007204 132776 000002 000004 9$:   BITB   #BIT1,@4(SP)    ;GET EXPECTED STATE OF REOM
703 007212 001413                BEQ    12$            ;BR IF EXPECTED REOM = 0
704 007214 132737 000002 002372      BITB   #REOM,RAX16     ;SEE IF REOM = 1
705 007222 001031                BNE    20$            ;BR IF REOM = 1
706 007224 004737 004500                JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
707                                ;REPORT REOM NOT SET
708 007230                ERRDF   31,EM31,ERR6
                                TRAP   C$ERDF
                                .WORD  31
                                .WORD  EM31
                                .WORD  ERR6
709 007240 000412                BR     16$            ;TAKE ERROR RETURN
710 007242 132737 000002 002372 12$:  BITB   #REOM,RAX16     ;SEE IF REOM = 0
711 007250 001416                BEQ    20$            ;BR IF REOM = 0
712 007252 004737 004500                JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
713                                ;REPORT REOM NOT CLEARED
714 007256                ERRDF   30,EM30,ERR6
                                TRAP   C$ERDF
                                .WORD  30
                                .WORD  EM30
                                .WORD  ERR6
715 007266 016637 000002 002402 16$:  MOV    2(SP),AXNUM     ;RESTORE AX BYTE NO.
716 007274 013706 002346                MOV    PSTACK,SP      ;RESTORE STACK POINTER TO BASE LEVEL
717 007300 013746 002362                MOV    RETADR,-(SP)   ;FIX ERROR RETURN PC
718 007304 000407                BR     23$
719 007306 062766 000002 000004 20$:  ADD    #2,4(SP)        ;FIX UP ERROR-FREE RETURN PC
720 007314 012637 002352                MOV    (SP)+,SUBRPC
721 007320 012637 002402                MOV    (SP)+,AXNUM    ;RESTORE AX BYTE NO.
722 007324 000207 23$:   RTS    PC              ;RETURN
723
724
725
726
727
728
729
730
*****
;* RDRXSI - THIS SUBROUTINE READS THE RCV SILO (REGS 10,12) AND RETURNS THE
;* SILO ENTRY IN BITS 0-11 OF RXWORD.

```

```

731
732 007326 013746 002400
733 007332 012737 000012 002400
734 007340 004737 003644
735 007344 113737 002364 002425
736 007352 042737 170000 002424
737 007360 012737 000010 002400
738 007366 004737 003644
739 007372 113737 002364 002424
740 007400 012637 002400
741 007404 000207
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756 007406 010146
757 007410 010346
758 007412 013746 002400
759 007416 016637 000006 002352
760 007424 162737 000004 002352
761 007432 012737 000012 002400
762 007440 005001
763 007442 017603 000006
764 007446 062703 000003
765 007452 005776 000006
766 007456 001414
767 007460 004737 006666
768 007464 000000
769 007466 004737 006402
770 007472 000001
771 007474 004737 007054
772 007500 000000
773 007502 004737 005226
774 007506 000001
775 007510 004737 005120
776 007514 005201
777 007516 004737 003644
778 007522 132737 000020 002364
779 007530 001005
780 007532 020103
781 007534 002762
782 007536 004737 006402
783 007542 000003
784 007544 020176 000006
785 007550 002003
786 007552 004737 006402
787 007556 000001

:*****
RDRXSI: MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV #12,REGNUM ;SET REG NO. = 12
JSR PC,READLU ;READ LU REG 12
MOVB REDBYT,RXWORD+1 ;GET HI BITS OF SILO ENTRY
BIC #170000,RXWORD ;CLEAR UNUSED BITS
MOV #10,REGNUM ;SET REG NO. = 10
JSR PC,READLU ;READ REG 10
MOVB REDBYT,RXWORD ;GET LOW BITS OF SILO ENTRY
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
RTS PC ;RETURN

:*****
;* RCV1ST - THIS SUBROUTINE RECEIVES THE FIRST CHAR OF A MESSAGE, AND MONITORS
;* STATUS OF THE RECEIVER. FIRST, A CHECK IS MADE FOR IACT = 0, IRDY = 0,
;* ICIR = 1, AND RSOM = 0. THEN, THE LINE UNIT IS CLOCKED USING
;* STEPLU UNTIL IRDY = 1. THE PROGRAM CHECKS FOR THIS TO OCCUR WITHIN 3
;* CYCLES AFTER THE NO. OF CYCLES PASSED IN THE WORD FOLLOWING THE CALL.
;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
;* CONTAINED IN RETADR.
:*****
RCV1ST: MOV R1,-(SP) ;SAVE R1
MOV R3,-(SP) ;SAVE R3
MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV 6(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBROUTINE CALL
MOV #12,REGNUM ;SET LU REG NO. = 12
CLR R1 ;INIT CYCLE COUNT TO 0
MOV @6(SP),R3 ;GET CYCLE COUNT LIMIT
ADD #3,R3
TST @6(SP) ;SEE IF DESIRED CYCLES = 0
BEQ 8$ ;BR IF YES
JSR PC,IACTIV ;CHK FOR IACT = 0
0
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 0
1
JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0 IN AX0-16
0
6$: JSR PC,STPLU ;CLOCK LU FOR 1 CYCLE
1
8$: JSR PC,WAIT50 ;ALLOW SILO DATA TO RIPPLE
INC R1 ;INCREMENT CYCLE COUNT
JSR PC,READLU ;READ REG 12
BITB #IRDY,REDBYT ;SEE IF IRDY = 1 YET
BNE 9$ ;BR IF IRDY = 1
CMP R1,R3 ;SEE IF LIMIT EXCEEDED
BLT 6$ ;BR IF NOT YET
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 1
3
9$: CMP R1,@6(SP) ;SEE IF LESS THAN REQUIRED CYCLES
BGE 12$ ;BR IF NOT
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 0
1

```

```

788 007560 004737 006666      12$: JSR   PC,IACTIV      ;CHK FOR IACT = 1
789 007564 000001              1
790 007566 004737 006402      JSR   PC,ISIRDY      ;CHK FOR ICIR = 1, IRDY = 1
791 007572 000003              3
792 007574 062766 000002 000006  ADD   #2,6(SP)       ;FIX UP RETURN PC
793 007602 012637 002400      MOV   (SP)+,REGNUM   ;RESTORE LU REG NO.
794 007606 012603              MOV   (SP)+,R3       ;RESTORE R3
795 007610 012601              MOV   (SP)+,R1       ;RESTORE R1
796 007612 005037 002352      CLR   SUBRPC        ;CLEAR SUBR CALL PC
797 007616 000207              RTS    PC            ;RETURN
798
799
800
801
802
803

```

```

:*****
:* STPERR - THIS SUBROUTINE LOADS THE CONTENTS OF THE FIRST WORD FOLLOWING THE
:* CALL INTO REG 17, AND SETS THE IERR BIT, AND CLOCKS THE LINE UNIT
:* FOR THE NO. OF CYCLES PASSED IN THE 2ND WORD FOLLOWING THE CALL. THEN,
:* IT RESTORES REG 17 TO ITS ORIGINAL CONTENTS, CLEARING THE IERR BIT.
:*****

```

```

809 007620 013746 002400      STPERR: MOV   REGNUM,-(SP)  ;SAVE LU REG NO.
810 007624 012737 000017 002400  MOV   #17,REGNUM    ;SET LU REG NO. = 17
811 007632 017637 000002 002366  MOV   @2(SP),WRIBYT
812 007640 152737 000002 002366  BISB  #IERR,WRIBYT
813 007646 004737 003722      JSR   PC,WRITLU     ;SET IERR BIT IN REG 17
814 007652 062766 000002 000002  ADD   #2,2(SP)      ;INCREMENT SUBR ARGUMENT POINTER
815 007660 017637 000002 007672  MOV   @2(SP),3$     ;GET DESIRED NO. OF CYCLES
816 007666 004737 005226      JSR   PC,STPLU     ;CLOCK LU FOR DESIRED NO. OF CYCLES
817 007672 000000              3$: .WORD 0           ;NO. OF CYCLES GOES HERE
818 007674 142737 000002 002366  BICB  #IERR,WRIBYT
819 007702 004737 003722      JSR   PC,WRITLU     ;CLEAR IERR BIT IN REG 17
820 007706 062766 000002 000002  ADD   #2,2(SP)      ;FIX UP RETURN PC
821 007714 012637 002400      MOV   (SP)+,REGNUM  ;RESTORE LU REG NO.
822 007720 000207              RTS    PC            ;RETURN
823
824
825
826
827
828

```

```

:*****
:* CKDATA - THIS SUBROUTINE READS THE RCV SILO AND COMPARES THE SILO ENTRY
:* TO BITS 0-11 OF THE FIRST WORD FOLLOWING THE CALL. IF THERE IS A
:* MISMATCH, THE ERROR IS REPORTED AND A RETURN IS MADE TO THE TEST AT THE
:* ADDRESS CONTAINED IN RETADR. IF BIT 15 = 0 IN THE FIRST WORD
:* FOLLOWING THE CALL, THE SUBROUTINE WILL NOT CHECK THE BCC BIT (SILO
:* BIT 8). IF THERE ARE NO ERRORS, THE LINE UNIT IS CLOCKED FOR THE
:* NUMBER OF CYCLES PASSED IN THE SECOND WORD FOLLOWING THE CALL.
:*****

```

```

837 007722 010146              CKDATA: MOV   R1,-(SP)    ;SAVE R1
838 007724 013746 002400      MOV   REGNUM,-(SP)  ;SAVE LU REG NO.
839 007730 016637 000004 002352  MOV   4(SP),SUBRPC
840 007736 162737 000004 002352  SUB   #4,SUBRPC     ;GET PC OF SUBR CALL
841 007744 017601 000004      MOV   @4(SP),R1     ;GET EXPECTED SILO ENTRY
842 007750 042701 170000      BIC   #170000,R1    ;CLEAR UNUSED BITS FOR COMPARE
843 007754 004737 007326      JSR   PC,RDRXSI     ;READ RCV SILO
844 007760 023727 002432 000347  CMP   SAVLEN,#TXLEN2!TXLEN1!TXLEN0!RXLEN2!RXLEN1!RXLEN0

```

```

845 007766 001005          BNE      4$          ;BR IF CHAR LENGTH NOT = 7
846 007770 042701 000200    BIC      #BIT7,R1     ;MASK OFF BIT 8TH BIT
847 007774 042737 000200    BIC      #BIT7,RXWORD
848 010002 120137 002424    4$:      CMPB     R1,RXWORD ;COMPARE EXPECTED BITS 0-7 TO ACTUAL
849 010006 001445          BEQ      6$          ;BR IF MATCH
850 010010 005037 002404    CLR      GOODAT
851 010014 110137 002404    MOVB     R1,GOODAT    ;GET EXPECTED DATA
852 010020 005037 002406    CLR      BADDAT
853 010024 113737 002424    MOVB     RXWORD,BADDAT ;GET ACTUAL DATA
854 010032 012737 000011    MOV      #11,REGNUM   ;SET REG NO. = 11
855 010040 004737 003644    JSR      PC,READLU    ;READ REG 11
856 010044 132737 000001    BITB     #UNRR,REDBYT ;SEE IF TX UNDERRUN ERROR
857 010052 001410          BEQ      5$          ;BR IF NOT
858 010054 004737 004500    JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
859          ;REPORT TX UNDERRUN ERROR
860 010060          ERRDF    54,EM54,ERR4
860 010060 104455          TRAP    C$ERDF
860 010062 000066          .WORD   54
860 010064 014162          .WORD   EM54
860 010066 015576          .WORD   ERR4
861 010070 000137 010552    JMP      36$         ;TAKE ERROR EXIT
862 010074 012737 000010    5$:      MOV      #10,REGNUM ;SET REG NO. = 10
863 010102 004737 004500    JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
864          ;REPORT RCV'D DATA MISCOMPARE
865 010106          ERRDF    34,EM34,ERR8
865 010106 104455          TRAP    C$ERDF
865 010110 000042          .WORD   34
865 010112 013274          .WORD   EM34
865 010114 020076          .WORD   ERR8
866 010116 000137 010552    JMP      36$         ;TAKE ERROR EXIT
867 010122 000301          6$:      SWAB     R1
868 010124 012737 000012    MOV      #12,REGNUM   ;SET LU REG NO. FOR ERROR REPORTS
869 010132 120137 002425    CMPB     R1,RXWORD+1 ;COMPARE EXPECTED SILO BITS 8-11 TO ACTUAL
870 010136 001002          BNE      7$          ;BR IF MISMATCH
871 010140 000137 010526    JMP      22$         ;CONTINUE
872 010144 005037 002404    7$:      CLR      GOODAT
873 010150 110137 002404    MOVB     R1,GOODAT    ;SET EXPECTED DATA
874 010154 005037 002406    CLR      BADDAT
875 010160 113737 002425    MOVB     RXWORD+1,BADDAT ;SET ACTUAL DATA
876 010166 032776 100000    BIT      #BCCCHK,@4(SP) ;SEE IF BCC SHOULD BE IGNORED
877 010174 001433          BEQ      10$         ;BR IF YES
878 010176 132701 000001    BITB     #BCC,R1      ;SEE IF EXPECTED BIT = 1
879 010202 001014          BNE      8$          ;BR IF YES
880 010204 132737 000001    BITB     #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 0
881 010212 001424          BEQ      10$         ;BR IF YES
882 010214 004737 004500    JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
883          ;REPORT BCC NOT CLEARED
884 010220          ERRDF    35,EM35,ERR8
884 010220 104455          TRAP    C$ERDF
884 010222 000043          .WORD   35
884 010224 013322          .WORD   EM35
884 010226 020076          .WORD   ERR8
885 010230 000137 010552    JMP      36$         ;TAKE ERROR EXIT
886 010234 132737 000001    8$:      BITB     #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 1
887 010242 001010          BNE      10$         ;BR IF YES
888 010244 004737 004500    JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
889          ;REPORT BCC NOT SET

```



```

927 010456 004737 004500      JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
928                          ;REPORT OVRR NOT CLEARED
929 010462                      ERRDF  41,EM41,ERR8
                                TRAP   C$ERDF
                                .WORD  41
                                .WORD  EM41
                                .WORD  ERR8
010462 104455
010464 000051
010466 013450
010470 020076
930 010472 000137 010552      JMP    36$            ;TAKE ERROR EXIT
931 010476 132737 000010 002425 20$: BITB  #OVRR,RXWORD+1 ;SEE IF ACTUAL BIT = 1
932 010504 001010          BNE   22$            ;BR IF YES
933 010506 004737 004500      JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
934                          ;REPORT OVRR NOT SET
935 010512                      ERRDF  42,EM42,ERR8
                                TRAP   C$ERDF
                                .WORD  42
                                .WORD  EM42
                                .WORD  ERR8
010512 104455
010514 000052
010516 013471
010520 020076
936 010522 000137 010552      JMP    36$            ;TAKE ERROR EXIT
937 010526          22$:
938 010526 062766 000002 000004 ADD  #2,4(SP)         ;INCR SUBROUTINE ARGUMENT POINTER
939 010534 017637 000004 010546 MOV  @4(SP),24$       ;GET DESIRED CYCLE COUNT
940 010542 004737 005226      JSR    PC,STPLU       ;CLOCK LU FOR DESIRED CYCLES
941 010546 000000          24$: .WORD  0
942 010550 000407          BR    38$            ;TAKE ERROR-FREE EXIT
943 010552 011637 002400          36$: MOV  (SP),REGNUM     ;RESTORE LU REG NO.
944 010556 013706 002346          MOV  PSTACK,SP       ;RESTORE PROGRAM STACK TO BASE LEVEL
945 010562 013746 002362          MOV  RETADR,-(SP)    ;FIX UP ERROR RETURN PC
946 010566 000406          BR    40$
947 010570 062766 000002 000004 38$: ADD  #2,4(SP)         ;FIX UP ERROR-FREE RETURN PC
948 010576 012637 002400          MOV  (SP)+,REGNUM    ;RESTORE LU REG NO.
949 010602 012601          MOV  (SP)+,R1        ;RESTORE R1
950 010604 005037 002352          40$: CLR  SUBRPC       ;CLEAR SUBROUTINE PC
951 010610 000207          RTS   PC            ;RETURN
952
953
954
955
956
957
958
959
960
961
*****
;* SETUP - THIS SUBROUTINE LOADS THE FIRST WORD AFTER THE CALL INTO AX2-15
;* (SYNCH CHAR), LOADS THE SECOND WORD AFTER THE CALL INTO REG 17
;* LOADS THE THIRD WORD INTO AX3-15, AND LOADS THE FOURTH INTO AX3-16.
*****
962 010612 013746 002402      SETUP: MOV  AXNUM,-(SP)    ;SAVE AX BYTE NO.
963 010616 013746 002400      MOV  REGNUM,-(SP)    ;SAVE LU REG NO.
964 010622 012737 000004 002402 MOV  #4,AXNUM        ;SET AX BYTE NO. FOR AX2
965 010630 017637 000004 002374 MOV  @4(SP),WAX15
966 010636 005037 002376      CLR  WAX16
967 010642 004737 004264      JSR  PC,WRITAX       ;SET SYNCH CHAR IN AX2-15, CLEAR AX2-16
968 010646 012737 000017 002400 MOV  #17,REGNUM      ;SET LU REG NO. = 17
969 010654 062766 000002 000004 ADD  #2,4(SP)         ;INCREMENT ARGUMENT POINTER
970 010662 017637 000004 002366 MOV  @4(SP),WRIBYT
971 010670 004737 003722      JSR  PC,WRITLU       ;LOAD REG 17
972 010674 012737 000006 002402 MOV  #6,AXNUM        ;SET AX BYTE NO. FOR AX3
973 010702 062766 000002 000004 ADD  #2,4(SP)         ;INCREMENT ARGUMENT POINTER
974 010710 017637 000004 002374 MOV  @4(SP),WAX15
975 010716 062766 000002 000004 ADD  #2,4(SP)         ;INCR ARGUMENT POINTER

```

```

976 010724 017637 000004 002376      MOV    @4(SP),WAX16
977 010732 013737 002376 002432      MOV    WAX16,SAVLEN      ;STORE TX AND RCV CHAR LENGTH
978 010740 004737 004264                JSR    PC,WRITAX        ;LOAD AX3-15, AX3-16
979 010744 062766 000002 000004      ADD    #2,4(SP)         ;FIX RETURN PC
980 010752 012637 002400                MOV    (SP)+,REGNUM     ;RESTORE LU REG NO.
981 010756 012637 002402                MOV    (SP)+,AXNUM     ;RESTORE AX BYTE NO.
982 010762 005037 002352                CLR    SUBRPC          ;CLEAR SUBROUTINE PC STORAGE
983 010766 000207                RTS     PC              ;RETURN

```

984
985
986
987
988
989

```

:*****
:* LODMSG - THIS SUBROUTINE LOADS THE NO. OF WORDS PASSED IN THE SECOND WORD
:* FOLLOWING THE CALL FROM THE MSG BUFFER WHOSE ADDRESS IS IN THE FIRST
:* WORD FOLLOWING THE CALL, INTO THE TRANSMITTER SILO.
:*****

```

```

994 010770 010146                LODMSG: MOV    R1,-(SP)      ;SAVE R1
995 010772 010246                MOV    R2,-(SP)      ;SAVE R2
996 010774 017601 000004                MOV    @4(SP),R1     ;GET MSG POINTER INTO R1
997 011000 062766 000002 000004      ADD    #2,4(SP)     ;INCR ARG POINTER
998 011006 017602 000004                MOV    @4(SP),R2     ;GET WORD COUNT INTO R2
999 011012 062766 000002 000004      ADD    #2,4(SP)     ;FIX UP RETURN PC
1000 011020 012137 002422 6$:      MOV    (R1)+,TXWORD  ;GET NEXT MSG WORD
1001 011024 004737 005146                JSR    PC,LDTXSI     ;LOAD A WORD INTO TX SILO
1002 011030 005302                DEC    R2            ;DECR COUNT
1003 011032 001372                BNE    6$           ;BR IF NOT DONE YET
1004 011034 004737 005120                JSR    PC,WAIT50     ;WAIT FOR SILO TO RIPPLE
1005 011040 012602                MOV    (SP)+,R2     ;RESTORE R2
1006 011042 012601                MOV    (SP)+,R1     ;RESTORE R1
1007 011044 000207                RTS     PC              ;RETURN

```

1008
1009
1010
1011
1012
1013

```

:*****
:* RXCHAR - THIS SUBROUTINE READS THE RCV SILO AND CLOCKS THE LINE UNIT.
:* FIRST, IT READS THE CHAR FROM THE RCV SILO, AND CHECKS FOR ICIR
:* = 1, IRDY = 0. IT THEN CLOCKS THE LINE UNIT FOR THE NO. OF CYCLES
:* PASSED IN THE WORD FOLLOWING THE CALL, AND THEN CHECKS FOR ICIR
:* = 1, IRDY = 1.
:*****

```

```

1020 011046 010146                RXCHAR: MOV    R1,-(SP)      ;SAVE R1
1021 011050 016637 000002 002352      MOV    2(SP),SUBRPC   ;GET PC OF SUBR CALL
1022 011056 162737 000004 002352      SUB    #4,SUBRPC     ;READ RCV SILO
1023 011064 004737 007326                JSR    PC,RDRXSI     ;ALLOW SILO TO RIPPLE
1024 011070 004737 005120                JSR    PC,WAIT50     ;INIT CYCLE COUNT
1025 011074 005001                CLR    R1            ;SEE IF REQUIRED CYCLES DONE YET
1026 011076 020176 000002 9$:      CMP    R1,@2(SP)     ;BR IF YES
1027 011102 001410                BEQ    12$           ;CHK ICIR = 1, IRDY = 0
1028 011104 004737 006402                JSR    PC,ISIRU?     ;CLK LU 1 CYCLE
1029 011110 000001                1
1030 011112 004737 005226                JSR    PC,STPLU     ;INCR CYCLE COUNT
1031 011116 000001                1
1032 011120 005201                INC    R1

```

```

1033 011122 000765          BR      9$
1034 011124 004737 006402 12$: JSR    PC,ISIRDY      ;CHK ICIR = 1 IRDY = 1
1035 011130 000003          3
1036 011132 062766 000002 000002 ADD    #2,2(SP)      ;FIX RETURN PC
1037 011140 005037 002352 CLR    SUBRPC        ;CLEAR SUBR CALL PC
1038 011144 012601          MOV    (SP)+,R1      ;RESTORE R1
1039 011146 000207          RTS     PC           ;RETURN
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052 011150 013746 002400  CKTBIT: MOV    REGNUM,-(SP) ;SAVE LU REG NO.
1053 011154 010146          MOV    R1,-(SP)     ;SAVE R1
1054 011156 016637 000004 002352 MOV    4(SP),SUBRPC
1055 011164 162737 000004 002352 SUB    #4,SUBRPC     ;GET PC OF SUBROUTINE CALL
1056 011172 012701 000001          MOV    #BIT0,R1     ;INIT BIT POINTER
1057 011176 012737 000017 002400 MOV    #17,REGNUM   ;SET REG NO. = 17
1058 011204 004737 005226 4$: JSR    PC,STPLU     ;CLOCK LINE UNIT 1 CYCLE
1059 011210 000001          1
1060 011212 004737 003644          JSR    PC,READLU    ;READ REG 17
1061 011216 030176 000004          BIT    R1,#4(SP)    ;SEE IF EXPECTED BIT = 1
1062 011222 001013          BNE    9$           ;BR IF YES
1063 011224 132737 000040 002364 BITB   #TXDATA,REDBYT ;SEE IF ACTUAL BIT = 0
1064 011232 001422          BEQ    12$          ;BR IF YES
1065 011234 004737 004500          JSR    PC,GETALL    ;GET REGS FOR PRINTOUT
1066
1067 011240          ;REPORT TXDATA NOT CLEARED
011240 104455          ERRDF 32,EM32,ERR4
011242 000040          TRAP  C$ERDF
011244 013222          .WORD 32
011246 015576          .WORD EM32
1068 011250 000420          .WORD ERR4
1069 011252 132737 000040 002364 9$: BR      20$           ;TAKE ERROR RETURN
1070 011260 001007          BITB   #TXDATA,REDBYT ;SEE IF ACTUAL BIT = 1
1071 011262 004737 004500          BNE    12$          ;BR IF YES
1072          JSR    PC,GETALL    ;GET REGS FOR PRINTOUT
1073          ;REPORT TXDATA BIT NOT SET
011266          ERRDF 33,EM33,ERR4
011266 104455          TRAP  C$ERDF
011270 000041          .WORD 33
011272 013251          .WORD EM33
011274 015576          .WORD ERR4
1074 011276 000405          BR      20$           ;TAKE ERROR EXIT
1075 011300 006301          ASL    R1           ;SHIFT BIT POINTER
1076 011302 020127 000400 12$: CMP    R1,#400      ;SEE IF 8 BITS SCANNED YET
1077 011306 001336          BNE    4$           ;BR IF NO
1078 011310 000405          BR      22$
1079 011312 013706 002346 20$: MOV    PSTACK,SP     ;RESTORE PROGRAM STACK POINTER TO BASE LEVEL
1080 011316 013746 002362          MOV    RETADR,-(SP) ;FIX UP RETURN PC
1081 011322 000406          BR      40$

```


1082 011324 062766 000002 000004 22\$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
1083 011332 012601 MOV (SP)+,R1 ;RESTORE R1
1084 011334 012637 002400 MOV (SP)+,REGNUM ;RESTORE REG NO.
1085 011340 005037 002352 40\$: CLR SUBRPC ;CLEAR SUBR CALL PC
1086 011344 000207 RTS PC ;RETURN

1087
1088
1089
1090
1091
1092
1093
1094
1095
1096

;* LDMSG1 - THIS SUBROUTINE LOADS THE TRANSMITTER SILO WITH MSG1, AND LOADS
;* THE DATA CHARS INTO THE RCV MSG BUFFER (RCVBUF:), AS EXPECTED DATA
;* FOR LATER COMPARISON.

1097 011346 010146 LDMSG1: MOV R1,-(SP) ;SAVE R1
1098 011350 010246 MOV R2,-(SP) ;SAVE R2
1099 011352 004737 010770 JSR PC,LODMSG ;LOAD MSG1 INTO TX SILO
1100 011356 003220 MSG1
1101 011360 000011 9.
1102 011362 012701 003224 MOV #MSG1+4,R1 ;GET POINTER TO MSG1
1103 011366 012702 003262 MOV #RCVBUF,R2 ;GET POINTER TO MSG BUF
1104 011372 012122 3\$: MOV (R1)+,(R2)+ ;LOAD A CHAR INTO MSG BUF
1105 011374 020127 003236 CMP R1,#MSG1+14. ;SEE IF DID LAST DATA CHAR YET
1106 011400 103774 BLO 3\$;BR IF NOT
1107 011402 052762 100400 177776 BIS #CRCCHK!RXBCC,-2(R2) ;SET EXPECTED BCC
1108 011410 012726 000160 MOV #160,(SP)+ ;LOAD HI CRC BYTE
1109 011414 012726 000034 MOV #034,(SP)+ ;LOAD LO CRC BYTE
1110 011420 012602 MOV (SP)+,R2 ;RESTORE R2
1111 011422 012601 MOV (SP)+,R1 ;RESTORE R1
1112 011424 000207 RTS PC ;RETURN

1113
1114
1115
1116
1117

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

.SBTTL GLOBAL ERROR REPORT SECTION

:/
:/ THE GLOBAL ERROR REPORT SECTION CONTAINS ERROR MESSAGES
:/ THAT ARE USED IN MORE THAN ONE TEST.
:/

011426	045	124	045	FMT1: .ASCIZ /%T%06%N/
011431	117	066	045	
011434	116	000		
011436	045	116	045	FMT2: .ASCIZ /%N%AFAILING REG: /
011441	101	106	101	
011444	111	114	111	
011447	116	107	040	
011452	122	105	107	
011455	072	040	000	
011460	045	101	105	FMT3: .ASCIZ /%AEXPECTED: %03%S5%AACTUAL: %03%N/
011463	130	120	105	
011466	103	124	105	
011471	104	072	040	
011474	045	117	063	
011477	045	123	065	
011502	045	101	101	
011505	103	124	125	
011510	101	114	072	
011513	040	045	117	
011516	063	045	116	
011521	000			
011522	045	116	045	FMT4: .ASCIZ /%N%T%N%T%N/
011525	124	045	116	
011530	045	124	045	
011533	116	000		
011535	045	117	063	FMT5: .ASCIZ /%03%S5%03%S5%03%S5%03%N/
011540	045	123	065	
011543	045	117	063	
011546	045	123	065	
011551	045	117	063	
011554	045	123	065	
011557	045	117	063	
011562	045	116	000	
011565	045	123	064	FMT6: .ASCIZ /%S4%03%S5%03%S5%03%S5%03%N/
011570	045	117	063	
011573	045	123	065	
011576	045	117	063	
011601	045	123	065	
011604	045	117	063	
011607	045	123	065	
011612	045	117	063	
011615	045	116	000	
011620	045	124	045	FMT7: .ASCIZ /%T%02%N/
011623	117	062	045	
011626	116	000		
011630	045	101	105	FMT8: .ASCIZ /%AEXTENDED REG AX%01%A-%T%N/
011633	130	124	105	
011636	116	104	105	
011641	104	040	122	

	011644	105	107	040	
	011647	101	130	045	
	011652	117	061	045	
	011655	101	055	045	
	011660	124	045	116	
	011663	000			
17	011664	045	124	045	FMT9: .ASCIZ /%T%N/
	011667	116	000		
18	011671	045	101	120	FMT10: .ASCIZ /%APC OF SUBR CALL: %06%N/
	011674	103	040	117	
	011677	106	040	123	
	011702	125	102	122	
	011705	040	103	101	
	011710	114	114	072	
	011713	040	045	117	
	011716	066	045	116	
	011721	000			
19	011722	045	101	122	FMT11: .ASCIZ /%AREG %02%A LOADED WITH: %03%N/
	011725	105	107	040	
	011730	045	117	062	
	011733	045	101	040	
	011736	114	117	101	
	011741	104	105	104	
	011744	040	127	111	
	011747	124	110	072	
	011752	040	045	117	
	011755	063	045	116	
	011760	000			
20	011761	045	116	045	FMT19: .ASCIZ /%N%ATEST %D2%A NOT RUN%N/
	011764	101	124	105	
	011767	123	124	040	
	011772	045	104	062	
	011775	045	101	040	
	012000	116	117	124	
	012003	040	122	125	
	012006	116	045	116	
	012011	000			
21	012012	045	116	045	FMT24: .ASCIZ /%N%PLEASE INSURE RUN SWITCH ON MICROPROCESSOR IS ON%N/
	012015	101	120	114	
	012020	105	101	123	
	012023	105	040	111	
	012026	116	123	125	
	012031	122	105	040	
	012034	122	125	116	
	012037	040	123	127	
	012042	111	124	103	
	012045	110	040	117	
	012050	116	040	115	
	012053	111	103	122	
	012056	117	120	122	
	012061	117	103	105	
	012064	123	123	117	
	012067	122	040	111	
	012072	123	040	117	
	012075	116	045	116	
	012100	000			

23					
24					
25	012101	103	123	122	EM1: .ASCIZ /CSR ADDRESS TIME-OUT (SELO)/
	012104	040	101	104	
	012107	104	122	105	
	012112	123	123	040	
	012115	124	111	115	
	012120	105	055	117	
	012123	125	124	040	
	012126	050	123	105	
	012131	114	060	051	
	012134	000			
26	012135	122	105	107	EM2: .ASCIZ /REG NOT INITIALIZED BY MST CLR/
	012140	040	116	117	
	012143	124	040	111	
	012146	116	111	124	
	012151	111	101	114	
	012154	111	132	105	
	012157	104	040	102	
	012162	131	040	115	
	012165	123	124	040	
	012170	103	114	122	
	012173	000			
27	012174	122	105	107	EM3: .ASCIZ /REG MISCOMPARE/
	012177	040	115	111	
	012202	123	103	117	
	012205	115	120	101	
	012210	122	105	000	
28	012213	122	105	107	EM4: .ASCIZ /REG NOT INITIALIZED BY UNIBUS RESET (INIT)/
	012216	040	116	117	
	012221	124	040	111	
	012224	116	111	124	
	012227	111	101	114	
	012232	111	132	105	
	012235	104	040	102	
	012240	131	040	125	
	012243	116	111	102	
	012246	125	123	040	
	012251	122	105	123	
	012254	105	124	040	
	012257	050	111	116	
	012262	111	124	051	
	012265	000			
29	012266	115	101	111	EM5: .ASCIZ /MAINT CLK BIT STUCK AT 0/
	012271	116	124	040	
	012274	103	114	113	
	012277	040	102	111	
	012302	124	040	123	
	012305	124	125	103	
	012310	113	040	101	
	012313	124	040	060	
	012316	000			
30	012317	115	101	111	EM6: .ASCIZ /MAINT CLK BIT STUCK AT 1/
	012322	116	124	040	
	012325	103	114	113	
	012330	040	102	111	
	012333	124	040	123	

	012336	124	125	103	
	012341	113	040	101	
	012344	124	040	061	
	012347	000			
31	012350	117	122	104	EM7: .ASCIZ /ORDY NOT SET/
	012353	131	040	116	
	012356	117	124	040	
	012361	123	105	124	
	012364	000			
32	012365	117	122	104	EM8: .ASCIZ /ORDY NOT CLEARED/
	012370	131	040	116	
	012373	117	124	040	
	012376	103	114	105	
	012401	101	122	105	
	012404	104	000		
33	012406	117	103	117	EM9: .ASCIZ /OCOR NOT SET/
	012411	122	040	116	
	012414	117	124	040	
	012417	123	105	124	
	012422	000			
34	012423	117	103	117	EM10: .ASCIZ /OCOR NOT CLEARED/
	012426	122	040	116	
	012431	117	124	040	
	012434	103	114	105	
	012437	101	122	105	
	012442	104	000		
35	012444	117	101	103	EM11: .ASCIZ /OACT NOT SET/
	012447	124	040	116	
	012452	117	124	040	
	012455	123	105	124	
	012460	000			
36	012461	117	101	103	EM12: .ASCIZ /OACT NOT CLEARED/
	012464	124	040	116	
	012467	117	124	040	
	012472	103	114	105	
	012475	101	122	105	
	012500	104	000		
37	012502	125	116	122	EM13: .ASCIZ /UNRR NOT CLEARED BY SOM/
	012505	122	040	116	
	012510	117	124	040	
	012513	103	114	105	
	012516	101	122	105	
	012521	104	040	102	
	012524	131	040	123	
	012527	117	115	000	
38	012532	125	116	122	EM14: .ASCIZ /UNRR NOT SET/
	012535	122	040	116	
	012540	117	124	040	
	012543	123	105	124	
	012546	000			
39	012547	125	116	122	EM15: .ASCIZ /UNRR NOT CLEARED BY OC/
	012552	122	040	116	
	012555	117	124	040	
	012560	103	114	105	
	012563	101	122	105	
	012566	104	040	102	
	012571	131	040	117	

	012574	103	000		
40	012576	125	116	122	EM16: .ASCIZ /UNRR NOT CLEARED/
	012601	122	040	116	
	012604	117	124	040	
	012607	103	114	105	
	012612	101	122	105	
	012615	104	000		
41	012617	111	122	104	EM17: .ASCIZ /IRDY NOT SET/
	012622	131	040	116	
	012625	117	124	040	
	012630	123	105	124	
	012633	000			
42	012634	111	122	104	EM18: .ASCIZ /IRDY NOT CLEARED/
	012637	131	040	116	
	012642	117	124	040	
	012645	103	114	105	
	012650	101	122	105	
	012653	104	000		
43	012655	111	103	111	EM19: .ASCIZ /ICIR NOT SET/
	012660	122	040	116	
	012663	117	124	040	
	012666	123	105	124	
	012671	000			
44	012672	111	103	111	EM20: .ASCIZ /ICIR NOT CLEARED/
	012675	122	040	116	
	012700	117	124	040	
	012703	103	114	105	
	012706	101	122	105	
	012711	104	000		
45	012713	111	101	103	EM21: .ASCIZ /IACT NOT SET/
	012716	124	040	116	
	012721	117	124	040	
	012724	123	105	124	
	012727	000			
46	012730	111	101	103	EM22: .ASCIZ /IACT NOT CLEARED/
	012733	124	040	116	
	012736	117	124	040	
	012741	103	114	105	
	012744	101	122	105	
	012747	104	000		
47	012751	104	123	123	EM23: .ASCIZ /DSSI NOT CLEARED/
	012754	111	040	116	
	012757	117	124	040	
	012762	103	114	105	
	012765	101	122	105	
	012770	104	000		
48	012772	104	123	123	EM24: .ASCIZ /DSSI NOT SET/
	012775	111	040	116	
	013000	117	124	040	
	013003	123	105	124	
	013006	000			
49	013007	104	123	123	EM25: .ASCIZ /DSSI NOT CLEARED BY MST CLR/
	013012	111	040	116	
	013015	117	124	040	
	013020	103	114	105	
	013023	101	122	105	
	013026	104	040	102	

	013031	131	040	115	
	013034	123	124	040	
	013037	103	114	122	
	013042	000			
50	013043	111	116	103	EM26: .ASCIZ /INCORRECT DATA CHAR RCV'D/
	013046	117	122	122	
	013051	105	103	124	
	013054	040	104	101	
	013057	124	101	040	
	013062	103	110	101	
	013065	122	040	122	
	013070	103	126	047	
	013073	104	000		
51	013075	111	116	103	EM27: .ASCIZ /INCORRECT CRC BYTE RCV'D/
	013100	117	122	122	
	013103	105	103	124	
	013106	040	103	122	
	013111	103	040	102	
	013114	131	124	105	
	013117	040	122	103	
	013122	126	047	104	
	013125	000			
52	013126	122	123	117	EM28: .ASCIZ /RSOM NOT CLEARED/
	013131	115	040	116	
	013134	117	124	040	
	013137	103	114	105	
	013142	101	122	105	
	013145	104	000		
53	013147	122	123	117	EM29: .ASCIZ /RSOM NOT SET/
	013152	115	040	116	
	013155	117	124	040	
	013160	123	105	124	
	013163	000			
54	013164	122	105	117	EM30: .ASCIZ /REOM NOT CLEARED/
	013167	115	040	116	
	013172	117	124	040	
	013175	103	114	105	
	013200	101	122	105	
	013203	104	000		
55	013205	122	105	117	EM31: .ASCIZ /REOM NOT SET/
	013210	115	040	116	
	013213	117	124	040	
	013216	123	105	124	
	013221	000			
56	013222	124	130	104	EM32: .ASCIZ /TXDATA BIT NOT CLEARED/
	013225	101	124	101	
	013230	040	102	111	
	013233	124	040	116	
	013236	117	124	040	
	013241	103	114	105	
	013244	101	122	105	
	013247	104	000		
57	013251	124	130	104	EM33: .ASCIZ /TXDATA BIT NOT SET/
	013254	101	124	101	
	013257	040	102	111	
	013262	124	040	116	
	013265	117	124	040	

	013270	123	105	124	
	013273	000			
58	013274	122	103	126	EM34: .ASCIZ /RCV'D DATA MISCOMPARE/
	013277	047	104	040	
	013302	104	101	124	
	013305	101	040	115	
	013310	111	123	103	
	013313	117	115	120	
	013316	101	122	105	
	013321	000			
59	013322	102	103	103	EM35: .ASCIZ /BCC NOT CLEARED/
	013325	040	116	117	
	013330	124	040	103	
	013333	114	105	101	
	013336	122	105	104	
	013341	000			
60	013342	102	103	103	EM36: .ASCIZ /BCC NOT SET/
	013345	040	116	117	
	013350	124	040	123	
	013353	105	124	000	
61	013356	105	102	114	EM37: .ASCIZ /EBLK NOT CLEARED/
	013361	113	040	116	
	013364	117	124	040	
	013367	103	114	105	
	013372	101	122	105	
	013375	104	000		
62	013377	105	102	114	EM38: .ASCIZ /EBLK NOT SET/
	013402	113	040	116	
	013405	117	124	040	
	013410	123	105	124	
	013413	000			
63	013414	122	101	102	EM39: .ASCIZ /RAB NOT CLEARED/
	013417	040	116	117	
	013422	124	040	103	
	013425	114	105	101	
	013430	122	105	104	
	013433	000			
64	013434	122	101	102	EM40: .ASCIZ /RAB NOT SET/
	013437	040	116	117	
	013442	124	040	123	
	013445	105	124	000	
65	013450	117	126	122	EM41: .ASCIZ /OVRR NOT CLEARED/
	013453	122	040	116	
	013456	117	124	040	
	013461	103	114	105	
	013464	101	122	105	
	013467	104	000		
66	013471	117	126	122	EM42: .ASCIZ /OVRR NOT SET/
	013474	122	040	116	
	013477	117	124	040	
	013502	123	105	124	
	013505	000			
67	013506	123	127	040	EM43: .ASCIZ /SW PACK #1 INCORRECT/
	013511	120	101	103	
	013514	113	040	043	
	013517	061	040	111	
	013522	116	103	117	

	013525	122	122	105	
	013530	103	124	000	
68	013533	123	127	040	EM44: .ASCIZ /SW PACK #2 INCORRECT/
	013536	120	101	103	
	013541	113	040	043	
	013544	062	040	111	
	013547	116	103	117	
	013552	122	122	105	
	013555	103	124	000	
69	013560	123	127	040	EM45: .ASCIZ /SW PACK #3 INCORRECT/
	013563	120	101	103	
	013566	113	040	043	
	013571	063	040	111	
	013574	116	103	117	
	013577	122	122	105	
	013602	103	124	000	
70	013605	122	103	126	EM46: .ASCIZ /RCV SILO NOT CLEARED BY IC/
	013610	040	123	111	
	013613	114	117	040	
	013616	116	117	124	
	013621	040	103	114	
	013624	105	101	122	
	013627	105	104	040	
	013632	102	131	040	
	013635	111	103	000	
71	013640	101	123	123	EM47: .ASCIZ /ASSEMB BIT COUNT INCORRECT/
	013643	105	115	102	
	013646	040	102	111	
	013651	124	040	103	
	013654	117	125	116	
	013657	124	040	111	
	013662	116	103	117	
	013665	122	122	105	
	013670	103	124	000	
72	013673	117	104	104	EM48: .ASCIZ /ODD VRC PARITY BIT NOT SET/
	013676	040	126	122	
	013701	103	040	120	
	013704	101	122	111	
	013707	124	131	040	
	013712	102	111	124	
	013715	040	116	117	
	013720	124	040	123	
	013723	105	124	000	
73	013726	117	104	104	EM49: .ASCIZ /ODD VRC PARITY BIT NOT CLEARED/
	013731	040	126	122	
	013734	103	040	120	
	013737	101	122	111	
	013742	124	131	040	
	013745	102	111	124	
	013750	040	116	117	
	013753	124	040	103	
	013756	114	105	101	
	013761	122	105	104	
	013764	000			
74	013765	105	126	105	EM50: .ASCIZ /EVEN VRC PARITY BIT NOT SET/
	013770	116	040	126	
	013773	122	103	040	

	013776	120	101	122	
	014001	111	124	131	
	014004	040	102	111	
	014007	124	040	116	
	014012	117	124	040	
	014015	123	105	124	
	014020	000			
75	014021	105	126	105	EM51: .ASCIZ /EVEN VRC PARITY BIT NOT CLEARED/
	014024	116	040	126	
	014027	122	103	040	
	014032	120	101	122	
	014035	111	124	131	
	014040	040	102	111	
	014043	124	040	116	
	014046	117	124	040	
	014051	103	114	105	
	014054	101	122	105	
	014057	104	000		
76	014061	122	105	101	EM52: .ASCIZ /READY NOT SET AFTER AX REG WRITE/
	014064	104	131	040	
	014067	116	117	124	
	014072	040	123	105	
	014075	124	040	101	
	014100	106	124	105	
	014103	122	040	101	
	014106	130	040	122	
	014111	105	107	040	
	014114	127	122	111	
	014117	124	105	000	
77	014122	122	105	101	EM53: .ASCIZ /READY NOT SET AFTER AX REG READ/
	014125	104	131	040	
	014130	116	117	124	
	014133	040	123	105	
	014136	124	040	101	
	014141	106	124	105	
	014144	122	040	101	
	014147	130	040	122	
	014152	105	107	040	
	014155	122	105	101	
	014160	104	000		
78	014162	124	130	040	EM54: .ASCIZ /TX UNDERRUN ERROR/
	014165	125	116	104	
	014170	105	122	122	
	014173	125	116	040	
	014176	105	122	122	
	014201	117	122	000	
79	014204	122	124	123	EM60: .ASCIZ /RTS NOT SET/
	014207	040	116	117	
	014212	124	040	123	
	014215	105	124	000	
80	014220	122	124	123	EM65: .ASCIZ /RTS NOT CLEARED/
	014223	040	116	117	
	014226	124	040	103	
	014231	114	105	101	
	014234	122	105	104	
81	014237	000			

82					
83					
84	014240	111	116	102	DH1: .ASCIZ &INBUS/OUTBUS REG &
	014243	125	123	057	
	014246	117	125	124	
	014251	102	125	123	
	014254	040	122	105	
	014257	107	040	000	
85	014262	114	111	116	DH2: .ASCIZ /LINE UNIT INBUS REGS :/
	014265	105	040	125	
	014270	116	111	124	
	014273	040	111	116	
	014276	102	125	123	
	014301	040	122	105	
	014304	107	123	040	
	014307	072	000		
86	014311	122	105	107	DH3: .ASCIZ /REG10 REG11 REG12 REG13/
	014314	061	060	040	
	014317	040	040	122	
	014322	105	107	061	
	014325	061	040	040	
	014330	040	122	105	
	014333	107	061	062	
	014336	040	040	040	
	014341	122	105	107	
	014344	061	063	000	
87	014347	040	040	040	DH4: .ASCIZ / REG14 REG15 REG16 REG17/
	014352	040	122	105	
	014355	107	061	064	
	014360	040	040	040	
	014363	122	105	107	
	014366	061	065	040	
	014371	040	040	122	
	014374	105	107	061	
	014377	066	040	040	
	014402	040	122	105	
	014405	107	061	067	
	014410	000			
88	014411	061	065	000	DH5: .ASCIZ /15/
89	014414	061	066	000	DH6: .ASCIZ /16/
90	014417	114	111	116	DH7: .ASCIZ /LINE UNIT EXTENDED REGS :/
	014422	105	040	125	
	014425	116	111	124	
	014430	040	105	130	
	014433	124	105	116	
	014436	104	105	104	
	014441	040	122	105	
	014444	107	123	040	
	014447	072	000		
91	014451	101	130	060	DH8: .ASCIZ /AX0-15 AX0-16 AX1-15 AX1-16/
	014454	055	061	065	
	014457	040	040	101	
	014462	130	060	055	
	014465	061	066	040	
	014470	040	101	130	
	014473	061	055	061	
	014476	065	040	040	

	014501	101	130	061				
	014504	055	061	066				
	014507	000						
92	014510	040	040	040	DH9:	.ASCIZ /	AX2-15	AX2-16 AX3-15 AX3-16/
	014513	040	101	130				
	014516	062	055	061				
	014521	065	040	040				
	014524	101	130	062				
	014527	055	061	066				
	014532	040	040	101				
	014535	130	063	055				
	014540	061	065	040				
	014543	040	101	130				
	014546	063	055	061				
	014551	066	000					
93								
94						.EVEN		
95								
96								
97								
98								
99								
100	014554				BGNMSG	ERR1		
	014554							ERR1::
101	014554				PRINTB	#FMT1,#ADDRES,MPCSR		
	014554	013746	002446					MOV MPCSR,-(SP)
	014560	012746	035574					MOV #ADDRES,-(SP)
	014564	012746	011426					MOV #FMT1,-(SP)
	014570	012746	000003					MOV #3,-(SP)
	014574	010600						MOV SP,R0
	014576	104414						TRAP C\$PNTB
	014600	062706	000010					ADD #10,SP
102	014604				ENDMSG			
	014604							L10002:
	014604	104423						TRAP C\$MSG
103								
104								
105								
106	014606				BGNMSG	ERR2		
	014606							ERR2::
107	014606				PRINTB	#FMT1,#ADDRES,MPCSR		
	014606	013746	002446					MOV MPCSR,-(SP)
	014612	012746	035574					MOV #ADDRES,-(SP)
	014616	012746	011426					MOV #FMT1,-(SP)
	014622	012746	000003					MOV #3,-(SP)
	014626	010600						MOV SP,R0
	014630	104414						TRAP C\$PNTB
	014632	062706	000010					ADD #10,SP
108	014636				PRINTB	#FMT2		
	014636	012746	011436					MOV #FMT2,-(SP)
	014642	012746	000001					MOV #1,-(SP)
	014646	010600						MOV SP,R0
	014650	104414						TRAP C\$PNTB
	014652	062706	000004					ADD #4,SP
109	014656				PRINTB	#FMT7,#DH1,REGNUM		
	014656	013746	002400					MOV REGNUM,-(SP)
	014662	012746	014240					MOV #DH1,-(SP)

	014666	012746	011620		MOV	#FMT7,-(SP)
	014672	012746	000003		MOV	#3,-(SP)
	014676	010600			MOV	SP,R0
	014700	104414			TRAP	C\$PNTB
	014702	062706	000010		ADD	#10,SP
110	014706			PRINTB	#FMT3,GOODAT,BADDAT	
	014706	013746	002406		MOV	BADDAT,-(SP)
	014712	013746	002404		MOV	GOODAT,-(SP)
	014716	012746	011460		MOV	#FMT3,-(SP)
	014722	012746	000003		MOV	#3,-(SP)
	014726	010600			MOV	SP,R0
	014730	104414			TRAP	C\$PNTB
	014732	062706	000010		ADD	#10,SP
111	014736			PRINTX	#FMT4,#DH2,#DH3	
	014736	012746	014311		MOV	#DH3,-(SP)
	014742	012746	014262		MOV	#DH2,-(SP)
	014746	012746	011522		MOV	#FMT4,-(SP)
	014752	012746	000003		MOV	#3,-(SP)
	014756	010600			MOV	SP,R0
	014760	104415			TRAP	C\$PNTX
	014762	062706	000010		ADD	#10,SP
112	014766			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
	014766	013746	002310		MOV	LUR13,-(SP)
	014772	013746	002306		MOV	LUR12,-(SP)
	014776	013746	002304		MOV	LUR11,-(SP)
	015002	013746	002302		MOV	LUR10,-(SP)
	015006	012746	011535		MOV	#FMT5,-(SP)
	015012	012746	000005		MOV	#5,-(SP)
	015016	010600			MOV	SP,R0
	015020	104415			TRAP	C\$PNTX
	015022	062706	000014		ADD	#14,SP
113	015026			PRINTX	#FMT9,#DH4	
	015026	012746	014347		MOV	#DH4,-(SP)
	015032	012746	011664		MOV	#FMT9,-(SP)
	015036	012746	000002		MOV	#2,-(SP)
	015042	010600			MOV	SP,R0
	015044	104415			TRAP	C\$PNTX
	015046	062706	000006		ADD	#6,SP
114	015052			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17	
	015052	013746	002320		MOV	LUR17,-(SP)
	015056	013746	002316		MOV	LUR16,-(SP)
	015062	013746	002314		MOV	LUR15,-(SP)
	015066	013746	002312		MOV	LUR14,-(SP)
	015072	012746	011565		MOV	#FMT6,-(SP)
	015076	012746	000005		MOV	#5,-(SP)
	015102	010600			MOV	SP,R0
	015104	104415			TRAP	C\$PNTX
	015106	062706	000014		ADD	#14,SP
115	015112			ENDMSG		
	015112					
	015112	104423				
116						
117						
118						
119						
120						
121	015114			BGNMSG	ERR3	

L10003:

TRAP C\$MSG

ERR3::

122	015114			PRINTB #FMT1,#ADDRES,MPCSR	
	015114	013746	002446		MOV MPCSR,-(SP)
	015120	012746	035574		MOV #ADDRES,-(SP)
	015124	012746	011426		MOV #FMT1,-(SP)
	015130	012746	000003		MOV #3,-(SP)
	015134	010600			MOV SP,R0
	015136	104414			TRAP C\$PNTB
	015140	062706	000010		ADD #10,SP
123	015144			PRINTB #FMT2	
	015144	012746	011436		MOV #FMT2,-(SP)
	015150	012746	000001		MOV #1,-(SP)
	015154	010600			MOV SP,R0
	015156	104414			TRAP C\$PNTB
	015160	062706	000004		ADD #4,SP
124	015164			PRINTB #FMT8,TMP1,TMP0	
	015164	013746	002530		MOV TMP0,-(SP)
	015170	013746	002532		MOV TMP1,-(SP)
	015174	012746	011630		MOV #FMT8,-(SP)
	015200	012746	000003		MOV #3,-(SP)
	015204	010600			MOV SP,R0
	015206	104414			TRAP C\$PNTB
	015210	062706	000010		ADD #10,SP
125	015214			PRINTB #FMT3,GOODAT,BADDAT	
	015214	013746	002406		MOV BADDAT,-(SP)
	015220	013746	002404		MOV GOODAT,-(SP)
	015224	012746	011460		MOV #FMT3,-(SP)
	015230	012746	000003		MOV #3,-(SP)
	015234	010600			MOV SP,R0
	015236	104414			TRAP C\$PNTB
	015240	062706	000010		ADD #10,SP
126	015244			PRINTX #FMT4,#DH2,#DH3	
	015244	012746	014311		MOV #DH3,-(SP)
	015250	012746	014262		MOV #DH2,-(SP)
	015254	012746	011522		MOV #FMT4,-(SP)
	015260	012746	000003		MOV #3,-(SP)
	015264	010600			MOV SP,R0
	015266	104415			TRAP C\$PNTX
	015270	062706	000010		ADD #10,SP
127	015274			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13	
	015274	013746	002310		MOV LUR13,-(SP)
	015300	013746	002306		MOV LUR12,-(SP)
	015304	013746	002304		MOV LUR11,-(SP)
	015310	013746	002302		MOV LUR10,-(SP)
	015314	012746	011535		MOV #FMT5,-(SP)
	015320	012746	000005		MOV #5,-(SP)
	015324	010600			MOV SP,R0
	015326	104415			TRAP C\$PNTX
	015330	062706	000014		ADD #14,SP
128	015334			PRINTX #FMT9,#DH4	
	015334	012746	014347		MOV #DH4,-(SP)
	015340	012746	011664		MOV #FMT9,-(SP)
	015344	012746	000002		MOV #2,-(SP)
	015350	010600			MOV SP,R0
	015352	104415			TRAP C\$PNTX
	015354	062706	000006		ADD #6,SP
129	015360			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17	

015360	013746	002320			MOV	LUR17,-(SP)
015364	013746	002316			MOV	LUR16,-(SP)
015370	013746	002314			MOV	LUR15,-(SP)
015374	013746	002312			MOV	LUR14,-(SP)
015400	012746	011565			MOV	#FMT6,-(SP)
015404	012746	000005			MOV	#5,-(SP)
015410	010600				MOV	SP,R0
015412	104415				TRAP	C\$PNTX
015414	062706	000014			ADD	#14,SP
130 015420			PRINTX	#FMT4,#DH7,#DH8		
015420	012746	014451			MOV	#DH8,-(SP)
015424	012746	014417			MOV	#DH7,-(SP)
015430	012746	011522			MOV	#FMT4,-(SP)
015434	012746	000003			MOV	#3,-(SP)
015440	010600				MOV	SP,R0
015442	104415				TRAP	C\$PNTX
015444	062706	000010			ADD	#10,SP
131 015450			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16		
015450	013746	002330			MOV	AX1.16,-(SP)
015454	013746	002326			MOV	AX1.15,-(SP)
015460	013746	002324			MOV	AX0.16,-(SP)
015464	013746	002322			MOV	AX0.15,-(SP)
015470	012746	011535			MOV	#FMT5,-(SP)
015474	012746	000005			MOV	#5,-(SP)
015500	010600				MOV	SP,R0
015502	104415				TRAP	C\$PNTX
015504	062706	000014			ADD	#14,SP
132 015510			PRINTX	#FMT9,#DH9		
015510	012746	014510			MOV	#DH9,-(SP)
015514	012746	011664			MOV	#FMT9,-(SP)
015520	012746	000002			MOV	#2,-(SP)
015524	010600				MOV	SP,R0
015526	104415				TRAP	C\$PNTX
015530	062706	000006			ADD	#6,SP
133 015534			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16		
015534	013746	002340			MOV	AX3.16,-(SP)
015540	013746	002336			MOV	AX3.15,-(SP)
015544	013746	002334			MOV	AX2.16,-(SP)
015550	013746	002332			MOV	AX2.15,-(SP)
015554	012746	011565			MOV	#FMT6,-(SP)
015560	012746	000005			MOV	#5,-(SP)
015564	010600				MOV	SP,R0
015566	104415				TRAP	C\$PNTX
015570	062706	000014			ADD	#14,SP
134 015574			ENDMSG			
015574						
015574	104423				L10004:	TRAP C\$MSG
135						
136						
137						
138						
139						
140 015576			BGNMSG	ERR4		
015576					ERR4::	
141 015576			PRINTB	#FMT10,SUBRPC		
015576	013746	002352			MOV	SUBRPC,-(SP)
015602	012746	011671			MOV	#FMT10,-(SP)

	015606	012746	000002		MOV	#2,-(SP)
	015612	010600			MOV	SP,R0
	015614	104414			TRAP	C\$PNTB
	015616	062706	000006		ADD	#6,SP
142	015622			PRINTB	#FMT1,#ADDRES,MPCSR	
	015622	013746	002446		MOV	MPCSR,-(SP)
	015626	012746	035574		MOV	#ADDRES,-(SP)
	015632	012746	011426		MOV	#FMT1,-(SP)
	015636	012746	000003		MOV	#3,-(SP)
	015642	010600			MOV	SP,R0
	015644	104414			TRAP	C\$PNTB
	015646	062706	000010		ADD	#10,SP
143	015652			PRINTB	#FMT2	
	015652	012746	011436		MOV	#FMT2,-(SP)
	015656	012746	000001		MOV	#1,-(SP)
	015662	010600			MOV	SP,R0
	015664	104414			TRAP	C\$PNTB
	015666	062706	000004		ADD	#4,SP
144	015672			PRINTB	#FMT7,#DH1,REGNUM	
	015672	013746	002400		MOV	REGNUM,-(SP)
	015676	012746	014240		MOV	#DH1,-(SP)
	015702	012746	011620		MOV	#FMT7,-(SP)
	015706	012746	000003		MOV	#3,-(SP)
	015712	010600			MOV	SP,R0
	015714	104414			TRAP	C\$PNTB
	015716	062706	000010		ADD	#10,SP
145	015722			PRINTX	#FMT4,#DH2,#DH3	
	015722	012746	014311		MOV	#DH3,-(SP)
	015726	012746	014262		MOV	#DH2,-(SP)
	015732	012746	011522		MOV	#FMT4,-(SP)
	015736	012746	000003		MOV	#3,-(SP)
	015742	010600			MOV	SP,R0
	015744	104415			TRAP	C\$PNTX
	015746	062706	000010		ADD	#10,SP
146	015752			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
	015752	013746	002310		MOV	LUR13,-(SP)
	015756	013746	002306		MOV	LUR12,-(SP)
	015762	013746	002304		MOV	LUR11,-(SP)
	015766	013746	002302		MOV	LUR10,-(SP)
	015772	012746	011535		MOV	#FMT5,-(SP)
	015776	012746	000005		MOV	#5,-(SP)
	016002	010600			MOV	SP,R0
	016004	104415			TRAP	C\$PNTX
	016006	062706	000014		ADD	#14,SP
147	016012			PRINTX	#FMT9,#DH4	
	016012	012746	014347		MOV	#DH4,-(SP)
	016016	012746	011664		MOV	#FMT9,-(SP)
	016022	012746	000002		MOV	#2,-(SP)
	016026	010600			MOV	SP,R0
	016030	104415			TRAP	C\$PNTX
	016032	062706	000006		ADD	#6,SP
148	016036			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17	
	016036	013746	002320		MOV	LUR17,-(SP)
	016042	013746	002316		MOV	LUR16,-(SP)
	016046	013746	002314		MOV	LUR15,-(SP)
	016052	013746	002312		MOV	LUR14,-(SP)
	016056	012746	011565		MOV	#FMT6,-(SP)

016062 012746 000005
016066 010600
016070 104415
149 016072 062706 000014
016076 012746 014451
016102 012746 014417
016106 012746 011522
016112 012746 000003
016116 010600
016120 104415
150 016122 062706 000010
016126 013746 002330
016132 013746 002326
016136 013746 002324
016142 013746 002322
016146 012746 011535
016152 012746 000005
016156 010600
016160 104415
151 016162 062706 000014
016166 012746 014510
016172 012746 011664
016176 012746 000002
016202 010600
016204 104415
152 016206 062706 000006
016212 013746 002340
016216 013746 002336
016222 013746 002334
016226 013746 002332
016232 012746 011565
016236 012746 000005
016242 010600
016244 104415
153 016246 062706 000014
016252
016252 104423
154
155
156
157
158
159 016254
016254
160 016254
016254 013746 002446
016260 012746 035574
016264 012746 011426
016270 012746 000003
016274 010600
016276 104414
016300 062706 000010

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

PRINTX #FMT9,#DH9

PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16

ENDMSG

BGNMSG ERR5

PRINTB #FMT1,#ADDRES,MPCSR

MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP
MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP
MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP
MOV #DH9,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP
MOV AX3.16,-(SP)
MOV AX3.15,-(SP)
MOV AX2.16,-(SP)
MOV AX2.15,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP
L10005:
TRAP C\$MSG
ERR5::
MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

161	016304			PRINTB #FMT11,REGNUM,LOADAT			
	016304	013746	002410		MOV	LOADAT,-(SP)	
	016310	013746	002400		MOV	REGNUM,-(SP)	
	016314	012746	011722		MOV	#FMT11,-(SP)	
	016320	012746	000003		MOV	#3,-(SP)	
	016324	010600			MOV	SP,R0	
	016326	104414			TRAP	C\$PNTB	
	016330	062706	000010		ADD	#10,SP	
162	016334			PRINTB #FMT2			
	016334	012746	011436		MOV	#FMT2,-(SP)	
	016340	012746	000001		MOV	#1,-(SP)	
	016344	010600			MOV	SP,R0	
	016346	104414			TRAP	C\$PNTB	
	016350	062706	000004		ADD	#4,SP	
163	016354			PRINTB #FMT8,TMP1,TMP0			
	016354	013746	002530		MOV	TMP0,-(SP)	
	016360	013746	002532		MOV	TMP1,-(SP)	
	016364	012746	011630		MOV	#FMT8,-(SP)	
	016370	012746	000003		MOV	#3,-(SP)	
	016374	010600			MOV	SP,R0	
	016376	104414			TRAP	C\$PNTB	
	016400	062706	000010		ADD	#10,SP	
164	016404			PRINTB #FMT3,GOODAT,BADDAT			
	016404	013746	002406		MOV	BADDAT,-(SP)	
	016410	013746	002404		MOV	GOODAT,-(SP)	
	016414	012746	011460		MOV	#FMT3,-(SP)	
	016420	012746	000003		MOV	#3,-(SP)	
	016424	010600			MOV	SP,R0	
	016426	104414			TRAP	C\$PNTB	
	016430	062706	000010		ADD	#10,SP	
165	016434			PRINTX #FMT4,#DH2,#DH3			
	016434	012746	014311		MOV	#DH3,-(SP)	
	016440	012746	014262		MOV	#DH2,-(SP)	
	016444	012746	011522		MOV	#FMT4,-(SP)	
	016450	012746	000003		MOV	#3,-(SP)	
	016454	010600			MOV	SP,R0	
	016456	104415			TRAP	C\$PNTX	
	016460	062706	000010		ADD	#10,SP	
166	016464			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13			
	016464	013746	002310		MOV	LUR13,-(SP)	
	016470	013746	002306		MOV	LUR12,-(SP)	
	016474	013746	002304		MOV	LUR11,-(SP)	
	016500	013746	002302		MOV	LUR10,-(SP)	
	016504	012746	011535		MOV	#FMT5,-(SP)	
	016510	012746	000005		MOV	#5,-(SP)	
	016514	010600			MOV	SP,R0	
	016516	104415			TRAP	C\$PNTX	
	016520	062706	000014		ADD	#14,SP	
167	016524			PRINTX #FMT9,#DH4			
	016524	012746	014347		MOV	#DH4,-(SP)	
	016530	012746	011664		MOV	#FMT9,-(SP)	
	016534	012746	000002		MOV	#2,-(SP)	
	016540	010600			MOV	SP,R0	
	016542	104415			TRAP	C\$PNTX	
	016544	062706	000006		ADD	#6,SP	
168	016550			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17			
	016550	013746	002320		MOV	LUR17,-(SP)	

	016554	013746	002316		MOV	LUR16,-(SP)
	016560	013746	002314		MOV	LUR15,-(SP)
	016564	013746	002312		MOV	LUR14,-(SP)
	016570	012746	011565		MOV	#FMT6,-(SP)
	016574	012746	000005		MOV	#5,-(SP)
	016600	010600			MOV	SP,R0
	016602	104415			TRAP	C\$PNTX
	016604	062706	000014		ADD	#14,SP
169	016610			PRINTX		#FMT4,#DH7,#DH8
	016610	012746	014451		MOV	#DH8,-(SP)
	016614	012746	014417		MOV	#DH7,-(SP)
	016620	012746	011522		MOV	#FMT4,-(SP)
	016624	012746	000003		MOV	#3,-(SP)
	016630	010600			MOV	SP,R0
	016632	104415			TRAP	C\$PNTX
	016634	062706	000010		ADD	#10,SP
170	016640			PRINTX		#FMT5,AX0.15,AX0.16,AX1.15,AX1.16
	016640	013746	002330		MOV	AX1.16,-(SP)
	016644	013746	002326		MOV	AX1.15,-(SP)
	016650	013746	002324		MOV	AX0.16,-(SP)
	016654	013746	002322		MOV	AX0.15,-(SP)
	016660	012746	011535		MOV	#FMT5,-(SP)
	016664	012746	000005		MOV	#5,-(SP)
	016670	010600			MOV	SP,R0
	016672	104415			TRAP	C\$PNTX
	016674	062706	000014		ADD	#14,SP
171	016700			PRINTX		#FMT9,#DH9
	016700	012746	014510		MOV	#DH9,-(SP)
	016704	012746	011664		MOV	#FMT9,-(SP)
	016710	012746	000002		MOV	#2,-(SP)
	016714	010600			MOV	SP,R0
	016716	104415			TRAP	C\$PNTX
	016720	062706	000006		ADD	#6,SP
172	016724			PRINTX		#FMT6,AX2.15,AX2.16,AX3.15,AX3.16
	016724	013746	002340		MOV	AX3.16,-(SP)
	016730	013746	002336		MOV	AX3.15,-(SP)
	016734	013746	002334		MOV	AX2.16,-(SP)
	016740	013746	002332		MOV	AX2.15,-(SP)
	016744	012746	011565		MOV	#FMT6,-(SP)
	016750	012746	000005		MOV	#5,-(SP)
	016754	010600			MOV	SP,R0
	016756	104415			TRAP	C\$PNTX
	016760	062706	000014		ADD	#14,SP
173	016764			ENDMSG		
	016764					
	016764	104423			L10006:	TRAP C\$MSG
174						
175						
176						
177						
178						
179	016766			BGNMSG	ERR6	
	016766					ERR6::
180	016766			PRINTB	#FMT10,SUBRPC	
	016766	013746	002352		MOV	SUBRPC,-(SP)
	016772	012746	011671		MOV	#FMT10,-(SP)
	016776	012746	000002		MOV	#2,-(SP)

	017002	010600			MOV	SP,R0
	017004	104414			TRAP	C\$PNTB
	017006	062706	000006		ADD	#6,SP
181	017012			PRINTB #FMT1,#ADDRES,MPCSR		
	017012	013746	002446		MOV	MPCSR,-(SP)
	017016	012746	035574		MOV	#ADDRES,-(SP)
	017022	012746	011426		MOV	#FMT1,-(SP)
	017026	012746	000003		MOV	#3,-(SP)
	017032	010600			MOV	SP,R0
	017034	104414			TRAP	C\$PNTB
	017036	062706	000010		ADD	#10,SP
182	017042			PRINTB #FMT2		
	017042	012746	011436		MOV	#FMT2,-(SP)
	017046	012746	000001		MOV	#1,-(SP)
	017052	010600			MOV	SP,R0
	017054	104414			TRAP	C\$PNTB
	017056	062706	000004		ADD	#4,SP
183	017062			PRINTB #FMT8,TMP1,TMP0		
	017062	013746	002530		MOV	TMP0,-(SP)
	017066	013746	002532		MOV	TMP1,-(SP)
	017072	012746	011630		MOV	#FMT8,-(SP)
	017076	012746	000003		MOV	#3,-(SP)
	017102	010600			MOV	SP,R0
	017104	104414			TRAP	C\$PNTB
	017106	062706	000010		ADD	#10,SP
184	017112			PRINTX #FMT4,#DH2,#DH3		
	017112	012746	014311		MOV	#DH3,-(SP)
	017116	012746	014262		MOV	#DH2,-(SP)
	017122	012746	011522		MOV	#FMT4,-(SP)
	017126	012746	000003		MOV	#3,-(SP)
	017132	010600			MOV	SP,R0
	017134	104415			TRAP	C\$PNTX
	017136	062706	000010		ADD	#10,SP
185	017142			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13		
	017142	013746	002310		MOV	LUR13,-(SP)
	017146	013746	002306		MOV	LUR12,-(SP)
	017152	013746	002304		MOV	LUR11,-(SP)
	017156	013746	002302		MOV	LUR10,-(SP)
	017162	012746	011535		MOV	#FMT5,-(SP)
	017166	012746	000005		MOV	#5,-(SP)
	017172	010600			MOV	SP,R0
	017174	104415			TRAP	C\$PNTX
	017176	062706	000014		ADD	#14,SP
186	017202			PRINTX #FMT9,#DH4		
	017202	012746	014347		MOV	#DH4,-(SP)
	017206	012746	011664		MOV	#FMT9,-(SP)
	017212	012746	000002		MOV	#2,-(SP)
	017216	010600			MOV	SP,R0
	017220	104415			TRAP	C\$PNTX
	017222	062706	000006		ADD	#6,SP
187	017226			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17		
	017226	013746	002320		MOV	LUR17,-(SP)
	017232	013746	002316		MOV	LUR16,-(SP)
	017236	013746	002314		MOV	LUR15,-(SP)
	017242	013746	002312		MOV	LUR14,-(SP)
	017246	012746	011565		MOV	#FMT6,-(SP)
	017252	012746	000005		MOV	#5,-(SP)

017256	010600			MOV	SP,R0
017260	104415			TRAP	C\$PNTX
017262	062706	000014		ADD	#14,SP
188 017266			PRINTX	#FMT4,#DH7,#DH8	
017266	012746	014451		MOV	#DH8,-(SP)
017272	012746	014417		MOV	#DH7,-(SP)
017276	012746	011522		MOV	#FMT4,-(SP)
017302	012746	000003		MOV	#3,-(SP)
017306	010600			MOV	SP,R0
017310	104415			TRAP	C\$PNTX
017312	062706	000010		ADD	#10,SP
189 017316			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16	
017316	013746	002330		MOV	AX1.16,-(SP)
017322	013746	002326		MOV	AX1.15,-(SP)
017326	013746	002324		MOV	AX0.16,-(SP)
017332	013746	002322		MOV	AX0.15,-(SP)
017336	012746	011535		MOV	#FMT5,-(SP)
017342	012746	000005		MOV	#5,-(SP)
017346	010600			MOV	SP,R0
017350	104415			TRAP	C\$PNTX
017352	062706	000014		ADD	#14,SP
190 017356			PRINTX	#FMT9,#DH9	
017356	012746	014510		MOV	#DH9,-(SP)
017362	012746	011664		MOV	#FMT9,-(SP)
017366	012746	000002		MOV	#2,-(SP)
017372	010600			MOV	SP,R0
017374	104415			TRAP	C\$PNTX
017376	062706	000006		ADD	#6,SP
191 017402			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
017402	013746	002340		MOV	AX3.16,-(SP)
017406	013746	002336		MOV	AX3.15,-(SP)
017412	013746	002334		MOV	AX2.16,-(SP)
017416	013746	002332		MOV	AX2.15,-(SP)
017422	012746	011565		MOV	#FMT6,-(SP)
017426	012746	000005		MOV	#5,-(SP)
017432	010600			MOV	SP,R0
017434	104415			TRAP	C\$PNTX
017436	062706	000014		ADD	#14,SP
192 017442			ENDMSG		
017442					
017442	104423				
193					
194					
195					
196					
197					
198 017444			BGNMSG	ERR7	
017444					
199 017444			PRINTB	#FMT1,#ADDRES,MPCSR	
017444	013746	002446			
017450	012746	035574			
017454	012746	011426			
017460	012746	000003			
017464	010600				
017466	104414				
017470	062706	000010			
200 017474			PRINTB	#FMT2	

L10007:

TRAP C\$MSG

ERR7::

MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

	017474	012746	011436		MOV	#FMT2,-(SP)
	017500	012746	000001		MOV	#1,-(SP)
	017504	010600			MOV	SP,R0
	017506	104414			TRAP	C\$PNTB
	017510	062706	000004		ADD	#4,SP
201	017514			PRINTB	#FMT7,#DH1,REGNUM	
	017514	013746	002400		MOV	REGNUM,-(SP)
	017520	012746	014240		MOV	#DH1,-(SP)
	017524	012746	011620		MOV	#FMT7,-(SP)
	017530	012746	000003		MOV	#3,-(SP)
	017534	010600			MOV	SP,R0
	017536	104414			TRAP	C\$PNTB
	017540	062706	000010		ADD	#10,SP
202	017544			PRINTX	#FMT4,#DH2,#DH3	
	017544	012746	014311		MOV	#DH3,-(SP)
	017550	012746	014262		MOV	#DH2,-(SP)
	017554	012746	011522		MOV	#FMT4,-(SP)
	017560	012746	000003		MOV	#3,-(SP)
	017564	010600			MOV	SP,R0
	017566	104415			TRAP	C\$PNTX
	017570	062706	000010		ADD	#10,SP
203	017574			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
	017574	013746	002310		MOV	LUR13,-(SP)
	017600	013746	002306		MOV	LUR12,-(SP)
	017604	013746	002304		MOV	LUR11,-(SP)
	017610	013746	002302		MOV	LUR10,-(SP)
	017614	012746	011535		MOV	#FMT5,-(SP)
	017620	012746	000005		MOV	#5,-(SP)
	017624	010600			MOV	SP,R0
	017626	104415			TRAP	C\$PNTX
	017630	062706	000014		ADD	#14,SP
204	017634			PRINTX	#FMT9,#DH4	
	017634	012746	014347		MOV	#DH4,-(SP)
	017640	012746	011664		MOV	#FMT9,-(SP)
	017644	012746	000002		MOV	#2,-(SP)
	017650	010600			MOV	SP,R0
	017652	104415			TRAP	C\$PNTX
	017654	062706	000006		ADD	#6,SP
205	017660			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17	
	017660	013746	002320		MOV	LUR17,-(SP)
	017664	013746	002316		MOV	LUR16,-(SP)
	017670	013746	002314		MOV	LUR15,-(SP)
	017674	013746	002312		MOV	LUR14,-(SP)
	017700	012746	011565		MOV	#FMT6,-(SP)
	017704	012746	000005		MOV	#5,-(SP)
	017710	010600			MOV	SP,R0
	017712	104415			TRAP	C\$PNTX
	017714	062706	000014		ADD	#14,SP
206	017720			PRINTX	#FMT4,#DH7,#DH8	
	017720	012746	014451		MOV	#DH8,-(SP)
	017724	012746	014417		MOV	#DH7,-(SP)
	017730	012746	011522		MOV	#FMT4,-(SP)
	017734	012746	000003		MOV	#3,-(SP)
	017740	010600			MOV	SP,R0
	017742	104415			TRAP	C\$PNTX
	017744	062706	000010		ADD	#10,SP
207	017750			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16	

	017750	013746	002330		MOV	AX1.16,-(SP)
	017754	013746	002326		MOV	AX1.15,-(SP)
	017760	013746	002324		MOV	AX0.16,-(SP)
	017764	013746	002322		MOV	AX0.15,-(SP)
	017770	012746	011535		MOV	#FMT5,-(SP)
	017774	012746	000005		MOV	#5,-(SP)
	020000	010600			MOV	SP,R0
	020002	104415			TRAP	C\$PNTX
	020004	062706	000014		ADD	#14,SP
208	020010			PRINTX	#FMT9,#DH9	
	020010	012746	014510		MOV	#DH9,-(SP)
	020014	012746	011664		MOV	#FMT9,-(SP)
	020020	012746	000002		MOV	#2,-(SP)
	020024	010600			MOV	SP,R0
	020026	104415			TRAP	C\$PNTX
	020030	062706	000006		ADD	#6,SP
209	020034			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
	020034	013746	002340		MOV	AX3.16,-(SP)
	020040	013746	002336		MOV	AX3.15,-(SP)
	020044	013746	002334		MOV	AX2.16,-(SP)
	020050	013746	002332		MOV	AX2.15,-(SP)
	020054	012746	011565		MOV	#FMT6,-(SP)
	020060	012746	000005		MOV	#5,-(SP)
	020064	010600			MOV	SP,R0
	020066	104415			TRAP	C\$PNTX
	020070	062706	000014		ADD	#14,SP
210	020074			ENDMSG		
	020074					L10010:
	020074	104423			TRAP	C\$MSG
211						
212						
213						
214						
215						
216	020076			BGNMSG	ERR8	
	020076					ERR8::
217	020076			PRINTB	#FMT10,SUBRPC	
	020076	013746	002352		MOV	SUBRPC,-(SP)
	020102	012746	011671		MOV	#FMT10,-(SP)
	020106	012746	000002		MOV	#2,-(SP)
	020112	010600			MOV	SP,R0
	020114	104414			TRAP	C\$PNTB
	020116	062706	000006		ADD	#6,SP
218	020122			PRINTB	#FMT1,#ADDRES,MPCSR	
	020122	013746	002446		MOV	MPCSR,-(SP)
	020126	012746	035574		MOV	#ADDRES,-(SP)
	020132	012746	011426		MOV	#FMT1,-(SP)
	020136	012746	000003		MOV	#3,-(SP)
	020142	010600			MOV	SP,R0
	020144	104414			TRAP	C\$PNTB
	020146	062706	000010		ADD	#10,SP
219	020152			PRINTB	#FMT2	
	020152	012746	011436		MOV	#FMT2,-(SP)
	020156	012746	000001		MOV	#1,-(SP)
	020162	010600			MOV	SP,R0
	020164	104414			TRAP	C\$PNTB
	020166	062706	000004		ADD	#4,SP

220	020172		PRINTB #FMT7,#DH1,REGNUM		
	020172	013746		MOV	REGNUM,-(SP)
	020176	012746		MOV	#DH1,-(SP)
	020202	012746		MOV	#FMT7,-(SP)
	020206	012746		MOV	#3,-(SP)
	020212	010600		MOV	SP,R0
	020214	104414		TRAP	C\$PNTB
	020216	062706		ADD	#10,SP
221	020222		PRINTB #FMT3,GOODAT,BADDAT		
	020222	013746		MOV	BADDAT,-(SP)
	020226	013746		MOV	GOODAT,-(SP)
	020232	012746		MOV	#FMT3,-(SP)
	020236	012746		MOV	#3,-(SP)
	020242	010600		MOV	SP,R0
	020244	104414		TRAP	C\$PNTB
	020246	062706		ADD	#10,SP
222	020252		PRINTX #FMT4,#DH2,#DH3		
	020252	012746		MOV	#DH3,-(SP)
	020256	012746		MOV	#DH2,-(SP)
	020262	012746		MOV	#FMT4,-(SP)
	020266	012746		MOV	#3,-(SP)
	020272	010600		MOV	SP,R0
	020274	104415		TRAP	C\$PNTX
	020276	062706		ADD	#10,SP
223	020302		PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13		
	020302	013746		MOV	LUR13,-(SP)
	020306	013746		MOV	LUR12,-(SP)
	020312	013746		MOV	LUR11,-(SP)
	020316	013746		MOV	LUR10,-(SP)
	020322	012746		MOV	#FMT5,-(SP)
	020326	012746		MOV	#5,-(SP)
	020332	010600		MOV	SP,R0
	020334	104415		TRAP	C\$PNTX
	020336	062706		ADD	#14,SP
224	020342		PRINTX #FMT9,#DH4		
	020342	012746		MOV	#DH4,-(SP)
	020346	012746		MOV	#FMT9,-(SP)
	020352	012746		MOV	#2,-(SP)
	020356	010600		MOV	SP,R0
	020360	104415		TRAP	C\$PNTX
	020362	062706		ADD	#6,SP
225	020366		PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17		
	020366	013746		MOV	LUR17,-(SP)
	020372	013746		MOV	LUR16,-(SP)
	020376	013746		MOV	LUR15,-(SP)
	020402	013746		MOV	LUR14,-(SP)
	020406	012746		MOV	#FMT6,-(SP)
	020412	012746		MOV	#5,-(SP)
	020416	010600		MOV	SP,R0
	020420	104415		TRAP	C\$PNTX
	020422	062706		ADD	#14,SP
226	020426		PRINTX #FMT4,#DH7,#DH8		
	020426	012746		MOV	#DH8,-(SP)
	020432	012746		MOV	#DH7,-(SP)
	020436	012746		MOV	#FMT4,-(SP)
	020442	012746		MOV	#3,-(SP)
	020446	010600		MOV	SP,R0

020450	104415			TRAP	C\$PNTX
020452	062706	000010		ADD	#10,SP
227 020456			PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16		
020456	013746	002330		MOV	AX1.16,-(SP)
020462	013746	002326		MOV	AX1.15,-(SP)
020466	013746	002324		MOV	AX0.16,-(SP)
020472	013746	002322		MOV	AX0.15,-(SP)
020476	012746	011535		MOV	#FMT5,-(SP)
020502	012746	000005		MOV	#5,-(SP)
020506	010600			MOV	SP,R0
020510	104415			TRAP	C\$PNTX
020512	062706	000014		ADD	#14,SP
228 020516			PRINTX #FMT9,#DH9		
020516	012746	014510		MOV	#DH9,-(SP)
020522	012746	011664		MOV	#FMT9,-(SP)
020526	012746	000002		MOV	#2,-(SP)
020532	010600			MOV	SP,R0
020534	104415			TRAP	C\$PNTX
020536	062706	000006		ADD	#6,SP
229 020542			PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16		
020542	013746	002340		MOV	AX3.16,-(SP)
020546	013746	002336		MOV	AX3.15,-(SP)
020552	013746	002334		MOV	AX2.16,-(SP)
020556	013746	002332		MOV	AX2.15,-(SP)
020562	012746	011565		MOV	#FMT6,-(SP)
020566	012746	000005		MOV	#5,-(SP)
020572	010600			MOV	SP,R0
020574	104415			TRAP	C\$PNTX
020576	062706	000014		ADD	#14,SP
230 020602			ENDMSG		
020602					
020602	104423			L10011:	TRAP C\$MSG
231					
232					
233					
234					
235					
236 020604			BGNMSG ERR9		
020604				ERR9::	
237 020604			PRINTB #FMT1,#ADDRES,MPCSR		
020604	013746	002446		MOV	MPCSR,-(SP)
020610	012746	035574		MOV	#ADDRES,-(SP)
020614	012746	011426		MOV	#FMT1,-(SP)
020620	012746	000003		MOV	#3,-(SP)
020624	010600			MOV	SP,R0
020626	104414			TRAP	C\$PNTB
020630	062706	000010		ADD	#10,SP
238 020634			PRINTB #FMT2		
020634	012746	011436		MOV	#FMT2,-(SP)
020640	012746	000001		MOV	#1,-(SP)
020644	010600			MOV	SP,R0
020646	104414			TRAP	C\$PNTB
020650	062706	000004		ADD	#4,SP
239 020654			PRINTB #FMT7,#DH1,REGNUM		
020654	013746	002400		MOV	REGNUM,-(SP)
020660	012746	014240		MOV	#DH1,-(SP)
020664	012746	011620		MOV	#FMT7,-(SP)

020670 012746 000003
020674 010600
020676 104414
240 020700 062706 000010
020704
020704 012746 014311
020710 012746 014262
020714 012746 011522
020720 012746 000003
020724 010600
020726 104415
241 020730 062706 000010
020734
020734 013746 002310
020740 013746 002306
020744 013746 002304
020750 013746 002302
020754 012746 011535
020760 012746 000005
020764 010600
020766 104415
242 020770 062706 000014
020774
020774 012746 014347
021000 012746 011664
021004 012746 000002
021010 010600
021012 104415
021014 062706 000006
243 021020
021020 013746 002320
021024 013746 002316
021030 013746 002314
021034 013746 002312
021040 012746 011565
021044 012746 000005
021050 010600
021052 104415
021054 062706 000014
244 021060
021060
021060 104423

PRINTX #FMT4,#DH2,#DH3

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

ENDMSG

MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

L10012:
TRAP C\$MSG


```

1      .SBTTL INITIALIZE SECTION
2
3      :////////////////////
4      :/ THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
5      :/ AT THE BEGINNING OF THE TEST SEQUENCE ON THE NEXT UNIT.
6      :////////////////////
7
8 021072      BGNINIT
9 021072      LSINIT::
10 021072 010637 002346      MOV      SP,PSTACK      ;SAVE BASE-LEVEL STACK POINTER
11 021076 005037 002352      CLR      SUBRPC      ;CLEAR SUBR CALL PC
12 021102 005037 002426      CLR      DISILO      ;CLEAR CURRENT STATE OF DISSI
13 021106 005037 002430      CLR      CHPTYP      ;CLEAR USYRT CHIP TYPE INDICATOR
14 021112 005037 002420      CLR      ERROR1      ;CLEAR ERROR FLAG
15 021116 005037 002432      CLR      SAVLEN      ;CLEAR CHAR LENGTH FROM SETUP
16 021122 005737 002412      TST      FRSTIM      ;SEE IF FIRST TIME THROUGH AFTER LOAD
17 021126 001007      BNE      6$          ;BR IF NOT
18 021130 013737 000004 002414      MOV      @#4,SAVE4      ;SAVE ERROR TRAP VECTOR
19 021136 013737 000006 002416      MOV      @#6,SAVE6
20 021144 000406      BR      9$
21 021146 013737 002414 000004 6$:      MOV      SAVE4,@#4      ;RESTORE ERROR TRAP VECTOR
22 021154 013737 002416 000006      MOV      SAVE6,@#6
23 021162 012737 000001 002412 9$:      MOV      #1,FRSTIM      ;MARK FLAG FOR NEXT TIME THROUGH
24      ;SEE IF PROGRAM JUST STARTED, BR IF YES
25 021170      READEF #EF.START
26 021170 012700 000040      MOV      #EF.START,RO
27 021174 104447      TRAP    C$REFG
28 021176      BCOMPLETE      STARST
29 021176 103415      BCS      STARST
30      ;SEE IF PROGRAM JUST RESTARTED, BR IF YES
31 021200      READEF #EF.RESTART
32 021200 012700 000037      MOV      #EF.RESTART,RO
33 021204 104447      TRAP    C$REFG
34 021206      BCOMPLETE      STARST
35 021206 103411      BCS      STARST
36      ;SEE IF THIS IS A NEW PASS, BR IF YES
37 021210      READEF #EF.NEW
38 021210 012700 000035      MOV      #EF.NEW,RO
39 021214 104447      TRAP    C$REFG
40 021216      BCOMPLETE      NEWST
41 021216 103411      BCS      NEWST
42      ;SEE IF PROGRAM WAS JUST CONTINUED
43 021220      READEF #EF.CONTINUE
44 021220 012700 000036      MOV      #EF.CONTINUE,RO
45 021224 104447      TRAP    C$REFG
46 021226      BCOMPLETE      ENDIT
47 021226 103504      BCS      ENDIT
48 021230 000414      BR      GETPRM
49 STARST:
50 021232 005037 002444      CLR      STARES      ;CLEAR FLAG TO SHOW JUST HAD STA OR RES
51 ;CLEAR DEVICE MAP
52 021236 005037 002434      CLR      DEVMAP
53 NEWST:
54 021242 012737 177777 002344      MOV      #-1,LOGDEV      ;RESET LOGICAL DEVICE TO -1
55 021250 005237 002444      INC      STARES      ;INCR NO. OF PASSES SINCE STA OR RES
56 021254 012737 000001 002436      MOV      #BIT0,DEVPTR      ;INIT DEVICE MAP BIT POINTER

```

```

45      : GET UNIBUS ADDRESS, VECTOR, PRIORITY LEVEL, SWITCH PACKS, TEST
46      : CONNECTOR INFORMATION FOR THIS LOGICAL DEVICE
47 021262      GETPRM:
48 021262 005237 002344      INC      LOGDEV      ;INCREMENT LOGICAL DEVICE NUMBER
49 021266 023737 002344 002012  CMP      LOGDEV,LSUNIT ;SEE IF MAXIMUM UNIT NO. EXCEEDED
50 021274 002362      BGE      NEWST      ;BR IF YES
51 021276      GPHARD LOGDEV,R1 ;GET P-TABLE POINTER INTO R1
      021276 013700 002344      MOV      LOGDEV,R0
      021302 104442      TRAP     C$GPHRD
      021304 010001      MOV      R0,R1
52 021306      BCOMPLETE 10$ ;BR IF DEVICE AVAILABLE
      021306 103403      BCS      10$
53 021310 006337 002436      ASL      DEVPTR      ;SHIFT DEVICE POINTER
54 021314 000762      BR       GETPRM      ;SKIP THIS DEVICE
55 021316 053737 002436 002434 10$: BIS      DEVPTR,DEVMAP ;SET BIT FOR THIS DEVICE
56 021324 006337 002436      ASL      DEVPTR      ;SHIFT BIT POINTER
57 021330 062701 000002      ADD      #2,R1      ;INCREMENT R1 PAST MICROPROCESSOR TYPE
58 021334 011137 002446      MOV      (R1),MPCSR ;STORE POINTER TO MICROPROCESSOR CSR'S
59 021340 011137 002450      MOV      (R1),BSEL1
60 021344 005237 002450      INC      BSEL1      ;GET POINTER TO BSEL1 (MAINTENANCE REGISTER)
61 021350 013737 002450 002452  MOV      BSEL1,BSEL2
62 021356 005237 002452      INC      BSEL2      ;GET POINTER TO BSEL2
63 021362 011137 002454      MOV      (R1),SEL4
64 021366 062737 000004 002454  ADD      #4,SEL4      ;GET POINTER TO SEL4
65 021374 012137 002456      MOV      (R1)+,SEL6
66 021400 062737 000006 002456  ADD      #6,SEL6      ;STORE POINTER TO SEL6
67 021406 011137 002460      MOV      (R1),MPIVEC ;GET MICROPROCESSOR INPUT INTRPT VECTOR
68 021412 012137 002462      MOV      (R1)+,MPOVEC
69 021416 062737 000004 002462  ADD      #4,MPOVEC    ;GET MICROPROCESSOR OUTPUT INTRPT VECTOR
70 021424 012137 002464      MOV      (R1)+,MPRIOR ;GET MICROPROCESSOR DEVICE PRIORITY
71 021430 062701 000014      ADD      #14,R1      ;POINT R1 TO RUN SWITCH INDICATOR
72 021434 011137 002476      MOV      (R1),RUNINH ;GET STATE OF MICROPROCESSOR RUN SWITCH
73 021440      ENDIT:
74 021440      ENDINIT
      021440 104411      L10015: TRAP     C$INIT
75
76
77
78
79
80

```

```
1      .SBTTL  AUTO DROP UNIT SECTION
2
3      ://////
4      :// THE AUTO DROP CODING DETERMINES WHETHER OR NOT THE DEVICE WHOSE P-TABLE
5      :// WAS JUST OBTAINED IS READY FOR TESTING, AND IT IS DROPPED IF NOT READY.
6      ://////
7 021442      BGNAUTO
8
9 021442      :ESTABLISH CPU PRIORITY = 7
021442      SETPRI  #PRI07
021446      104441
10 021450      012700 000340
11 021456      012737 021472 000004      MOV      #6$,@#4      ;SET UP NON-EXISTENT MEMORY ERROR TRAP VECTOR
12 021464      005777 160756 000006      MOV      #PRI07,@#6
13 021470      000405      TST      @MPCSR      ;ADDRESS SELO
14      :COME HERE IF DEVICE CSR IS NON-EXISTENT      BR      9$      ;TAKE THIS BR IF DEVICE RESPONDS
15 021472      062706 000004      6$:      ADD      #4,SP      ;CLEAN UP THE STACK POINTER
16 021476      013700 002344      DODU     LOGDEV      ;DROP THIS UNIT FROM TESTING
17 021504      013737 002414 000004      9$:      MOV      SAVE4,@#4      ;RESTORE ERROR TRAP VECTOR
18 021512      013737 002416 000006      MOV      SAVE6,@#6
19 021520      ENDAUTO
021520      104461
20
21
22
23
24
```

1
2
3
4
5
6
7
8 021522
021522
9
10
11 021522
021522
021522 104412
12
13
14
15
16

.SBTTL CLEANUP CODING SECTION

:///
:// THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
:// AT THE END OF THE TEST SEQUENCE ON A PARTICULAR UNIT.
:///

BGNCLN

L\$CLEAN::

ENDCLN

L10017: TRAP C\$CLEAN


```
1          .SBTTL  DROP UNIT SECTION
2
3          ://////////
4          :// THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
5          :// TO NO LONGER BE TESTED.
6          ://////////
7
8 021524          BGNDU
9                L$DU::
10 021524          :ISSUE UNIBUS RESET TO CLEAN UP
11 021524 104433          BRESET
12 021524          :PRINT 'UNIT XX DROPPED'
13 021524          PRINTF #FMT27,LOGDEV
14 021526 013746 002344
15 021532 012746 021554
16 021536 012746 000002
17 021542 010600
18 021544 104417
19 021546 062706 000006
20
21          ENDDU
                L10020:
                TRAP  C$DU
                TRAP  C$RESET
                MOV   LOGDEV,-(SP)
                MOV   #FMT27,-(SP)
                MOV   #2,-(SP)
                MOV   SP,R0
                TRAP  C$PNTF
                ADD   #6,SP
021552          L10020:
021552 104453          TRAP  C$DU
16 021554          045      116      045  FMT27: .ASCIZ /%N%AUNIT %D2%A DROPPED%N/
17 021557          101      125      116
18 021562          111      124      040
19 021565          045      104      062
20 021570          045      101      040
21 021573          104      122      117
021576          120      120      105
021601          104      045      116
021604          000
                .EVEN
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

.SBTTL ADD UNIT SECTION

:/ THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
:/ TO BE (A) TESTED FOR THE FIRST TIME, OR (B) RESUMED IN TESTING. IF
:/ 'EF.AUNIT' IS SET, THE UNIT WILL BE TESTED AS A NEW UNIT.

021606
021606
021606
021606 104452

BGNAU
ENDAU

LSAU::
L10021: TRAP CSAU

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47

.SBTTL HARDWARE TESTS

```

:*****
.SBTTL      TEST 1 - MICROPROCESSOR CSR ADDRESSING TEST (SELO)
:*
:* THIS TEST ADDRESSES THE FIRST MICROPROCESSOR CSR (SELO), TO MAKE SURE
:* THAT A NON-EXISTENT MEMORY TIME-OUT TRAP DOES NOT OCCUR WHILE
:* ATTEMPTING TO ADDRESS THE MICROPROCESSOR.
:*****

```

```

BGNTST
                                T1::
:ESTABLISH CPU PRIORITY = 7
      SETPRI #PRI07
                                MOV #PRI07,R0
                                TRAP C$SPRI
17 021616 012700 000340
      021614 104441
18 021616 012737 021640 000004      MOV #6$,@#4 ;SET UP NON-EXISTENT MEMORY ERROR TRAP VECTOR
19 021624 012737 000340 000006      MOV #PRI07,@#6
20 021632 005777 160610      TST @MPCSR ;ADDRESS SELO
21 021636 000406      BR 9$ ;TAKE THIS BR IF DEVICE RESPONDS
:COME HERE IF DEVICE CSR IS NON-EXISTENT
22 021640 062706 000004 6$: ADD #4,SP ;CLEAN UP THE STACK POINTER
:REPORT CSR ADDRESS TIME-OUT
      ERRDF 1,EM1,ERR1
                                TRAP C$ERDF
                                .WORD 1
                                .WORD EM1
                                .WORD ERR1
25 021654 013737 002414 000004 9$: MOV SAVE4,@#4 ;RESTORE ERROR TRAP VECTOR
26 021662 013737 002416 000006      MOV SAVE6,@#6
27 021670      ENDTST
                                L10022:
                                TRAP C$ETST
      021670 104401

```

```

:*****
.SBTTL      TEST 2 - INBUS/OUTBUS REG 14 INITIALIZATION TEST
:*
:* MASTER CLEAR (MCLR) IS SET IN THE MICROPROCESSOR, IBUS REG 14 IS READ
:* AND COMPARED TO 200.
:*****

```

```

BGNTST
                                T2::
40 021672 012737 000014 002400      MOV #14,REGNUM ;SET LU REG NO. = 14
41 021700 004737 003576      JSR PC,MSTCLR ;ISSUE MASTER CLEAR
42 021704 004737 003644      JSR PC,READLU ;READ REG 14
43 021710 123737 002364 003026      CMPB REDBYT,PATM+4 ;CHK FOR INITIALIZED STATE
44 021716 001416      BEQ 6$ ;BR IF YES
45 021720 005037 002404      CLR GOODAT ;SET EXPECTED REG CONTENTS = 000
46 021724 113737 003026 002404      MOVB PATM+4,GOODAT ;SET EXPECTED DATA
47 021732 013737 002364 002406      MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS

```

```

48 021740 004737 003770      JSR    PC,GETREG      ;GET REGS FOR PRINTOUT
49                          ;REPORT REG NOT CLEARED BY MASTER CLEAR
50 021744      104455      ERRDF  2,EM2,ERR2
                                TRAP    C$ERDF
                                .WORD   2
                                .WORD   EM2
                                .WORD   ERR2
51 021754      6$:
52 021754      ENDTST
                                L10023:
                                TRAP    C$ETST
021754      104401
    
```

53
54
55
56
57

```

58 *****
59 .SBTTL      TEST 3 - INBUS/OUTBUS REG 14 READ/WRITE BIT TEST
60 .*
61 .* WRITE, READ, AND COMPARE ALL WORDS OF DATA PATTERN A INTO REG 14,
62 .* A BYTE AT A TIME. NON-R/W BITS ARE MASKED OFF TO 0 BEFORE WRITING AND
63 .* READING.
64 .* DATA PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
65 .*                   375,373,367,357,337,277,177.
66 *****
67 BGNTST
    
```

```

68 021756 004737 003576      JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR      T3::
69 021762 012737 000014 002400  MOV    #14,REGNUM    ;SET LU REG NO. = 14
70 021770 012701 002571      MOV    #PATA,R1      ;GET POINTER TO DATA PAT IN R1
71 021774      3$:
72 021774      BGNSEG
                                TRAP    C$BSEG
73 021776 104404 002366      MOVB   (R1),WRIBYT   ;GET A BYTE OF PAT A
74 022002 143737 002560 002366  BICB   R14NRW,WRIBYT ;MASK OFF NON-READ/WRITE BITS
75 022010 004737 003722      JSR    PC,WRITLU    ;WRITE DATA BYTE INTO REG 14
76 022014 004737 003644      JSR    PC,READLU    ;READ DATA BYTE FROM REG 14
77 022020 143737 002560 002364  BICB   R14NRW,REDBYT ;MASK OFF NON-READ/WRITE BITS
78 022026 123737 002364 002366  CMPB   REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
79 022034 001414 6$
80 022036 013737 002366 002404  MOV    WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
81 022044 013737 002364 002406  MOV    REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
82 022052 004737 003770      JSR    PC,GETREG    ;GET REGS FOR PRINTOUT
    
```

```

83 ;REPORT LINE UNIT REG MISCOMPARE
84 022056      ERRDF  3,EM3,ERR2
                                TRAP    C$ERDF
                                .WORD   3
                                .WORD   EM3
                                .WORD   ERR2
022056      104455
022060      000003
022062      012174
022064      014606
    
```

```

85 022066      6$:
86 022066      ENDSEG
                                10000$:
                                TRAP    C$ESEG
022066      104405
87 022070 005201      INC    R1      ;INCREMENT DATA PATTERN POINTER
88 022072 020127 002615  CMP    R1,#PATB    ;SEE IF ALL WORDS OF PATTERN A USED YET
89 022076 103736      BLO   3$      ;BR IF NOT DONE YET
90 022100      ENDTST
    
```

L10024: TRAP C\$ETST

022100
 022100 104401
 91
 92
 93
 94
 95
 96
 97
 98
 99

```

:*****
:SBTTL      TEST 4 - REG 14 MASTER CLEAR TEST
:* WRITE 377 INTO REG 14, ISSUE MASTER CLEAR, READ REG 14 AND COMPARE
:* TO 200.
:*****
BGNTST
    
```

100
 101 022102
 022102
 102 022102 004737 003576
 103 022106 012737 000014 002400
 104 022114 112737 000377 002366
 105 022122 004737 003722
 106 022126 004737 003576
 107 022132 004737 003644
 108 022136 123737 002364 003026
 109 022144 001416
 110 022146 005037 002404
 111 022152 113737 003026 002404
 112 022160 013737 002364 002406
 113 022166 004737 003770
 114
 115 022172
 022172 104455
 022174 000002
 022176 012135
 022200 014606
 116 022202
 117 022202
 022202
 022202 104401

```

T4::
JSR PC,MSTCLR ;PERFORM MASTER CLEAR
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB #377,WRIBYT ;SET DATA BYTE = 377
JSR PC,WRITLU ;WRITE 377 INTO REG 14
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,READLU ;READ REG 14
CMPB REDBYT,PATM+4 ;CHK FOR INIT'D STATE
BEQ 6$ ;BR IF REG GOT CLEARED
CLR GOODAT
MOVB PATM+4,GOODAT ;SET EXPECTED DATA
MOV REDBYT,BADDAT ;SET ACTUAL DATA
JSR PC,GETREG ;GET REGS FOR PRINTOUT
:REPORT REG NOT CLEARED BY MASTER CLEAR
ERRDF 2,EM2,ERR2
    
```

TRAP C\$ERDF
 .WORD 2
 .WORD EM2
 .WORD ERR2

6\$:
 ENDTST

L10025: TRAP C\$ETST

118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128 022204
 022204
 129 022204 004737 003576
 130 022210 012737 000014 002400
 131 022216 112737 000377 002366
 132 022224 004737 003722
 133 022230
 022230 104433
 134 022232 142777 000200 160210
 135 022240 012701 000024
 136 022244 000240

```

:*****
:SBTTL      TEST 5 - REG 14 UNIBUS RESET (INIT) TEST
:* WRITE 377 INTO REG 14, ISSUE UNIBUS RESET (INIT), READ REG 14 AND COMPARE
:* TO 200.
:*****
BGNTST
    
```

T5::

```

JSR PC,MSTCLR ;PERFORM MASTER CLEAR
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB #377,WRIBYT ;SET DATA BYTE = 377
JSR PC,WRITLU ;WRITE 377 INTO REG 14
BRESET ;ISSUE UNIBUS RESET (INIT)
BICB #RUN,@BSEL1 ;CLEAR RUN BIT
MOV #20.,R1 ;INIT LOOP COUNTER
2$: NOP
    
```

TRAP C\$RESET

```
137 022246 005301          DEC      R1          ;DECR COUNTER
138 022250 001375          BNE     2$          ;BR TO STALL
139 022252 004737 003644   JSR     PC,READLU  ;READ REG 14
140 022256 142737 000100 002364   BICB   #TXEN,REDBYT ;CLEAR UNPREDICTABLE BIT
141 022264 123737 002364 003026   CMPB   REDBYT,PATM+4 ;CHK FOR INIT'D STATE
142 022272 001416          BEQ     6$          ;BR IF REG GOT CLEARED
143 022274 005037 002404   CLR    GOODAT      ;SET EXPECTED DATA
144 022300 113737 003026 002404   MOVB   PATM+4,GOODAT ;SET ACTUAL DATA
145 022306 013737 002364 002406   MOV    REDBYT,BADDAT ;GET REGS FOR PRINTOUT
146 022314 004737 003770   JSR    PC,GETREG   ;GET REGS FOR PRINTOUT
147          ;REPORT REG NOT CLEARED BY UNIBUS RESET (INIT)
148 022320          ERRDF  4,EM4,ERR2
          022320 104455          TRAP   C$ERDF
          022322 000004          .WORD  4
          022324 012213          .WORD  EM4
          022326 014606          .WORD  ERR2
149 022330          6$:
150 022330          ENDTST
          022330          L10026:
          022330 104401          TRAP   C$ETST
```

151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184

```
*****  
:SBTTL      TEST 6 - LINE UNIT FALSE SELECTION TEST  
:*  
:* FIRST, A MASTER CLEAR IS PERFORMED. THEN, THE PROGRAM SINGLE-STEPS THE  
:* MICROPROCESSOR THROUGH AN INSTRUCTION WHICH LOADS 041 (OCT) INTO THE MAR  
:* REGISTER (OBUS* ADRS 14). THEN, THE LINE UNIT REGISTER 14 IS READ AND CHECKED  
:* TO BE UNAFFECTED (STILL = 0). THIS TEST IS INTENDED TO DETECT A FALSE  
:* SELECTION OF THE LINE UNIT REGISTERS, WHEN THE LINE UNIT IS NOT BEING  
:* ACCESSED.  
:*****
```

```
166 022332          BGNTST  
          022332          T6::  
167 022332 004737 003576   JSR    PC,MSTCLR   ;ISSUE A MASTER CLEAR  
168 022336 012737 000014 002400   MOV    #14,REGNUM ;SET LU REG NO. = 14  
169 022344 013701 002400   MOV    REGNUM,R1  ;SET DESTINATION = OBUS* REG 14  
170 022350 052701 000100   BIS    #100,R1    ;SET SOURCE = BSEL4  
171 022354 052701 121000   BIS    #MVIXOX,R1 ;SET REST OF MOVE INSTRUCTION  
172 022360 010137 022376   MOV    R1,2$     ;SET INSTRUCTION AS SUBROUTINE ARGUMENT  
173 022364 112777 000041 160062   MOVB   #041,@BSEL4 ;SET DATA BYTE = 041  
174 022372 004737 003540   JSR    PC,STPCLK  ;EXECUTE MOVE INSTRUCTION  
175 022376 000000          2$: .WORD  0          ;INSTRUCTION GOES HERE  
176 022400 004737 003644   JSR    PC,READLU  ;READ LU REG 14  
177 022404 123737 002364 003026   CMPB   REDBYT,PATM+4 ;CHECK FOR LU REG 14 UNCHANGED  
178 022412 001416          BEQ     4$          ;BR IF LU REG 14 UNCHANGED  
179 022414 005037 002404   CLR    GOODAT      ;SET EXPECTED DATA  
180 022420 113737 003026 002404   MOVB   PATM+4,GOODAT ;SET ACTUAL DATA  
181 022426 013737 002364 002406   MOV    REDBYT,BADDAT ;GET REGS FOR PRINTOUT  
182 022434 004737 003770   JSR    PC,GETREG   ;GET REGS FOR PRINTOUT  
183          ;REPORT REGISTER MISCOMPARE  
184 022440          ERRDF  3,EM3,ERR2  
          022440 104455          TRAP   C$ERDF  
          022442 000003          .WORD  3
```

185	022444	012174						
186	022446	014606						
	022450		4\$:					
	022450		ENDTST					
	022450	104401				L10027:	TRAP	C\$ETST

187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

```

:*****
:SBTTL      TEST 7 - INBUS REG MASTER CLEAR TEST
:
:* FIRST, ALL READ/WRITE BITS OF REGS 10-17 ARE SET BY LOADING A
:* DIFFERENT WORD OF PATTERN G INTO EACH REG. THEN,
:* A MASTER CLEAR IS ISSUED AND EACH REG IS READ AND COMPARED TO A WORD OF
:* PATTERN M, WHICH CONTAINS THE INITIALIZED STATES OF THE REGS. (UNPREDICTABLE
:* BITS ARE MASKED OFF TO 0 BEFORE COMPARISON).
:* PATTERN G = 000,000,240,120,177,000,000,001
:* PATTERN M = 000,020,000,000,200,000,000,051
:*****
    
```

```

BGNTST
T7::
204 022452 004737 003576          JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
205 022456 012737 000010 002400  MOV    #10,REGNUM    ;INIT REG NO. TO 10
206 022464 012701 002644          MOV    #PATG,R1      ;INIT DATA PATTERN POINTER
207 022470 112137 002366 2$:    MOV    (R1)+,WRIBYT   ;SET DATA PATTERN BYTE TO BE WRITTEN
208 022474 004737 003722          JSR    PC,WRITLU     ;WRITE BYTE INTO REG
209 022500 005237 002400          INC    REGNUM        ;INCREMENT REG NO. FOR WRITING
210 022504 020127 002654          CMP    R1,#PATH     ;SEE IF ALL BYTES WRITTEN YET
211 022510 103767                BLO    2$           ;BR IF NOT DONE WRITING YET
212 022512 004737 003576          JSR    PC,MSTCLR     ;ISSUE MASTER CLEAR
213 022516 012737 000010 002400  MOV    #10,REGNUM    ;INIT LU REG NO. TO 10
214 022524 012702 003022          MOV    #PATM,R2     ;INIT DATA PATTERN POINTER
215 022530 012701 002550          MOV    #UPBITS,R1   ;INIT POINTER TO UNPREDICTABLE BITS
216 022534                3$:
217 022534 004737 003644          JSR    PC,READLU    ;READ A LINE UNIT REG
218 022540 142137 002364          BICB  (R1)+,REDBYT  ;MASK OUT UNPREDICTABLE BITS FOR THIS REG
219 022544 123712 002364          CMPB  REDBYT,(R2)   ;COMPARE MASKED DATA TO EXPECTED
220 022550 001417                BEQ    6$           ;BR IF DATA READ IS OK
221 022552 005037 002404          CLR    GOODAT       ;CLEAR GOOD DATA
222 022556 111237 002404          MOV    (R2),GOODAT  ;SET EXPECTED DATA
223 022562 013737 002364 002406  MOV    REDBYT,BADDAT ;SET ACTUAL DATA
224 022570 004737 003770          JSR    PC,GETREG    ;GET REGS FOR PRINTOUT
225                ;REPORT REG NOT CLEARED BY MASTER CLEAR
226                ERRDF 2,EM2,ERR2
    
```

						TRAP	C\$ERDF
	022574	104455				.WORD	2
	022576	000002				.WORD	EM2
	022600	012135				.WORD	ERR2
	022602	014606					
227	022604		ESCAPE	TST		TRAP	C\$ESCAPE
	022604	104410				.WORD	L10030-
	022606	000016					

```

6$: INC    REGNUM      ;INCREMENT REG NO.
    INC    R2         ;INCR DATA PATTERN POINTER
    CMP    R2,#PATN  ;SEE IF ALL DONE YET
    
```

```

231 022622 103744          BLO      3$          ;BR IF NOT DONE YET
232 022624          ENDTST
    022624
    022624 104401          L10030: TRAP C$ETST
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247 022626          BGNST
    022626
248 022626 004737 003576      JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
249 022632 012701 002500      MOV      #REDDAT,R1    ;INIT POINTER TO EXPECTED DATA AREA
250 022636 012702 003022      MOV      #PATM,R2      ;GET POINTER TO PATTERN M
251 022642 012703 000010      MOV      #8,R3         ;SET COUNTER
252 022646 112221          3$:      MOV      (R2)+,(R1)+  ;LOAD BYTE OF PATRN INTO EXPECTED DATA AREA
253 022650 005303          DEC      R3            ;DECR COUNTER
254 022652 001375          BNE      3$           ;BR IF NOT DONE LOADING YET
255 022654 005001          CLR      R1           ;INIT DATA PATTERN INDEX FOR WRITING
256 022656 010137 002400      6$:      MOV      R1,REGNUM   ;GET REG NO. FOR WRITING
257 022662 062737 000010 002400  ADD      #10,REGNUM    ;GET REG NO. FOR WRITING
258 022670 116137 002615 002366  MOV      PATB(R1),WRIBYT ;SET DATA BYTE TO BE WRITTEN
259 022676 113761 002366 002500  MOV      WRIBYT,REDDAT(R1) ;SET EXPECTED DATA FOR READ
260 022704 146161 002550 002500  BICB    UPBITS(R1),REDDAT(R1) ;MASK OUT UNPREDICTABLE BITS
261 022712 004737 003722      JSR      PC,WRITLU     ;WRITE DATA BYTE INTO REG
262 022716 005003          CLR      R3           ;INIT DATA PAT INDEX FOR READS
263 022720 010337 002400      9$:      MOV      R3,REGNUM    ;GET REG NO. FOR READING
264 022724 062737 000010 002400  ADD      #10,REGNUM    ;GET REG NO. FOR READING
265 022732 004737 003644      JSR      PC,READLU     ;READ A LINE UNIT REG
266 022736 146337 002550 002364  BICB    UPBITS(R3),REDBYT ;MASK OUT UNPREDICTABLE BITS
267 022744 023727 002400 000011  CMP      REGNUM,#11    ;SEE IF READING REG 11
268 022752 001006          BNE      10$          ;BR IF NOT
269 022754 142737 000020 002364  BICB    #ORDY,REDBYT   ;MASK ORDY BIT IN ACTUAL BYTE
270 022762 142763 000020 002500  BICB    #ORDY,REDDAT(R3) ;MASK ORDY BIT IN EXPECTED BYTE
271 022770 123763 002364 002500  10$:    CMP      REDBYT,REDDAT(R3) ;COMPARE BYTE READ TO EXPECTED
272 022776 001420          BEQ      12$          ;BR IF DATA MATCHES
273 023000 005037 002404      CLR      GOODAT
274 023004 116337 002500 002404  MOV      REDDAT(R3),GOODAT ;SET EXPECTED DATA
275 023012 013737 002364 002406  MOV      REDBYT,BADDAT ;SET ACTUAL DATA
276 023020 004737 003770      JSR      PC,GETREG     ;READ AND STORE REGS 10-17 FOR PRINTOUT
277
278 023024          ;REPORT REG MISCOMPARE
    023024 104455          ERRDF   3,EM3,ERR2
    023026 000003          TRAP   C$ERDF
    023030 012174          .WORD  3
    023032 014606          .WORD  EM3
279 023034          .WORD  ERR2
    023034 104410          TRAP   C$ESCAPE
    ESCAPE TST
    
```



```

280 023036 000022          .WORD L10031-.
281 023040 005203          12$: INC R3          ;INCREMENT DATA PATTERN INDEX FOR READING
282 023042 020327 000010  CMP R3,#10        ;SEE IF ALL REGS READ YET
283 023046 002724          BLT 9$           ;BR IF NOT
284 023050 005201          INC R1           ;INCREMENT DATA PAT INDEX FOR WRITING
285 023052 020127 000010  CMP R1,#10        ;SEE IF ALL REGS WRITTEN YET
286 023056 002677          BLT 6$           ;BR IF NOT
286 023060          ENDTST
287 023060          L10031:
288 023060 104401          TRAP C$ETST
    
```

287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323

```

:*****
:SBTTL      TEST 9 - REG 11 READ/WRITE BIT TEST
:*
:* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN C INTO REG 11 :
:* DATA PATTERN C = 020,020,020.
:*****
BGNTST
    
```

T9::

```

299 023062 004737 003576          JSR PC,MSTCLR      ;ISSUE MASTER CLEAR
300 023066 012737 000011 002400  MOV #11,REGNUM     ;SET LU REG NO. = 11
301 023074 012701 002625          MOV #PATC,R1      ;GET POINTER TO DATA PAT IN R1
302 023100          3$:
303 023100          BGNSEG
304 023100 104404          TRAP C$BSEG
304 023102 111137 002366          MOVB (R1),WRIBYT  ;GET A BYTE OF PAT C
305 023106 004737 003722          JSR PC,WRITLU     ;WRITE DATA BYTE INTO REG 11
306 023112 004737 003644          JSR PC,READLU     ;READ DATA BYTE FROM REG 11
307 023116 143737 002551 002364  BICB UPBITS+1,REDBYT ;MASK OUT UNPREDICTABLE BITS
308 023124 123737 002364 002366  CMPB REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
309 023132 001414          BEQ 6$           ;BR IF BYTES MATCH
310 023134 013737 002366 002404  MOV WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
311 023142 013737 002364 002406  MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
312 023150 004737 003770          JSR PC,GETREG     ;GET REGS FOR PRINTOUT
313          ;REPORT LINE UNIT REG MISCOMPARE
314 023154          ERRDF 3,EM3,ERR2
314 023154 104455          TRAP C$ERDF
314 023156 000003          .WORD 3
314 023160 012174          .WORD EM3
314 023162 014606          .WORD ERR2
    
```

```

315 023164          6$:
316 023164          ENDSEG
316 023164          10000$:
317 023166 005201          TRAP C$ESEG
318 023170 020127 002630          INC R1           ;INCREMENT DATA PATTERN POINTER
319 023174 103741          CMP R1,#PATD     ;SEE IF ALL WORDS OF PATTERN C USED YET
320 023176          BLO 3$           ;BR IF NOT DONE YET
320 023176          ENDTST
320 023176          L10032:
320 023176 104401          TRAP C$ETST
    
```

```

324
325
326
327
328
329
330
331
332 023200
    023200
333 023200 004737 003576          JSR    PC,MSTCLR          ;ISSUE MASTER CLEAR
334 023204 012737 000012 002400  MOV    #12,REGNUM        ;SET LU REG NO. = 12
335 023212 012701 002630          MOV    #PATD,R1          ;GET POINTER TO DATA PAT IN R1
336 023216
337 023216
    023216 104404
338 023220 111137 002366          MOVB   (R1),WRIBYT        ;GET A BYTE OF PAT D
339 023224 004737 003722          JSR    PC,WRITLU          ;WRITE DATA BYTE INTO REG 12
340 023230 004737 003644          JSR    PC,READLU         ;READ DATA BYTE FROM REG 12
341 023234 143737 002552 002364  BICB   UPBITS+2,REDBYT    ;MASK OUT UNPREDICTABLE BITS
342 023242 123737 002364 002366  CMPB   REDBYT,WRIBYT      ;COMPARE BYTE READ TO BYTE WRITTEN
343 023250 001414
344 023252 013737 002366 002404  BEQ    6$
345 023260 013737 002364 002406  MOV    WRIBYT,GOODAT      ;SET EXPECTED REG CONTENTS
346 023266 004737 003770          MOV    REDBYT,BADDAT     ;SET ACTUAL REG CONTENTS
347
348 023272
    023272
    023274 000003
    023276 012174
    023300 014606
349 023302
350 023302
    023302
    023302 104405
351 023304 005201
352 023306 020127 002633          INC    R1
353 023312 103741
354 023314
    023314
    023314 104401
355
356
357
358
359
360
361
362
363
364
365
366 023316
    023316
367 023316 004737 003576          JSR    PC,MSTCLR          ;ISSUE MASTER CLEAR
368 023322 012737 000013 002400  MOV    #13,REGNUM        ;SET LU REG NO. = 13
369 023330 012701 002633          MOV    #PATE,R1          ;GET POINTER TO DATA PAT IN R1
    
```

```

:*****
:SBTTL      TEST 10 - REG 12 READ/WRITE BIT TEST
:
:* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN D INTO REG 12 :
:* DATA PATTERN D = 000,040,000.
:*****
    
```

```

;REPORT LINE UNIT REG MISCOMPARE
ERRDF 3,EM3,ERR2
    
```

T10:: TRAP C\$BSEG

TRAP C\$ERDF
 .WORD 3
 .WORD EM3
 .WORD ERR2

10000\$: TRAP C\$ESEG

L10033: TRAP C\$ETST

```

:*****
:SBTTL      TEST 11 - REG 13 READ/WRITE BIT TEST
:
:* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN E INTO REG 13 :
:* DATA PATTERN E = 000,120,020,100,120,000.
:*****
    
```

T11::

```

370 023334          3$:
371 023334          BGNSEG
    023334 104404          TRAP  C$BSEG
372 023336 111137 002366  MOVB  (R1),WRIBYT  ;GET A BYTE OF PAT E
373 023342 004737 003722  JSR   PC,WRITLU   ;WRITE DATA BYTE INTO REG 13
374 023346 004737 003644  JSR   PC,READLU   ;READ DATA BYTE FROM REG 13
375 023352 143737 002553 002364 BICB  UPBITS+3,REDBYT ;MASK OUT UNPREDICTABLE BITS
376 023360 123737 002364 002366 CMPB  REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
377 023366 001414          BEQ   6$          ;BR IF BYTES MATCH
378 023370 013737 002366 002404 MOV   WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
379 023376 013737 002364 002406 MOV   REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
380 023404 004737 003770  JSR   PC,GETREG   ;GET REGS FOR PRINTOUT
381          ;REPORT LINE UNIT REG MISCOMPARE
382 023410          ERRDF  3,EM3,ERR2
    023410 104455          TRAP  C$ERDF
    023412 000003          .WORD  3
    023414 012174          .WORD  EM3
    023416 014606          .WORD  ERR2
383 023420          6$:
384 023420          ENDSEG
    023420          10000$:
385 023422 005201          INC   R1          ;INCREMENT DATA PATTERN POINTER
386 023424 020127 002641  CMP   R1,#PATF   ;SEE IF ALL WORDS OF PATTERN E USED YET
387 023430 103741          BLO  3$          ;BR IF NOT DONE YET
388 023432          ENDTST
    023432          L10034:
    023432 104401          TRAP  C$ETST
389
390
391
392
393
394
395          ;*****
396          .SBTTL  TEST 12 - REG 17 READ/WRITE BIT TEST
397          ;*
398          ;* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN F INTO REG 17 :
399          ;* DATA PATTERN F = 050,051,050.
400          ;*****
401 023434          BGNTST
    023434          T12::
402 023434 004737 003576  JSR   PC,MSTCLR   ;ISSUE MASTER CLEAR
403 023440 012737 000017 002400 MOV   #17,REGNUM  ;SET LU REG NO. = 17
404 023446 012701 002641  MOV   #PATF,R1   ;GET POINTER TO DATA PAT IN R1
405 023452          3$:
    023452          BGNSEG
    023452 104404          TRAP  C$BSEG
406 023454 111137 002366  MOVB  (R1),WRIBYT  ;GET A BYTE OF PAT F
407 023460 004737 003722  JSR   PC,WRITLU   ;WRITE DATA BYTE INTO REG 17
408 023464 004737 003644  JSR   PC,READLU   ;READ DATA BYTE FROM REG 17
409 023470 143737 002557 002364 BICB  UPBITS+7,REDBYT ;MASK OUT UNPREDICTABLE BITS
410 023476 123737 002364 002366 CMPB  REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
411 023504 001414          BEQ   6$          ;BR IF BYTES MATCH
412 023506 013737 002366 002404 MOV   WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
413 023514 013737 002364 002406 MOV   REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
414 023522 004737 003770  JSR   PC,GETREG   ;GET REGS FOR PRINTOUT
415          ;REPORT LINE UNIT REG MISCOMPARE
    
```

```

416 023526          ERRDF  3,EM3,ERR2
      023526 104455
      023530 000003
      023532 012174
      023534 014606
417 023536          6$:
418 023536          ENDSEG
      023536
      023536 104405
419 023540 005201
420 023542 020127 002644
421 023546 103741
422 023550          ENDTST
      023550
      023550 104401
      10000$:
      TRAP  C$ERDF
      .WORD 3
      .WORD EM3
      .WORD ERR2
      10000$:
      TRAP  C$ESEG
      INC  R1          ;INCREMENT DATA PATTERN POINTER
      CMP  R1,#PATG   ;SEE IF ALL WORDS OF PATTERN F USED YET
      BLO  3$         ;BR IF NOT DONE YET
      L10035:
      TRAP  C$ETST
    
```

423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448

```

;*****
;SBTTL      TEST 13 - MAINTENANCE CLOCK BIT TEST
;
;* FIRST, A MASTER CLEAR IS ISSUED TO INIT ALL REGS. THEN, THE MICROPROCESSOR
;* IS PLACED IN A LOOP ON AN INSTRUCTION, BY SETTING THE INSTRUCTION IN SEL6
;* AND SETTING ROMI AND RUN IN BSEL1. THE INSTRUCTION IS ONE WHICH REPETITIVELY
;* READS LINE UNIT REG 17 INTO BSEL2. THE PDP-11 CAN THEN SCAN BSEL2 TO MONITOR
;* THE MAINTENANCE CLOCK BIT, MCLK. THE FOLLOWING SEQUENCE IS THEN PERFORMED
;* TO MONITOR MCLK :
;* - THE PROGRAM REPEATEDLY CHECKS THE MCLK BIT FOR THE 1 STATE, AND IF IT IS
;* NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC (DEPENDING ON THE PROCESSOR)
;* AN ERROR IS REPORTED. (THE MAINTENANCE CLOCK HAS A PERIOD OF 41.6 MICRO-
;* SEC).
;* - THE PROGRAM NEXT REPEATEDLY CHECKS THE MCLK BIT FOR THE 0 STATE, AND IF
;* IT IS NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC AN ERROR IS REPORTED.
;* - THE PROGRAM NEXT REPEATEDLY CHECKS MCLK BIT FOR THE 1 STATE AGAIN, AND
;* IF IT IS NOT FOUND WITHIN SEVERAL HUNDRED MILLI-SEC, AN ERROR IS REPORTED.
;*
;* IF THE P-TABLE FOR THIS UNIT INDICATES THAT THE MICROPROCESSOR RUN SWITCH
;* IS NOT ON, THE TEST WILL BE SKIPPED.
;*****
BGNTST
    
```

```

449 023552
      023552
450 023552 004737 003576
451 023556 012737 000015 002442
452 023564 005737 002476
453 023570 001020
454 023572 023727 002444 000001
455 023600 001012
456 023602
      023602 013746 002442
      023606 012746 011761
      023612 012746 000002
      023616 010600
      023620 104417
      023622 062706 000006
457 023626 000137 024174
      JSR  PC,MSTCLR   ;PERFORM MASTER CLEAR
      MOV  #13,TSTNUM ;SET TEST NO.
      TST  RUNINH     ;SEE IF RUN SWITCH IS SET
      BNE  1$         ;BR IF YES, TO RUN TEST
      CMP  STARES,#1  ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
      BNE  40$        ;BR IF NOT, TO SKIP PRINTING
      PRINTF #FMT19,TSTNUM ;PRINT MSG TO SAY TEST NOT RUN
      MOV  TSTNUM,-(SP)
      MOV  #FMT19,-(SP)
      MOV  #2,-(SP)
      MOV  SP,R0
      TRAP C$PNTF
      ADD  #6,SP
      40$:
      JMP  A1         ;GO TO SKIP TEST
      T13::
    
```

```

458 023632 012737 000017 002400 1$: MOV #17,REGNUM ;SET REG NO. = 17
459 023640 012701 000360 MOV #360,R1 ;SET INSTRUCTION SOURCE = INBUS REG 17
460 023644 052701 000002 BIS #2,R1 ;SET INSTRUCTION DESTINATION = BSEL2
461 023650 052701 021000 BIS #MVIOX,R1 ;SET REST OF MOVE INSTRUCTION
462 023654 010137 023664 MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
463 023660 004737 004046 JSR PC,LOOPIN ;GET MICROPROCESSOR LOOPING ON MOVE INSTRUCTION
464 023664 000000 2$: .WORD 0 ;INSTRUCTION GOES HERE
    
```

```

465
466 -----
467 ; WAIT FOR MCLK BIT TO BE SET TO 1
468 -----
    
```

```

469 023666 005037 002510 CLR REG0 ;INIT PROGRAM TIMER
470 023672 117737 156554 002364 3$: MOVB @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
471 023700 132737 000002 002364 BITB #MCLK,REDBYT ;SEE IF MCLK BIT = 1 YET
472 023706 001031 BNE 6$ ;BR IF MCLK = 1
473 023710 005237 002510 INC REG0 ;INCREMENT TIMER
474 023714 001366 BNE 3$ ;BR IF PROGRAM TIMER DID NOT TIME OUT YET
475 ; (TIME OUT = SEVERAL HUNDRED MILLI-SEC)
476 023716 012737 000002 002404 MOV #MCLK,GOODAT ;SET EXPECTED REG CONTENTS
477 023724 013737 002364 002406 MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
478 023732 004737 003770 JSR PC,GETREG ;GET REGS FOR PRINTOUT
479 ;REPORT MCLK BIT STUCK AT 0
480 023736 ERRDF 5,EM5,ERR2
    
```

```

023736 104455 TRAP C$ERDF
023740 000005 .WORD 5
023742 012266 .WORD EM5
023744 014606 .WORD ERR2
    
```

```

481 ;TYPE 'PLEASE INSURE MICROPROCESSOR RUN SWITCH IS ON'
482 023746 PRINTF #FMT24
023746 012746 012012 MOV #FMT24,-(SP)
023752 012746 000001 MOV #1,-(SP)
023756 010600 MOV SP,R0
023760 104417 TRAP C$PNTF
023762 062706 000004 TRAP C$PNTF
023766 000137 024174 ADD #4,SP
    
```

```

483 023766 000137 024174 JMP A1 ;ESCAPE TO END OF TEST
484
485
486 -----
487 ; WAIT FOR MCLK BIT TO BE CLEARED TO 0
488 -----
    
```

```

488 023772 6$: CLR REG0 ;INIT PROGRAM TIMER
489 023772 005037 002510 MOVB @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
490 023776 117737 156450 002364 8$: BITB #MCLK,REDBYT ;SEE IF MCLK BIT = 0 YET
491 024004 132737 000002 002364 BEQ 10$ ;BR IF MCLK = 0
492 024012 001430 INC REG0 ;INCREMENT TIMER
493 024014 005237 002510 BNE 8$ ;BR IF TIMER DID NOT TIME OUT YET
494 024020 001366 CLR GOODAT ;SET EXPECTED REG CONTENTS
495 024022 005037 002404 MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
496 024026 013737 002364 002406 JSR PC,GETREG ;GET REGS FOR PRINTOUT
497 024034 004737 003770 ;REPORT MCLK BIT STUCK AT 1
498 ERRDF 6,EM6,ERR2
    
```

```

499 024040 TRAP C$ERDF
024040 104455 .WORD 6
024042 000006 .WORD EM6
024044 012317 .WORD ERR2
024046 014606
    
```

```

500 ;TYPE 'PLEASE INSURE MICROPROCESSOR RUN SWITCH IS ON'
501 024050 PRINTF #FMT24
    
```

```

024050 012746 012012
024054 012746 000001
024060 010600
024062 104417
024064 062706 000004
502 024070 000137 024174          JMP      A1          ;ESCAPE TO END OF TEST
503
504
505          ;-----
506          ; WAIT FOR MCLK BIT TO BE SET TO 1 AGAIN
507          ;-----
508 024074 005037 002510          10$:      CLR      REG0          ;INIT PROGRAM TIMER
509 024100 117737 156346 002364 12$:      MOVB   @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
510 024106 132737 000002 002364      BITB   #MCLK,REDBYT ;SEE IF MCLK BIT = 1 YET
511 024114 001027          BNE    A1          ;BE IF MCLK = 1
512 024116 005237 002510          INC    REG0          ;INCREMENT TIMER
513 024122 001366          BNE    12$         ;BR IF TIMER DID NOT TIME OUT YET
514 024124 012737 000002 002404      MOV    #MCLK,GOODAT ;SET EXPECTED REG CONTENTS
515 024132 013737 002364 002406      MOV    REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
516 024140 004737 003770          JSR    PC,GETREG    ;GET REGS FOR PRINTOUT
517          ;REPORT MCLK BIT STUCK AT 0
518 024144          ERRDF 5,EM5,ERR2
          TRAP   C$ERDF
          .WORD 5
          .WORD EM5
          .WORD ERR2
          ;TYPE 'PLEASE INSURE MICROPROCESSOR RUN SWITCH IS ON'
519          ;TYPE 'PLEASE INSURE MICROPROCESSOR RUN SWITCH IS ON'
520 024154          PRINTF #FMT24
          MOV    #FMT24,-(SP)
          MOV    #1,-(SP)
          MOV    SP,R0
          TRAP   C$PNTF
          ADD    #4,SP
          024154 012746 012012
          024160 012746 000001
          024164 010600
          024166 104417
          024170 062706 000004
521 024174
522 024174 004737 003576          A1:      JSR    PC,MSTCLR ;ISSUE MASTER CLEAR
523 024200          ENDTST
          L10036:
          TRAP   C$ETST
524
525
526
527
528
529
530          ;*****
531          ;SBTTL      TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST
532          ;*
533          ;* FIRST, ALL READ/WRITE BITS OF EXTENDED RECS AX0-AX3 ARE SET BY LOADING
534          ;* A DIFFERENT WORD OF PATTERN H INTO EACH REG. THEN, A MASTER CLEAR IS
535          ;* ISSUED AND EACH REG IS READ AND COMPARED TO A WORD OF PATTERN I, WHICH
536          ;* CONTAINS THE INITIALIZED STATES OF ALL THE EXTENDED REGS.
537          ;* PATTERN H = 000,000,377,017,377,377,375,377
538          ;* PATTERN I = 000,000,000,000,000,103,000,000
539          ;*****
          BGNTST
          T14::
540 024202 004737 003576          JSR    PC,MSTCLR ;ISSUE AN INITIAL MASTER CLEAR
541 024206 005037 002402          CLR    AXNUM      ;INIT AX REG BYTE NO. TO 0
    
```

```

542 024212 012701 002654      MOV    #PATH,R1      ;INIT DATA PATTERN POINTER
543 024216 112137 002374      1$:  MOVVB (R1)+,WAX15  ;SET BITS TO LOAD INTO LO BYTE
544 024222 112137 002376      MOVVB (R1)+,WAX16  ;SET BITS TO LOAD INTO HI BYTE
545 024226 004737 004264      JSR   PC,WRITAX     ;WRITE EXTENDED REG
546 024232 032737 000002 002420  BIT    #WRDYTO,ERROR1 ;SEE IF READY FAILED TO SET
547 024240 001413                BEQ    8$           ;BR IF NOT
548 024242 012737 000014 002400  MOV    #14,REGNUM   ;SET LU REG NO = 14
549 024250 004737 003770      JSR   PC,GETREG     ;GET REGS FOR PRINTOUT
550                                ;REPORT READY NOT SET AFTER AX REG WRITE
551 024254      ERRDF 52,EM52,ERR9
                                TRAP  C$ERDF
                                .WORD  52
                                .WORD  EM52
                                .WORD  ERR9
552 024264      ESCAPE TST
                                TRAP  C$ESCAPE
                                .WORD  L10037-.
553 024270 062737 000002 002402  8$:  ADD    #2,AXNUM     ;INCR REG BYTE NO.
554 024276 020127 002664      CMP    R1,#PATI     ;SEE IF ALL REGS WRITTEN YET
555 024302 103745                BLO   1$           ;BR IF NOT DONE WRITING YET
556 024304 004737 003576      JSR   PC,MSTCLR     ;ISSUE MASTER CLEAR
557 024310 005037 002402      CLR   AXNUM        ;INIT EXTENDED REG BYTE NO. TO 0
558 024314 012701 002664      MOV    #PATI,R1     ;INIT DATA PAT POINTER FOR READS
559 024320 004737 004076      2$:  JSR   PC,READAX   ;READ AN EXTENDED REG
560 024324 032737 000001 002420  BIT    #RRDYTO,ERROR1 ;SEE IF READY FAILED TO SET
561 024332 001413                BEQ    10$          ;BR IF NOT
562 024334 012737 000014 002400  MOV    #14,REGNUM   ;SET LU REG NO. = 14
563 024342 004737 003770      JSR   PC,GETREG     ;GET REGS FOR PRINTOUT
564                                ;REPORT READY NOT SET AFTER AX REG READ
565 024346      ERRDF 53,EM53,ERR9
                                TRAP  C$ERDF
                                .WORD  53
                                .WORD  EM53
                                .WORD  ERR9
566 024356      ESCAPE TST
                                TRAP  C$ESCAPE
                                .WORD  L10037-.
567 024362 023727 002402 000006  10$:  CMP    AXNUM,#6    ;SEE IF AX3-15
568 024370 001003                BNE   3$           ;BR IF NOT
569 024372 142737 000372 002370  3$:  BICB  #AX315U,RAX15 ;MASK OFF UNPREDICTABLE BITS
570 024400 123711 002370      CMPB  RAX15,(R1)   ;COMPARE LO BYTE TO EXPECTED VALUE
571 024404 001417                BEQ   4$           ;BR IF DATA MATCHES
572 024406 005037 002404      CLR   GOODAT       ;
573 024412 111137 002404      MOVVB (R1),GOODAT  ;GET EXPECTED DATA BYTE
574 024416 013737 002370 002406  MOV    RAX15,BADDAT ;GET ACTUAL DATA BYTE
575 024424 004737 004500      JSR   PC,GETALL    ;GET REGS FOR PRINTOUT
576                                ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
577 024430      ERRDF 2,EM2,ERR3
                                TRAP  C$ERDF
                                .WORD  2
                                .WORD  EM2
                                .WORD  ERR3
578 024440      ESCAPE TST
                                TRAP  C$ESCAPE
                                .WORD  L10037-.
579 024444 005237 002402  4$:  INC    AXNUM       ;INCREMENT AX BYTE NO.
580 024450 005201                INC    R1          ;INCREMENT PAT POINTER
    
```

TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST

```

581 024452 123711 002372          CMPB   RAX16,(R1)      ;COMPARE HI BYTE TO EXPECTED VALUE
582 024456 001417                   BEQ    6$              ;BR IF DATA MATCHES
583 024460 005037 002404          CLR    GOODAT
584 024464 111137 002404          MOVB   (R1),GOODAT    ;GET EXPECTED DATA BYTE
585 024470 013737 002372 002406  MOV    RAX16,BADDAT   ;GET ACTUAL DATA BYTE
586 024476 004737 004500          JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
587                                     ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
588 024502                                ERRDF  2,EM2,ERR3
                                TRAP   C$ERDF
                                .WORD  2
                                TRAP   C$EM2
                                .WORD  EM2
                                .WORD  ERR3
589 024512                                ESCAPE TST
                                TRAP   C$ESCAPE
                                .WORD  L10037-
590 024516 005237 002402 6$:    INC    AXNUM          ;INCR AX BYTE NO.
591 024522 005201                   INC    R1              ;INCR PAT POINTER
592 024524 020127 002674          CMP    R1,#PATJ      ;SEE IF ALL REGS READ YET
593 024530 103673                   BLO   2$              ;BR IF NOT DONE READING YET
594 024532                                ENDTST
                                L10037:
                                TRAP   C$SETST
024532 104401
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610

```

```

:*****
:SBTTL      TEST 15 - EXTENDED REGISTER ADDRESSING TEST
:
:* FIRST, ISSUE A MASTER CLEAR TO PUT REGS INTO INITIALIZED STATES SHOWN IN
:* PATTERN I. THEN, WRITE A DIFFERENT WORD OF PATTERN J INTO EACH EXTENDED (AX)
:* REG, AND AFTER EACH WRITE, READ AND COMPARE ALL EXTENDED REGS TO EXPECTED
:* VALUES.
:* PATTERN I = 000,000,000,000,000,103,000,000
:* PATTERN J = 000,000,010,002,004,103,001,100
:*****
BGNTST

```

```

611 024534 004737 003576          JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
612 024540 012701 002664          MOV    #PATI,R1      ;INIT POINTER TO PAT I
613 024544 012702 002500          MOV    #REDDAT,R2    ;INIT POINTER TO EXPECTED DATA AREA
614 024550 112122 002674 3$:    MOVB   (R1)+,(R2)+    ;MOVE PAT I INTO REDDAT TABLE
615 024552 020127 002674          CMP    R1,#PATJ
616 024556 103774                   BLO   3$
617 024560 005001                   CLR    R1              ;INIT INDEX FOR WRITING
618 024562 010137 002402 002374 6$:    MOV    R1,AXNUM      ;GET AX BYTE NO. FOR WRITING
619 024566 116137 002674 002376  MOVB   PATJ(R1),WAX15 ;SET LO AX BYTE TO BE WRITTEN
620 024574 116137 002675 002376  MOVB   PATJ+1(R1),WAX16 ;SET HI AX BYTE TO BE WRITTEN
621 024602 113761 002374 002500  MOVB   WAX15,REDDAT(R1) ;SET EXPECTED LO BYTE IN TABLE
622 024610 113761 002376 002501  MOVB   WAX16,REDDAT+1(R1) ;SET EXPECTED HI BYTE IN TABLE
623 024616 004737 004264          JSR    PC,WRITAX     ;WRITE DATA BYTES INTO EXTENDED REG
624 024622 005003                   CLR    R3              ;INIT INDEX FOR READING
625 024624 010337 002402 000006 9$:    MOV    R3,AXNUM      ;GET AX BYTE NO. FOR READING
626 024630 004737 004076          JSR    PC,READAX     ;READ 2 AX BYTES
627 024634 023727 002402          CMP    AXNUM,#6      ;SEE IF AX3-15
628 024642 001003                   BNE   10$            ;BR IF NOT
                                T15::

```



```

629 024644 142737 000372 002370          BICB    #AX315U,RAX15    ;MASK OFF UNPREDICTABLE BITS
630 024652 123763 002370 002500 10$:    CMPB    RAX15,REDDAT(R3) ;COMPARE LO BYTE READ TO EXPECTED
631 024660 001420          BEQ     12$              ;BR IF LO BYTE MATCHES EXPECTED
632 024662 005037 002404          CLR     GOODAT
633 024666 116337 002500 002404    MOVB    REDDAT(R3),GOODAT ;GET EXPECTED LO BYTE
634 024674 013737 002370 002406    MOV     RAX15,BADDAT    ;GET ACTUAL DATA
635 024702 004737 004500          JSR     PC,GETALL       ;GET REGS FOR PRINTOUT
636                                     ;REPORT REG MISCMPARE
637 024706          ERRDF  3,EM3,ERR3
                                     TRAP   C$ERDF
024706 104455                                     .WORD  3
024710 000003                                     .WORD  EM3
024712 012174                                     .WORD  ERR3
024714 015114
638 024716          ESCAPE  TST
                                     TRAP   C$ESCAPE
024716 104410                                     .WORD  L10040-.
024720 000102
639 024722 005237 002402 002501 12$:    INC     AXNUM            ;INCR AX BYTE NO.
640 024726 123763 002372 002501    CMPB    RAX16,REDDAT+1(R3) ;COMPARE HI BYTE READ TO EXPECTED
641 024734 001420          BEQ     15$              ;BR IF HI BYTE MATCHES EXPECTED
642 024736 005037 002404          CLR     GOODAT
643 024742 116337 002501 002404    MOVB    REDDAT+1(R3),GOODAT ;GET EXPECTED HI BYTE
644 024750 013737 002372 002406    MOV     RAX16,BADDAT    ;GET ACTUAL DATA
645 024756 004737 004500          JSR     PC,GETALL       ;GET REGS FOR PRINTOUT
646                                     ;REPORT REG MISCMPARE
647 024762          ERRDF  3,EM3,ERR3
                                     TRAP   C$ERDF
024762 104455                                     .WORD  3
024764 000003                                     .WORD  EM3
024766 012174                                     .WORD  ERR3
024770 015114
648 024772          ESCAPE  TST
                                     TRAP   C$ESCAPE
024772 104410                                     .WORD  L10040-.
024774 000026
649 024776 062703 000002 000010 15$:    ADD     #2,R3            ;INCR INDEX FOR READS
650 025002 020327 000010          CMP     R3,#10          ;SEE IF ALL AX BYTES READ YET
651 025006 002706          BLT     9$              ;BR IF NOT DONE READING YET
652 025010 062701 000002          ADD     #2,R1            ;INCR INDEX FOR WRITES
653 025014 020127 000010          CMP     R1,#10          ;SEE IF ALL AX BYTES WRITTEN YET
654 025020 002660          BLT     6$              ;BR IF NOT DONE WRITING YET
655 025022          ENDTST
                                     L10040: TRAP   C$ETST
025022 104401

```

656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671

```

:*****
:SBITL      TEST 16 - REGS 15,16 / AX2-15,AX2-16 READ/WRITE BIT TEST
:*
:* USING REGS 15,16, THE INDIRECT REGS AX2-15,AX2-16 (USYRT REGS 4,5) ARE
:* WRITTEN AND READ USING EACH WORD OF PATTERN K. AX2-15 IS COMPARED
:* TO THE WORD WRITTEN, AND AX2-16 IS ALWAYS COMPARED TO 103. (AX2-16 IS NOT
:* WRITEABLE).
:* PATTERN K =
:* FOR REG 15: 000,377,125,252,001,002,004,010,020,040,100,200,000,000,
:*              000,000,000,000,000,000,376,375,373,367,357,337,277,177,
:*              377,377,377,377,377,377,377,377
:*****

```

```

672          :*      FOR REG 16: 000,377,125,252,000,000,000,000,000,000,001,002,
673          :*      004,010,020,040,100,200,377,377,377,377,377,377,377,
674          :*      376,375,373,367,357,337,277,177.
675          :*      *****
676 025024    BGNTST
677 025024    004737 003576          JSR    PC,MSTCLR          ;ISSUE MASTER CLEAR          T16::
678 025030    012701 002704          MOV    #PATK,R1          ;INIT DATA PATTERN POINTER
679 025034    3$:
680 025034    BGNSEG
681 025036    104404          TRAP   C$BSEG
682 025036    012737 000004 002402    MOV    #4,AXNUM          ;SET BYTE NO. = 4
683 025044    111137 002374          MOVB   (R1),WAX15        ;SET DATA TO WRITE INTO LO BYTE
684 025050    116137 000001 002376    MOVB   1(R1),WAX16       ;SET DATA TO WRITE INTO HI BYTE
685 025056    143737 002565 002374    BICB   ANBITS+4,WAX15    ;MASK OFF NON-READ/WRITE BITS IN LO BYTE
686 025064    143737 002566 002376    BICB   ANBITS+5,WAX16    ;MASK OFF NON-READ/WRITE BITS IN HI BYTE
687 025072    004737 004264          JSR    PC,WRITAX         ;LOAD DATA INTO AX2-15,AX2-16
688 025076    004737 004076          JSR    PC,READAX        ;READ AX2-15 AND AX2-16
689 025102    123737 002370 002374    CMPB   RAX15,WAX15       ;COMPARE LO BYTE DATA READ
690 025110    001416          BEQ    6$                ;BR IF DATA MATCHES
691 025112    013737 002374 002404    MOV    WAX15,GOODAT     ;SET EXPECTED DATA
692 025120    013737 002370 002406    MOV    RAX15,BADDAT     ;SET ACTUAL DATA
693 025126    004737 004500          JSR    PC,GETALL        ;GET REGS FOR PRINTOUT
694          ;REPORT REG MISCOMPARE
695          ERRDF 3,EM3,ERR3
696 025132    104455          TRAP   C$ERDF
697 025132    000003          .WORD 3
698 025134    012174          .WORD EM3
699 025136    015114          .WORD ERR3
700 025140    ESCAPE SEG
701 025142    104410          TRAP   C$ESCAPE
702 025142    000052          .WORD 10000$-.
703 025146    005237 002402 002671    6$:  INC    AXNUM          ;SET AX BYTE NO. = 5
704 025152    123737 002372          CMPB   RAX16,PATI+5     ;COMPARE HI BYTE DATA READ
705 025160    001416          BEQ    9$                ;BR IF DATA MATCHES
706 025162    005037 002404          CLR    GOODAT
707 025166    113737 002671 002404    MOVB   PATI+5,GOODAT    ;SET EXPECTED DATA
708 025174    013737 002372 002406    MOV    RAX16,BADDAT     ;SET ACTUAL DATA
709 025202    004737 004500          JSR    PC,GETALL        ;GET REGS FOR PRINTOUT
710          ;REPORT REG MISCOMPARE
711          ERRDF 3,EM3,ERR3
712 025206    104455          TRAP   C$ERDF
713 025206    000003          .WORD 3
714 025210    012174          .WORD EM3
715 025212    015114          .WORD ERR3
716 025216    9$:
717 025216    ENDSEG
718 025216    10000$:
719 025216    104405          TRAP   C$ESEG
720 025220    062701 000002          ADD    #2,R1            ;INCR PATTERN POINTER
721 025224    020127 003014          CMP    R1,#PATL        ;SEE IF ALL DATA WRITTEN YET
722 025230    103701          BLO    3$                ;BR IF NOT DONE YET
723          ENDTST
724 025232    104401          L10041:
725          TRAP   C$ETST
    
```

713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756

```

:*****
:SBTTL      TEST 17 - AX0-15,AX0-16 READ/WRITE BIT TEST
:*
:* IN THIS TEST, A MASTER CLEAR IS DONE, AND THEN A WRITE, READ, AND COMPARE
:* ARE PERFORMED IN REGS AX0-15,AX0-16 USING EACH WORD OF PATTERN L.
:* ANY BITS IN AX0-15,AX0-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)
:* IN THE EXPECTED VALUE BEFORE COMPARISON.
:* PATTERN L =
:*   FOR REG 15: 000,377,000
:*   FOR REG 16: 000,377,000.
:*****
    
```

```

BGNTST
T17::
JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
MOV    #PATL,R1      ;INIT DATA PATTERN POINTER
3$:
BGNSEG
TRAP   C$BSEG
MOV    #0,AXNUM      ;SET BYTE NO. = 0
MOVB   (R1),WAX15    ;SET DATA TO WRITE INTO LO BYTE
MOVB   1(R1),WAX16   ;SET DATA TO WRITE INTO HI BYTE
BICB   ANBITS+0,WAX15 ;MASK OFF NON-READ/WRITE BITS IN LO BYTE
BICB   ANBITS+1,WAX16 ;MASK OFF NON-READ/WRITE BITS IN HI BYTE
JSR    PC,WRITAX     ;LOAD DATA INTO AX0-15,AX0-16
JSR    PC,READAX     ;READ AX0-15 AND AX0-16
CMPB   RAX15,WAX15   ;COMPARE LO BYTE DATA READ
BEQ    6$            ;BR IF DATA MATCHES
MOV    WAX15,GOODAT  ;SET EXPECTED DATA
MOV    RAX15,BADDAT  ;SET ACTUAL DATA
JSR    PC,GETALL     ;GET REGS FOR PRINTOUT
;REPORT REG MISCOMPARE
ERRDF  3,EM3,ERR3
TRAP   C$ERDF
.WORD  3
.WORD  EM3
.WORD  ERR3
746:
ESCAPE SEG
TRAP   C$ESCAPE
.WORD  10000$-
6$:
INC    AXNUM         ;SET AX BYTE NO. = 1
CMPB   RAX16,WAX16  ;COMPARE HI BYTE DATA READ
BEQ    9$            ;BR IF DATA MATCHES
MOV    WAX16,GOODAT ;SET EXPECTED DATA
MOV    RAX16,BADDAT ;SET ACTUAL DATA
JSR    PC,GETALL     ;GET REGS FOR PRINTOUT
;REPORT REG MISCOMPARE
ERRDF  3,EM3,ERR3
TRAP   C$ERDF
.WORD  3
.WORD  EM3
.WORD  ERR3
9$:
ENDSEG
10000$:
    
```


025620	000003						.WORD	3
025622	012174						.WORD	EM3
025624	015114						.WORD	ERR3
802 025626				9\$:				
803 025626					ENDSEG			
025626							10000\$:	
025626	104405						TRAP	C\$ESEG
804 025630	062701	000002			ADD #2,R1			
805 025634	020127	003014			CMP R1,#PATL			
806 025640	103703				BLO 3\$			
807 025642				ENDTST				
025642							L10043:	
025642	104401						TRAP	C\$ETST

808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831

```

:*****
:SBTTL      TEST 19 - AX3-15,AX3-16 READ/WRITE BIT TEST
:
:* IN THIS TEST A MASTER CLEAR IS DONE AND THEN A WRITE, READ, AND COMPARE ARE
:* PERFORMED IN REGS AX3-15,AX3-16 USING EACH WORD OF PATTERN V FOR WRITING,
:* AND PATTERN U FOR COMPARING.
:* ANY BITS IN AX3-15,AX3-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)
:* IN THE EXPECTED VALUE BEFORE COMPARISON.
:* PATTERN V =
:*   FOR REG 15 : 000,333,331,323,313,233,133,000,000,000,000,
:*                000,000,000,000,000,000,000,000
:*   FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,
:*                100,200,346,345,343,307,247,147
:* PATTERN U =
:*   FOR REG 15 : 000,001,013,011,021,101,301,000,000,000,000,
:*                000,000,000,000,000,000,000,000
:*   FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,
:*                100,200,346,345,343,307,247,147
:*****
    
```

832 025644								
025644								
833 025644	004737	003576			JSR PC,MSTCLR		T19::	
834 025650	142777	000010	154572		BICB #LULoop,@BSEL1			
835 025656	012702	003152			MOV #PATV,R2			
836 025662	012701	003104			MOV #PATU,R1			
837 025666				3\$:				
838 025666					BGNSEG			
025666	104404						TRAP	C\$BSEG
839 025670	012737	000006	002402		MOV #6,AXNUM			
840 025676	111237	002374			MOVB (R2),WAX15			
841 025702	116237	000001	002376		MOVB 1(R2),WAX16			
842 025710	004737	004264			JSR PC,WRITAX			
843 025714	004737	004076			JSR PC,READAX			
844 025720	132737	000001	002374		BITB #TEST,WAX15			
845 025726	001003				BNE 4\$			
846 025730	142737	000372	002370		BICB #AX315U,RAX15			
847 025736	142737	000040	002370	4\$:	BICB #C32BCC,RAX15			
848 025744	123711	002370			CMPB RAX15,(R1)			
849 025750	001417				BEQ 6\$			

```

850 025752 005037 002404          CLR    GOODAT
851 025756 111137 002404          MOV    (R1),GOODAT      ;SET EXPECTED DATA
852 025762 013737 002370 002406  MOV    RAX15,BADDAT    ;SET ACTUAL DATA
853 025770 004737 004500          JSR    PC,GETALL       ;GET REGS FOR PRINTOUT
854                                     ;REPORT REG MISC
855 025774          ERRDF 3,EM3,ERR3
                                     TRAP  C$ERDF
025774 104455          .WORD 3
025776 000003          .WORD EM3
026000 012174          .WORD ERR3
026002 015114
856 026004          ESCAPE SEG
026004 104410          TRAP  C$ESCAPE
026006 000052          .WORD 10000$.
857 026010 005237 002402 6$:    INC    AXNUM            ;SET AX BYTE NO. = 7
858 026014 123761 002372 000001  CMPB  RAX16,1(R1)      ;COMPARE HI BYTE DATA READ
859 026022 001416          BEQ    9$              ;BR IF DATA MATCHES
860 026024 005037 002404          CLR    GOODAT
861 026030 116137 000001 002404  MOV    1(R1),GOODAT    ;SET EXPECTED DATA
862 026036 013737 002372 002406  MOV    RAX16,BADDAT    ;SET ACTUAL DATA
863 026044 004737 004500          JSR    PC,GETALL       ;GET REGS FOR PRINTOUT
864                                     ;REPORT REG MISC
865 026050          ERRDF 3,EM3,ERR3
                                     TRAP  C$ERDF
026050 104455          .WORD 3
026052 000003          .WORD EM3
026054 012174          .WORD ERR3
026056 015114
866 026060          9$:
867 026060          ENDSEG
                                     10000$:
026060 104405          TRAP  C$ESEG
868 026062 062702 000002          ADD    #2,R2           ;INCR PATTERN V POINTER
869 026066 062701 000002          ADD    #2,R1           ;INCR PATTERN U POINTER
870 026072 020127 003152          CMP    R1,#PATV       ;SEE IF ALL DATA WRITTEN YET
871 026076 103673          BLO    3$              ;BR IF NOT DONE YET
872 026100          ENDTST
                                     L10044:
026100 104401          TRAP  C$ETST
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889 026102          BGNST
026102
890 026102 004737 003576          JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
891 026106 012701 003050          MOV    #PATO,R1      ;INIT PAT 0 POINTER
    
```

```

*****
:SBTTL      TEST 20 - REG 17 - AX2-16 READ/WRITE, MASTER CLEAR TEST
:*
:* THIS TEST CONSISTS OF 2 SUBTESTS. IN THE FIRST SUBTEST, EACH BYTE OF PAT 0
:* IS WRITTEN INTO REG 17 AND AFTER EACH WRITE, AX2-16 IS READ AND COMPARED
:* TO A BYTE OF PAT P.
:* PATTERN O = 000,041,004,010,020,040,100,101,200,201,300,111,301,375
:* PATTERN P = 000,113,200,040,020,010,001,104,007,105,007,144,107,157
:* IN THE SECOND SUBTEST, REG 17 IS LOADED WITH 375, A MASTER CLEAR IS ISSUED,
:* AND AX2-16 IS COMPARED TO ITS INITIALIZED STATE (103).
*****
    
```

```

892 026112 012702 003066      MOV    #PATP,R2      ;INIT PAT P POINTER
893 026116 012737 000017 002400  MOV    #17,REGNUM   ;SET LU REG NO. = 17
894 026124 012737 000005 002402  MOV    #5,AXNUM     ;SET AX BYTE NO. = 5 FOR AX2-16
895                                     -----
896                                     ; WRITE REG 17, READ AND COMPARE AX2-16
897                                     -----
898 026132      BGNSUB
      026132
      026132 104402      T20.1: TRAP    C$BSUB
899 026134      3$:
900 026134      BGNSEG
      026134 104404      TRAP    C$BSEG
901 026136 111137 002366      MOVB   (R1),WRIBYT  ;SET BYTE TO BE WRITTEN
902 026142 004737 003722      JSR    PC,WRITLU   ;WRITE DATA BYTE INTO REG 17
903 026146 004737 004076      JSR    PC,READAX   ;READ AX2
904 026152 123712 002372      CMPB  RAX16,(R2)   ;COMPARE AX2-16 TO EXPECTED DATA
905 026156 001421      BEQ   6$          ;BR IF DATA MATCHES
906 026160 005037 002410      CLR   LOADAT
907 026164 111137 002410      MOVB  (R1),LOADAT ;SET DATA WHICH WAS WRITTEN
908 026170 005037 002404      CLR   GOODAT
909 026174 111237 002404      MOVB  (R2),GOODAT ;SET EXPECTED DATA READ
910 026200 013737 002372 002406  MOV   RAX16,BADDAT ;SET ACTUAL DATA READ
911 026206 004737 004500      JSR   PC,GETALL   ;GET REGS FOR PRINTOUT
912                                     ;REPORT REG MISCOMPARE
913 026212      ERRDF 3,EM3,ERR5
      026212 104455      TRAP   C$ERDF
      026214 000003      .WORD 3
      026216 012174      .WORD EM3
      026220 016254      .WORD ERR5
914 026222      6$:
915 026222      ENDSEG
      026222      10000$: TRAP   C$ESEG
916 026224 005201      INC   R1          ;INCR PAT O POINTER
917 026226 005202      INC   R2          ;INCR PAT P POINTER
918 026230 020127 003066      CMP   R1,#PATP    ;SEE IF ALL BYTES LOADED YET
919 026234 103737      BLO  3$          ;BR IF NOT
920 026236      ENDSUB
      026236      L10046: TRAP   C$ESUB
      026236 104403
921                                     -----
922                                     ; LOAD REG 17, DO MASTER CLEAR, READ AND COMPARE AX2-16
923                                     -----
924 026240      BGNSUB
      026240
      026240 104402      T20.2: TRAP   C$BSUB
925 026242 112737 000375 002366  MOVB  #375,WRIBYT  ;SET DATA TO BE LOADED
926 026250 004737 003722      JSR   PC,WRITLU   ;LOAD DATA INTO REG 17
927 026254 004737 003576      JSR   PC,MSTCLR   ;PERFORM MASTER CLEAR
928 026260 004737 004076      JSR   PC,READAX   ;READ AX2-15,AX2-16
929 026264 123737 002372 002671  CMPB  RAX16,PATI+5 ;SEE IF AX2-16 WAS INIT'D CORRECTLY
930 026272 001416      BEQ   6$          ;BR IF YES
931 026274 005037 002404      CLR   GOODAT
932 026300 113737 002671 002404  MOVB  PATI+5,GOODAT ;SET EXPECTED DATA
933 026306 013737 002372 002406  MOV   RAX16,BADDAT ;SET ACTUAL DATA
934 026314 004737 004500      JSR   PC,GETALL   ;GET REGS FOR PRINTOUT
935                                     ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
    
```

```

936 026320          ERRDF  2,EM2,ERR3
    026320 104455
    026322 000002
    026324 012135
    026326 015114
937 026330          6$:
938 026330          ENDSUB
    026330
    026330 104403
939 026332          ENDTST
    026332
    026332 104401
    
```

940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956

```

:*****
:SBTTL      TEST 21 - TRANSMITTER BUFFER DATA TEST
:* A MASTER CLEAR IS DONE FIRST, AND THEN A BYTE OF PATTERN N IS LOADED INTO
:* REG 11 AND THE NEXT BYTE IS LOADED TWICE INTO REG 10. THE PROGRAM THEN WAITS
:* AT LEAST 50 MICRO-SEC, AND THEN IT READS AND COMPARES AX1-15 TO THE BYTE
:* WHICH WAS LOADED INTO REG 10, AND IT READS AND COMPARES AX1-16 TO THE BYTE
:* WHICH WAS LOADED INTO REG 11. THIS PROCESS IS REPEATED (INCLUDING THE MASTER
:* CLEAR) FOR EACH PAIR OF BYTES IN PATTERN N.
:* PATTERN N =
:*   FOR REG 10: 000,125,252,377,000,000,000
:*   FOR REG 11: 000,000,000,000,005,012,017
:*****
    
```

```

957 026334          BGNST
    026334
958 026334 012701 003032
959 026340 012737 026602 002362
960 026346
    026346
    026346 104402
961 026350 004737 003576
962 026354 004737 004634
963 026360 000001
964 026362 012737 000011 002400
965 026370 111137 002366
966 026374 004737 003722
967 026400 012737 000010 002400
968 026406 116137 000001 002366
969 026414 004737 003722
970 026420 004737 003722
971 026424 004737 005120
972 026430 004737 004634
973 026434 000003
974 026436 012737 000002 002402
975 026444 004737 004076
976 026450 123761 002370 000001
977 026456 001420
978 026460 005037 002404
979 026464 116137 000001 002404
980 026472 013737 002370 002406
981 026500 004737 004500
    
```

```

T21::
MOV #PATN,R1 ;INIT PATTERN POINTER
MOV #A2,RETADR ;SET SUBROUTINE ERROR RETURN ADDRESS
BGNSUB
T21.1: TRAP C$BSUB
3$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR
    JSR PC,OSIRDY ;CHECK ORDY AND OCOR FOR EXPECTED STATES
    1
    MOV #11,REGNUM ;SET LU REG NO. = 11
    MOVB (R1),WRIBYT ;SET DATA BYTE TO BE WRITTEN
    JSR PC,WRITLU ;WRITE BYTE INTO REG 11
    MOV #10,REGNUM ;SET LU REG NO. = 10
    MOVB 1(R1),WRIBYT ;SET DATA BYTE TO BE WRITTEN
    JSR PC,WRITLU ;WRITE BYTE INTO REG 10
    JSR PC,WRITLU ;WRITE IT AGAIN (SO 2 ENTRIES ARE IN SILO)
    JSR PC,WAIT50 ;WAIT FOR SILO DATA TO RIPPLE
    JSR PC,OSIRDY ;CHECK ORDY AND OCOR FOR EXPECTED STATES
    3
    MOV #2,AXNUM ;SET BYTE NO. FOR AX1-15
    JSR PC,READAX ;READ AX1-15, AX1-16
    CMPB RAX15,1(R1) ;COMPARE AX1-15 TO EXPECTED
    BEQ 6$ ;BR IF MATCH
    CLR GOODAT
    MOVB 1(R1),GOODAT ;SET EXPECTED DATA
    MOV RAX15,BADDAT ;SET ACTUAL DATA
    JSR PC,GETALL ;GET REGS FOR PRINTOUT
    
```



```

026606
1023 026606 012737 027262 002362          MOV    #A3,RETADR      ;SET SUBR ERROR RETURN ADDR
1024
1025          ;-----
1026          ; SET MASTER CLEAR, CHECK FOR ORDY=1, OCOR=0
1027 026614 004737 003576          JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
1028 026620 004737 004634          JSR    PC,OSIRDY      ;CHK ORDY=1, OCOR=0
1029 026624 000001
1030
1031          ;-----
1032          ; LOAD 2 SOM CHARS, ALLOW SILO TO RIPPLE, CHK ORDY=1, OCOR=1
1033 026626 012737 000400 002422          MOV    #TXSOM,TXWORD  ;SET DATA TO WRITE INTO SILO
1034 026634 004737 005146          JSR    PC,LDTXSI      ;LOAD THE SILO WITH SOM
1035 026640 004737 005146          JSR    PC,LDTXSI      ;LOAD ANOTHER SOM
1036 026644 004737 005120          JSR    PC,WAIT50      ;WAIT FOR DATA TO RIPPLE
1037 026650 004737 004634          JSR    PC,OSIRDY      ;CHK ORDY=1, OCOR=1
1038 026654 000003
1039
1040          ;-----
1041          ; CLOCK LINE UNIT, CHK FOR OCOR = 0 WITHIN 3 CYCLES
1042 026656 005001
1043 026660 012737 000017 002400          CLR    R1              ;INIT CYCLE COUNTER TO 0
1044 026666 004737 005226          MOV    #17,REGNUM     ;SET REG NO. = 17
1045 026672 000001          3$: JSR    PC,STPLU      ;STEP LU 1 CYCLE
1046 026674 004737 003644          JSR    PC,READLU      ;READ REG 17
1047 026700 132737 000020 002364          BITB   #OCOR,REDBYT   ;SEE IF OCOR = 0 YET
1048 026706 001404          BEQ    6$              ;BR IF OCOR = 0
1049 026710 005201          INC    R1              ;INCR CYCLE COUNT
1050 026712 020127 000003          CMP    R1,#3          ;SEE IF 3 CYCLES DONE YET
1051 026716 002763          BLT    3$              ;BR IF NO
1052 026720 004737 004634          6$: JSR    PC,OSIRDY      ;CHK ORDY=1, OCOR=0
1053 026724 000001
1054
1055          ;-----
1056          ; LOAD 64 BINARY COUNT CHARS INTO SILO, CHK ORDY=0
1057 026726 005003
1058 026730 010337 002422          8$: CLR    R3              ;INIT PATTERN FOR WRITING
1059 026734 004737 005146          MOV    R3,TXWORD      ;SET DATA TO BE WRITTEN
1060 026740 004737 005120          JSR    PC,LDTXSI      ;LOAD DATA CHAR INTO TX SILO
1061 026744 020327 000077          JSR    PC,WAIT50      ;WAIT FOR DATA TO RIPPLE IN SILO
1062 026750 001004          CMP    R3,#63.        ;SEE IF 64TH CHAR JUST LOADED
1063 026752 012737 000002 026774          BNE    9$              ;BR IF NO
1064 026760 000403          MOV    #2,14$         ;SET UP TO CHK ORDY=0,OCOR=1
1065 026762 012737 000003 026774          BR     12$            ;BR IF NO
1066 026770 004737 004634          9$: MOV    #3,14$         ;SET UP TO CHK ORDY=1,OCOR=1
1067 026774 000000          12$: JSR    PC,OSIRDY      ;CHK ORDY, OCOR
1068 026776 005203          14$: .WORD 0
1069 027000 020327 000100          INC    R3              ;INCR PATTERN FOR WRITES
1070 027004 002751          CMP    R3,#64.        ;SEE IF 64 CHARS LOADED YET
1071          BLT    8$              ;BR IF NO
1072
1073          ;-----
1074 027006 012737 000011 002400          ; CLOCK LINE UNIT, CHECK ORDY = 1 WITHIN 8 CYCLES
1075 027014 005001          MOV    #11,REGNUM     ;SET REG NO. = 11
1076 027016 004737 005226          16$: CLR    R1              ;INIT CYCLE COUNT
1077 027022 000001          JSR    PC,STPLU      ;CLOCK LU FOR 1 CYCLE
1078 027024 004737 003644          JSR    PC,READLU      ;READ REG 11
    
```

```

1079 027030 132737 000020 002364          BITB  #ORDY,REDBYT  ;SEE IF ORDY = 1 YET
1080 027036 001004          BNE    19$         ;BR IF YES
1081 027040 005201          INC    R1          ;INCR CYCLE COUNT
1082 027042 C20127 000010          CMP    R1,#8.     ;SEE IF 8 CYCLES YET
1083 027046 002763          BLT   16$         ;BR IF NOT YET
1084 027050 004737 004634          19$: JSR    PC,OSIRDY ;CHK ORDY = 1, OCOR = 1
1085 027054 000003          3
1086
1087          ;-----
1088          ; READ AND COMPARE FIRST CHARACTER IN AX1-15
1089          ;-----
1089 027056 005004          CLR    R4         ;INIT PATTERN FOR READING
1090 027060 012737 000002 002402          MOV    #2,AXNUM   ;SET AX BYTE NO. FOR AX1-15
1091 027066 004737 004076          JSR    PC,READAX  ;READ AX1-15
1092 027072 123704 002370          CMPB  RAX15,R4    ;COMPARE AX1-15 TO EXPECTED
1093 027076 001415          BEQ   20$         ;BR IF MATCH
1094 027100 010437 002404          MOV    R4,GOODAT ;SET EXPECTED DATA
1095 027104 013737 002370 002406          MOV    RAX15,BADDAT ;SET ACTUAL DATA
1096 027112 004737 004500          JSR    PC,GETALL  ;GET REGS FOR PRINTOUT
1097          ;REPORT REG MISCMPARE
1098 027116          ERRDF 3,EM3,ERR3
1098 027116 104455          TRAP  C$ERDF
1098 027120 000003          .WORD 3
1098 027122 012174          .WORD EM3
1098 027124 015114          .WORD ERR3
1099 027126          ESCAPE TST
1099 027126 104410          TRAP  C$ESCAPE
1099 027130 000132          .WORD L10052-.
1100 027132          20$:
1101          ;-----
1102          ; LOAD AND COMPARE REST OF CHARS, MONITOR ORDY, OCOR
1103          ;-----
1104 027132 020327 000377          24$: CMP    R3,#255.  ;SEE IF ALL CHARS LOADED YET
1105 027136 003010          BGT   26$         ;BR IF YES
1106 027140 010337 002422          MOV    R3,TXWORD  ;SET DATA TO BE WRITTEN
1107 027144 004737 005146          JSR    PC,LDTXSI  ;LOAD DATA CHAR INTO TX SILO
1108 027150 005203          INC    R3         ;INCR DATA TO BE WRITTEN
1109 027152 004737 004634          JSR    PC,OSIRDY  ;CHK ORDY=0, OCOR=1
1110 027156 000002          2
1111 027160 005204          26$: INC    R4         ;INCR PAT FOR READING
1112 027162 004737 005226          JSR    PC,STPLU   ;CLOCK LINE UNIT FOR 8 CYCLES
1113 027166 000010          8.
1114 027170 004737 004076          JSR    PC,READAX  ;READ AX1-15
1115 027174 123704 002370          CMPB  RAX15,R4    ;COMPARE AX1-15 TO EXPECTED
1116 027200 001415          BEQ   27$         ;BR IF MATCH
1117 027202 010437 002404          MOV    R4,GOODAT ;SET EXPECTED DATA
1118 027206 013737 002370 002406          MOV    RAX15,BADDAT ;SET ACTUAL DATA
1119 027214 004737 004500          JSR    PC,GETALL  ;GET REGS FOR PRINTOUT
1120          ;REPORT REG MISCMPARE
1121 027220          ERRDF 3,EM3,ERR3
1121 027220 104455          TRAP  C$ERDF
1121 027222 000003          .WORD 3
1121 027224 012174          .WORD EM3
1121 027226 015114          .WORD ERR3
1122 027230          ESCAPE TST
1122 027230 104410          TRAP  C$ESCAPE
1122 027232 000030          .WORD L10052-.
1123 027234 020427 000377          27$: CMP    R4,#255.  ;SEE IF WE READ LAST CHAR YET
    
```

```

1124 027240 001004          BNE      29$          ;BR :F NOT
1125 027242 004737 004634  JSR      PC,OSIRDY    ;CHK ORDY=1, OCOR=0
1126 027246 000001          1
1127 027250 000404          BR       A3
1128 027252 004737 004634  29$:    JSR      PC,OSIRDY    ;CHK ORDY=1, OCOR=1
1129 027256 000003          3
1130 027260 000724          BR       24$          ;CONTINUE
1131 027262
1132 027262          A3:
      027262          ENDTST
      027262 104401
                                           L10052:
                                           TRAP   C$ETST
    
```

1133
 1134
 1135
 1136
 1137
 1138
 1139
 1140
 1141
 1142
 1143
 1144
 1145
 1146
 1147
 1148
 1149
 1150
 1151

```

:*****
:SBTTL      TEST 23 - TX MSG TIMING TEST, CHAR MODE, WITH CRC
:*
:* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)
:* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS
:* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN
:* CHARACTER ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND CRC-16.
:* THE FOLLOWING STEPS ARE DONE:
:* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN CHAR MODE.
:* SOM IS SET TWICE TO SEND 2 SYNCH CHARS. THEN, 2 000 CHARS ARE SENT, AND
:* THEN 2 TERMINATING SYNCHS ARE SENT.
:* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE
:* CLEARED STATE AFTER THE 3RD SYNCH COMPLETES.
:*****
BGNTST
    
```

```

1152 027264
      027264
1153 027264 012737 027376 002362  MOV     #A4,RETADR    T23::
1154 027272 004737 005512          JSR     PC,INITRN    ;SET TEST EXIT ADDRESS FOR ERRORS
1155 027276 000226          SYNCH          ;DO MASTER CLR, LOAD 2 SOM'2
1156 027300 000011          STRIP!DDCMP
1157 027302 004737 006074  JSR     PC,TXCHAR    ;LOAD A 000 CHAR, TX FIRST SYNCH (226)
1158 027306 000000          000
1159 027310 100010          CHPCHK!8.
1160 027312 004737 006074  JSR     PC,TXCHAR    ;LOAD 2ND 000 CHAR, TX 2ND SYNCH
1161 027316 000000          000
1162 027320 000010          8.
1163 027322 004737 006074  JSR     PC,TXCHAR    ;LOAD EOM CHAR, TX FIRST 000 CHAR
1164 027326 001000          TXEOM
1165 027330 000010          8.
1166 027332 004737 006074  JSR     PC,TXCHAR    ;LOAD EOM CHAR, TX 2ND 000 CHAR
1167 027336 001000          TXEOM
1168 027340 000010          8.
1169 027342 004737 006074  JSR     PC,TXCHAR    ;LOAD EOM, TX CRC-16 CHAR
1170 027346 001000          TXEOM
1171 027350 000020          16.
1172 027352 004737 006074  JSR     PC,TXCHAR    ;LOAD EOM, TX FIRST TERMINATING SYNCH
1173 027356 001000          TXEOM
1174 027360 000010          8.
1175 027362 004737 006074  JSR     PC,TXCHAR    ;LOAD EOM, TX 2ND TERMINATING SYNCH
1176 027366 001000          TXEOM
1177 027370 000010          8.
    
```

1178 027372 004737 006246 JSR PC,ENDTRN ;SET OC, MONITOR OCOR
1179 027376 A4: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
1180 027376 004737 003576
1181 027402 ENDTST
027402 L10053:
027402 104401 TRAP C\$ETST

1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200

```
*****  
:SBTTL TEST 24 - TX MSG TIMING TEST, BIT MODE, WITH CRC  
:  
:* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)  
:* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS  
:* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN  
:* BIT ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND CRC-CCITT-1.  
:* THE FOLLOWING STEPS ARE DONE:  
:* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN BIT MODE.  
:* SOM IS SET TWICE TO SEND 2 FLAG CHARS. THEN, 2 000 CHARS ARE SENT, AND  
:* THEN 2 TERMINATING FLAGS ARE SENT.  
:* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE  
:* CLEARED STATE.  
:*****  
BGNTST
```

1201 027404
027404
1202 027404 012737 027506 002362 MOV #A5,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS T24::
1203 027412 004737 005512 JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'2
1204 027416 000000
1205 027420 000000
1206 027422 004737 006074 JSR PC,TXCHAR ;LOAD A 000 CHAR, TX FIRST FLAG
1207 027426 000000
1208 027430 100010
1209 027432 004737 006074 JSR PC,TXCHAR ;LOAD 2ND 000 CHAR, TX 2ND FLAG
1210 027436 000000
1211 027440 000010
1212 027442 004737 006074 JSR PC,TXCHAR ;LOAD EOM CHAR, TX FIRST 000 CHAR
1213 027446 001000
1214 027450 000010
1215 027452 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX 2ND 000 CHAR AND CRC-CCITT-1 CHAR
1216 027456 001000
1217 027460 000030
1218 027462 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX FIRST TERMINATING FLAG
1219 027466 001000
1220 027470 000010
1221 027472 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX 2ND TERMINATING FLAG
1222 027476 001000
1223 027500 000010
1224 027502 004737 006246 JSR PC,ENDTRN ;SET OC, MONITOR OCOR
1225 027506 A5: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
1226 027506 004737 003576
1227 027512 ENDTST
027512 L10054:
027512 104401 TRAP C\$ETST

1228
1229

1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1268
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1277
 1278
 1279
 1280
 1281
 1282
 1283

027514
 027514
 027514 012737 027702 002362
 027522 004737 005512
 027526 000226
 027530 000311
 027532 012737 000014 002400
 027540 012737 000100 002366
 027546 004737 003722
 027552 004737 006074
 027556 000000
 027560 100010
 027562 004737 006074
 027566 000000
 027570 000010
 027572 004737 006074
 027576 001000
 027600 000010
 027602 004737 006074
 027606 001000
 027610 000010
 027612 004737 006074
 027616 001000
 027620 000010
 027622 004737 006074
 027626 001000
 027630 000010
 027632 004737 005226
 027636 000030
 027640 012737 000013 002400
 027646 004737 003644
 027652 032737 000040 002364
 027660 001006
 027662 004737 004500
 027666
 027666 104455
 027670 000074

```

:*****
:SBITL      TEST 25 - TX MSG TIMING TEST, CHAR MODE, WITH NO CRC
:*
:* IN THIS TEST, AN ENTIRE MESSAGE IS TRANSMITTED (USING STEPLU AND LULOOP)
:* AND THE PROGRAM MONITORS THE OCCURRENCE OF USYRT TX BUFFER EMPTY FLAGS
:* (BY SCANNING ORDY AND OCOR) AND OACT, AT EACH STEP. THE TEST IS DONE IN
:* CHARACTER ORIENTED PROTOCOL MODE, USING 8-BIT CHARS AND NO ERROR CHECKING.
:* THE FOLLOWING STEPS ARE DONE:
:* A MASTER CLEAR IS DONE, AND THE LINE UNIT IS PLACED IN CHAR MODE.
:* SOM IS SET TWICE TO SEND 2 SYNCH CHARS. THEN, 2 000 CHARS ARE SENT, AND
:* THEN 2 TERMINATING SYNCHS ARE SENT.
:* THE TEST IS PERFORMED WITH TXEN (REG14, BIT6) SET, AND THE PROGRAM CHECKS
:* THAT THIS HOLDS RTS HIGH PAST THE END OF THE MESSAGE.
:* THE TRANSMITTER IS THEN DISABLED, USING OC, AND OACT IS MONITORED FOR THE
:* CLEARED STATE.
:*****
BGNTST
T25::
MOV      #A6,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
JSR      PC,INITRN      ;DO MASTER CLR, LOAD 2 SOM*2
SYNCH
CRC2!CRC1!STRIP!DDCMP
MOV      #14,REGNUM      ;SET REG NO. = 14
MOV      #TXEN,WRIBYT    ;SET TXEN BIT
JSR      PC,WRITLU      ;LOAD TXEN BIT IN REG 14
JSR      PC,TXCHAR      ;LOAD A 000 CHAR, TX FIRST SYNCH (226)
000
CHPCHK!8.
JSR      PC,TXCHAR      ;LOAD 2ND 000 CHAR, TX 2ND SYNCH
000
8.
JSR      PC,TXCHAR      ;LOAD EOM CHAR, TX FIRST 000 CHAR
TXEOM
8.
JSR      PC,TXCHAR      ;LOAD EOM CHAR, TX 2ND 000 CHAR
TXEOM
8.
JSR      PC,TXCHAR      ;LOAD EOM, TX FIRST TERMINATING SYNCH
TXEOM
8.
JSR      PC,TXCHAR      ;LOAD EOM, TX 2ND TERMINATING SYNCH
TXEOM
8.
JSR      PC,STPLU      ;CLK PAST END OF MSG
24.
MOV      #13,REGNUM      ;SET REG NO. = 13
JSR      PC,READLU      ;READ REG 13
BIT      #RTS,REDBYT     ;CHK FOR RTS STILL SET
BNE      3$             ;BR IF RTS SET
JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
;REPORT RTS NOT SET
ERRDF   60,EM60,ERR7

TRAP    C$ERDF
        .WORD 60
    
```

1284	027672	014204						.WORD	EM60
	027674	017444						.WORD	ERR7
1285	027676	004737	006246	3\$:	JSR	PC,ENDTRN	;SET OC, MONITOR OCOR		
1286	027702	004737	003576	A6:	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR TO CLEAN UP		
1287	027706			ENDTST					
	027706	104401					L10055:	TRAP	C\$ETST

1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306

```

:*****
:SBTTL      TEST 26 - TX UNDERRUN SET AND CLEAR TEST - CHAR MODE
:
:* IN THIS TEST, A TX UNDERRUN ERROR IS FORCED IN EACH OF 2 SITUATIONS,
:* AND THEN CLEARED DIFFERENTLY IN EACH.
:* IN THE FIRST, A MESSAGE IS INITIATED, A 000 CHAR IS SENT, AND THE TX
:* BUFFER IS NOT SERVICED IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG,
:* WHICH CAUSES UNRR TO SET IN REG 11. THEN, SOM IS SET TO CLEAR THE ERROR,
:* AND THIS IS VERIFIED.
:* IN THE SECOND SITUATION, A MSG IS INITIATED, A 000 CHAR IS SENT, AND THE TX
:* BUFFER IS NOT SERVICED IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG, WHICH
:* AGAIN CAUSES UNRR TO SET. THEN, A MASTER CLEAR IS DONE, AND THE UNRR BIT
:* IS CHECKED TO BE CLEARED.
:*****
BGNTST
    
```

1307 027710
027710

T26::

 : CAUSE TX UNDERRUN, CHK UNRR = 1; SET SOM, CHK UNRR = 0

1311	027710	012737	030246	002362	MOV	#A7,RETADR	;SET TEST EXIT ADDRESS FOR ERRORS		
1312	027716	004737	005512		JSR	PC,INITRN	;DO MASTER CLEAR, LOAD 2 SOM'S		
1313	027722	000226			SYNCH				
1314	027724	000011			STRIP!DDCMP				
1315	027726	004737	006074		JSR	PC,TXCHAR	;LOAD A 000 CHAR, TX FIRST SYNCH		
1316	027732	000000			000				
1317	027734	100010			CHPCHK!8.				
1318	027736	004737	005226		JSR	PC,STPLU	;CLOCK THE TRANSMITTER UNTIL 7 BITS OF		
1319	027742	000016			14.		; THE 000 CHAR HAVE BEEN TRANSMITTED		
1320	027744	012737	000011	002400	MOV	#11,REGNUM	;SET LU REG NO. = 11		
1321	027752	004737	003644		JSR	PC,READLU	;READ REG 11		
1322	027756	132737	000001	002364	BITB	#UNRR,REDBYT	;CHK FOR UNRR = 0		
1323	027764	001410			BEQ	6\$;BR IF UNRR = 0		
1324	027766	004737	004500		JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
1325					;REPORT UNRR NOT CLEARED				
1326	027772				ERRDF	16,EM16,ERR7		TRAP	C\$ERDF
	027772	104455						.WORD	16
	027774	000020						.WORD	EM16
	027776	012576						.WORD	ERR7
	030000	017444							
1327	030002	000137	030246		JMP	A7	;SKIP TO END OF TEST		
1328	030006	004737	005226		6\$:	JSR	PC,STPLU	;CLOCK LAST BIT OF 000 CHAR	
1329	030012	000003			3				
1330	030014	004737	003644		JSR	PC,READLU	;READ REG 11		
1331	030020	132737	000001	002364	BITB	#UNRR,REDBYT	;CHK FOR UNRR = 1		

```

1332 030026 001010          BNE      9$          ;BR IF UNRR = 1
1333 030030 004737 004500  JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
1334          ;REPORT UNRR NOT SET
1335 030034          ERRDF  14,EM14,ERR7
                                TRAP      C$ERDF
                                .WORD    14
                                .WORD    EM14
                                .WORD    ERR7
1336 030044 000137 030246  JMP      A7          ;SKIP TO END OF TEST
1337 030050 012737 000400 002422 9$:  MOV      #TXSOM,TXWORD ;SET SOM CHAR TO BE WRITTEN
1338 030056 004737 005146  JSR      PC,LDTXSI  ;LOAD SOM CHAR INTO TX SILO
1339 030062 004737 005120  JSR      PC,WAIT50  ;WAIT FOR SILO DATA TO RIPPLE
1340 030066 004737 005226  JSR      PC,STPLU   ;CLOCK LU FOR 2 CYCLES
1341 030072 000002 2
1342 030074 004737 003644  JSR      PC,READLU  ;READ REG 11
1343 030100 132737 000001 002364 BITB     #UNRR,REDBYT ;CHK FOR UNRR = 0
1344 030106 001410 BEQ      12$        ;BR IF UNRR = 0
1345 030110 004737 004500  JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
1346          ;REPORT UNRR NOT CLEARED BY SOM
1347 030114          ERRDF  13,EM13,ERR7
                                TRAP      C$ERDF
                                .WORD    13
                                .WORD    EM13
                                .WORD    ERR7
1348 030124 000137 030246  JMP      A7          ;SKIP TO END OF TEST
1349 030130 12$:
1350 -----
1351 ;CAUSE TX UNDERRUN, CHK UNRR = 1; DO MASTER CLR, CHK UNRR = 0
1352 -----
1353 030130 004737 005512  JSR      PC,INITRN  ;DO MASTER CLEAR, LOAD 2 SOM'S
1354 030134 000226 SYNCH
1355 030136 000051 IDLE!STRIP!DDCMP
1356 030140 004737 006074  JSR      PC,TXCHAR  ;LOAD A 000 CHAR, TX FIRST SYNCH
1357 030144 000000 000
1358 030146 100010 CHPCHK!8.
1359 030150 004737 005226  JSR      PC,STPLU   ;STEP THE LU UNTIL 000 HAS BEN TRANSMITTED
1360 030154 000021 17.
1361 030156 004737 003644  JSR      PC,READLU  ;READ REG 11
1362 030162 132737 000001 002364 BITB     #UNRR,REDBYT ;CHK FOR UNR = 1
1363 030170 001010 BNE      16$        ;BR IF UNRR = 1
1364 030172 004737 004500  JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
1365          ;REPORT UNRR NOT SET
1366 030176          ERRDF  14,EM14,ERR7
                                TRAP      C$ERDF
                                .WORD    14
                                .WORD    EM14
                                .WORD    ERR7
1367 030206 000137 030246  JMP      A7          ;SKIP TO END OF TEST
1368 030212 004737 003576 16$:  JSR      PC,MSTCLR  ;ISSUE MASTER CLEAR
1369 030216 004737 003644  JSR      PC,READLU  ;READ REG 11
1370 030222 132737 000001 002364 BITB     #UNRR,REDBYT ;CHK FOR UNRR = 0
1371 030230 001406 BEQ      A7          ;BR IF UNRR = 0
1372 030232 004737 004500  JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
1373          ;REPORT UNRR NOT CLEARED BY OC
1374 030236          ERRDF  15,EM15,ERR7
                                TRAP      C$ERDF
                                .WORD    15
    
```



```

1427 030446 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX FIRST TERMINATING SYNCH
1428 030452 001000 TXEOM
1429 030454 000010 8.
1430 030456 004737 006246 JSR PC,ENDTRN ;CLEAR TRANSMITTER
1431 030462 A8:
1432 030462 ENDTST
030462 L10057:
030462 104401 TRAP C$ETST
    
```

1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448

```

:*****
:SBTTL TEST 28 - TRANSMIT CHAR LENGTH TIMING TEST - BIT MODE, CRC
:*
:* THE LINE UNIT IS PLACED IN BIT MODE AND A MESSAGE IS INITIATED
:* WITH 2 FLAG CHARS. NEXT, 2 8-BIT 000 CHARS ARE SENT, FOLLOWED BY 000 CHARS
:* WITH EACH OF THE FOLLOWING TRANSMITTER CHAR LENGTHS:
:* 1 BIT, 2 BITS, 3 BITS, 4 BITS, 5 BITS, 6 BITS, 7 BITS, AND 8 BITS.
:* (FOR EXAMPLE, A 5-BIT CHAR REQUIRES 5 CLOCK CYCLES TO BE TRANSMITTED).
:* TWO TERMINATING FLAGS ARE SENT AFTER THE DATA.
:*****
BGNTST
    
```

```

1449 030464 012737 031012 002362 MOV #A9,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
1450 030472 004737 005512 JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'S
1451 030476 000000 000
1452 030500 000000 000
1453 030502 004737 006074 JSR PC,TXCHAR ;LOAD FIRST 8-BIT 000 CHAR, TX FIRST FLAG
1454 030506 000000 000
1455 030510 100010 CHPCHK!8.
1456 030512 004737 006074 JSR PC,TXCHAR ;LOAD 2ND 8-BIT 000 CHAR, TX 2ND FLAG
1457 030516 000000 000
1458 030520 000010 8.
1459 030522 004737 006074 JSR PC,TXCHAR ;LOAD 1-BIT 000 CHAR, TX FIRST 8-BIT 000 CHAR
1460 030526 000000 000
1461 030530 000010 8.
1462 030532 012737 000006 002402 MOV #6,AXNUM ;SET BYTE NO. = 6 FOR AX3
1463 030540 012737 000000 002374 MOV #000,WAX15 ;SET DATA FOR AX3-15 = 0
1464 030546 012737 000040 002376 MOV #TXLENO,WAX16 ;SET TX CHAR LENGTH = 1 FOR AX3-16
1465 030554 004737 004264 JSR PC,WRITAX ;LOAD AX3-15,AX3-16
1466 030560 004737 006074 JSR PC,TXCHAR ;LOAD 2-BIT 000 CHAR, TX 2ND 8-BIT 000 CHAR
1467 030564 000000 000
1468 030566 000010 8.
1469 030570 012737 000100 002376 MOV #TXLEN1,WAX16
1470 030576 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 2
1471 030602 004737 006074 JSR PC,TXCHAR ;LOAD 3-BIT 000 CHAR, TX 1-BIT 000 CHAR
1472 030606 000000 000
1473 030610 000001 1
1474 030612 012737 000140 002376 MOV #TXLEN1!TXLENO,WAX16
1475 030620 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 3
1476 030624 004737 006074 JSR PC,TXCHAR ;LOAD 4-BIT 000 CHAR, TX 2-BIT 000 CHAR
1477 030630 000000 000
1478 030632 000002 2
1479 030634 012737 000200 002376 MOV #TXLEN2,WAX16
1480 030642 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 4
    
```

```

1481 030646 004737 006074 JSR PC,TXCHAR ;LOAD 5-BIT 000 CHAR, TX 3-BIT 000 CHAR
1482 030652 000000 000
1483 030654 000003 3
1484 030656 012737 000240 002376 MOV #TXLEN2!TXLENO,WAX16
1485 030664 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 5
1486 030670 004737 006074 JSR PC,TXCHAR ;LOAD 6-BIT 000 CHAR, TX 4-BIT 000 CHAR
1487 030674 000000 000
1488 030676 000004 4
1489 030700 012737 000300 002376 MOV #TXLEN2!TXLEN1,WAX16
1490 030706 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 6
1491 030712 004737 006074 JSR PC,TXCHAR ;LOAD 7-BIT 000 CHAR, TX 5-BIT 000 CHAR
1492 030716 000000 000
1493 030720 000005 5
1494 030722 012737 000340 002376 MOV #TXLEN2!TXLEN1!TXLENO,WAX16
1495 030730 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 7
1496 030734 004737 006074 JSR PC,TXCHAR ;LOAD 8-BIT 000 CHAR, TX 6-BIT 000 CHAR
1497 030740 000000 000
1498 030742 000006 6
1499 030744 012737 000000 002376 MOV #000,WAX16
1500 030752 004737 004264 JSR PC,WRITAX ;SET TX CHAR LENGTH = 8
1501 030756 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX 7-BIT 000 CHAR
1502 030762 001000 TXEOM
1503 030764 000007 7
1504 030766 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX 8-BIT 000 CHAR, CRC-CCITT-1 CHAR
1505 030772 001000 TXEOM
1506 030774 000031 25.
1507 030776 004737 006074 JSR PC,TXCHAR ;LOAD EOM, TX FIRST TERMINATING FLAG
1508 031002 001000 TXEOM
1509 031004 000010 8.
1510 031006 004737 006246 JSR PC,ENDTRN ;CLEAR TRANSMITTER
1511 031012 A9:
1512 031012 ENDTST
031012 L10060:
031012 104401 TRAP C$ETST
    
```

1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526

```

*****
.SBTTL TEST 29 - TXDATA BIT TEST - CHAR MODE, NO CRC
:*
:* THE LINE UNIT IS INITIALIZED AND A MSG IS INITIATED (USING STEPLU) WITH CRC-
:* 16 SELECTED IN CHAR MODE. TWO SYNCHS, 000,125,252,377,000, AND 2 TERMINATING
:* SYNCHS ARE THEN SENT. THE PROGRAM CHECKS EACH BIT OF THE TRANSMITTED
:* DATA CHARS, BY MONITORING TXDATA (REG 17) AS THE DATA IS CLOCKED OUT OF
:* THE USYRT TRANSMITTER.
*****
BGNTST
    
```

```

1527 031014 T29::
031014 JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'S
1528 031014 004737 005512
1529 031020 000226 SYNCH
1530 031022 000011 STRIP!DDCMP
1531 031024 012701 003224 MOV #MSG1+4,R1 ;GET POINTER TO DATA
1532 031030 010103 MOV R1,R3
1533 031032 012137 002422 3$: MOV (R1)+,TXWORD
1534 031036 004737 005146 JSR PC,LDIXSI ;LOAD A DATA CHAR INTO TX SILO
    
```

```

1535 031042 020127 003242      CMP      R1,#MSG1+18.      ;SEE IF ALL CHARS LOADED YET
1536 031046 103771              BLO      3$                ;BR IF NOT YET
1537 031050 004737 005120      JSR      PC,WAIT50         ;WAIT FOR SILO TO RIPPLE
1538 031054 004737 005226      JSR      PC,STPLU          ;CLOCK LU UNTIL SYNCHS ARE TX'D
1539 031060 100020              CHPCHK!16.
1540 031062 011337 031072      6$:     MOV      (R3),8$      ;GET EXPECTED DATA CHAR
1541 031066 004737 011150      JSR      PC,CKTBIT         ;CHECK TXDATA FOR CHAR BITS
1542 031072 000000              8$:     .WORD    0            ;EXPECTED CHAR GOES HERE
1543 031074 062703 000002      ADD      #2,R3             ;INCR PATTERN POINTER
1544 031100 020327 003236      CMP      R3,#MSG1+14.     ;SEE IF ALL CHARS CHECKED YET
1545 031104 103766              BLO      6$                ;BR IF NOT YET
1546 031106 004737 003576      16$:    JSR      PC,MSTCLR     ;ISSUE MASTER CLEAR TO CLEAN UP
1547 031112              ENDTST
    031112
    031112 104401
    
```

L10061: TRAP C\$ETST

1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562

```

:*****
:SBTTL      TEST 30 - USYRT RECEIVER MSG TEST - CHAR MODE, CRC
:*
:* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH
:* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH CRC-16
:* SELECTED. TWO SYNCHS, 000,125,252,377,000, AND FOUR TERMINATING SYNCHS ARE
:* SENT. THE PROGRAM MONITORS IACT, AND THE RCV'D CHARS AND CRC BYTES ARE READ
:* FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR IACT
:* STILL SET AFTER THE MESSAGE.
:*****
BGNTST
    
```

```

1563 031114              T30::
    031114
1564 031114 012737 031456 002362      MOV      #24$,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
1565 031122 004737 003576              JSR      PC,MSTCLR         ;ISSUE MASTER CLEAR
1566 031126 004737 010612              JSR      PC,SETUP          ;PROGRAM THE USYRT
1567 031132 000226              SYNCH
1568 031134 000013              STRIP! IERR!DDCMP
1569 031136 000000              000
1570 031140 000000              000
1571 031142 012737 000012 002400      MOV      #12,REGNUM       ;SET LU REG NO. = 12
1572 031150 005037 002402              CLR      AXNUM             ;SET AX BYTE NO. = 0 FOR AX0
1573 031154 112737 000040 002366      MOV      #LULP,WRIBYT
1574 031162 004737 003722              JSR      PC,WRITLU         ;SET LULP IN REG 12
1575 031166 012701 003220              MOV      #MSG1,R1         ;GET POINTER TO MSG DATA TABLE
1576 031172 012137 002422      3$:     MOV      (R1)+,TXWORD     ;GET CHAR TO BE LOADED
1577 031176 004737 005146              JSR      PC,LDTXSI         ;LOAD CHAR INTO TX SILO
1578 031202 020127 003246      CMP      R1,#MSG1+22.     ;SEE IF ALL MSG CHARS LOADED YET
1579 031206 103771              BLO      3$                ;BR IF NOT YET
1580 031210 004737 005120              JSR      PC,WAIT50         ;ALLOW DATA TO RIPPLE IN SILO
1581 031214 004737 005226              JSR      PC,STPLU          ;CLOCK LU FOR 40 CYCLES (UNTIL FIRST
1582 031220 000050              40.                        ; DATA CHAR IS ABOUT TO BE RECEIVED)
1583 031222 004737 006666              JSR      PC,IACTIV         ;CHK IACT = 0
1584 031226 000000              0
1585 031230 004737 005226              JSR      PC,STPLU          ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
1586 031234 000006              6
1587 031236 012701 003224              MOV      #MSG1+4,R1
1588 031242 004737 006666      10$:    JSR      PC,IACTIV         ;CHK IACT = 1
    
```

```

1589 031246 000001 1
1590 031250 004737 004076 JSR PC,READAX ;READ AX0
1591 031254 023721 002370 CMP RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
1592 031260 001415 BEQ 12$ ;BR IF RCV'D DATA OK
1593 031262 016137 177776 002404 MOV -2(R1),GOODAT ;GET EXPECTED DATA
1594 031270 013737 002370 002404 MOV RAX15,BADDAT ;GET ACTUAL DATA
1595 031276 004737 004500 JSR PC,GETALL ;GET REGS FOR PRINTOUT
1596 ;REPORT INCORRECT DATA CHAR RCV'D
1597 031302 ERRDF 26,EM26,ERR3
031302 104455 TRAP C$ERDF
031304 000032 .WORD 26
031306 013043 .WORD EM26
031310 015114 .WORD ERR3
1598 031312 000461 BR 24$
1599 031314 004737 005226 12$: JSR PC,STPLU ;CLOCK LU 8 CYCLES
1600 031320 000010 8.
1601 031322 020127 003236 CMP R1,#MSG1+14. ;SEE IF CHECKING HI CRC BYTE YET
1602 031326 103745 BLO 10$ ;BR IF NOT YET
1603 031330 004737 006666 JSR PC,IACTIV ;CHK IACT = 1
1604 031334 000001 1
1605 031336 004737 004076 JSR PC,READAX ;READ AX0
1606 031342 123727 002370 000160 CMPB RAX15,#160 ;CMP RCV'D CHAR TO EXPECTED HI CRC BYTE
1607 031350 001415 BEQ 16$ ;BR IF HI CRC BYTE RCV'D OK
1608 031352 012737 000160 002404 MOV #160,GOODAT ;GET EXPECTED DATA
1609 031360 013737 002370 002404 14$: MOV RAX15,BADDAT ;SET ACTUAL DATA
1610 031366 004737 004500 JSR PC,GETALL ;GET REGS FOR PRINTOUT
1611 ;REPORT INCORRECT CRC BYTE RCV'D
1612 031372 ERRDF 27,EM27,ERR3
031372 104455 TRAP C$ERDF
031374 000033 .WORD 27
031376 013075 .WORD EM27
031400 015114 .WORD ERR3
1613 031402 000425 BR 24$
1614 031404 004737 005226 16$: JSR PC,STPLU ;CLOCK LU FOR 8 CYCLES
1615 031410 000010 8.
1616 031412 004737 006666 JSR PC,IACTIV ;CHK IACT = 1
1617 031416 000001 1
1618 031420 012737 000034 002404 MOV #034,GOODAT ;GET EXPECTED LO CRC BYTE
1619 031426 004737 004076 JSR PC,READAX ;READ AX0
1620 031432 123727 002370 000034 CMPB RAX15,#034 ;CMP RCV'D CHAR TO EXPECTED LO CRC BYTE
1621 031440 001347 BNE 14$ ;BR IF LO CRC INCORRECT
1622 031442 004737 005226 JSR PC,STPLU ;CLOCK LU 8 CYCLES
1623 031446 000010 8.
1624 031450 004737 006666 JSR PC,IACTIV ;CHK IACT STILL = 1
1625 031454 000001 1
1626 031456 004737 003576 24$: JSR PC,MSTCLR ;ISSUE CLEAN-UP MASTER CLEAR
1627 031462 ENDTST
031462 L10062: TRAP C$ETST
031462 104401
1628
1629
1630
1631
1632
1633 ;*****
1634 .SBTTL TEST 31 - USYRT RECEIVER MSG TEST - BIT MODE, CRC
1635 ;*

```

```

1636      : * THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH
1637      : * LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH CRC-
1638      : * CCITT-1. TWO FLAGS, 000,125,252,377,000, AND TWO TERMINATING FLAGS ARE THEN
1639      : * SENT. THE PROGRAM MONITORS IACT, RSOM, AND THE RCV'D CHARS ARE READ
1640      : * FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR
1641      : * IACT = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT STILL = 0.
1642      : *****
1643 031464  BGNTST
1644 031464 012737 032020 002362      MOV      #24$,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
1645 031472 004737 003576      JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
1646 031476 004737 010612      JSR      PC,SETUP      ;PROGRAM THE USYRT
1647 031502 000000      000
1648 031504 000002      IERR
1649 031506 000000      000
1650 031510 000000      000
1651 031512 012737 000012 002400      MOV      #12,REGNUM      ;SET LU REG NO. = 12
1652 031520 005037 002402      CLR      AXNUM          ;SET AX BYTE NO. = 0 FOR AX0
1653 031524 112737 000040 002366      MOV      #LULP,WRIBYT
1654 031532 004737 003722      JSR      PC,WRITLU      ;SET LULP IN REG 12
1655 031536 012701 003220      MOV      #MSG1,R1      ;GET POINTER TO MSG DATA TABLE
1656 031542 012137 002422      3$: MOV      (R1)+,TXWORD    ;GET CHAR TO BE LOADED
1657 031546 004737 005146      JSR      PC,LDTXSI      ;LOAD CHAR INTO TX SILO
1658 031552 020127 003242      CMP      R1,#MSG1+18.   ;SEE IF ALL MSG CHARS LOADED YET
1659 031556 103771      BLO      3$            ;BR IF NOT YET
1660 031560 004737 005120      JSR      PC,WAIT50      ;ALLOW DATA TO RIPPLE IN SILO
1661 031564 004737 005226      JSR      PC,STPLU      ;CLOCK LU FOR 50 CYCLES (UNTIL FIRST
1662 031570 000062      50.                  ; DATA CHAR IS ABOUT TO BE RECEIVED)
1663 031572 004737 006666      JSR      PC,IACTIV      ;CHK IACT = 0
1664 031576 000000      0
1665 031600 004737 007054      JSR      PC,RSEOM      ;CHK RSOM = 0, REOM = 0
1666 031604 000000      0
1667 031606 004737 005226      JSR      PC,STPLU      ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
1668 031612 000006      6
1669 031614 012701 003224      MOV      #MSG1+4,R1
1670 031620 020127 003224      5$: CMP      R1,#MSG1+4   ;SEE IF 1ST CHAR RCV'D
1671 031624 001007      BNE      6$            ;BR IF NO
1672 031626 004737 006666      JSR      PC,IACTIV      ;CHK IACT = 1
1673 031632 000001      1
1674 031634 004737 007054      JSR      PC,RSEOM      ;CHK RSOM = 1, REOM = 0
1675 031640 000001      1
1676 031642 000420      BR      9$
1677 031644 020127 003234      6$: CMP      R1,#MSG1+12. ;SEE IF LAST CHAR RCV'D
1678 031650 001007      BNE      8$            ;BR IF NO
1679 031652 004737 006666      JSR      PC,IACTIV      ;CHK FOR IACT = 0
1680 031656 000000      0
1681 031660 004737 007054      JSR      PC,RSEOM      ;CHK RSOM = 0, REOM = 0
1682 031664 000000      0
1683 031666 000406      BR      9$
1684 031670 004737 006666      8$: JSR      PC,IACTIV      ;CHK FOR IACT = 1
1685 031674 000001      1
1686 031676 004737 007054      JSR      PC,RSEOM      ;CHK RSOM = 0, REOM = 0
1687 031702 000000      0
1688 031704 004737 004076      9$: JSR      PC,READAX      ;READ AX0
1689 031710 023721 002370      CMP      RAX15,(R1)+   ;COMPARE RCV'D CHAR TO EXPECTED
1690 031714 001415      BEQ      12$          ;BR IF RCV'D DATA OK
1691 031716 016137 177776 002404      MOV      -2(R1),GOODAT ;GET EXPECTED DATA
    
```

```

1692 031724 013737 002370 002406      MOV    RAX15,BADDAT    ;GET ACTUAL DATA
1693 031732 004737 004500              JSR    PC,GETALL      ;GET REGS FOR PRINTOUT
1694                                     ;REPORT INCORRECT DATA CHAR RCV'D
1695 031736 0104455                      ERRDF  26,EM26,ERR3
                                TRAP    C$ERDF
                                .WORD   26
                                .WORD   EM26
                                .WORD   ERR3
1696 031746 000424                      BR     24$
1697 031750 004737 005226      12$:  JSR    PC,STPLU      ;CLOCK LU 8 CYCLES
1698 031754 000010                      .
1699 031756 020127 003236      CMP    R1,#MSG1+14.   ;SEE IF ALL DATA CHARS CHECKED YET
1700 031762 103716                      BLO   5$              ;BR IF NOT YET
1701 031764 004737 006666      JSR    PC,IACTIV      ;CHK IACT = 0
1702 031770 000000                      .
1703 031772 004737 007054      JSR    PC,RSEOM       ;CHK RSOM = 0, REOM = 0
1704 031776 000000                      .
1705 032000 012737 000200 002366      MOV    #IC,WRIBYT
1706 032006 004737 003722      JSR    PC,WRITLU      ;SET IC (INPUT CLEAR) IN REG 12
1707 032012 004737 006666      JSR    PC,IACTIV      ;CHK IACT = 0
1708 032016 000000                      .
1709 032020 004737 003576      24$:  JSR    PC,MSTCLR   ;ISSUE CLEAN-UP MASTER CLEAR
1710 032024 000000                      .
                                ENDTST
                                L10063:
                                TRAP    C$ETST
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726 032026 0104401
032026
032026

```

```

:*****
.SBTTL      TEST 32 - USYRT RECEIVER MSG TEST - CHAR MODE, NO CRC
:*
:* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH
:* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH NO
:* ERROR DETECTION. TWO SYNCHS, 000,125,252,377,000, AND TWO SYNCHS ARE
:* THEN SENT. THE PROGRAM MONITORS IACT, AND THE RECEIVED CHARS ARE READ FROM
:* AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR IACT
:* STILL = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT = 0.
:*****
BGNTST

```

```

1727 032026 012737 032270 002362      MOV    #24$,RETADR    ;SET TEST EXIT ADDRESS FOR ERRORS
1728 032034 004737 003576      JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
1729 032040 004737 010612      JSR    PC,SETUP       ;PROGRAM THE USYRT
1730 032044 000226                      SYNCH
1731 032046 000313                      CRC2!CRC1!STRIP!IERR!DDCMP
1732 032050 000000                      000
1733 032052 000000                      000
1734 032054 012737 000012 002400      MOV    #12,REGNUM     ;SET LU REG NO. = 12
1735 032062 005037 002402          CLR    AXNUM          ;SET AX BYTE NO. = 0 FOR AX0
1736 032066 112737 000040 002366      MOVB  #LULP,WRIBYT
1737 032074 004737 003722      JSR    PC,WRITLU      ;SET LULP IN REG 12
1738 032100 012701 003220          MOV    #MSG1,R1      ;GET POINTER TO MSG DATA TABLE
1739 032104 012137 002422      3$:  MOV    (R1)+,TXWORD  ;GET CHAR TO BE LOADED
1740 032110 004737 005146      JSR    PC,LDTXSI     ;LOAD CHAR INTO TX SILO
1741 032114 020127 003242      CMP    R1,#MSG1+18.  ;SEE IF ALL MSG CHARS LOADED YET

```

```

1742 032120 103771          BLO      3$          ;BR IF NOT YET
1743 032122 004737 005120   JSR      PC,WAIT50   ;ALLOW DATA TO RIPPLE IN SILO
1744 032126 004737 005226   JSR      PC,STPLU    ;CLOCK LU FOR 24 CYCLES (UNTIL FIRST
1745 032132 000030          24.          ; DATA CHAR IS ABOUT TO BE RECEIVED)
1746 032134 004737 006666   JSR      PC,IACTIV   ;CHK IACT = 0
1747 032140 000000          0
1748 032142 004737 005226   JSR      PC,STPLU    ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
1749 032146 000006          6
1750 032150 012701 003224   MOV      #MSG1+4,R1
1751 032154 004737 006666   10$: JSR      PC,IACTIV   ;CHK IACT = 1
1752 032160 000001          1
1753 032162 004737 004076   JSR      PC,READAX   ;READ AX0
1754 032166 023721 002370   CMP      RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
1755 032172 001415          BEQ      12$        ;BR IF RCV'D DATA OK
1756 032174 016137 177776 002404  MOV      -2(R1),GOODAT ;GET EXPECTED DATA
1757 032202 013737 002370 002406  MOV      RAX15,BADDAT ;GET ACTUAL DATA
1758 032210 004737 004500   JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
1759          ;REPORT INCORRECT DATA CHAR RCV'D
1760 032214          ERRDF  26,EM26,ERR3
                                TRAP  C$ERDF
                                .WORD 26
                                .WORD EM26
                                .WORD  ERR3
1761 032224 000421          BR      24$
1762 032226 004737 005226   12$: JSR      PC,STPLU    ;CLOCK LU 8 CYCLES
1763 032232 000010          8.
1764 032234 020127 003236   CMP      R1,#MSG1+14. ;SEE IF ALL 5 DATA CHARS RCV'D YET
1765 032240 103745          BLO     10$        ;BR IF NOT YET
1766 032242 004737 006666   JSR      PC,IACTIV   ;CHK FOR IACT STILL = 1
1767 032246 000001          1
1768 032250 012737 000200 002366  MOV      #IC,WRIBYT
1769 032256 004737 003722   JSR      PC,WRITLU   ;SET IC (INPUT CLEAR) IN REG 12
1770 032262 004737 006666   JSR      PC,IACTIV   ;CHK IACT = 0
1771 032266 000000          0
1772 032270 004737 003576   24$: JSR      PC,MSTCLR  ;ISSUE CLEAN-UP MASTER CLEAR
1773 032274          ENDTST
                                L10064: TRAP  C$ETST
                                032274 104401
1774
1775
1776
1777
1778
1779
1780          ;*****
1781          ;SBTTL      TEST 33 - USYRT RECEIVER MSG TEST - BIT MODE, NO CRC
1782          ;*
1783          ;* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH
1784          ;* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH ERROR
1785          ;* DETECTION INHIBITED. TWO FLAGS, 000,125,252,377,000, AND TWO TERMINATING FLAGS
1786          ;* ARE THEN SENT. THE PROGRAM MONITORS IACT, RSOM, AND THE RCV'D CHARS ARE
1787          ;* READ FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR
1788          ;* IACT = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT STILL = 0.
1789          ;*****
1790 032276          BGNTST
1791 032304 004737 003576   MOV      #24$,RETADR
                                JSR      PC,MSTCLR  ;ISSUE MASTER CLEAR
                                T33::
    
```



```

1792 032310 004737 010612 JSR PC,SETUP ;PROGRAM THE USYRT
1793 032314 000000 000 ;
1794 032316 000302 CRC2!CRC1!IERR
1795 032320 000000 000 ;
1796 032322 000000 000 ;
1797 032324 012737 000012 002400 MOV #12,REGNUM ;SET LU REG NO. = 12
1798 032332 005037 002402 CLR AXNUM ;SET AX BYTE NO. = 0 FOR AX0
1799 032336 112737 000040 002366 MOVB #LULP,WRIBYT
1800 032344 004737 003722 JSR PC,WRITLU ;SET LULP IN REG 12
1801 032350 012701 003220 MOV #MSG1,R1 ;GET POINTER TO MSG DATA TABLE
1802 032354 012137 002422 3$: MOV (R1)+,TXWORD ;GET CHAR TO BE LOADED
1803 032360 004737 005146 JSR PC,LDTXSI ;LOAD CHAR INTO TX SILO
1804 032364 020127 003242 CMP R1,#MSG1+18. ;SEE IF ALL MSG CHARS LOADED YET
1805 032370 103771 BLO 3$ ;BR IF NOT YET
1806 032372 004737 005120 JSR PC,WAIT50 ;ALLOW DATA TO RIPPLE IN SILO
1807 032376 004737 005226 JSR PC,STPLU ;CLOCK LU FOR 33 CYCLES (UNTIL FIRST
1808 03240? 000041 33. ; DATA CHAR IS ABOUT TO BE RECEIVED)
1809 032404 004737 006666 JSR PC,IACTIV ;CHK IACT = 0
1810 032410 000000 0 ;
1811 032412 004737 007054 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
1812 032416 000000 0 ;
1813 032420 004737 005226 JSR PC,STPLU ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
1814 032424 000006 6 ;
1815 032426 012701 003224 MOV #MSG1+4,R1
1816 032432 020127 003224 5$: CMP R1,#MSG1+4 ;SEE IF 1ST CHAR RCV'D
1817 032436 001007 BNE 6$ ;BR IF NO
1818 032440 004737 006666 JSR PC,IACTIV ;CHK IACT = 1
1819 032444 000001 1 ;
1820 032446 004737 007054 JSR PC,RSEOM ;CHK RSOM = 1, REOM = 0
1821 032452 000001 1 ;
1822 032454 000420 BR 9$
1823 032456 020127 003234 6$: CMP R1,#MSG1+12. ;SEE IF LAST CHAR RCV'D
1824 032462 001007 BNE 8$ ;BR IF NO
1825 032464 004737 006666 JSR PC,IACTIV ;CHK FOR IACT = 0
1826 032470 000000 0 ;
1827 032472 004737 007054 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
1828 032476 000000 0 ;
1829 032500 000406 BR 9$
1830 032502 004737 006666 8$: JSR PC,IACTIV ;CHK FOR IACT = 1
1831 032506 000001 1 ;
1832 032510 004737 007054 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
1833 032514 000000 0 ;
1834 032516 004737 004076 9$: JSR PC,READAX ;READ AX0
1835 032522 023721 002370 CMP RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
1836 032526 001415 BEQ 12$ ;BR IF RCV'D DATA OK
1837 032530 016137 177776 002404 MOV -2(R1),GOODAT ;GET EXPECTED DATA
1838 032536 013737 002370 002406 MOV RAX15,BADDAT ;GET ACTUAL DATA
1839 032544 004737 004500 JSR PC,GETALL ;GET REGS FOR PRINTOUT
1840 ;REPORT INCORRECT DATA CHAR RCV'D
1841 032550 ERRDF 26,EM26,ERR3
    032550 104455 TRAP C$ERDF
    032552 000032 .WORD 26
    032554 013043 .WORD EM26
    032556 015114 .WORD ERR3
1842 032560 000424 BR 24$
1843 032562 004737 005226 12$: JSR PC,STPLU ;CLOCK LU 8 CYCLES
1844 032566 000010 8.
    
```

```

1845 032570 020127 003236      CMP      R1,#MSG1+14.      ;SEE IF ALL DATA CHARS CHECKED YET
1846 032574 103716      BLO      5$                ;BR IF NOT YET
1847 032576 004737 006666      JSR      PC,IACTIV        ;CHK IACT = 0
1848 032602 000000      0
1849 032604 004737 007054      JSR      PC,RSEOM        ;CHK RSOM = 0, REOM = 0
1850 032610 000000      0
1851 032612 012737 000200 002366  MOV      #IC,WRIBYT
1852 032620 004737 003722      JSR      PC,WRITLU       ;SET IC (INPUT CLEAR) IN REG 12
1853 032624 004737 006666      JSR      PC,IACTIV        ;CHK IACT = 0
1854 032630 000000      0
1855 032632 004737 003576 24$:    JSR      PC,MSTCLR       ;ISSUE CLEAN-UP MASTER CLEAR
1856 032636      ENDTST
                                L10065:
                                TRAP      C$ETST
    
```

1857
1858
1859
1860
1861

```

1862      ;:*****
1863      .SBTTL      TEST 34 - SILO-DISABLED TRANSMITTER LOAD TEST
1864      ;*
1865      ;* THIS TEST DISABLES THE SILOS, LOADS A 125 CHARACTER INTO THE TX SILO, AND
1866      ;* READS AX1-15 AND CHECKS THAT THE DATA DID NOT GET LOADED INTO THE USYRT TX
1867      ;* BUFFER.
1868      ;:*****
1869      BGNTST
    
```

```

1870 032640 004737 003576      JSR      PC,MSTCLR       ;ISSUE MASTER CLEAR
1871 032644 012737 000014 002400  MOV      #14,REGNUM      ;SET REG NO. = 14
1872 032652 012737 000040 002366  MOV      #DISSI,WRIBYT   ;SET DISSI BIT
1873 032660 004737 003722      JSR      PC,WRITLU
1874 032664 012737 000040 002426  MOV      #DISSI,DISILO   ;SET DISABLE SILO FLAG
1875 032672 012737 000125 002422  MOV      #125,TXWORD     ;LOAD 125 INTO TX SILO
1876 032700 004737 005146      JSR      PC,LDTXSI
1877 032704 012737 000002 002402  MOV      #2,AXNUM        ;SET REG NO. FOR AX1
1878 032712 004737 004076      JSR      PC,READAX       ;READ AX1-15, AX1-16
1879 032716 123727 002370 000000  CMPB     RAX15,#000      ;CHECK FOR AX1-15 UNCHANGED
1880 032724 001414      BEQ      3$              ;BR IF UNCHANGED
1881 032726 012737 000000 002404  MOV      #000,GOODAT     ;GET EXPECTED DATA
1882 032734 013737 002370 002406  MOV      RAX15,BADDAT    ;GET ACTUAL DATA
1883 032742 004737 004500      JSR      PC,GETALL       ;GET REGS FOR PRINTOUT
1884      ;REPORT REG MISCOMPARE
1885 032746      ERRDF 3,EM3,ERR3
                                TRAP      C$ERDF
                                .WORD    3
                                .WORD    EM3
                                .WORD    ERR3
    
```

```

1886 032756 005037 002426 3$:     CLR      DISILO         ;CLEAR DISABLE SILO FLAG
1887 032762 004737 003576      JSR      PC,MSTCLR       ;ISSUE MASTER CLEAR TO CLEAN UP
1888 032766      ENDTST
                                L10066:
                                TRAP      C$ETST
1889
1890
1891
1892
    
```

```

1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905 032770
      032770
1906 032770 012737 033670 002362
1907 032776 004737 005512
1908 033002 000000
1909 033004 000000
1910 033006 004737 003576
1911 033012 004737 010612
1912 033016 000000
1913 033020 000302
1914 033022 000000
1915 033024 000000
1916 033026 012737 000014 002400
1917 033034 012737 000140 002366
1918 033042 004737 003722
1919 033046 012737 000140 002426
1920 033054 012737 000012 002400
1921 033062 112737 000040 002366
1922 033070 004737 003722
1923 033074 012701 003220
1924 033100 004737 004634
1925 033104 000001
1926 033106 012737 000002 002402
1927 033114 112137 002374
1928 033120 112137 002376
1929 033124 004737 004264
1930 033130 004737 004634
1931 033134 000000
1932 033136 004737 005324
1933 033142 000000
1934 033144 004737 005226
1935 033150 000003
1936 033152 004737 005324
1937 033156 000001
1938 033160 012703 000004
1939 033164 112137 002374 4$:
1940 033170 112137 002376
1941 033174 020327 000001
1942 033200 001006
1943 033202 005737 002430
1944 033206 001403
1945 033210 112737 000002 002376
1946 033216 004737 004634 5$:
1947 033222 000001
1948 033224 004737 004264

```

```

:*****
:SBTTL      TEST 35 - SILO-DISABLED MESSAGE TEST - BIT MODE, NO CRC
:*
:* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU)
:* WITH LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, WITH SILO
:* DISABLE SET, AND WITH NO ERROR DETECTION. TWO FLAGS, 000,125,252, AND
:* TERMINATING FLAGS ARE THEN SENT BY LOADING THE TRANSMITTED CHARS INTO
:* REG AX1. THE PROGRAM MONITORS OACT, IACT, RSOM, REOM, ORDY, OCOR, ICIR,
:* IRDY, AND THE RECEIVED CHARS ARE READ FROM AX0 AND COMPARED TO EXPECTED
:* VALUES.
:*****
BGNTST

```

```

T35::
MOV #18$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,INITRN ;FIND OUT WHICH USYRT CHIP
0
0
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM THE USYRT
000
CRC2!CRC1!IERR
000
000
MOV #14,REGNUM ;SET REG NO. = 14
MOV #TXEN!DISSI,WRIBYT
JSR PC,WRITLU ;SET TXEN AND DISSI IN REG 14
MOV #TXEN!DISSI,DISILO ;SET DISABLE SILO FLAG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOVB #LULP,WRIBYT
JSR PC,WRITLU ;SET LULP IN REG 12
MOV #MSG1,R1 ;GET POINTER TO MSG
JSR PC,OSIRDY ;CHK ORDY = 1
1
MOV #2,AXNUM ;SET AX BYTE NO. FOR AX1
MOVB (R1)+,WAX15 ;GET A CHAR
MOVB (R1)+,WAX16
JSR PC,WRITAX ;LOAD CHAR INTO USYRT TX BUFFER
JSR PC,OSIRDY ;CHK ORDY = 0
0
JSR PC,OACTIV ;CHK OACT = 0
0
JSR PC,STPLU ;CLOCK LU FOR 3 CYCLES
3
JSR PC,OACTIV ;CHK OACT = 1
1
MOV #4,R3 ;INIT COUNTER
MOVB (R1)+,WAX15 ;GET ANOTHER CHAR
MOVB (R1)+,WAX16
CMP R3,#1 ;SEE IF LOADING LAST DATA CHAR YET
BNE 5$ ;BR IF NOT
TST CHPTYP ;SEE IF SIG USYRT
BEQ 5$ ;BR IF YES
MOVB #TEOM,WAX16 ;SET TEOM WITH LAST DATA CHAR
JSR PC,OSIRDY ;CHK ORDY = 1
1
JSR PC,WRITAX ;LOAD ANOTHER CHAR INTO USYRT TX BUFFER

```

1949	033230	004737	004634		JSR	PC,OSIRDY		;CHK ORDY = 0	
1950	033234	000000			0				
1951	033236	004737	006666		JSR	PC,IACTIV		;CHK IACT = 0	
1952	033242	000000			0				
1953	033244	004737	007054		JSR	PC,RSEOM		;CHK RSOM = 0, REOM = 0	
1954	033250	000000			0				
1955	033252	004737	005226		JSR	PC,STPLU		;CLOCK LU FOR 8 CYCLES	
1956	033256	000010			8.				
1957	033260	004737	005324		JSR	PC,DACTIV		;CHK OACT = 1	
1958	033264	000001			1				
1959	033266	004737	004634		JSR	PC,OSIRDY		;CHK ORDY = 1	
1960	033272	000001			1				
1961	033274	005303			DEC	R3		;DECR COUNTER	
1962	033276	001332			BNE	4\$;BR IF NOT DONE YET	
1963	033300	004737	006402		JSR	PC,ISIRDY		;CHK IRDY = 0	
1964	033304	000000			0				
1965	033306	005737	002430		TST	CHPTYP		;SEE IF SIG USYRT	
1966	033312	001007			BNE	11\$;BR IF NOT	
1967	033314	105037	002374		CLRB	WAX15			
1968	033320	112737	000002	002376	MOVB	#TEOM,WAX16		;LOAD EOM CHAR	
1969	033326	004737	004264		JSR	PC,WRITAX		;LOAD ANOTHER CHAR INTO USYRT TX BUFFER	
1970	033332	004737	005226		11\$: JSR	PC,STPLU		;CLOCK LU FOR 3 CYCLES	
1971	033336	000003			3				
1972	033340	004737	006402		JSR	PC,ISIRDY		;CHK IRDY = 1	
1973	033344	000002			2				
1974	033346	004737	005324		JSR	PC,DACTIV		;CHK OACT = 1	
1975	033352	000001			1				
1976	033354	004737	006666		JSR	PC,IACTIV		;CHK IACT = 1	
1977	033360	000001			1				
1978	033362	004737	007054		JSR	PC,RSEOM		;CHK RSOM = 1, REOM = 0	
1979	033366	000001			1				
1980	033370	004737	006402		JSR	PC,ISIRDY		;CHK IRDY = 0	
1981	033374	000000			0				
1982	033376	012737	000000	002402	MOV	#0,AXNUM		;SET AX BYTE NO. FOR AX0	
1983	033404	123727	002370	000000	CMPB	RAX15,#000		;COMPARE RCV'D CHAR TO 000	
1984	033412	001415			BEQ	9\$;BR IF MATCH	
1985	033414	012737	000000	002404	MOV	#0,GOODAT		;SET EXPECTED DATA	
1986	033422	012737	002370	002406	6\$: MOV	#RAX15,BADAT		;SET ACTUAL DATA	
1987	033430	004737	004500		JSR	PC,GETALL		;GET REGS FOR PRINTOUT	
1988									
1989	033434								
	033434	104455							
	033436	000032							TRAP C\$ERDF
	033440	013043							.WORD 26
	033442	015114							.WORD EM26
									.WORD ERR3
1990	033444	000511			BR	18\$			
1991	033446	004737	005226		9\$: JSR	PC,STPLU		;CLOCK LU FOR 8 CYCLES	
1992	033452	000010			8.				
1993	033454	004737	006402		JSR	PC,ISIRDY		;CHK IRDY = 1	
1994	033460	000002			2				
1995	033462	004737	005324		JSR	PC,DACTIV		;CHK OACT = 1	
1996	033466	000001			1				
1997	033470	004737	006666		JSR	PC,IACTIV		;CHK IACT = 1	
1998	033474	000001			1				
1999	033476	004737	007054		JSR	PC,RSEOM		;CHK RSOM = 1, REOM = 0	
2000	033502	000000			0				
2001	033504	004737	006402		JSR	PC,ISIRDY		;CHK IRDY = 0	

```
2002 033510 000000 0
2003 033512 123727 002370 000125 CMPB RAX15,#125 ;COMPARE 2ND RCV'D CHAR TO 125
2004 033520 001404 BEQ 12$ ;BR IF MATCH
2005 033522 012737 000125 002404 MOV #125,GOODAT ;SET EXPECTED DATA
2006 033530 000734 BR 6$ ;BR TO REPORT ERROR
2007 033532 004737 005226 12$: JSR PC,STPLU ;CLOCK LU FOR 8 CYCLES
2008 033536 000010 8.
2009 033540 004737 006402 JSR PC,ISIRDY ;CHK IRDY = 1
2010 033544 000002 2
2011 033546 004737 005324 JSR PC,DACTIV ;CHK OACT = 1
2012 033552 000001 1
2013 033554 004737 006666 JSR PC,IACTIV ;CHK IACT = 0
2014 033560 000000 0
2015 033562 004737 007054 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 1
2016 033566 000002 2
2017 033570 004737 006402 JSR PC,ISIRDY ;CHK IRDY = 0
2018 033574 000000 0
2019 033576 123727 002370 000252 CMPB RAX15,#252 ;COMPARE 3RD RCV'D CHAR TO 252
2020 033604 001404 BEQ 14$ ;BR IF MATCH
2021 033606 012737 000252 002404 MOV #252,GOODAT ;SET EXPECTED DATA
2022 033614 000702 BR 6$ ;BR TO REPORT ERROR
2023 033616 012737 000014 002400 14$: MOV #14,REGNUM ;SET REG NO. = 14
2024 033624 012737 000040 002366 MOV #DISSI,WRIBYT
2025 033632 004737 003722 JSR PC,WRITLU ;CLEAR TX ENABLE
2026 033636 012737 000011 002400 MOV #11,REGNUM ;SET REG NO. = 11
2027 033644 012737 000200 002366 MOV #OC,WRIBYT
2028 033652 004737 003722 JSR PC,WRITLU ;SET OC TO SHUT DOWN TRANSMITTER
2029 033656 004737 005120 JSR PC,WAIT50 ;WAIT FOR SHUTDOWN
2030 033662 004737 005324 JSR PC,DACTIV ;CHK FOR OACT = 0
2031 033666 000000 0
2032 033670 005037 002426 18$: CLR DISILO ;CLEAR DISABLE SILO FLAG
2033 033674 004737 003576 JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
2034 033700
033700 ENDTST
033700 104401 L10067: TRAP C$ETST
```

2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056

```
*****
SBTTL TEST 36 - RECEIVER BUFFER TEST - CHAR MODE, CRC
*
* FIRST, A MASTER CLEAR IS DONE AND THE PROGRAM CHECKS FOR ICIR = 1 AND IRDY
* = 0. THEN, 2 SOM CHARS ARE LOADED AND CLOCKED INTO THE USYRT, AND 64
* BYTES OF A 256-BYTE BINARY COUNT DATA PATTERN (000-377) ARE LOADED INTO
* THE TX SILO.
* THE LINE UNIT IS THEN CLOCKED UNTIL IRDY = 1, AND THE PROGRAM CHECKS FOR
* THIS TO OCCUR WITHIN 40-43 CYCLES. THE PROGRAM READS THE RCV SILO, CHECKS THE
* CHAR FOR 000, AND CHECKS FOR IRDY = 0 AGAIN.
* THE LINE UNIT IS THEN CLOCKED IN GROUPS OF 8 CYCLES, AND AFTER EACH, THE
* PROGRAM CHECKS FOR ICIR = 1, IRDY = 1, UNTIL THE 64TH GROUP, AFTER WHICH
* IT CHECKS FOR ICIR = 0, IRDY = 1. THE SECOND DATA CHAR IS READ FROM THE
* RECEIVER SILO AND COMPARED TO 001. THEN, THE PROGRAM CHECKS FOR ICIR = 1,
* IRDY = 1 AGAIN.
* THE REST OF THE BINARY COUNT DATA BYTES ARE CYCLED 8 CLOCKS AND READ AND
* COMPARED A BYTE AT A TIME.
```

```

2057 .....
2058 033702 BGNTST
      033702
2059 033702 012737 034156 002362          MOV    #A10,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
2060 -----
2061      DO MASTER CLR, CHK FOR ICIR = 1, IRDY = 0
2062 -----
2063 033710 004737 003576          JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
2064 033714 004737 006402          JSR    PC,ISIRDY     ;CHK ICIR = 1, IRDY = 0
2065 033720 000001
2066 -----
2067      LOAD AND CLOCK 2 SOM'S, LOAD 64 BYTES OF BINARY COUNT PATTERN INTO TX SILO,
2068      CLOCK LINE UNIT, CHK FOR IRDY = 1 WITHIN 40-43 CYCLES
2069 -----
2070 033722 004737 005512          JSR    PC,INITRN     ;LOAD 2 SOM'S, CLOCK THEM INTO USYRT
2071 033726 000226          SYNCH
2072 033730 000011          STRIP!DDCMP
2073 033732 005003          CLR    R3            ;INIT BINARY COUNT DATA FOR WRITING
2074 033734 010337 002422 2$:    MOV    R3,TXWORD
2075 033740 004737 005146          JSR    PC,LDTXSI     ;LOAD A DATA BYTE INTO TX SILO
2076 033744 005203          INC    R3            ;INCR DATA
2077 033746 020327 000100          CMP    R3,#64.      ;SEE IF 64 BYTES LOADED YET
2078 033752 002770          BLT   2$            ;BR IF NOT YET
2079 033754 004737 007406          JSR    PC,RCV1ST     ;RECEIVE AND TIME FIRST CHARACTER
2080 033760 000050          40.
2081 -----
2082      READ RCV SILO, COMPARE FIRST CHAR TO 000, CHK FOR ICIR = 1, IRDY = 0
2083 -----
2084 033762 005004 9$:    CLR    R4            ;INIT PATTERN FOR READING
2085 033764 004737 007722          JSR    PC,CKDATA     ;READ RCV SILO, COMPARE DATA
2086 033770 000000          0            ;EXPECTED DATA = 000
2087 033772 000000          0            ;DON'T CLOCK LINE UNIT
2088 033774 005204 16$:    INC    R4            ;INCR DATA FOR READING
2089 033776 004737 006402          JSR    PC,ISIRDY     ;CHK FOR ICIR = 1, IRDY = 0
2090 034002 000001          1
2091 -----
2092      CLOCK 63 CHARS INTO RCV SILO, CHK ICIR = 1, IRDY = 1
2093 -----
2094 034004 004737 005226 18$:    JSR    PC,STPLU      ;CLOCK LU FOR 8 CYCLES
2095 034010 000010          8.
2096 034012 010337 002422          MOV    R3,TXWORD
2097 034016 004737 005146          JSR    PC,LDTXSI     ;LOAD ANOTHER WORD INTO TX SILO
2098 034022 005203          INC    R3            ;INCR PATTERN FOR WRITING
2099 034024 004737 006402          JSR    PC,ISIRDY     ;CHK ICIR = 1, IRDY = 1
2100 034030 000003          3
2101 034032 020327 000177          CMP    R3,#127.     ;SEE IF 63 MORE CHARS CLOCKED YET
2102 034036 002762          BLT   18$          ;BR IF NOT YET
2103 -----
2104      CLOCK 1 MORE CHAR INTO RCV SILO, CHK ICIR = 0, IRDY = 1
2105 -----
2106 034040 004737 005226          JSR    PC,STPLU      ;CLOCK LU FOR 8 CYCLES
2107 034044 000010          8.
2108 034046 004737 006402          JSR    PC,ISIRDY     ;CHK ICIR = 0, IRDY = 1
2109 034052 000002          2
2110 -----
2111      READ, COMPARE, CLOCK REST OF DATA CHARS
2112 -----

```

```

2113 034054 010437 034064      20$:  MOV    R4,21$      ;SET EXPECTED DATA
2114 034060 004737 007722      JSR    PC,CKDATA    ;READ AND COMPARE DATA
2115 034064 000000                21$:   0              ;EXPECTED SILO ENTRY GOES HERE
2116 034066 000000                0      ;DON'T CLOCK LINE UNIT
2117 034070 005204                22$:   INC    R4        ;INCR DATA PATTERN FOR READS
2118 034072 020427 000400      CMP    R4,#400      ;SEE IF ALL DONE READING YET
2119 034076 001427                BEQ    32$          ;BR IF DONE READING
2120 034100 004737 006402      JSR    PC,ISIRDY    ;CHK ICIR = 1, IRDY = 1
2121 034104 000003                3      ;
2122 034106 004737 005226      JSR    PC,STPLU     ;CLOCK LU FOR 8 CYCLES
2123 034112 000010                8.     ;
2124 034114 020327 000377      CMP    R3,#377      ;SEE IF ALL CHARS LOADED INTO TX SILO YET
2125 034120 003007                BGT    24$          ;BR IF YES
2126 034122 010337 002422      MOV    R3, TXWORD   ;
2127 034126 004737 005146      JSR    PC,LDTXSI    ;LOAD ANOTHER CHAR INTO TX SILO
2128 034132 005203                INC    R3            ;INCR DATA PATTERN FOR WRITING
2129 034134 000137 034054      JMP    20$          ;
2130 034140 012737 001000 002422 24$:  MOV    #TXEOM, TXWORD ;LOAD EQM INTO TX SILO
2131 034146 004737 005146      JSR    PC,LDTXSI    ;
2132 034152 000137 034054      JMP    20$          ;
2133 034156                32$:   ;
2134 034156 004737 003576      A10:  JSR    PC,MSTCLR  ;ISSUE MASTER CLEAR TO CLEAN UP
2135 034162                ENDTST ;
        034162                L10070:
        034162 104401                TRAP   C$ETST
    
```

2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152

```

*****
.SBTTL      TEST 37 - RECEIVER CHAR LENGTH TIMING TEST - CHAR MODE, NO CRC
*
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH NO ERROR DETECTION, AND A MSG IS
* INITIATED WITH 2 SYNCH CHARS. NEXT, FIFTEEN 000 CHARS ARE LOADED INTO THE
* TRANSMITTER SILO. THE LINE UNIT IS THEN CLOCKED USING STEPLU WITH LULOOP
* SET, WHILE THE RECEIVER CHAR LENGTH IS SET TO THE FOLLOWING VALUES : 5,6,7,8.
* FOR EACH RCV CHAR LENGTH, THE PROGRAM CHECKS TO MAKE SURE THET USYRT RECEIVER
* FLAGS OCCUR THE PROPER NO. OF CYCLES APART, FOR EACH RCV CHAR LENGTH.
* (FOR EXAMPLE A 5-BIT CHAR TAKES 5 CLOCK CYCLES TO BE RECEIVED). A MASTER
* CLEAR IS THEN DONE TO TERMINATE THE OPERATION.
*****
    
```

```

2153 034164                BGNTST ;
        034164                T37::
2154 034164 012737 034356 002362      MOV    #24$,RETADR  ;SET TEST EXIT ADRS FOR ERRORS
2155 034172 004737 005512      JSR    PC,INITRN    ;DO MASTER CLR, LOAD 2 SOM'S
2156 034176 000000                000    ;
2157 034200 000341                CRC2!CRC1!IDLE!DDCMP ;
2158 034202 012701 000017      MOV    #15.,R1      ;INIT COUNTER
2159 034206 005037 002422      CLR    TXWORD       ;
2160 034212 004737 005146      3$:   JSR    PC,LDTXSI    ;LOAD A 000 CHAR INTO TX SILO
2161 034216 005301                DEC    R1            ;DECR COUNTER
2162 034220 001374                BNE    3$           ;BR IF NOT DONE LOADING YET
2163 034222 004737 005120      JSR    PC,WAIT50    ;WAIT FOR SILO TO RIPPLE
2164 034226 012737 000006 002402      MOV    #6,AXNUM     ;SET BYTE NO. = 6 FOR AX3
2165 034234 012737 000000 002374      MOV    #000,WAX15   ;SET DATA FOR AX3-15 = 0
2166 034242 012737 000005 002376      MOV    #RXLEN2!RXLENO,WAX16 ;SET RCV LEN = 5
    
```

```

2167 034250 004737 004264 JSR PC,WRITAX ;LOAD AX3
2168 034254 004737 005226 JSR PC,STPLU ;CLK LU UNTIL TX'ING 1ST DATA CHAR
2169 034260 100012 CHPCHK!10.
2170 034262 012737 000006 002376 MOV #RXLEN2!RXLEN1,WAX16 ;SET RCV LEN = 6
2171 034270 004737 004264 JSR PC,WRITAX ;LOAD AX3
2172 034274 004737 007406 JSR PC,RCV1ST ;CLOCK 5-BIT DATA CHAR
2173 034300 000005 5
2174 034302 012737 000007 002376 MOV #RXLEN2!RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 7
2175 034310 004737 004264 JSR PC,WRITAX ;LOAD AX3
2176 034314 004737 011046 JSR PC,RXCHAR ;RCV 5-BIT DATA CHAR, CLK 6-BIT
2177 034320 000006 6
2178 034322 012737 000000 002376 MOV #0,WAX16 ;SET RCV LEN = 8
2179 034330 004737 004264 JSR PC,WRITAX ;LOAD AX3
2180 034334 004737 011046 JSR PC,RXCHAR ;RCV 6-BIT DATA CHAR, CLK 7-BIT
2181 034340 000007 7
2182 034342 004737 011046 JSR PC,RXCHAR ;RCV 7-BIT DATA CHAR, CLK 8-BIT
2183 034346 000010 8.
2184 034350 004737 011046 JSR PC,RXCHAR ;RCV 8-BIT DATA CHAR
2185 034354 000010 8.
2186 034356 004737 003576 24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
2187 034362 034362 104401 ENDTST L10071: TRAP C$ETST
    
```

2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204

```

:*****
:SBTTL TEST 38 - RECEIVER CHAR LENGTH TIMING TEST - BIT MODE, NO CRC
:*
:* THE LINE UNIT IS PLACED IN BIT MODE WITH NO ERROR DETECTION, AND A MESSAGE IS
:* INITIATED WITH 2 FLAG CHARS. NEXT, FIFTEEN 000 CHARS ARE LOADED INTO THE
:* TRANSMITTER SILO. THE LINE UNIT IS THEN CLOCKED USING STEPLU WITH LULOOP
:* SET, WHILE THE RCV CHAR LENGTH IS SET TO THE FOLLOWING VALUES : 8,8,8,7,6,5,
:* 4,3,2,1. FOR EACH RCV CHAR LENGTH, THE PROGRAM CHECKS TO MAKE SURE THAT THE
:* USYRT RECEIVER FLAGS OCCUR THE PROPER NO. OF CYCLES APART, FOR EACH RCV
:* CHAR LENGTH. (FOR EXAMPLE, A 5 BIT CHAR TAKES 5 CLOCK CYCLES TO BE RECEIVED).
:* A MASTER CLEAR IS THEN DONE TO TERMINATE THE OPERATION.
:*****
    
```

```

2205 034364 034364 BGNTST
2206 034364 012737 034656 002362 MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS T38::
2207 034372 004737 005512 JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'S
2208 034376 000000 000
2209 034400 000300 CRC2!CRC1
2210 034402 012701 000017 MOV #15.,R1 ;INIT COUNTER
2211 034406 005037 002422 CLR TXWORD
2212 034412 004737 005146 3$: JSR PC,LDTXSI ;LOAD A 000 CHAR INTO TX SILO
2213 034416 005301 DEC R1 ;DECR COUNTER
2214 034420 001374 BNE 3$ ;BR IF NOT DONE LOADING YET
2215 034422 004737 005120 JSR PC,WAIT50 ;WAIT FOR SILO TO RIPPLE
2216 034426 012737 000006 002402 MOV #6,AXNUM ;SET BYTE NO. = 6 FOR AX3
2217 034434 012737 000000 002374 MOV #000,WAX15 ;SET DATA FOR AX3-15 = 0
2218 034442 004737 007406 JSR PC,RCV1ST ;CLOCK FIRST 8-BIT DATA CHAR
2219 034446 000040 32.
2220 034450 012737 000000 002376 MOV #0,WAX16 ;SET RCV LEN = 8
    
```



```

2221 034456 004737 004264 JSR PC,WRITAX ;LOAD AX3
2222 034462 004737 011046 JSR PC,RXCHAR ;RCV FIRST 8-BIT DATA CHAR, CLK SECOND 8-BIT
2223 034466 000010 8.
2224 034470 012737 000007 002376 MOV #RXLEN2!RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 7
2225 034476 004737 004264 JSR PC,WRITAX ;LOAD AX3
2226 034502 004737 011046 JSR PC,RXCHAR ;RCV SECOND 8-BIT DATA CHAR, CLK 3RD 8-BIT
2227 034506 000010 8.
2228 034510 012737 000006 002376 MOV #RXLEN2!RXLEN1,WAX16 ;SET RCV LEN = 6
2229 034516 004737 004264 JSR PC,WRITAX ;LOAD AX3
2230 034522 004737 011046 JSR PC,RXCHAR ;RCV 3RD 8-BIT DATA CHAR, CLK 7-BIT
2231 034526 000007 7
2232 034530 012737 000005 002376 MOV #RXLEN2!RXLENO,WAX16 ;SET RCV LEN = 5
2233 034536 004737 004264 JSR PC,WRITAX ;LOAD AX3
2234 034542 004737 011046 JSR PC,RXCHAR ;RCV 7-BIT DATA CHAR, CLK 6-BIT
2235 034546 000006 6
2236 034550 012737 000004 002376 MOV #RXLEN2,WAX16 ;SET RCV LEN = 4
2237 034556 004737 004264 JSR PC,WRITAX ;LOAD AX3
2238 034562 004737 011046 JSR PC,RXCHAR ;RCV 6-BIT DATA CHAR, CLK 5-BIT
2239 034566 000005 5
2240 034570 012737 000003 002376 MOV #RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 3
2241 034576 004737 004264 JSR PC,WRITAX ;LOAD AX3
2242 034602 004737 011046 JSR PC,RXCHAR ;RCV 5-BIT DATA CHAR, CLR 4-BIT
2243 034606 000004 4
2244 034610 012737 000002 002376 MOV #RXLEN1,WAX16 ;SET RCV LEN = 2
2245 034616 004737 004264 JSR PC,WRITAX ;LOAD AX3
2246 034622 004737 011046 JSR PC,RXCHAR ;RCV 4-BIT DATA CHAR, CLK 3-BIT
2247 034626 000003 3
2248 034630 012737 000001 002376 MOV #RXLENO,WAX16 ;SET RCV LEN = 1
2249 034636 004737 004264 JSR PC,WRITAX ;LOAD AX3
2250 034642 004737 011046 JSR PC,RXCHAR ;RCV 3-BIT DATA CHAR, CLK 2-BIT
2251 034646 000002 2
2252 034650 004737 011046 JSR PC,RXCHAR ;RCV 2-BIT DATA CHAR, CLK 1-BIT
2253 034654 000001 1
2254 034656 004737 003576 24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
2255 034662 034662 104401 ENDTST L10072: TRAP C$ETST
    034662
    
```

2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269

```

:*****
:SBTTL TEST 39 - TRANSMITTER UNDERRUN ERROR, IDLE MARKING, CHAR MODE,NO CRC
:*
:* THE LINE UNIT IS PLACED IN CHAR MODE, AND THE IDLE BIT IS SET. THEN, A
:* MSG IS INITIATED, A 000 CHAR IS SENT, AND THE TX BUFFER IS NOT SERVICED
:* IN RESPONSE TO THE USYRT TX BUFFER EMPTY FLAG, WHICH CAUSES A TX UNDERRUN
:* ERROR. THEN, THE RECEIVER IS CLOCKED AND CHECKED FOR TWO 377 CHARS TO BE
:* RECEIVED (LINE MARKING) BEFORE SHUTTING DOWN WITH A MASTER CLEAR.
:*****
BGNTST
    
```

```

2270 034664 034664
2271 034664 012737 034746 002362 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
2272 034672 004737 005512 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
2273 034676 000226 SYNCH
2274 034700 000341 CRC2!CRC1!IDLE!DDCMP
    
```

```

2275 034702 012737 000000 002422      MOV      #000, TXWORD
2276 034710 004737 005146                JSR      PC, LDTXSI      ;LOAD 000 CHAR INTO TX SILO
2277 034714 004737 005226                JSR      PC, STPLU       ;CLOCK LINE UNIT UNTIL LINE GOES MARKING
2278 034720 000063 51.
2279 034722 004737 007326                JSR      PC, RDRXSI      ;READ 000 CHAR
2280 034726 004737 007722                JSR      PC, CKDATA      ;READ AND CHECK FOR MARK CHAR (377)
2281 034732 000377 377
2282 034734 000000 000
2283 034736 004737 007722                JSR      PC, CKDATA      ;READ AND CHECK FOR ANOTHER MARK CHAR
2284 034742 000377 377
2285 034744 000000 000
2286 034746 004737 003576      24$:    JSR      PC, MSTCLR      ;ISSUE MASTER CLEAR TO CLEAN UP
2287 034752 000000 000
      034752 104401      L10073:
      034752 104401      TRAP      C$ETST
    
```

2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307

```

:*****
:SBTTL      TEST 40 - MSG TERMINATION WITH GA CHARS - BIT MODE, NO CRC
:*
:* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
:* INITIATED IN BIT MODE.
:* 2 FLAG CHARACTERS ARE SENT, FOLLOWED BY
:* THE FOLLOWING DATA CHARACTERS : 000, 125, 252, 377, 000. THEN THE LOOP
:* MODE BIT (STRIP) IS SET AND 2 TERMINATING GO-AHEAD CHARACTERS ARE
:* SENT. EACH USYRT RCV FLAG IS TIMED AS IT IS RECEIVED, AND THE 5 DATA
:* WORDS ARE READ AND COMPARED TO EXPECTED VALUES.
:* ALSO, THE FIRST GA CHAR IS CHECKED BY SCANNING THE TXDATA BIT AS THE GA
:* IS BEING TRANSMITTED (GA = 376 OCTAL).
:* THE TEST ALSO CHECKS FOR SETTING OF RAB AND EBLK
:* IN LOOP MODE.
:*****
BGNTST
    
```

```

2308 034754 000000 000
      034754 000000 000
2309 034754 012737 035110 002362      MOV      #24$, RETADR      ;SET TEST EXIT ADRS FOR ERRORS
2310 034762 004737 005512                JSR      PC, INITRN      ;MST CLR, LOAD 2 SOM'S
2311 034766 000000 000
2312 034770 000310 000
2313 034772 004737 010770                JSR      PC, LODMSG      ;LOAD DATA CHARS INTO TX SILO
2314 034776 003224 5
2315 035000 000005 5
2316 035002 012737 005000 002422      MOV      #TXGOA!TXEOM, TXWORD
2317 035010 004737 005146                JSR      PC, LDTXSI      ;LOAD A GA CHAR INTO TX SILO
2318 035014 004737 005146                JSR      PC, LDTXSI      ;LOAD ANOTHER GA
2319 035020 004737 005120                JSR      PC, WAIT50      ;ALLOW SILO TO RIPPLE
2320 035024 004737 005226                JSR      PC, STPLU       ;CLOCK LU UNTIL GA CHAR IS TX'ING
2321 035030 100071 57.
2322 035032 004737 011150                JSR      PC, CKTBIT      ;SCAN TXDATA BIT FOR GA CHAR
2323 035036 000376 376
2324 035040 004737 007722                JSR      PC, CKDATA      ;RCV 000 CHAR, CLK 125
2325 035044 000000 000
2326 035046 000000 0
2327 035050 004737 007722                JSR      PC, CKDATA      ;RCV 125 CHAR, CLK 252
2328 035054 000125 125
    
```

T40::

```

2329 035056 000000          0
2330 035060 004737 007722    JSR    PC,CKDATA      ;RCV 252 CHAR, CLK 377
2331 035064 000252          252
2332 035066 000000          0
2333 035070 004737 007722    JSR    PC,CKDATA      ;RCV 377 CHAR
2334 035074 000377          377
2335 035076 000010          8.
2336 035100 004737 007722    JSR    PC,CKDATA      ;RCV 000 CHAR, CHK RAB = 1, EBLK = 1
2337 035104 003000          3000
2338 035106 000010          8.
2339 035110 004737 003576    24$: JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
2340 035114          ENDTST
      035114          L10074: TRAP    C$ETST
      035114 104401
    
```

2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358

```

:*****
:SBTTL      TEST 41 - IDLE SYNCHS TEST - CHAR MODE
:*
:* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
:* INITIATED IN CHAR MODE. 24(DEC) SYNCHS ARE SENT.
:* EACH SYNCH IS TIMED AS IT IS RECEIVED, AND THE BITS ARE CHECKED
:* FOR A VALID SYNCH CHAR FOR EACH OF THE 22 SYNCHS WHICH FOLLOW
:* THE FIRST TWO (THESE PERFORM SYNCHRONIZATION, AND ARE NOT READ).
:* WHILE THE LAST SYNCH IS BEING TRANSMITTED, OC IS SET, AND THE
:* NEXT CHAR RCV'D AFTER THE SYNCH IS CHECKED TO BE 377 (LINE MARKING).
:* THEN, A MASTER CLEAR IS ISSUED.
:* THE SYNCH CHAR USED IS 226 (OCTAL).
:*****
BGNTST
    
```

```

2359 035116          T41::
      035116
2360 035116 012737 035256 002362    MOV    #24$,RETADR      ;SET TEST EXIT ADRS FOR ERRORS
2361 035124 004737 005512          JSR    PC,INITRN        ;MST CLR, LOAD 2 SOM'S
2362 035130 000226          SYNCH
2363 035132 000341          CRC2!CRC1!IDLE!DDCMP
2364 035134 012701 000026          MOV    #22.,R1          ;INIT COUNTER
2365 035140 012737 000226 002422    MOV    #SYNCH,TXWORD
2366 035146 004737 005146          6$: JSR    PC,LDTXSI      ;LOAD AN SOM INTO TX SILO
2367 035152 005301          DEC    R1              ;DECR COUNTER
2368 035154 001374          BNE   6$              ;BR IF MORE TO LOAD
2369 035156 004737 005120          JSR    PC,WAIT50       ;ALLOW SILO TO RIPPLE
2370 035162 004737 007406          JSR    PC,RCV1ST      ;CLK LU UNTIL 3RD SYNCH RCV'D (RCVR IS ACTIVE)
2371 035166 000030          24.
2372 035170 012701 000025          MOV    #21.,R1          ;INIT COUNTER
2373 035174 004737 007722          8$: JSR    PC,CKDATA      ;READ A SYNCH, CLK NEXT ONE
2374 035200 000226          SYNCH
2375 035202 000010          8.
2376 035204 005301          DEC    R1              ;DECR COUNTER
2377 035206 001372          BNE   8$              ;BR IF NOT ALL CHECKED
2378 035210 004737 007722          JSR    PC,CKDATA      ;CHECK LAST SYNCH
2379 035214 000226          SYNCH
2380 035216 000004          4
2381 035220 012737 000011 002400    MOV    #11,REGNUM      ;SET REG NO. = 11
2382 035226 112737 000200 002366    MOVB  #OC,WRIBYT
    
```

TEST 41 - IDLE SYNCHS TEST - CHAR MODE

```

2383 035234 004737 003722      JSR    PC,WRITLU      ;SET OC IN REG 11
2384 035240 004737 005226      JSR    PC,STPLU      ;FINISH CLOCKING CHAR, AND THEN SOME
2385 035244 000014              12.
2386 035246 004737 007722      JSR    PC,CKDATA     ;RCV A MARK CHAR, CHK IT
2387 035252 000377              377
2388 035254 000000              0
2389 035256 004737 003576      24$: JSR    PC,MSTCLR   ;ISSUE MASTER CLEAR TO CLEAN UP
2390 035262 000000              0
      035262 104401
      035262
      L10075: TRAP    C$ETST

```

2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413

.SBTTL TEST 42 - STRIP SYNCH TEST

```

;*
;* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
;* INITIATED IN CHAR MODE AND WITH THE STRIP SYNCH
;* BIT SET. THEN 24 (DEC) SYNCHS ARE SENT
;* FOLLOWED BY THE FOLLOWING DATA CHARACTERS : 377, 000, 125, 252,
;* AND 2 TERMINATING SYNCHS
;* EACH OF THE 23 SYNCHS A, ER THE FIRST ARE CHECKED AT THE TRANSMITTER OUTPUT,
;* BY SCANNING THE TXDATA BIT.
;* EACH USYRT RCV FLAG IS TIMED AS IT IS RECEIVED, AND THE 4 DATA WORDS
;* ARE READ AND COMPARED TO EXPECTED VALUES.
;* FINALLY, THE LINE UNIT IS CLOCKED FOR SEVERAL CHAR TIMES, AND A CHECK
;* IS MADE FOR OACT = 0 (TEOM SHOULD CAUSE TX ENABLE TO DROP).
;* THE ABOVE TEST IS REPEATED FOR EACH OF THE FOLLOWING SYNCH CHAR DATA
;* PATTERNS : 226,000,125,252,376,177.

```

BGNTST

```

2414 035264 012737 035504 002362      MOV    #24$,RETADR   ;SET TEST EXIT ADRS FOR ERRORS
2415 035272 012701 035512              MOV    #SYNPAT,R1    ;GET POINTER TO DATA
2416 035276 011137 035306      2$: MOV    (R1),3$     ;GET A SYNCH PATTERN
2417 035302 004737 005512      JSR    PC,INITRN     ;MST CLR, LOAD 2 SOM'S
2418 035306 000000              3$: .WORD 0           ;SYNCH PATTERN GOES HERE
2419 035310 000311              CRC2!CRC1!STRIP!DDCMP
2420 035312 012737 000400 002422      MOV    #TXSOM,TXWORD
2421 035320 012702 000026              MOV    #22.,R2       ;LOAD 22 SOM'S INTO TX SILO
2422 035324 004737 005146      6$: JSR    PC,LDTXSI   ;LOAD DATA CHARS INTO TX SILO
2423 035330 005302              DEC    R2
2424 035332 001374              BNE   6$
2425 035334 004737 010770      JSR    PC,LODMSG     ;LOAD DATA CHARS INTO TX SILO
2426 035340 003246              MSG4
2427 035342 000006              6
2428 035344 012737 001000 002422      MOV    #TXEOM,TXWORD
2429 035352 004737 005146              JSR    PC,LDTXSI     ;LOAD A TEOM
2430 035356 004737 005146              JSR    PC,LDTXSI     ;LOAD ANOTHER TEOM
2431 035362 004737 005120              JSR    PC,WAIT50     ;ALLOW SILO TO RIPPLE
2432 035366 011137 035410              MOV    (R1),16$     ;GET CURRENT SYNCH PATTERN
2433 035372 012702 000027              MOV    #23.,R2       ;INIT COUNTER
2434 035376 004737 005226              JSR    PC,STPLU     ;CLOCK OUT FIRST SYNCH
2435 035402 100010              CHPCHK!8.
2436 035404 004737 011150      14$: JSR    PC,CKTBIT   ;CHECK TX'D SYNCH

```

```
2437 035410 000000          16$: .WORD 0          ;SYNCH PATTERN GOES HERE
2438 035412 005302          DEC R2          ;DECR COUNTER
2439 035414 001373          BNE 14$        ;BR IF NOT DONE CHECKING LAST 23 SYNCHS
2440 035416 004737 007406   JSR PC,RCV1ST ;CLOCK UNTIL 000 CHAR RCV'D
2441 035422 000010          8.
2442 035424 004737 007722   JSR PC,CKDATA ;RCV 377, CLOCK 000
2443 035430 000377          377
2444 035432 000010          8.
2445 035434 004737 007722   JSR PC,CKDATA ;RCV 000, CLK 125
2446 035440 000000          000
2447 035442 000010          8.
2448 035444 004737 007722   JSR PC,CKDATA ;RCV 125, CLK 252
2449 035450 000125          125
2450 035452 000010          8.
2451 035454 004737 007722   JSR PC,CKDATA ;RCV 252, CLK END OF MSG
2452 035460 000252          252
2453 035462 000040          32.
2454 035464 004737 005324   JSR PC,OACTIV ;CHK FOR OACT = 0
2455 035470 000000          0
2456 035472 062701 000002   ADD #2,R1     ;INIT SYNCH PATTERN POINTER
2457 035476 020127 035526   CMP R1,#SYNPAT+12. ;SEE IF ALL PATTERNS CHECKED YET
2458 035502 103675          BLO 2$        ;BR IF NOT YET
2459 035504 004737 003576   JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
2460 035510          ENDTST
      035510          L10076:
      035510 104401          TRAP C$ETST
2461
2462 035512 000226          SYNPAT: 226
2463 035514 000000          000
2464 035516 000125          125
2465 035520 000252          252
2466 035522 000376          376
2467 035524 000177          177
2468
2469
2470
2471
2472
```

.SBTTL HARDWARE PARAMETER CODING SECTION

```
////////////////////////////////////  
:/ THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS  
:/ THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE  
:/ MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE  
:/ INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE  
:/ MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS  
:/ WITH THE OPERATOR.  
////////////////////////////////////
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14 035526          BGNHRD  
    035526 000022  
    035530  
                                .WORD L10077-L$HARD/2  
                                L$HARD::  
15  
16 035530          GPRMA  ADDRES,2,0,160000,177776,YES  
    035530 001031  
    035532 035574  
    035534 160000  
    035536 177776  
                                .WORD  T$CODE  
                                .WORD  ADDRES  
                                .WORD  T$LLOLIM  
                                .WORD  T$HILIM  
17 035540          GPRMA  VECTOR,4,0,0,674,YES  
    035540 002031  
    035542 035622  
    035544 000000  
    035546 000674  
                                .WORD  T$CODE  
                                .WORD  VECTOR  
                                .WORD  T$LLOLIM  
                                .WORD  T$HILIM  
18 035550          GPRMD  PRIRTY,6,0,7000,4,7,YES  
    035550 003032  
    035552 035653  
    035554 007000  
    035556 000004  
    035560 000007  
                                .WORD  T$CODE  
                                .WORD  PRIRTY  
                                .WORD  7000  
                                .WORD  T$LLOLIM  
                                .WORD  T$HILIM  
19 035562          GPRMD  ISRUN,24,0,7,0,7,YES  
    035562 012032  
    035564 035704  
    035566 000007  
    035570 000000  
    035572 000007  
                                .WORD  T$CODE  
                                .WORD  ISRUN  
                                .WORD  7  
                                .WORD  T$LLOLIM  
                                .WORD  T$HILIM  
20  
21 035574          ENDHRD  
                                .EVEN  
                                L10077:  
22  
23 035574          104    105    126  ADDRES: .ASCIZ /DEVICE CSR ADDRESS : /  
    035577          111    103    105  
    035602          040    103    123  
    035605          122    040    101  
    035610          104    104    122  
    035613          105    123    123  
    035616          040    072    040  
    035621          000  
24 035622          104    105    126  VECTOR: .ASCIZ /DEVICE VECTOR ADDRESS : /  
    035625          111    103    105  
    035630          040    126    105  
    035633          103    124    117  
    035636          122    040    101
```

	035641	104	104	122	
	035644	105	123	123	
	035647	040	072	040	
	035652	000			
25	035653	104	105	126	PRIPTY: .ASCIZ /DEVICE PRIORITY LEVEL : /
	035656	111	103	105	
	035661	040	120	122	
	035664	111	117	122	
	035667	111	124	131	
	035672	040	114	105	
	035675	126	105	114	
	035700	040	072	040	
	035703	000			
26	035704	115	111	103	ISRUN: .ASCIZ /MICROPROCESSOR RUN SWITCH - TYPE 0 IF OFF, 1 IF ON : /
	035707	122	117	120	
	035712	122	117	103	
	035715	105	123	123	
	035720	117	122	040	
	035723	122	125	116	
	035726	040	123	127	
	035731	111	124	103	
	035734	110	040	055	
	035737	040	124	131	
	035742	120	105	040	
	035745	060	040	111	
	035750	106	040	117	
	035753	106	106	054	
	035756	040	061	040	
	035761	111	106	040	
	035764	117	116	040	
	035767	072	040	000	
27					.EVEN
28					
29					
30					
31					
32					
33					

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41

.SBTTL SOFTWARE PARAMETER CODING SECTION

:/ THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
:/ THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
:/ MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
:/ INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
:/ MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
:/ WITH THE OPERATOR.

BGNSFT

.WORD L10100-L\$SOFT/2
L\$SOFT::

ENDSFT

.EVEN
L10100:

.EVEN

***** PATCH AREA FOR DEBUG *****

PATCH:
. = +200
NOP
NOP
NOP

ENDMOD

LASTAD

.EVEN
.WORD 0
.WORD 0

L\$LAST::

.END

000000

035772
035772
035774

035774
035774

035774

036174
000240
000240
000240

036174
036176
036200

036202
036202

000000
000000
036206

000001

ABORT = 000004	BIT7 = 000200 G	C\$OPEN= 000034	EM23 012751	EVL = 000004 G
ADDRES 035574	BIT8 = 000400 G	C\$PNTB= 000014	EM24 012772	E\$END = 002100
ADR = 000020 G	BIT9 = 001000 G	C\$PNTF= 000017	EM25 013007	E\$LOAD= 000035
ANBITS 002561	BOE = 000400 G	C\$PNTS= 000016	EM26 013043	FMT1 011426
APA = 000200	BPOLL = 000100	C\$PNTX= 000015	EM27 013075	FMT10 011671
ASBC0 = 000020	BSEL1 002450	C\$QIO = 000377	EM28 013126	FMT11 011722
ASBC1 = 000040	BSEL2 002452	C\$RDBU= 000007	EM29 013147	FMT19 011761
ASBC2 = 000100	BSEL4 002454	C\$REFG= 000047	EM3 012174	FMT2 011436
ASSEMB= 000010	CARR = 000001	C\$RESE= 000033	EM30 013164	FMT24 012012
AXNUM 002402	CHPCHK= 100000	C\$REVI= 000003	EM31 013205	FMT27 021554
AX0.15= 002322	CHPTYP 002430	C\$RFLA= 000021	EM32 013222	FMT3 011460
AX0.16= 002324	CKDATA 007722	C\$RPT = 000025	EM33 013251	FMT4 011522
AX1 = 000001	CKTBIT 011150	C\$SEFG= 000046	EM34 013274	FMT5 011535
AX1.15= 002326	CRCCHK= 100000	C\$SPRI= 000041	EM35 013322	FMT6 011565
AX1.16= 002330	CRCTY0= 000001	C\$SVEC= 000037	EM36 013342	FMT7 011620
AX2 = 000002	CRCTY1= 000002	C\$TPRI= 000013	EM37 013356	FMT8 011630
AX2.15= 002332	CRCTY2= 000004	C32BCC= 000040	EM38 013377	FMT9 011664
AX2.16= 002334	CRC1 = 000100	C32ENB= 000004	EM39 013414	FRSTIM 002412
AX3.15= 002336	CRC2 = 000200	DDC = 000100	EM4 012213	F\$AU = 000015
AX3.16= 002340	CS = 000004	DDCMP = 000001	EM40 013434	F\$AUTO= 000020
AX315U= 000372	C\$AU = 000052	DEVMAP 002434	EM41 013450	F\$BGN = 000040
A1 024174	C\$AUTO= 000061	DEVPTR 002436	EM42 013471	F\$CLEA= 000007
A10 034156	C\$BRK = 000022	DFPTBL 002252 G	EM43 013506	F\$DU = 000016
A2 026602	C\$BSEG= 000004	DH1 014240	EM44 013533	F\$END = 000041
A3 027262	C\$BSUB= 000002	DH2 014262	EM45 013560	F\$HARD= 000004
A4 027376	C\$CEFG= 000045	DH3 014311	EM46 013605	F\$HW = 000013
A5 027506	C\$CLCK= 000062	DH4 014347	EM47 013640	F\$INIT= 000006
A6 027702	C\$CLEA= 000012	DH5 014411	EM48 013673	F\$JMP = 000050
A7 030246	C\$CLOS= 000035	DH6 014414	EM49 013726	F\$MOD = 000000
A8 030462	C\$CLP1= 000006	DH7 014417	EM5 012266	F\$MSG = 000011
A9 031012	C\$CVEC= 000036	DH8 014451	EM50 013765	F\$PROT= 000021
BADDAT 002406	C\$DCLN= 000044	DH9 014510	EM51 014021	F\$PWR = 000017
BCC = 000001	C\$DODU= 000051	DIAGMC= 000000	EM52 014061	F\$RPT = 000012
BCCCHK= 100000	C\$DRPT= 000024	DISILO 002426	EM53 014122	F\$SEG = 000003
BIT0 = 000001 G	C\$DU = 000053	DISSI = 000040	EM54 014162	F\$SOFT= 000005
BIT00 = 000001 G	C\$EDIT= 000003	DTR = 000100	EM6 012317	F\$SRV = 000010
BIT01 = 000002 G	C\$ERDF= 000055	EBLK = 000002	EM60 014204	F\$SUB = 000002
BIT02 = 000004 G	C\$ERHR= 000056	EF.CON= 000036 G	EM65 014220	F\$SW = 000014
BIT03 = 000010 G	C\$ERRO= 000060	EF.NEW= 000035 G	EM7 012350	F\$TEST= 000001
BIT04 = 000020 G	C\$ERSF= 000054	EF.PWR= 000034 G	EM8 012365	GETALL 004500
BIT05 = 000040 G	C\$ERSO= 000057	EF.RES= 000037 G	EM9 012406	GETPRM 021262
BIT06 = 000100 G	C\$ESCA= 000010	EF.STA= 000040 G	ENAX = 000004	GETREG 003770
BIT07 = 000200 G	C\$ESEG= 000005	EM1 012101	ENDIT 021440	GOAH = 000010
BIT08 = 000400 G	C\$ESUB= 000003	EM10 012423	ENDPAT 003220	GOODAT 002404
BIT09 = 001000 G	C\$ETST= 000001	EM11 012444	ENDTRN 006246	G\$CNT0= 000200
BIT1 = 000002 G	C\$EXIT= 000032	EM12 012461	EOM = 000002	G\$DELM= 000372
BIT10 = 002000 G	C\$GETB= 000026	EM13 012502	ERRFLG 002356	G\$DISP= 000003
BIT11 = 004000 G	C\$GETW= 000027	EM14 012532	ERROR1 002420	G\$EXCP= 000400
BIT12 = 010000 G	C\$GMAN= 000043	EM15 012547	ERR1 014554 G	G\$HILI= 000002
BIT13 = 020000 G	C\$GPHR= 000042	EM16 012576	ERR2 014606 G	G\$LOLI= 000001
BIT14 = 040000 G	C\$GPLO= 000030	EM17 012617	ERR3 015114 G	G\$NO = 000000
BIT15 = 100000 G	C\$GPRI= 000040	EM18 012634	ERR4 015576 G	G\$OFFS= 000400
BIT2 = 000004 G	C\$INIT= 000011	EM19 012655	ERR5 016254 G	G\$OF SI= 000376
BIT3 = 000010 G	C\$INLP= 000020	EM2 012135	ERR6 016766 G	G\$PRMA= 000001
BIT4 = 000020 G	C\$MANI= 000050	EM20 012672	ERR7 017444 G	G\$PRMD= 000002
BIT5 = 000040 G	C\$MEM = 000031	EM21 012713	ERR8 020076 G	G\$PRML= 000000
BIT6 = 000100 G	C\$MSG = 000023	EM22 012730	ERR9 020604 G	G\$RADA= 000140

G\$RADE= 000000	LUREG 002302	L\$SOFT 035774 G	L10061 031112	PATH 002654
G\$RADD= 000040	LUR10 = 002302	L\$SPC 002056 G	L10062 031462	PATI 002664
G\$RADL= 000120	LUR11 = 002304	L\$SPCP 002020 G	L10063 032024	PATJ 002674
G\$RADO= 000020	LUR12 = 002306	L\$SPTP 002024 G	L10064 032274	PATK 002704
G\$XFER= 000004	LUR13 = 002310	L\$STA 002030 G	L10065 032636	PATL 003014
G\$YES = 000010	LUR14 = 002312	L\$SW 002302 G	L10066 032766	PATM 003022
HDX = 000020	LUR15 = 002314	L\$TEST 002114 G	L10067 033700	PATN 003032
HELP = 000001	LUR16 = 002316	L\$TIML 002014 G	L10070 034162	PATO 003050
HOE = 100000 G	LUR17 = 002320	L\$UNIT 002012 G	L10071 034362	PATP 003066
IACT = 000100	LUSWI1 002466	L10000 002300	L10072 034662	PATU 003104
IACTIV 006666	LUSWI2 002470	L10001 002302	L10073 034752	PATV 003152
IBE = 010000 G	LUSWI3 002472	L10002 014604	L10074 035114	PNT = 001000 G
IC = 000200	LU1MOD 002000 G	L10003 015112	L10075 035262	POLL = 000200
ICIR = 000010	L\$ACP 002110 G	L10004 015574	L10076 035510	PRI = 002000 G
IDL = 000010	L\$APT 002036 G	L10005 016252	L10077 035574	PRIOR 002350
IDLE = 000040	L\$AU 021606 G	L10006 016764	L10100 035774	PRIPTY 035653
IDU = 000040 G	L\$AUT 002070 G	L10007 017442	MAINT1= 000010	PRI00 = 000000 G
IER = 020000 G	L\$AUTO 021442 G	L10010 020074	MAINT2= 000004	PRI01 = 000040 G
IERR = 000002	L\$CCP 002106 G	L10011 020602	MCLK = 000002	PRI02 = 000100 G
INITRN 005512	L\$CLEA 021522 G	L10012 021060	MCLR = 000100	PRI03 = 000140 G
INTFLG 002354	L\$CO 002032 G	L10013 021062	MODR = 000010	PRI04 = 000200 G
INTGRL= 000010	L\$DEPO 002011 G	L10015 021440	MPCSR 002446	PRI05 = 000240 G
IRDY = 000020	L\$DESC 003470 G	L10016 021520	MPIVEC 002460	PRI06 = 000300 G
ISIRDY 006402	L\$DESP 002076 G	L10017 021522	MPOVEC 002462	PRI07 = 000340 G
ISR = 000100 G	L\$DEVP 002060 G	L10020 021552	MPRIOR 002464	PSTACK 002346
ISRUN 035704	L\$DISP 002124 G	L10021 021606	MSG1 003220	RAB = 000004
IXE = 004000 G	L\$DLY 002116 G	L10022 021670	MSG4 003246	RABT = 000004
ISAU = 000041	L\$DTP 002040 G	L10023 021754	MSTCLR 003576	RAX15 002370
ISAUTO= 000041	L\$DTYP 002034 G	L10024 022100	MVIOX = 021000	RAX16 002372
ISCLN = 000041	L\$DU 021524 G	L10025 022202	MVIXO = 122000	RCVBUF 003262
ISDU = 000041	L\$DUT 002072 G	L10026 022330	MVIXOX= 121000	RCV1ST 007406
ISHRD = 000041	L\$DVTY 003462 G	L10027 022450	NEWST 021242	RDALL = 000004
ISINIT= 000041	L\$EF 002052 G	L10030 022624	OACT = 000100	RDAX = 000020
ISMOD = 000041	L\$ENVI 002044 G	L10031 023060	OACTIV 005324	RDRXSI 007326
ISMSG = 000041	L\$ETP 002102 G	L10032 023176	OC = 000200	READAX 004076
ISPROT= 000040	L\$EXP1 002046 G	L10033 023314	OCOR = 000020	READLU 003644
ISPTAB= 000041	L\$EXP4 002064 G	L10034 023432	OP = 000002	READY = 000200
ISPWR = 000041	L\$EXP5 002066 G	L10035 023550	ORDY = 000020	REDBYT 002364
ISRPT = 000041	L\$HARD 035530 G	L10036 024200	OSIRDY 004634	REDDAT 002500
ISSEG = 000041	L\$HIME 002120 G	L10037 024532	OVRR = 000010	REGNUM 002400
ISSETU= 000041	L\$HPCP 002016 G	L10040 025022	OSAPTS= 000000	REG0 002510
ISSFT = 000041	L\$HPTP 002022 G	L10041 025232	OSAU = 000001	REG1 002512
ISSRV = 000041	L\$HW 002252 G	L10042 025436	OSBGNR= 000000	REG2 002514
ISSUB = 000041	L\$ICP 002104 G	L10043 025642	OSBGS= 000000	REG3 002516
ISTST = 000041	L\$INIT 021072 G	L10044 026100	OSDU = 000001	REG4 002520
I422 = 000200	L\$LADP 002026 G	L10045 026332	OSERRT= 000000	REG5 002522
J\$JMP = 000167	L\$LAST 036206 G	L10046 026236	OSGNSW= 000000	REG6 002524
LDMSG1 011346	L\$LOAD 002100 G	L10047 026330	OSPOIN= 000001	REG7 002526
LDTXSI 005146	L\$LUN 002074 G	L10050 026604	OSSETU= 000000	REOM = 000002
LOADAT 002410	L\$MREV 002050 G	L10051 026602	PATA 002571	RERR = 000200
LODMSG 010770	L\$NAME 002000 G	L10052 027262	PATB 002615	RETADR 002362
LOE = 040000 G	L\$PRIO 002042 G	L10053 027402	PATC 002625	RING = 000200
LOGDEV 002344	L\$PROT 021064 G	L10054 027512	PATCH 035774	ROMI = 000002
LOOPIN 004046	L\$PRT 002112 G	L10055 027706	PATD 002630	ROMO = 000004
LOT = 000010 G	L\$REPP 002062 G	L10056 030252	PATE 002633	ROR = 000010
LULOOP= 000010	L\$REV 002010 G	L10057 030462	PATF 002641	RRDYTO= 000001
LULP = 000040	L\$RPT 021062 G	L10060 031012	PATG 002644	RSEOM 007054

RSOM = 000001	STPCLK 003540	TXABT = 002000	T\$TEMP= 000000	T29 031014 G
RTS = 000040	STPERR 007620	TXCHAR 006074	T\$TEST= 000052	T3 021756 G
RUN = 000200	STPLU 005226	TXDATA= 000040	T\$TSTM= 177777	T30 031114 G
RUNINH 002476	STR = 000040	TXEN = 000100	T\$TSTS= 000001	T31 031464 G
RXABT = 002000	STRIP = 0000i0	TXEOM = 001000	T\$\$AU = 010021	T32 032026 G
RXBCC = 000400	SUBRPC 002352	TXGA = 000010	T\$\$AUT= 010016	T33 032276 G
RXCHAR 011046	SVCGBL= 000000	TXGOA = 004000	T\$\$CLE= 010017	T34 032640 G
RXEBL = 001000	SVCINS= 000001	TXLENO= 000040	T\$\$DU = 010020	T35 032770 G
RXLENO= 000001	SVCSUB= 000001	TXLEN1= 000100	T\$\$HAR= 010077	T36 033702 G
RXLEN1= 000002	SVCTAG= 000001	TXLEN2= 000200	T\$\$HW = 010000	T37 034164 G
RXLEN2= 000004	SVCTST= 000001	TXSOM = 000400	T\$\$INI= 010015	T38 034364 G
RXOVR = 004000	SW0 = 000002	TXWORD 002422	T\$\$MSG= 010012	T39 034664 G
RXWORD 002424	SW1 = 000004	TX0 = 000001	T\$\$PRO= 010014	T4 022102 G
RX0 = 000001	SW2 = 000010	TX1 = 000002	T\$\$RPT= 010013	T40 034754 G
RX1 = 000002	SW3 = 000040	TX2 = 000004	T\$\$SEG= 010000	T41 035116 G
RX2 = 000004	SYNCH = 000226	TX3 = 000010	T\$\$SOF= 010100	T42 035264 G
RX3 = 000010	SYNPAT 035512	TX4 = 000020	T\$\$SUB= 010051	T5 022204 G
RX4 = 000020	SYNO = 000001	TX5 = 000040	T\$\$SW = 010001	T6 022332 G
RX5 = 000040	SYN1 = 000002	TX6 = 000100	T\$\$TES= 010076	T7 022452 G
RX6 = 000100	SYN2 = 000004	TX7 = 000200	T1 021610 G	T8 022626 G
RX7 = 000200	SYN3 = 000010	T\$ARGC= 000001	T10 023200 G	T9 023062 G
R14NRW 002560	SYN4 = 000020	T\$CODE= 012032	T11 023316 G	UAM = 000200 G
SAVE4 002414	SYN5 = 000040	T\$ERRN= 000032	T12 023434 G	UNIT 002440
SAVE6 002416	SYN6 = 000100	T\$EXCP= 000000	T13 023552 G	UNRR = 000001
SAVLEN 002432	SYN7 = 000200	T\$FLAG= 000040	T14 024202 G	UPBITS 002550
SCRACH 002342	S\$LSYM= 010000	T\$GMAN= 000000	T15 024534 G	VECTOR 035622
SEC = 000020	TEOM = 000002	T\$HILI= 000007	T16 025024 G	V35 = 000020
SECA = 000020	TERR = 000200	T\$LAST= 000001	T17 025234 G	WAIT50 005120
SELFR = 000040	TEST = 000001	T\$LOLI= 000000	T18 025440 G	WAX = 000010
SELSBY= 000002	TESTMD= 000004	T\$LSYM= 010000	T19 025644 G	WAX15 002374
SEL4 002454	TIMFLG 002360	T\$LTNO= 000052	T2 021672 G	WAX16 002376
SEL6 002456	TMPO 002530	T\$NEST= 177777	T20 026102 G	WRDYTO= 000002
SETUP 010612	TMP1 002532	T\$NS0 = 000000	T20.1 026132	WRIBYT 002366
SFPTBL 002302 C	TMP2 002534	T\$NS1 = 000005	T20.2 026240	WRITAX 004264
SIGQ = 000100	TMP3 002536	T\$NS2 = 000002	T21 026334 G	WRITLU 003722
SIGR = 000200	TMP4 002540	T\$NS3 = 000003	T21.1 026346	XYZ = 000100
SOM = 000001	TMP5 002542	T\$PTNU= 000000	T22 026606 G	X\$ALWA= 000000
STALL 005136	TMP6 002544	T\$SAVL= 177777	T23 027264 G	X\$FALS= 000040
STARES 002444	TMP7 002546	T\$SEGL= 177777	T24 027404 G	X\$OFFS= 000400
STARST 021232	TSOM = 000001	T\$SEK0= 010000	T25 027514 G	X\$TRUE= 000020
STBY = 000002	TSTCON 002474	T\$SUBN= 000000	T26 027710 G	\$LSTIN= 000001
STEPLU= 000020	TSTNUM 002442	T\$TAGL= 177777	T27 030254 G	\$LSTTA= 000001
STEPMP= 000001	TXAB = 000004	T\$TAGN= 010101	T28 030464 G	

. ABS. 036206 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 21664 WORDS (85 PAGES)

DYNAMIC MEMORY AVAILABLE FOR 69 PAGES

CZDMRB.BIC,CZDMRB.SEQ/C/N:TOC=SVC34R.MLB,CZDMRB.P11

F\$END	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18	7-18
	7-18	7-18	7-18#	7-24	16-102	16-115	16-134	16-153	16-173	16-192	16-210	16-230	16-244	17-11
	19-74	20-19	21-11	22-13	23-10	24-14	24-14	24-14	24-27	24-27	24-39	24-39	24-39	24-52
	24-52	24-67	24-67	24-67	24-86	24-90	24-90	24-101	24-101	24-101	24-117	24-117	24-128	24-128
	24-128	24-150	24-150	24-166	24-166	24-166	24-186	24-186	24-203	24-203	24-203	24-227	24-232	24-232
	24-247	24-247	24-247	24-279	24-286	24-286	24-298	24-298	24-298	24-316	24-320	24-320	24-332	24-332
	24-332	24-350	24-354	24-354	24-366	24-366	24-366	24-384	24-388	24-388	24-400	24-400	24-400	24-418
	24-422	24-422	24-449	24-449	24-449	24-523	24-523	24-539	24-539	24-539	24-552	24-566	24-578	24-589
	24-594	24-594	24-610	24-610	24-610	24-638	24-648	24-655	24-655	24-676	24-676	24-676	24-695	24-706
	24-710	24-710	24-727	24-727	24-727	24-746	24-756	24-760	24-760	24-774	24-774	24-774	24-793	24-803
	24-807	24-807	24-832	24-832	24-832	24-856	24-867	24-872	24-872	24-889	24-889	24-889	24-898	24-898
	24-915	24-920	24-920	24-924	24-924	24-938	24-938	24-939	24-939	24-957	24-957	24-957	24-960	24-960
	24-984	24-994	24-999	24-999	24-:00	24-:00	24-:22	24-:22	24-:22	24-:99	24-:22	24-:32	24-:32	24-:52
	24-;52	24-;52	24-;81	24-;81	24-<01	24-<01	24-<01	24-<27	24-<27	24-<49	24-<49	24-<49	24-<87	24-<87
	24-=07	24-=07	24-=07	24-=76	24-=76	24-=91	24-=91	24-=91	24->32	24->32	24->48	24->48	24->48	24-?12
	24-?12	24-?27	24-?27	24-?27	24-?47	24-?47	24-?63	24-?63	24-?63	24-@27	24-@27	24-@43	24-@43	24-@43
	24-A10	24-A10	24-A26	24-A26	24-A26	24-A73	24-A73	24-A89	24-A89	24-A89	24-B56	24-B56	24-B69	24-B69
	24-B69	24-B88	24-B88	24-C05	24-C05	24-C05	24-D34	24-D34	24-D58	24-D58	24-D58	24-E35	24-E35	24-E53
	24-E53	24-E53	24-E87	24-E87	24-F05	24-F05	24-F05	24-F55	24-F55	24-F70	24-F70	24-F70	24-F87	24-F87
	24-G08	24-G08	24-G08	24-G40	24-G40	24-G59	24-G59	24-G59	24-G90	24-G90	24-H13	24-H13	24-H13	24-H60
	24-H60	25-21	26-16	26-37										
F\$HARD	7-18#	25-14	25-21											
F\$HW	7-18#	10-9	10-23											
F\$INIT	7-18#	19-8	19-74											
F\$JMP	7-18#													
F\$MOD	7-18#	7-24	26-37											
F\$MSG	7-18#	16-100	16-102	16-106	16-115	16-121	16-134	16-140	16-153	16-159	16-173	16-179	16-192	16-198
	16-210	16-216	16-230	16-236	16-244									
F\$PROT	7-18#	18-8	18-12											
F\$PWR	7-18#													
F\$RPT	7-18#	17-9	17-11											
F\$SEG	7-18#	24-72	24-86	24-303	24-316	24-337	24-350	24-371	24-384	24-405	24-418	24-680	24-706	24-731
	24-756	24-778	24-803	24-838	24-867	24-900	24-915							
F\$SOFT	7-18#	26-13	26-16											
F\$SRV	7-18#													
F\$SUB	7-18#	24-898	24-920	24-924	24-938	24-960	24-999							
F\$SW	7-18#	11-8	11-11											
F\$TEST	7-18#	24-14	24-27	24-39	24-52	24-67	24-90	24-101	24-117	24-128	24-150	24-166	24-186	24-203
	24-232	24-247	24-286	24-298	24-320	24-332	24-354	24-366	24-388	24-400	24-422	24-449	24-523	24-539
	24-594	24-610	24-655	24-676	24-710	24-727	24-760	24-774	24-807	24-832	24-872	24-889	24-939	24-957
	24-:00	24-:22	24-:32	24-:52	24-:81	24-<01	24-<27	24-<49	24-<87	24-=07	24-=76	24-=91	24->32	24->48
	24-?12	24-?27	24-?47	24-?63	24-@27	24-@43	24-A10	24-A26	24-A73	24-A89	24-B56	24-B69	24-B88	24-C05
	24-D34	24-D58	24-E35	24-E53	24-E87	24-F05	24-F55	24-F70	24-F87	24-G08	24-G40	24-G59	24-G90	24-H13
	24-H60													
FMT1	16-9#	16-101	16-107	16-122	16-142	16-160	16-181	16-199	16-218	16-237				
FMT10	16-18#	16-141	16-180	16-217										
FMT11	16-19#	16-161												
FMT19	16-20#	24-456												
FMT2	16-10#	16-108	16-123	16-143	16-162	16-182	16-200	16-219	16-238					
FMT24	16-21#	24-482	24-501	24-520										
FMT27	22-12	22-15#												
FMT3	16-11#	16-110	16-125	16-164	16-221									
FMT4	16-12#	16-111	16-126	16-130	16-145	16-149	16-165	16-169	16-184	16-188	16-202	16-206	16-222	16-226
	16-240													
FMT5	16-13#	16-112	16-127	16-131	16-146	16-150	16-166	16-170	16-185	16-189	16-203	16-207	16-223	16-227
	16-241													
FMT6	16-14#	16-114	16-129	16-133	16-148	16-152	16-168	16-172	16-187	16-191	16-205	16-209	16-225	16-229

FMT7	16-243													
FMT8	16-15#	16-109	16-144	16-201	16-220	16-239								
FMT9	16-16#	16-124	16-163	16-183										
	16-17#	16-113	16-128	16-132	16-147	16-151	16-167	16-171	16-186	16-190	16-204	16-208	16-224	16-228
	16-242													
FRSTIM	13-37#	19-16	19-23*											
GSCNTO	7-18#													
GSDLM	7-18#													
GSDISP	7-18#													
GSEXCP	7-18#													
GSHILI	7-18#													
GSLOLI	7-18#													
GSNO	7-18#													
GSOFFS	7-18#	25-16	25-17	25-18	25-19									
GSOFSI	7-18#	25-16	25-17	25-18	25-19									
GSPRMA	7-18#	25-16	25-17											
GSPRMD	7-18#	25-18	25-19											
GSPRML	7-18#													
GSRADA	7-18#													
GSRADB	7-18#													
GSRADD	7-18#													
GSRADL	7-18#													
GSRADO	7-18#	25-16	25-17	25-18	25-19									
GSXFER	7-18#													
GYES	7-18#	25-16	25-17	25-18	25-19									
GETALL	15-222#	15-270	15-276	15-286	15-292	15-399	15-405	15-563	15-595	15-601	15-611	15-617	15-652	15-658
	15-692	15-698	15-706	15-712	15-858	15-863	15-882	15-888	15-897	15-903	15-912	15-918	15-927	15-933
	15-:65	15-:71	24-575	24-586	24-635	24-645	24-692	24-702	24-743	24-752	24-790	24-799	24-853	24-863
	24-911	24-934	24-981	24-991	24-:96	24-:19	24-<81	24-=24	24-=33	24-=45	24-=64	24-=72	24-?95	24-@10
	24-@93	24-A58	24-B39	24-B83	24-C87									
GETPRM	19-36	19-47#	19-54											
GETREG	15-99#	15-228	24-48	24-82	24-113	24-146	24-182	24-224	24-276	24-312	24-346	24-380	24-414	24-478
	24-497	24-516	24-549	24-563										
GOAH	12-60#													
GOODAT	13-34#	15-850*	15-851*	15-872*	15-873*	16-110	16-125	16-164	16-221	24-45*	24-46*	24-80*	24-110*	24-111*
	24-143*	24-144*	24-179*	24-180*	24-221*	24-222*	24-273*	24-274*	24-310*	24-344*	24-378*	24-412*	24-476*	24-495*
	24-514*	24-572*	24-573*	24-583*	24-584*	24-632*	24-633*	24-642*	24-643*	24-690*	24-699*	24-700*	24-741*	24-750*
	24-788*	24-797*	24-850*	24-851*	24-860*	24-861*	24-908*	24-909*	24-931*	24-932*	24-978*	24-979*	24-988*	24-989*
	24-:94*	24-:17*	24-?93*	24-@08*	24-@18*	24-@91*	24-A56*	24-B37*	24-B81*	24-C85*	24-D05*	24-D21*		
HDX	12-78#	12-148#												
HELP	7-4#	8-9	8-19	9-10	14-24									
HOE	12-20#													
ISAU	7-18#	23-9#	23-10#											
ISAUTO	7-18#	20-7#	20-19#											
ISCLN	7-18#	21-8#	21-11#											
ISDU	7-18#	22-8#	22-13#											
ISHRD	25-14#	25-21#												
ISINIT	7-18#	19-8#	19-74#											
ISMOD	7-18#	7-24	7-24#	26-37	26-37#									
ISMSG	7-18#	16-100#	16-102#	16-106#	16-115#	16-121#	16-134#	16-140#	16-153#	16-159#	16-173#	16-179#	16-192#	16-198#
	16-210#	16-216#	16-230#	16-236#	16-244#									
ISPROT	7-18#	18-8#												
ISPTAB	7-18#													
ISPWR	7-18#													
ISRPT	7-18#	17-9#	17-11#											
ISSEG	7-18#	24-14	24-39	24-67	24-72#	24-86#	24-101	24-128	24-166	24-203	24-247	24-298	24-303#	24-316#
	24-332	24-337#	24-350#	24-366	24-371#	24-384#	24-400	24-405#	24-418#	24-449	24-539	24-610	24-676	24-680#

L\$CO	8-17#		
L\$DEPO	8-17#		
L\$DESC	8-17	14-17#	
L\$DESP	8-17#		
L\$DEVP	8-17#		
L\$DISP	8-17	9-8#	
L\$DLY	8-17#		
L\$DTP	8-17#		
L\$DTYP	8-17#		
L\$DU	8-17	22-8#	
L\$DUT	8-17#		
L\$DVTY	8-17	14-12#	
L\$EF	8-17#		
L\$ENVI	8-17#		
L\$ETP	8-17#		
L\$EXP1	8-17#		
L\$EXP4	8-17#		
L\$EXP5	8-17#		
L\$HARD	8-17	25-14	25-14#
L\$HIME	8-17#		
L\$HPCP	8-17#		
L\$HPTP	8-17#		
L\$HW	8-17	10-9	10-9#
L\$ICP	8-17#		
L\$INIT	8-17	19-8#	
L\$LADP	8-17#		
L\$LAST	8-17	26-39#	
L\$LOAD	8-17#		
L\$LUN	8-17#		
L\$MREV	8-17#		
L\$NAME	8-17#		
L\$PRIO	8-17#		
L\$PROT	8-17	18-8#	
L\$PRT	8-17#		
L\$REPP	8-17#		
L\$REV	8-17#		
L\$RPT	17-9#		
L\$SOFT	26-13	26-13#	
L\$SPC	8-17#		
L\$SPCP	8-17#		
L\$SPTP	8-17#		
L\$STA	8-17#		
L\$SW	11-8	11-8#	
L\$TEST	8-17#		
L\$TIML	8-17#		
L\$UNIT	8-17#	19-49	
L10000	10-9	10-23#	
L10001	11-8	11-11#	
L10002	16-102#		
L10003	16-115#		
L10004	16-134#		
L10005	16-153#		
L10006	16-173#		
L10007	16-192#		
L10010	16-210#		
L10011	16-230#		
L10012	16-244#		

TSSHAR	25-14	25-14#	25-21											
TSSHW	10-9	10-9#	10-23											
TSSINI	19-8#	19-74												
TSSMSG	16-100#	16-102	16-106#	16-115	16-121#	16-134	16-140#	16-153	16-159#	16-173	16-179#	16-192	16-198#	16-210
	16-216#	16-230	16-236#	16-244										
TSSPRO	18-8#													
TSSRPT	17-9#	17-11												
TSSSEG	24-72	24-72#	24-86	24-86#	24-303	24-303#	24-316	24-316#	24-337	24-337#	24-350	24-350#	24-371	24-371#
	24-384	24-384#	24-405	24-405#	24-418	24-418#	24-680	24-680#	24-695	24-706	24-706#	24-731	24-731#	24-746
	24-756	24-756#	24-778	24-778#	24-793	24-803	24-803#	24-838	24-838#	24-856	24-867	24-867#	24-900	24-900#
	24-915	24-915#												
TSSSOFF	26-13	26-13#	26-16											
TSSSUB	24-898#	24-920	24-924#	24-938	24-960#	24-984	24-994	24-999						
TSSSW	11-8	11-8#	11-11											
TSSTES	24-14#	24-27	24-39#	24-52	24-67#	24-90	24-101#	24-117	24-128#	24-150	24-166#	24-186	24-203#	24-227
	24-232	24-247#	24-279	24-286	24-298#	24-320	24-332#	24-354	24-366#	24-388	24-400#	24-422	24-449#	24-523
	24-539#	24-552	24-566	24-578	24-589	24-594	24-610#	24-638	24-648	24-655	24-676#	24-710	24-727#	24-760
	24-774#	24-807	24-832#	24-872	24-889#	24-939	24-957#	24-:00	24-:22#	24-:99	24-:22	24-:32	24-:52#	24-:81
	24-<01#	24-<27	24-<49#	24-<87	24-07#	24-76	24-91#	24->32	24->48#	24-?12	24-?27#	24-?47	24-?63#	24-?27
	24-@43#	24-A10	24-A26#	24-A73	24-A89#	24-B56	24-B69#	24-B88	24-C05#	24-D34	24-D58#	24-E35	24-E53#	24-E87
	24-F05#	24-F55	24-F70#	24-F87	24-G08#	24-G40	24-G59#	24-G90	24-H13#	24-H60				
TSARGC	8-17	8-17	8-17	8-17	8-17	8-17	8-17	8-17	8-17	8-17	8-17	8-17#	8-17#	8-17#
	8-17#	8-17#	8-17#	16-101	16-101	16-101	16-101	16-101#	16-101#	16-101#	16-101#	16-107	16-107	16-107
	16-107#	16-107#	16-107#	16-108	16-108	16-108#	16-109	16-109	16-109	16-109	16-109#	16-109#	16-109#	16-110
	16-110	16-110	16-110	16-110#	16-110#	16-110#	16-111	16-111	16-111	16-111	16-111#	16-111#	16-111#	16-112
	16-112	16-112	16-112	16-112	16-112	16-112#	16-112#	16-112#	16-112#	16-112#	16-112#	16-113	16-113	16-113
	16-113#	16-114	16-114	16-114	16-114	16-114	16-114#	16-114#	16-114#	16-114#	16-114#	16-114#	16-114#	16-122
	16-122	16-122	16-122#	16-122#	16-122#	16-123	16-123	16-123#	16-124	16-124	16-124	16-124	16-124#	16-124#
	16-124#	16-125	16-125	16-125	16-125	16-125#	16-125#	16-125#	16-126	16-126	16-126	16-126	16-126#	16-126#
	16-126#	16-127	16-127	16-127	16-127	16-127	16-127	16-127#	16-127#	16-127#	16-127#	16-127#	16-127#	16-128
	16-128	16-128#	16-128#	16-129	16-129	16-129	16-129	16-129	16-129	16-129#	16-129#	16-129#	16-129#	16-129#
	16-130	16-130	16-130	16-130	16-130#	16-130#	16-130#	16-131	16-131	16-131	16-131	16-131	16-131	16-131#
	16-131#	16-131#	16-131#	16-131#	16-132	16-132	16-132	16-132#	16-132#	16-132#	16-133	16-133	16-133	16-133
	16-133	16-133#	16-133#	16-133#	16-133#	16-133#	16-141	16-141	16-141	16-141#	16-141#	16-142	16-142	16-142
	16-142	16-142#	16-142#	16-142#	16-143	16-143	16-143#	16-144	16-144	16-144	16-144	16-144#	16-144#	16-144#
	16-145	16-145	16-145	16-145	16-145#	16-145#	16-145#	16-146	16-146	16-146	16-146	16-146	16-146	16-146#
	16-146#	16-146#	16-146#	16-146#	16-147	16-147	16-147#	16-147#	16-147#	16-148	16-148	16-148	16-148	16-148
	16-148	16-148#	16-148#	16-148#	16-148#	16-148#	16-149	16-149	16-149	16-149	16-149#	16-149#	16-149#	16-150
	16-150	16-150	16-150	16-150	16-150	16-150#	16-150#	16-150#	16-150#	16-150#	16-151	16-151	16-151	16-151#
	16-151#	16-152	16-152	16-152	16-152	16-152	16-152#	16-152#	16-152#	16-152#	16-152#	16-152#	16-152#	16-160
	16-160	16-160	16-160#	16-160#	16-160#	16-161	16-161	16-161	16-161#	16-161#	16-161#	16-161#	16-162	16-162
	16-162#	16-163	16-163	16-163	16-163	16-163#	16-163#	16-163#	16-164	16-164	16-164	16-164	16-164#	16-164#
	16-164#	16-165	16-165	16-165	16-165	16-165#	16-165#	16-165#	16-166	16-166	16-166	16-166	16-166	16-166
	16-166#	16-166#	16-166#	16-166#	16-166#	16-167	16-167	16-167	16-167#	16-167#	16-167#	16-168	16-168	16-168
	16-168	16-168	16-168#	16-168#	16-168#	16-168#	16-168#	16-169	16-169	16-169	16-169	16-169	16-169#	16-169#
	16-170	16-170	16-170	16-170	16-170	16-170	16-170#	16-170#	16-170#	16-170#	16-170#	16-171	16-171	16-171
	16-171#	16-171#	16-172	16-172	16-172	16-172	16-172	16-172#	16-172#	16-172#	16-172#	16-172#	16-172#	16-180
	16-180	16-180	16-180#	16-180#	16-181	16-181	16-181	16-181#	16-181#	16-181#	16-181#	16-182	16-182	16-182#
	16-183	16-183	16-183	16-183	16-183#	16-183#	16-183#	16-184	16-184	16-184	16-184	16-184#	16-184#	16-184#
	16-185	16-185	16-185	16-185	16-185	16-185	16-185#	16-185#	16-185#	16-185#	16-185#	16-186	16-186	16-186
	16-186#	16-186#	16-187	16-187	16-187	16-187	16-187	16-187	16-187#	16-187#	16-187#	16-187#	16-187#	16-188
	16-188	16-188	16-188	16-188#	16-188#	16-188#	16-189	16-189	16-189	16-189	16-189	16-189	16-189#	16-189#
	16-189#	16-189#	16-189#	16-190	16-190	16-190	16-190#	16-190#	16-191	16-191	16-191	16-191	16-191	16-191
	16-191#	16-191#	16-191#	16-191#	16-191#	16-199	16-199	16-199	16-199	16-199#	16-199#	16-199#	16-200	16-200
	16-200#	16-201	16-201	16-201	16-201	16-201#	16-201#	16-201#	16-202	16-202	16-202	16-202	16-202#	16-202#
	16-202#	16-203	16-203	16-203	16-203	16-203	16-203	16-203#	16-203#	16-203#	16-203#	16-203#	16-204	16-204
	16-204	16-204#	16-204#	16-205	16-205	16-205	16-205	16-205	16-205	16-205#	16-205#	16-205#	16-205#	16-205#

	16-206	16-206	16-206	16-206	16-206#	16-206#	16-206#	16-207	16-207	16-207	16-207	16-207	16-207	16-207#
	16-207#	16-207#	16-207#	16-207#	16-208	16-208	16-208	16-208#	16-208#	16-209	16-209	16-209	16-209	16-209
	16-209	16-209#	16-209#	16-209#	16-209#	16-209#	16-217	16-217	16-217	16-217#	16-217#	16-218	16-218	16-218
	16-218	16-218#	16-218#	16-218#	16-219	16-219	16-219#	16-220	16-220	16-220	16-220	16-220#	16-220#	16-220#
	16-221	16-221	16-221	16-221	16-221#	16-221#	16-221#	16-222	16-222	16-222	16-222	16-222#	16-222#	16-222#
	16-223	16-223	16-223	16-223	16-223	16-223	16-223#	16-223#	16-223#	16-223#	16-223#	16-224	16-224	16-224
	16-224#	16-224#	16-225	16-225	16-225	16-225	16-225	16-225	16-225#	16-225#	16-225#	16-225#	16-225#	16-226
	16-226	16-226	16-226	16-226#	16-226#	16-226#	16-227	16-227	16-227	16-227	16-227	16-227	16-227#	16-227#
	16-227#	16-227#	16-227#	16-228	16-228	16-228	16-228#	16-228#	16-229	16-229	16-229	16-229	16-229	16-229
	16-229#	16-229#	16-229#	16-229#	16-229#	16-237	16-237	16-237	16-237	16-237#	16-237#	16-237#	16-238	16-238
	16-238#	16-239	16-239	16-239	16-239	16-239#	16-239#	16-239#	16-240	16-240	16-240	16-240	16-240#	16-240#
	16-240#	16-241	16-241	16-241	16-241	16-241	16-241	16-241#	16-241#	16-241#	16-241#	16-241#	16-242	16-242
	16-242	16-242#	16-242#	16-243	16-243	16-243	16-243	16-243	16-243	16-243#	16-243#	16-243#	16-243#	16-243#
	22-12	22-12	22-12	22-12#	22-12#	24-456	24-456	24-456	24-456#	24-456#	24-482	24-482	24-482#	24-501
	24-501	24-501#	24-520	24-520	24-520#									
T\$CODE	25-16	25-16	25-16	25-16#	25-16#	25-16#	25-17	25-17	25-17	25-17#	25-17#	25-17#	25-18	25-18
	25-18	25-18#	25-18#	25-18#	25-19	25-19	25-19	25-19#	25-19#	25-19#	25-19#	25-19#	25-18	25-18
T\$ERRN	7-18#	15-272	15-272#	15-278	15-278#	15-288	15-294	15-294#	15-401	15-401#	15-407	15-407#	15-565	15-565
	15-565#	15-597	15-597#	15-603	15-603#	15-613	15-619	15-619#	15-654	15-654#	15-660	15-660#	15-694	15-694
	15-694#	15-700	15-700#	15-708	15-708#	15-714	15-714#	15-860	15-860#	15-865	15-865#	15-884	15-884#	15-890
	15-890#	15-899	15-899#	15-905	15-905#	15-914	15-914#	15-920	15-920#	15-929	15-929#	15-935	15-935#	15-:67
	15-:67#	15-:73	15-:73#	24-24	24-24#	24-50	24-50#	24-84	24-84#	24-115	24-115#	24-148	24-148#	24-184
	24-184#	24-226	24-226#	24-278	24-278#	24-314	24-314#	24-348	24-348#	24-382	24-382#	24-416	24-416#	24-480
	24-480#	24-499	24-499#	24-518	24-518#	24-551	24-551#	24-565	24-565#	24-577	24-577#	24-588	24-588#	24-637
	24-637#	24-647	24-647#	24-694	24-694#	24-704	24-704#	24-745	24-745#	24-754	24-754#	24-792	24-792#	24-801
	24-801#	24-855	24-855#	24-865	24-865#	24-913	24-913#	24-936	24-936#	24-983	24-983#	24-993	24-993#	24-:98
	24-:98#	24-:21	24-:21#	24-<83	24-<83#	24-:26	24-:26#	24-:35	24-:35#	24-:47	24-:47#	24-:66	24-:66#	24-:74
	24-:74#	24-?97	24-?97#	24-a12	24-a12#	24-a95	24-a95#	24-A60	24-A60#	24-B41	24-B41#	24-B85	24-B85#	24-C89
	24-C89#													
T\$EXCP	25-16	25-16#	25-17	25-17#	25-18	25-18#	25-19	25-19#						
T\$FLAG	24-227	24-227#	24-227#	24-279	24-279#	24-279#	24-552	24-552#	24-552#	24-566	24-566#	24-566#	24-578	24-578#
	24-578#	24-589	24-589#	24-589#	24-638	24-638#	24-638#	24-648	24-648#	24-648#	24-695	24-695#	24-695#	24-746
	24-746#	24-746#	24-793	24-793#	24-793#	24-856	24-856#	24-856#	24-984	24-984#	24-984#	24-994	24-994#	24-994#
	24-:99	24-:99#	24-:99#	24-:22	24-:22#	24-:22#	24-:22#	24-:22#						
T\$GMAN	7-18#													
T\$HILI	25-16	25-16#	25-17	25-17#	25-18	25-18#	25-19	25-19#						
T\$LAST	7-18#	26-39#												
T\$LOLI	25-16	25-16#	25-17	25-17#	25-18	25-18#	25-19	25-19#						
T\$LSYM	7-18	7-18#	10-23	11-11	16-102	16-115	16-134	16-153	16-173	16-192	16-210	16-230	16-244	17-11
	19-74	20-19	21-11	22-13	23-10	24-27	24-52	24-90	24-117	24-150	24-186	24-232	24-286	24-320
	24-354	24-388	24-422	24-523	24-594	24-655	24-710	24-760	24-807	24-872	24-920	24-938	24-939	24-999
	24-:00	24-:32	24-:81	24-<27	24-<87	24-:76	24->32	24-?12	24-?47	24-a27	24-A10	24-A73	24-B56	24-B88
	24-D34	24-E35	24-E87	24-F55	24-F87	24-G40	24-G90	24-H60	25-21	26-16				
T\$LTNO	26-39#													
T\$NEST	7-18#	7-24	7-24	7-24#	10-9	10-9	10-9#	10-23	10-23	10-23	10-23#	11-8	11-8	11-8#
	11-11	11-11	11-11	11-11#	16-100	16-100	16-100#	16-102	16-102	16-102	16-102#	16-106	16-106	16-106#
	16-115	16-115	16-115	16-115#	16-121	16-121	16-121#	16-134	16-134	16-134	16-134#	16-140	16-140	16-140#
	16-153	16-153	16-153	16-153#	16-159	16-159	16-159#	16-173	16-173	16-173	16-173#	16-179	16-179	16-179#
	16-192	16-192	16-192	16-192#	16-198	16-198	16-198#	16-210	16-210	16-210	16-210#	16-216	16-216	16-216#
	16-230	16-230	16-230	16-230#	16-236	16-236	16-236#	16-244	16-244	16-244	16-244#	17-9	17-9	17-9#
	17-11	17-11	17-11	17-11#	18-8	18-8	18-8#	18-12	18-12	18-12	18-12#	19-8	19-8	19-8#
	19-74	19-74	19-74	19-74#	20-7	20-7	20-7#	20-19	20-19	20-19	20-19#	21-8	21-8	21-8#
	21-11	21-11	21-11	21-11#	22-8	22-8	22-8#	22-13	22-13	22-13	22-13#	23-9	23-9	23-9#
	23-10	23-10	23-10	23-10#	24-14	24-14	24-14#	24-27	24-27	24-27	24-27#	24-39	24-39	24-39#
	24-52	24-52	24-52	24-52#	24-67	24-67	24-67#	24-72	24-72	24-72#	24-86	24-86	24-86	24-86#
	24-90	24-90	24-90	24-90#	24-101	24-101	24-101#	24-117	24-117	24-117	24-117#	24-128	24-128	24-128#
	24-150	24-150	24-150	24-150#	24-166	24-166	24-166#	24-186	24-186	24-186	24-186#	24-203	24-203	24-203#

CROSS REFERENCE TABLE (CREF V01-05)

	24-232	24-232	24-232	24-232#	24-247	24-247	24-247#	24-286	24-286	24-286	24-286#	24-298	24-298	24-298#
	24-303	24-303	24-303#	24-316	24-316	24-316	24-316#	24-320	24-320	24-320	24-320#	24-332	24-332	24-332#
	24-337	24-337	24-337#	24-350	24-350	24-350	24-350#	24-354	24-354	24-354	24-354#	24-366	24-366	24-366#
	24-371	24-371	24-371#	24-384	24-384	24-384	24-384#	24-388	24-388	24-388	24-388#	24-400	24-400	24-400#
	24-405	24-405	24-405#	24-418	24-418	24-418	24-418#	24-422	24-422	24-422	24-422#	24-449	24-449	24-449#
	24-523	24-523	24-523	24-523#	24-539	24-539	24-539#	24-594	24-594	24-594	24-594#	24-610	24-610	24-610#
	24-655	24-655	24-655	24-655#	24-676	24-676	24-676#	24-680	24-680	24-680#	24-706	24-706	24-706	24-706#
	24-710	24-710	24-710	24-710#	24-727	24-727	24-727#	24-731	24-731	24-731#	24-756	24-756	24-756	24-756#
	24-760	24-760	24-760	24-760#	24-774	24-774	24-774#	24-778	24-778	24-778#	24-803	24-803	24-803	24-803#
	24-807	24-807	24-807	24-807#	24-832	24-832	24-832#	24-838	24-838	24-838#	24-867	24-867	24-867	24-867#
	24-872	24-872	24-872	24-872#	24-889	24-889	24-889#	24-898	24-898	24-898#	24-900	24-900	24-900#	24-915
	24-915	24-915	24-915#	24-920	24-920	24-920	24-920#	24-924	24-924	24-924#	24-938	24-938	24-938	24-938#
	24-939	24-939	24-939	24-939#	24-957	24-957	24-957#	24-960	24-960	24-960#	24-999	24-999	24-999	24-999#
	24-:00	24-:00	24-:00	24-:00#	24-:22	24-:22	24-:22#	24-:32	24-:32	24-:32	24-:32#	24-:52	24-:52	24-:52#
	24-:81	24-:81	24-:81	24-:81#	24-<01	24-<01	24-<01#	24-<27	24-<27	24-<27	24-<27#	24-<49	24-<49	24-<49#
	24-<87	24-<87	24-<87	24-<87#	24-=07	24-=07	24-=07#	24-=76	24-=76	24-=76	24-=76#	24-=91	24-=91	24-=91#
	24->32	24->32	24->32	24->32#	24->48	24->48	24->48#	24-?12	24-?12	24-?12	24-?12#	24-?27	24-?27	24-?27#
	24-?47	24-?47	24-?47	24-?47#	24-?63	24-?63	24-?63#	24-@27	24-@27	24-@27	24-@27#	24-@43	24-@43	24-@43#
	24-A10	24-A10	24-A10	24-A10#	24-A26	24-A26	24-A26#	24-A73	24-A73	24-A73	24-A73#	24-A89	24-A89	24-A89#
	24-B56	24-B56	24-B56	24-B56#	24-B69	24-B69	24-B69#	24-B88	24-B88	24-B88	24-B88#	24-C05	24-C05	24-C05#
	24-D34	24-D34	24-D34	24-D34#	24-D58	24-D58	24-D58#	24-E35	24-E35	24-E35	24-E35#	24-E53	24-E53	24-E53#
	24-E87	24-E87	24-E87	24-E87#	24-F05	24-F05	24-F05#	24-F55	24-F55	24-F55	24-F55#	24-F70	24-F70	24-F70#
	24-F87	24-F87	24-F87	24-F87#	24-G08	24-G08	24-G08#	24-G40	24-G40	24-G40	24-G40#	24-G59	24-G59	24-G59#
	24-G90	24-G90	24-G90	24-G90#	24-H13	24-H13	24-H13#	24-H60	24-H60	24-H60	24-H60#	25-14	25-14	25-14#
	25-21	25-21	25-21	25-21#	26-13	26-13	26-13#	26-16	26-16	26-16	26-16#	26-37	26-37	26-37
	26-37#													
T\$NSO	7-24#	26-37												
T\$NS1	10-9#	10-23	11-8#	11-11	16-100#	16-102	16-106#	16-115	16-121#	16-134	16-140#	16-153	16-159#	16-173
	16-179#	16-192	16-198#	16-210	16-216#	16-230	16-236#	16-244	17-9#	17-11	18-8#	18-12	19-8#	19-74
	20-7#	20-19	21-8#	21-11	22-8#	22-13	23-9#	23-10	24-14#	24-27	24-39#	24-52	24-67#	24-90
	24-101#	24-117	24-128#	24-150	24-166#	24-186	24-203#	24-232	24-247#	24-286	24-298#	24-320	24-332#	24-354
	24-366#	24-388	24-400#	24-422	24-449#	24-523	24-539#	24-594	24-610#	24-655	24-676#	24-710	24-727#	24-760
	24-774#	24-807	24-832#	24-872	24-889#	24-939	24-957#	24-:00	24-:22#	24-:32	24-:52#	24-:81	24-<01#	24-<27
	24-<49#	24-<87	24-=07#	24-=76	24-=91#	24->32	24->48#	24-?12	24-?27#	24-?47	24-?63#	24-@27	24-@43#	24-A10
	24-A26#	24-A73	24-A89#	24-B56	24-B69#	24-B88	24-C05#	24-D34	24-D58#	24-E35	24-E53#	24-E87	24-F05#	24-F55
	24-F70#	24-F87	24-G08#	24-G40	24-G59#	24-G90	24-H13#	24-H60	25-14#	25-21	26-13#	26-16		
T\$NS2	24-72#	24-86	24-303#	24-316	24-337#	24-350	24-371#	24-384	24-405#	24-418	24-680#	24-706	24-731#	24-756
	24-778#	24-803	24-838#	24-867	24-898#	24-920	24-924#	24-938	24-960#	24-999				
T\$NS3	24-900#	24-915												
T\$PTNU	7-18#													
T\$SAVL	7-18#													
T\$SEGL	7-18#	24-72	24-72	24-72#	24-86	24-86	24-86	24-86	24-86#	24-303	24-303	24-303#	24-316	24-316
	24-316	24-316	24-316#	24-337	24-337	24-337#	24-350	24-350	24-350	24-350	24-350#	24-371	24-371	24-371#
	24-384	24-384	24-384	24-384#	24-405	24-405	24-405#	24-418	24-418	24-418	24-418	24-418	24-418#	24-680
	24-680	24-680#	24-695	24-706	24-706	24-706	24-706#	24-731	24-731	24-731	24-731#	24-746	24-756	24-756
	24-756	24-756	24-756#	24-778	24-778	24-778#	24-793	24-803	24-803	24-803	24-803	24-803#	24-838	24-838
	24-838#	24-856	24-867	24-867	24-867	24-867	24-867#	24-900	24-900	24-900#	24-915	24-915	24-915	24-915
	24-915#													
T\$SEKO	24-72#	24-86	24-303#	24-316	24-337#	24-350	24-371#	24-384	24-405#	24-418	24-680#	24-695	24-706	24-731#
	24-746	24-756	24-778#	24-793	24-803	24-838#	24-856	24-867	24-900#	24-915				
T\$SUBN	7-18#	24-14#	24-39#	24-67#	24-101#	24-128#	24-166#	24-203#	24-247#	24-298#	24-332#	24-366#	24-400#	24-449#
	24-539#	24-610#	24-676#	24-727#	24-774#	24-832#	24-889#	24-898	24-898	24-898#	24-924	24-924	24-924#	24-957#
	24-960	24-960	24-960#	24-:22#	24-:52#	24-<01#	24-<49#	24-=07#	24-=91#	24->48#	24-?27#	24-?63#	24-@43#	24-A26#
	24-A89#	24-B69#	24-C05#	24-D58#	24-E53#	24-F05#	24-F70#	24-G08#	24-G59#	24-H13#				
T\$TAGL	7-18#													
T\$TAGN	7-18#	10-9	10-9	10-9#	11-8	11-8	11-8#	16-100	16-100	16-100#	16-106	16-106	16-106#	16-121
	16-121	16-121#	16-140	16-140	16-140#	16-159	16-159	16-159#	16-179	16-179	16-179#	16-198	16-198	16-198#

	16-216	16-216	16-216#	16-236	16-236	16-236#	17-9	17-9	17-9#	18-8	18-8	18-8#	19-8	19-8
	19-8#	20-7	20-7	20-7#	21-8	21-8	21-8#	22-8	22-8	22-8#	23-9	23-9	23-9#	24-14
	24-14	24-14#	24-39	24-39	24-39#	24-67	24-67	24-67#	24-101	24-101	24-101#	24-128	24-128	24-128#
	24-166	24-166	24-166#	24-203	24-203	24-203#	24-247	24-247	24-247#	24-298	24-298	24-298#	24-332	24-332
	24-332#	24-366	24-366	24-366#	24-400	24-400	24-400#	24-449	24-449	24-449#	24-539	24-539	24-539#	24-610
	24-610	24-610#	24-676	24-676	24-676#	24-727	24-727	24-727#	24-774	24-774	24-774#	24-832	24-832	24-832#
	24-889	24-889	24-889#	24-898	24-898	24-898#	24-924	24-924	24-924#	24-957	24-957	24-957#	24-960	24-960
	24-960#	24-:22	24-:22	24-:22#	24-:52	24-:52	24-:52#	24-<01	24-<01	24-<01#	24-<49	24-<49	24-<49#	24-:07
	24-:07	24-:07#	24-:91	24-:91	24-:91#	24->48	24->48	24->48#	24-?27	24-?27	24-?27#	24-?63	24-?63	24-?63#
	24-@43	24-@43	24-@43#	24-A26	24-A26	24-A26#	24-A89	24-A89	24-A89#	24-B69	24-B69	24-B69#	24-C05	24-C05
	24-C05#	24-D58	24-D58	24-D58#	24-E53	24-E53	24-E53#	24-F05	24-F05	24-F05#	24-F70	24-F70	24-F70#	24-G08
	24-G08	24-G08#	24-G59	24-G59	24-G59#	24-H13	24-H13	24-H13#	25-14	25-14	25-14#	26-13	26-13	26-13#
T\$TEMP	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#
	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#
	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#	9-8#
	9-8#	10-23	10-23#	11-11	11-11#	16-102	16-102#	16-115	16-115#	16-134	16-134#	16-153	16-153#	16-173
	16-173#	16-192	16-192#	16-210	16-210#	16-230	16-230#	16-244	16-244#	17-11	17-11#	18-12	18-12#	19-74
	19-74#	20-19	20-19#	21-11	21-11#	22-13	22-13#	23-10	23-10#	24-27	24-27#	24-52	24-52#	24-86
	24-86#	24-90	24-90#	24-117	24-117#	24-150	24-150#	24-186	24-186#	24-227	24-227#	24-232	24-232#	24-279
	24-279#	24-286	24-286#	24-316	24-316#	24-320	24-320#	24-350	24-350#	24-354	24-354#	24-384	24-384#	24-388
	24-388#	24-418	24-418#	24-422	24-422#	24-523	24-523#	24-552	24-552#	24-566	24-566#	24-578	24-578#	24-589
	24-589#	24-594	24-594#	24-638	24-638#	24-648	24-648#	24-655	24-655#	24-695	24-695#	24-695#	24-706	24-706#
	24-710	24-710#	24-746	24-746#	24-746#	24-756	24-756#	24-760	24-760#	24-793	24-793#	24-793#	24-803	24-803#
	24-807	24-807#	24-856	24-856#	24-856#	24-867	24-867#	24-872	24-872#	24-915	24-915#	24-920	24-920#	24-938
	24-938#	24-939	24-939#	24-984	24-984#	24-994	24-994#	24-999	24-999#	24-:00	24-:00#	24-:99	24-:99#	24-:22
	24-:22#	24-:32	24-:32#	24-:81	24-:81#	24-<27	24-<27#	24-<87	24-<87#	24-:76	24-:76#	24->32	24->32#	24-?12
	24-?12#	24-?47	24-?47#	24-@27	24-@27#	24-A10	24-A10#	24-A73	24-A73#	24-B56	24-B56#	24-B88	24-B88#	24-D34
	24-D34#	24-E35	24-E35#	24-E87	24-E87#	24-F55	24-F55#	24-F87	24-F87#	24-G40	24-G40#	24-G90	24-G90#	24-H60
	24-H60#	25-16	25-16	25-16	25-16#	25-16#	25-16#	25-17	25-17	25-17	25-17#	25-17#	25-17#	25-18
	25-18	25-18	25-18#	25-18#	25-18#	25-19	25-19	25-19	25-19#	25-19#	25-19#	25-21	25-21#	26-16
	26-16#	26-37	26-37#											
T\$TEST	7-18#	24-14	24-14	24-14#	24-39	24-39	24-39#	24-67	24-67	24-67#	24-101	24-101	24-101#	24-128
	24-128	24-128#	24-166	24-166	24-166#	24-203	24-203	24-203#	24-247	24-247	24-247#	24-298	24-298	24-298#
	24-332	24-332	24-332#	24-366	24-366	24-366#	24-400	24-400	24-400#	24-449	24-449	24-449#	24-539	24-539
	24-539#	24-610	24-610	24-610#	24-676	24-676	24-676#	24-727	24-727	24-727#	24-774	24-774	24-774#	24-832
	24-832	24-832#	24-889	24-889	24-889#	24-898	24-898	24-924	24-924	24-957	24-957#	24-960	24-:22	24-:22#
	24-:52	24-:52	24-:52#	24-<01	24-<01	24-<01#	24-<49	24-<49	24-<49#	24-:07	24-:07#	24-:07#	24-:91	24-:91
	24-:91#	24->48	24->48	24->48#	24-?27	24-?27	24-?27#	24-?63	24-?63	24-?63#	24-@43	24-@43	24-@43#	24-A26
	24-A26	24-A26#	24-A89	24-A89	24-A89#	24-B69	24-B69	24-B69#	24-C05	24-C05	24-C05#	24-D58	24-D58	24-D58#
	24-E53	24-E53	24-E53#	24-F05	24-F05	24-F05#	24-F70	24-F70	24-F70#	24-F70#	24-G08	24-G08	24-G08#	24-G59
	24-G59#	24-H13	24-H13	24-H13#	26-39									
T\$STM	7-18#	15-272	15-278	15-288	15-294	15-401	15-407	15-565	15-597	15-603	15-613	15-619	15-654	15-660
	15-694	15-700	15-708	15-714	15-860	15-865	15-884	15-890	15-899	15-905	15-914	15-920	15-929	15-935
	15-:67	15-:73	16-101	16-102	16-107	16-108	16-109	16-110	16-111	16-112	16-113	16-114	16-115	16-122
	16-123	16-124	16-125	16-126	16-127	16-128	16-129	16-130	16-131	16-132	16-133	16-134	16-141	16-142
	16-143	16-144	16-145	16-146	16-147	16-148	16-149	16-150	16-151	16-152	16-153	16-160	16-161	16-162
	16-163	16-164	16-165	16-166	16-167	16-168	16-169	16-170	16-171	16-172	16-173	16-180	16-181	16-182
	16-183	16-184	16-185	16-186	16-187	16-188	16-189	16-190	16-191	16-192	16-199	16-200	16-201	16-202
	16-203	16-204	16-205	16-206	16-207	16-208	16-209	16-210	16-217	16-218	16-219	16-220	16-221	16-222
	16-223	16-224	16-225	16-226	16-227	16-228	16-229	16-230	16-237	16-238	16-239	16-240	16-241	16-242
	16-243	16-244	17-11	19-25	19-28	19-31	19-34	19-51	19-74	20-9	20-16	20-19	21-11	22-10

ENDSFT	1-568#	7-18#	26-16											
ENDSRV	1-580#	7-18#												
ENDSUB	1-596#	7-18#	24-920	24-938	24-999									
ENDSW	1-614#	7-18#	11-11											
ENDTST	1-624#	7-18#	24-27	24-52	24-90	24-117	24-150	24-186	24-232	24-286	24-320	24-354	24-388	24-422
	24-523	24-594	24-655	24-710	24-760	24-807	24-872	24-939	24-:00	24-:32	24-:81	24-<27	24-<87	24-=76
	24->32	24-?12	24-?47	24-a27	24-A10	24-A73	24-B56	24-B88	24-D34	24-E35	24-E87	24-F55	24-F87	24-G40
	24-G90	24-H60												
EQUALS	1-642#	7-18#	12-20											
ERRDF	1-714#	7-18#	15-272	15-278	15-288	15-294	15-401	15-407	15-565	15-597	15-603	15-613	15-619	15-654
	15-660	15-694	15-700	15-708	15-714	15-860	15-865	15-884	15-890	15-899	15-905	15-914	15-920	15-929
	15-935	15-:67	15-:73	24-24	24-50	24-84	24-115	24-148	24-184	24-226	24-278	24-314	24-348	24-382
	24-416	24-480	24-499	24-518	24-551	24-565	24-577	24-588	24-637	24-647	24-694	24-704	24-745	24-754
	24-792	24-801	24-855	24-865	24-913	24-936	24-983	24-993	24-:98	24-;21	24-<83	24-=26	24-=35	24-=47
	24-=66	24-=74	24-?97	24-a12	24-a95	24-A60	24-B41	24-B85	24-C89					
ERRHRD	1-718#	7-18#												
ERROR	1-722#	7-18#												
ERRSF	1-726#	7-18#												
ERRSOF	1-730#	7-18#												
ERRTBL	1-734#	7-18#												
ESCAPE	1-744#	7-18#	24-227	24-279	24-552	24-566	24-578	24-589	24-638	24-648	24-695	24-746	24-793	24-856
	24-984	24-994	24-:99	24-;22										
EXIT	1-771#	7-18#												
FEQUAL	1-810#	7-18#												
GETBYT	1-824#	7-18#												
GETPRI	1-834#	7-18#												
GETWOR	1-829#	7-18#												
GMANIA	1-839#	7-18#												
GMANID	1-848#	7-18#												
GMANIL	1-859#	7-18#												
GPHARD	1-868#	7-18#	19-51											
GPRMA	1-874#	7-18#	25-16	25-17										
GPRMD	1-903#	7-18#	25-18	25-19										
GPRML	1-934#	7-18#												
HEADER	1-954#	7-18#	8-17											
INLOOP	1-962#	7-18#												
IOSETU	1-966#	7-18#												
IOSTAR	1-974#	7-18#												
KT11	1-982#	7-18#												
LASTAD	1-:47#	7-18#	26-39											
MSBYTE	1-D00#	7-18#	8-17	8-17	8-17	8-17#								
MSCHEC	1-E18#	7-18#												
MSCNTO	1-E82#	7-18#	25-16	25-16#	25-17	25-17#	25-18	25-18#	25-19	25-19#				
MSCOUN	1-D66#	7-18#	16-101	16-101	16-101#	16-107	16-107	16-107#	16-108	16-108#	16-109	16-109	16-109#	16-110
	16-110	16-110#	16-111	16-111	16-111#	16-112	16-112	16-112#	16-112	16-112#	16-113	16-113#	16-114	16-114
	16-114	16-114	16-114#	16-122	16-122	16-122#	16-123	16-123#	16-124	16-124	16-124#	16-125	16-125	16-125#
	16-126	16-126	16-126#	16-127	16-127	16-127	16-127	16-127#	16-128	16-128#	16-129	16-129	16-129	16-129
	16-129#	16-130	16-130	16-130#	16-131	16-131	16-131	16-131#	16-131	16-131#	16-132	16-132#	16-133	16-133
	16-133	16-133#	16-141	16-141#	16-142	16-142	16-142#	16-143	16-143#	16-144	16-144	16-144#	16-145	16-145
	16-145#	16-146	16-146	16-146	16-146	16-146#	16-147	16-147#	16-148	16-148	16-148	16-148	16-148#	16-149
	16-149	16-149#	16-150	16-150	16-150	16-150#	16-151	16-151#	16-152	16-152	16-152	16-152	16-152#	16-152#
	16-160	16-160	16-160#	16-161	16-161	16-161#	16-162	16-162#	16-163	16-163	16-163#	16-164	16-164	16-164#
	16-165	16-165	16-165#	16-166	16-166	16-166	16-166	16-166#	16-167	16-167#	16-168	16-168	16-168	16-168
	16-168#	16-169	16-169	16-169#	16-170	16-170	16-170	16-170#	16-171	16-171#	16-172	16-172	16-172	16-172
	16-172	16-172#	16-180	16-180#	16-181	16-181	16-181#	16-182	16-182#	16-183	16-183	16-183#	16-184	16-184
	16-184#	16-185	16-185	16-185	16-185	16-185#	16-186	16-186#	16-187	16-187	16-187	16-187	16-187#	16-188
	16-188	16-188#	16-189	16-189	16-189	16-189	16-189#	16-190	16-190#	16-191	16-191	16-191	16-191	16-191#

19-34#	19-34#	19-35	19-35#	19-51	19-51	19-51	19-51#	19-51#	19-51#	19-51#	19-52	19-52#	19-74	19-74#
20-9	20-9	20-9#	20-9#	20-16	20-16	20-16#	20-16#	20-19	20-19#	20-19#	21-11	21-11#	22-10	22-10#
22-i2	22-12	22-12	22-12	22-12	22-12	22-12#	22-12#	22-12#	22-12#	22-12#	22-12#	22-13	22-13#	23-10
23-10#	24-16	24-16	24-16#	24-16#	24-24	24-24	24-24	24-24	24-24#	24-24#	24-24#	24-24#	24-24#	24-24#
24-27	24-27#	24-50	24-50	24-50	24-50	24-50#	24-50#	24-50#	24-50#	24-50#	24-50#	24-52	24-52#	24-72
24-72#	24-84	24-84	24-84	24-84	24-84#	24-84#	24-84#	24-84#	24-84#	24-84#	24-86	24-86#	24-90	24-90#
24-115	24-115	24-115	24-115	24-115#	24-115#	24-115#	24-115#	24-115#	24-117	24-117#	24-117#	24-133	24-133#	24-148
24-148	24-148	24-148	24-148#	24-148#	24-148#	24-148#	24-148#	24-150	24-150#	24-150#	24-184	24-184	24-184	24-184
24-184#	24-184#	24-184#	24-184#	24-184#	24-186	24-186#	24-186#	24-226	24-226	24-226	24-226	24-226#	24-226#	24-226#
24-226#	24-226#	24-227	24-227	24-227#	24-227#	24-227#	24-227#	24-232	24-232#	24-278	24-278	24-278	24-278#	24-278#
24-278#	24-278#	24-278#	24-279	24-279	24-279#	24-279#	24-279#	24-286	24-286#	24-303	24-303#	24-314	24-314	24-314
24-314	24-314#	24-314#	24-314#	24-314#	24-314#	24-316	24-316#	24-320	24-320#	24-320#	24-337	24-337#	24-348	24-348
24-348	24-348	24-348#	24-348#	24-348#	24-348#	24-348#	24-348#	24-350	24-350#	24-354	24-354#	24-371	24-371#	24-382
24-382	24-382	24-382	24-382#	24-382#	24-382#	24-382#	24-382#	24-384	24-384#	24-388	24-388#	24-405	24-405#	24-405#
24-416	24-416	24-416	24-416	24-416#	24-416#	24-416#	24-416#	24-416#	24-418	24-418#	24-422	24-422#	24-456	24-456
24-456	24-456	24-456	24-456	24-456	24-456#	24-456#	24-456#	24-456#	24-456#	24-456#	24-480	24-480	24-480	24-480
24-480#	24-480#	24-480#	24-480#	24-480#	24-482	24-482	24-482	24-482	24-482	24-482#	24-482#	24-482#	24-482#	24-482#
24-499	24-499	24-499	24-499	24-499#	24-499#	24-499#	24-499#	24-499#	24-499#	24-501	24-501	24-501	24-501	24-501
24-501#	24-501#	24-501#	24-501#	24-518	24-518	24-518	24-518	24-518	24-518#	24-518#	24-518#	24-518#	24-518#	24-520
24-520	24-520	24-520	24-520	24-520#	24-520#	24-520#	24-520#	24-523	24-523#	24-523#	24-551	24-551	24-551	24-551
24-551#	24-551#	24-551#	24-551#	24-551#	24-552	24-552	24-552	24-552	24-552#	24-552#	24-565	24-565	24-565	24-565#
24-565#	24-565#	24-565#	24-565#	24-566	24-566	24-566#	24-566#	24-566#	24-566#	24-577	24-577	24-577	24-577#	24-577#
24-577#	24-577#	24-577#	24-578	24-578	24-578#	24-578#	24-578#	24-588	24-588	24-588	24-588	24-588#	24-588#	24-588#
24-588#	24-588#	24-589	24-589	24-589#	24-589#	24-589#	24-589#	24-594	24-594#	24-637	24-637	24-637	24-637#	24-637#
24-637#	24-637#	24-637#	24-638	24-638	24-638#	24-638#	24-638#	24-647	24-647	24-647	24-647	24-647#	24-647#	24-647#
24-647#	24-647#	24-648	24-648	24-648#	24-648#	24-648#	24-648#	24-655	24-655#	24-680	24-680#	24-694	24-694	24-694
24-694#	24-694#	24-694#	24-694#	24-694#	24-695	24-695	24-695#	24-695#	24-695#	24-704	24-704	24-704	24-704	24-704#
24-704#	24-704#	24-704#	24-704#	24-706	24-706#	24-710	24-710#	24-731	24-731#	24-745	24-745	24-745	24-745	24-745
24-745#	24-745#	24-745#	24-745#	24-745#	24-746	24-746	24-746#	24-746#	24-746#	24-754	24-754	24-754	24-754	24-754#
24-754#	24-754#	24-754#	24-754#	24-756	24-756#	24-760	24-760#	24-778	24-778#	24-792	24-792	24-792	24-792	24-792
24-792#	24-792#	24-792#	24-792#	24-792#	24-793	24-793	24-793#	24-793#	24-793#	24-801	24-801	24-801	24-801	24-801#
24-801#	24-801#	24-801#	24-801#	24-803	24-803#	24-807	24-807#	24-838	24-838#	24-855	24-855	24-855	24-855	24-855
24-855#	24-855#	24-855#	24-855#	24-855#	24-856	24-856	24-856#	24-856#	24-856#	24-865	24-865	24-865	24-865	24-865#
24-865#	24-865#	24-865#	24-865#	24-867	24-867#	24-872	24-872#	24-898	24-898#	24-900	24-900#	24-913	24-913	24-913
24-913	24-913	24-913#	24-913#	24-913#	24-913#	24-913#	24-913#	24-915	24-915#	24-920	24-920#	24-924	24-924#	24-936
24-936	24-936	24-936	24-936#	24-936#	24-936#	24-936#	24-936#	24-936#	24-936#	24-938	24-938#	24-939	24-939#	24-960
24-983	24-983	24-983	24-983#	24-983#	24-983#	24-983#	24-983#	24-983#	24-983#	24-984	24-984	24-984#	24-984#	24-993
24-993	24-993	24-993	24-993#	24-993#	24-993#	24-993#	24-993#	24-993#	24-993#	24-994	24-994#	24-994#	24-994#	24-999
24-:00	24-:00#	24-:98	24-:98	24-:98	24-:98	24-:98#	24-:98#	24-:98#	24-:98#	24-:98#	24-:98#	24-:99	24-:99	24-:99#
24-:99#	24-:21	24-:21	24-:21	24-:21	24-:21#	24-:21#	24-:21#	24-:21#	24-:21#	24-:21#	24-:22	24-:22	24-:22#	24-:22#
24-:32	24-:32#	24-:81	24-:81#	24-:81#	24-:81#	24-:83	24-:83	24-:83	24-:83	24-:83	24-:83#	24-:83#	24-:83#	24-:83#
24-<83#	24-<87	24-<87#	24-==26	24-==26	24-==26	24-==26	24-==26#	24-==26#	24-==26#	24-==26#	24-==26#	24-==26#	24-==35	24-==35
24-==35	24-==35	24-==35#	24-==35#	24-==35#	24-==35#	24-==35#	24-==35#	24-==47	24-==47	24-==47	24-==47	24-==47#	24-==47#	24-==47#
24-==47#	24-==47#	24-==66	24-==66	24-==66	24-==66#	24-==66#	24-==66#	24-==66#	24-==66#	24-==66#	24-==66#	24-==74	24-==74	24-==74
24-==74	24-==74#	24-==74#	24-==74#	24-==74#	24-==74#	24-==74#	24-==74#	24-==76	24-==76#	24-==76#	24-?12	24-?12#	24-?47	24-?47#
24-?97	24-?97	24-?97	24-?97	24-?97#	24-?97#	24-?97#	24-?97#	24-?97#	24-?97#	24-?97#	24-@12	24-@12	24-@12	24-@12#
24-@12#	24-@12#	24-@12#	24-@12#	24-@27	24-@27#	24-@95	24-@95	24-@95	24-@95	24-@95	24-@95#	24-@95#	24-@95#	24-@95#
24-@95#	24-A10	24-A10#	24-A60	24-A60	24-A60	24-A60	24-A60#	24-A60#	24-A60#	24-A60#	24-A60#	24-A60#	24-A73	24-A73#
24-B41	24-B41	24-B41	24-B41	24-B41#	24-B41#	24-B41#	24-B41#	24-B41#	24-B41#	24-B56	24-B56#	24-B85	24-B85	24-B85
24-B85	24-B85#	24-B85#	24-B85#	24-B85#	24-B85#	24-B88	24-B88#	24-B88#	24-B88#	24-C89	24-C89	24-C89	24-C89#	24-C89#
24-C89#	24-C89#	24-C89#	24-D34	24-D34#	24-E35	24-E35#	24-E87	24-E87#	24-E87#	24-F55	24-F55#	24-F87	24-F87#	24-G40
24-G40#	24-G90	24-G90#	24-H60	24-H60#	25-14	25-14#	25-16	25-16	25-16	25-16	25-16	25-16#	25-17	25-17
25-17	25-17	25-17#	25-18	25-18	25-18	25-18	25-18	25-18	25-18#	25-19	25-19	25-19	25-19	25-19
25-19#	25-21	25-21#	26-13	26-13#	26-16	26-16#	26-39	26-39	26-39	26-39	26-39#	26-39#	26-39#	26-39#
M\$GNLS	1-C13#	7-18#	24-86	24-86#	24-316	24-316#	24-350	24-350#	24-350#	24-384	24-384#	24-418	24-418#	24-706#
M\$GNSU	24-756	24-756#	24-803	24-803#	24-867	24-867#	24-915	24-915#	24-915#					
	1-B98#	7-18#	24-898	24-898#	24-924	24-924#	24-960	24-960#	24-960#					

MSGNTA	1-B90#	7-18#	10-23	10-23#	11-11	11-11#	16-102	16-102#	16-115	16-115#	16-134	16-134#	16-153	16-153#
	16-173	16-173#	16-192	16-192#	16-210	16-210#	16-230	16-230#	16-244	16-244#	17-11	17-11#	19-74	19-74#
	20-19	20-19#	21-11	21-11#	22-13	22-13#	23-10	23-10#	24-27	24-27#	24-52	24-52#	24-90	24-90#
	24-117	24-117#	24-150	24-150#	24-186	24-186#	24-232	24-232#	24-286	24-286#	24-320	24-320#	24-354	24-354#
	24-388	24-388#	24-422	24-422#	24-523	24-523#	24-594	24-594#	24-655	24-655#	24-710	24-710#	24-760	24-760#
	24-807	24-807#	24-872	24-872#	24-920	24-920#	24-938	24-938#	24-939	24-939#	24-999	24-999#	24-:00	24-:00#
	24-:32	24-:32#	24-:81	24-:81#	24-<27	24-<27#	24-<87	24-<87#	24-=76	24-=76#	24->32	24->32#	24-?12	24-?12#
	24-?47	24-?47#	24-@27	24-@27#	24-A10	24-A10#	24-A73	24-A73#	24-B56	24-B56#	24-B88	24-B88#	24-D34	24-D34#
	24-E35	24-E35#	24-E87	24-E87#	24-F55	24-F55#	24-F87	24-F87#	24-G40	24-G40#	24-G90	24-G90#	24-H60	24-H60#
	25-21	25-21#	26-16	26-16#										
MSGNTE	1-B94#	7-18#	24-14	24-14#	24-39	24-39#	24-67	24-67#	24-101	24-101#	24-128	24-128#	24-166	24-166#
	24-203	24-203#	24-247	24-247#	24-298	24-298#	24-332	24-332#	24-366	24-366#	24-400	24-400#	24-449	24-449#
	24-539	24-539#	24-610	24-610#	24-676	24-676#	24-727	24-727#	24-774	24-774#	24-832	24-832#	24-889	24-889#
	24-957	24-957#	24-:22	24-:22#	24-:52	24-:52#	24-<01	24-<01#	24-<49	24-<49#	24-=07	24-=07#	24-=91	24-=91#
	24->48	24->48#	24-?27	24-?27#	24-?63	24-?63#	24-@43	24-@43#	24-A26	24-A26#	24-A89	24-A89#	24-B69	24-B69#
	24-C05	24-C05#	24-D58	24-D58#	24-E53	24-E53#	24-F05	24-F05#	24-F70	24-F70#	24-G08	24-G08#	24-G59	24-G59#
	24-H13	24-H13#												
MSHAPT	1-A39#	7-18#	8-17	8-17#										
MSHNAP	1-B24#	7-18#	8-17	8-17#										
MSINCR	1-D26#	7-18#	7-24	7-24#	10-9	10-9	10-9#	10-9#	11-8	11-8	11-8#	11-8#	15-272#	15-278#
	15-288#	15-294#	15-401#	15-407#	15-565#	15-597#	15-603#	15-613#	15-619#	15-654#	15-660#	15-694#	15-700#	15-708#
	15-714#	15-860#	15-865#	15-884#	15-890#	15-899#	15-905#	15-914#	15-920#	15-929#	15-935#	15-:67#	15-:73#	16-100
	16-100	16-100#	16-100#	16-101#	16-102#	16-106	16-106	16-106#	16-106#	16-107#	16-108#	16-109#	16-110#	16-111#
	16-112#	16-113#	16-114#	16-115#	16-121	16-121	16-121#	16-121#	16-122#	16-123#	16-124#	16-125#	16-126#	16-127#
	16-128#	16-129#	16-130#	16-131#	16-132#	16-133#	16-134#	16-140	16-140	16-140#	16-140#	16-141#	16-142#	16-143#
	16-144#	16-145#	16-146#	16-147#	16-148#	16-149#	16-150#	16-151#	16-152#	16-153#	16-159	16-159	16-159#	16-159#
	16-160#	16-161#	16-162#	16-163#	16-164#	16-165#	16-166#	16-167#	16-168#	16-169#	16-170#	16-171#	16-172#	16-173#
	16-179	16-179	16-179#	16-179#	16-180#	16-181#	16-182#	16-183#	16-184#	16-185#	16-186#	16-187#	16-188#	16-189#
	16-190#	16-191#	16-192#	16-198	16-198	16-198#	16-198#	16-199#	16-200#	16-201#	16-202#	16-203#	16-204#	16-205#
	16-206#	16-207#	16-208#	16-209#	16-210#	16-216	16-216	16-216#	16-216#	16-217#	16-218#	16-219#	16-220#	16-221#
	16-222#	16-223#	16-224#	16-225#	16-226#	16-227#	16-228#	16-229#	16-230#	16-236	16-236	16-236#	16-236#	16-237#
	16-238#	16-239#	16-240#	16-241#	16-242#	16-243#	16-244#	17-9	17-9	17-9#	17-9#	17-11#	18-8	18-8
	18-8#	18-8#	19-8	19-8	19-8#	19-8#	19-25#	19-28#	19-31#	19-34#	19-51#	19-74#	20-7	20-7
	20-7#	20-7#	20-9#	20-16#	20-19#	21-8	21-8	21-8#	21-8#	21-11#	22-8	22-8	22-8#	22-8#
	22-10#	22-12#	22-13#	23-9	23-9	23-9#	23-9#	23-10#	24-14	24-14	24-14	24-14#	24-14#	24-14#
	24-16#	24-24#	24-27#	24-39	24-39	24-39	24-39#	24-39#	24-39#	24-50#	24-52#	24-67	24-67	24-67
	24-67#	24-67#	24-67#	24-72	24-72	24-72	24-72#	24-72#	24-72#	24-72#	24-84#	24-86#	24-90#	24-101
	24-101	24-101	24-101#	24-101#	24-101#	24-115#	24-117#	24-128	24-128	24-128	24-128#	24-128#	24-128#	24-133#
	24-148#	24-150#	24-166	24-166	24-166	24-166#	24-166#	24-184#	24-186#	24-186#	24-203	24-203	24-203	24-203#
	24-203#	24-203#	24-226#	24-227#	24-232#	24-247	24-247	24-247#	24-247#	24-247#	24-247#	24-247#	24-278#	24-279#
	24-298	24-298	24-298	24-298#	24-298#	24-298#	24-303	24-303	24-303	24-303#	24-303#	24-303#	24-303#	24-314#
	24-316#	24-320#	24-332	24-332	24-332	24-332#	24-332#	24-332#	24-332#	24-337	24-337	24-337#	24-337#	24-337#
	24-337#	24-348#	24-350#	24-354#	24-366	24-366	24-366	24-366#	24-366#	24-366#	24-366#	24-371	24-371	24-371#
	24-371#	24-371#	24-371#	24-382#	24-384#	24-388#	24-400	24-400	24-400	24-400#	24-400#	24-400#	24-405	24-405
	24-405	24-405#	24-405#	24-405#	24-405#	24-416#	24-418#	24-422#	24-449	24-449	24-449	24-449#	24-449#	24-449#
	24-456#	24-480#	24-482#	24-499#	24-501#	24-518#	24-520#	24-523#	24-539	24-539	24-539	24-539#	24-539#	24-539#
	24-551#	24-552#	24-565#	24-566#	24-577#	24-578#	24-588#	24-589#	24-594#	24-610	24-610	24-610	24-610#	24-610#
	24-610#	24-637#	24-638#	24-647#	24-648#	24-655#	24-676	24-676	24-676	24-676#	24-676#	24-676#	24-680	24-680
	24-680	24-680#	24-680#	24-680#	24-680#	24-694#	24-695#	24-704#	24-706#	24-710#	24-727	24-727	24-727	24-727#
	24-727#	24-727#	24-731	24-731	24-731	24-731#	24-731#	24-731#	24-731#	24-745#	24-746#	24-754#	24-756#	24-760#
	24-774	24-774	24-774	24-774#	24-774#	24-774#	24-778	24-778	24-778	24-778#	24-778#	24-778#	24-778#	24-792#
	24-793#	24-801#	24-803#	24-807#	24-832	24-832	24-832	24-832#	24-832#	24-832#	24-838	24-838	24-838	24-838#
	24-838#	24-838#	24-838#	24-855#	24-856#	24-865#	24-867#	24-872#	24-889	24-889	24-889	24-889#	24-889#	24-889#
	24-898	24-898	24-898	24-898#	24-898#	24-898#	24-900	24-900	24-900	24-900#	24-900#	24-900#	24-900#	24-913#
	24-915#	24-920#	24-924	24-924	24-924	24-924#	24-924#	24-924#	24-936#	24-938#	24-939#	24-957	24-957	24-957
	24-957#	24-957#	24-957#	24-960	24-960	24-960	24-960#	24-960#	24-960#	24-983#	24-984#	24-993#	24-994#	24-999#
	24-:00#	24-:22	24-:22	24-:22	24-:22#	24-:22#	24-:22#	24-:22#	24-:22#	24-:21#	24-:22#	24-:32#	24-:52	24-:52

16-163#	16-163#	16-164	16-164	16-164	16-164	16-164#	16-164#	16-164#	16-164#	16-165	16-165	16-165	16-165
16-165#	16-165#	16-165#	16-165#	16-166	16-166	16-166	16-166	16-166	16-166	16-166#	16-166#	16-166#	16-166#
16-166#	16-166#	16-167	16-167	16-167	16-167#	16-167#	16-167#	16-168	16-168	16-168	16-168	16-168	16-168
16-168#	16-168#	16-168#	16-168#	16-168#	16-168#	16-169	16-169	16-169	16-169	16-169#	16-169#	16-169#	16-169#
16-170	16-170	16-170	16-170	16-170	16-170	16-170#	16-170#	16-170#	16-170#	16-170#	16-170#	16-170#	16-171
16-171	16-171#	16-171#	16-171#	16-172	16-172	16-172	16-172	16-172	16-172	16-172#	16-172#	16-172#	16-172#
16-172#	16-172#	16-180	16-180	16-180	16-180#	16-180#	16-180#	16-181	16-181	16-181	16-181	16-181#	16-181#
16-181#	16-181#	16-182	16-182	16-182#	16-182#	16-183	16-183	16-183	16-183	16-183#	16-183#	16-183#	16-183#
16-184	16-184	16-184	16-184	16-184#	16-184#	16-184#	16-184#	16-185	16-185	16-185	16-185	16-185	16-185
16-185#	16-185#	16-185#	16-185#	16-185#	16-185#	16-186	16-186	16-186	16-186#	16-186#	16-186#	16-187	16-187
16-187	16-187	16-187	16-187	16-187#	16-187#	16-187#	16-187#	16-187#	16-187#	16-188	16-188	16-188	16-188
16-188#	16-188#	16-188#	16-188#	16-189	16-189	16-189	16-189	16-189	16-189	16-189#	16-189#	16-189#	16-189#
16-189#	16-189#	16-190	16-190	16-190	16-190#	16-190#	16-190#	16-191	16-191	16-191	16-191	16-191	16-191
16-191#	16-191#	16-191#	16-191#	16-191#	16-191#	16-199	16-199	16-199	16-199	16-199#	16-199#	16-199#	16-199#
16-200	16-200	16-200#	16-200#	16-201	16-201	16-201	16-201	16-201#	16-201#	16-201#	16-201#	16-201#	16-202
16-202	16-202	16-202#	16-202#	16-202#	16-202#	16-203	16-203	16-203	16-203	16-203	16-203	16-203#	16-203#
16-203#	16-203#	16-203#	16-203#	16-204	16-204	16-204	16-204#	16-204#	16-204#	16-205	16-205	16-205	16-205
16-205	16-205	16-205#	16-205#	16-205#	16-205#	16-205#	16-205#	16-206	16-206	16-206	16-206	16-206#	16-206#
16-206#	16-206#	16-207	16-207	16-207	16-207	16-207	16-207	16-207#	16-207#	16-207#	16-207#	16-207#	16-207#
16-208	16-208	16-208	16-208#	16-208#	16-208#	16-209	16-209	16-209	16-209	16-209	16-209	16-209#	16-209#
16-209#	16-209#	16-209#	16-209#	16-217	16-217	16-217	16-217#	16-217#	16-217#	16-218	16-218	16-218	16-218
16-218#	16-218#	16-218#	16-218#	16-219	16-219	16-219#	16-219#	16-220	16-220	16-220	16-220	16-220#	16-220#
16-220#	16-220#	16-221	16-221	16-221	16-221	16-221#	16-221#	16-221#	16-221#	16-222	16-222	16-222	16-222
16-222#	16-222#	16-222#	16-222#	16-223	16-223	16-223	16-223	16-223	16-223	16-223#	16-223#	16-223#	16-223#
16-223#	16-223#	16-224	16-224	16-224	16-224#	16-224#	16-224#	16-225	16-225	16-225	16-225	16-225	16-225
16-225#	16-225#	16-225#	16-225#	16-225#	16-225#	16-226	16-226	16-226	16-226	16-226#	16-226#	16-226#	16-226#
16-227	16-227	16-227	16-227	16-227	16-227	16-227#	16-227#	16-227#	16-227#	16-227#	16-227#	16-227#	16-228
16-228	16-228#	16-228#	16-228#	16-229	16-229	16-229	16-229	16-229	16-229	16-229#	16-229#	16-229#	16-229#
16-229#	16-229#	16-237	16-237	16-237	16-237	16-237#	16-237#	16-237#	16-237#	16-238	16-238	16-238#	16-238#
16-239	16-239	16-239	16-239	16-239#	16-239#	16-239#	16-239#	16-240	16-240	16-240	16-240	16-240#	16-240#
16-240#	16-240#	16-241	16-241	16-241	16-241	16-241	16-241	16-241#	16-241#	16-241#	16-241#	16-241#	16-241#
16-242	16-242	16-242	16-242#	16-242#	16-242#	16-243	16-243	16-243	16-243	16-243	16-243	16-243#	16-243#
16-243#	16-243#	16-243#	16-243#	22-12	22-12	22-12	22-12#	22-12#	22-12#	24-456	24-456	24-456	24-456#
24-456#	24-456#	24-482	24-482	24-482#	24-482#	24-501	24-501	24-501#	24-501#	24-520	24-520	24-520#	24-520#
MSRADI	1-D77#	7-18#	25-16	25-16#	25-17	25-17#	25-18	25-18#	25-19	25-19#			
MSRBRO	1-C52#	7-18#											
MSRNRO	1-C62#	7-18#	19-51	19-51#									
MSSETS	1-D32#	7-18#	7-24	7-24#	10-9	10-9#	11-8	11-8#	16-100	16-100#	16-106	16-106#	16-121
	16-140	16-140#	16-159	16-159#	16-179	16-179#	16-198	16-198#	16-216	16-216#	16-236	16-236#	17-9
	18-8	18-8#	19-8	19-8#	20-7	20-7#	21-8	21-8#	22-8	22-8#	23-9	23-9#	24-14
	24-39	24-39#	24-67	24-67#	24-72	24-72	24-72#	24-72#	24-101	24-101#	24-128	24-128#	24-166
	24-203	24-203#	24-247	24-247#	24-298	24-298#	24-303	24-303	24-303#	24-303#	24-332	24-332#	24-337
	24-337#	24-337#	24-366	24-366#	24-371	24-371	24-371#	24-371#	24-400	24-400#	24-405	24-405#	24-405#
	24-449	24-449#	24-539	24-539#	24-610	24-610#	24-676	24-676#	24-680	24-680	24-680#	24-680#	24-727
	24-731	24-731	24-731#	24-731#	24-774	24-774#	24-778	24-778#	24-778#	24-778#	24-832	24-832#	24-838
	24-838#	24-838#	24-889	24-889#	24-898	24-898#	24-900	24-900	24-900#	24-900#	24-924	24-924#	24-957
	24-960	24-960#	24-:22	24-:22#	24-:52	24-:52#	24-<01	24-<01#	24-<49	24-<49#	24-=07	24-=07#	24-=91
	24->48	24->48#	24-?27	24-?27#	24-?63	24-?63#	24-a43	24-a43#	24-A26	24-A26#	24-A89	24-A89#	24-B69
	24-C05	24-C05#	24-D58	24-D58#	24-E53	24-E53#	24-F05	24-F05#	24-F70	24-F70#	24-G08	24-G08#	24-G59
	24-H13	24-H13#	25-14	25-14#	26-13	26-13#							
MSSTAR	1-A33#	7-18#											
MS SVC	1-C33#	7-18#	15-272	15-278	15-288	15-294	15-401	15-407	15-565	15-597	15-603	15-613	15-619
	15-660	15-694	15-700	15-708	15-714	15-860	15-865	15-884	15-890	15-899	15-905	15-914	15-920
	15-935	15-:67	15-:73	16-101	16-101#	16-102	16-102#	16-107	16-107#	16-108	16-108#	16-109	16-109#
	16-110#	16-111	16-111#	16-112	16-112#	16-113	16-113#	16-114	16-114#	16-115	16-115#	16-122	16-122#
	16-123#	16-124	16-124#	16-125	16-125#	16-126	16-126#	16-127	16-127#	16-128	16-128#	16-129	16-129#
	16-130#	16-131	16-131#	16-132	16-132#	16-133	16-133#	16-134	16-134#	16-141	16-141#	16-142	16-142#

	16-143#	16-144	16-144#	16-145	16-145#	16-146	16-146#	16-147	16-147#	16-148	16-148#	16-149	16-149#	16-150
	16-150#	16-151	16-151#	16-152	16-152#	16-153	16-153#	16-160	16-160#	16-161	16-161#	16-162	16-162#	16-163
	16-163#	16-164	16-164#	16-165	16-165#	16-166	16-166#	16-167	16-167#	16-168	16-168#	16-169	16-169#	16-170
	16-170#	16-171	16-171#	16-172	16-172#	16-173	16-173#	16-180	16-180#	16-181	16-181#	16-182	16-182#	16-183
	16-183#	16-184	16-184#	16-185	16-185#	16-186	16-186#	16-187	16-187#	16-188	16-188#	16-189	16-189#	16-190
	16-190#	16-191	16-191#	16-192	16-192#	16-199	16-199#	16-200	16-200#	16-201	16-201#	16-202	16-202#	16-203
	16-203#	16-204	16-204#	16-205	16-205#	16-206	16-206#	16-207	16-207#	16-208	16-208#	16-209	16-209#	16-210
	16-210#	16-217	16-217#	16-218	16-218#	16-219	16-219#	16-220	16-220#	16-221	16-221#	16-222	16-222#	16-223
	16-223#	16-224	16-224#	16-225	16-225#	16-226	16-226#	16-227	16-227#	16-228	16-228#	16-229	16-229#	16-230
	16-230#	16-237	16-237#	16-238	16-238#	16-239	16-239#	16-240	16-240#	16-241	16-241#	16-242	16-242#	16-243
	16-243#	16-244	16-244#	17-11	17-11#	19-25	19-25#	19-28	19-28#	19-31	19-31#	19-34	19-34#	19-51
	19-51#	19-74	19-74#	20-9	20-9#	20-16	20-16#	20-19	20-19#	21-11	21-11#	22-10	22-10#	22-12
	22-12#	22-13	22-13#	23-10	23-10#	24-16	24-16#	24-24	24-24#	24-27	24-27#	24-50	24-52	24-72
	24-72#	24-84	24-86	24-86#	24-90	24-90#	24-115	24-117	24-117#	24-133	24-133#	24-148	24-150	24-150#
	24-184	24-186	24-186#	24-226	24-227	24-227#	24-232	24-232#	24-278	24-279	24-279#	24-286	24-286#	24-303
	24-303#	24-314	24-316	24-316#	24-320	24-320#	24-337	24-337#	24-348	24-350	24-350#	24-354	24-354#	24-371
	24-371#	24-382	24-384	24-384#	24-388	24-388#	24-405	24-405#	24-416	24-418	24-418#	24-422	24-422#	24-456
	24-456#	24-480	24-482	24-482#	24-499	24-501	24-501#	24-518	24-520	24-520#	24-523	24-523#	24-551	24-552
	24-552#	24-565	24-566	24-566#	24-577	24-578	24-578#	24-588	24-589	24-589#	24-594	24-594#	24-637	24-638
	24-638#	24-647	24-648	24-648#	24-655	24-655#	24-680	24-680#	24-694	24-695	24-695#	24-704	24-706	24-706#
	24-710	24-710#	24-731	24-731#	24-745	24-746	24-746#	24-754	24-756	24-756#	24-760	24-760#	24-778	24-778#
	24-792	24-793	24-793#	24-801	24-803	24-803#	24-807	24-807#	24-838	24-838#	24-855	24-856	24-856#	24-865
	24-867	24-867#	24-872	24-872#	24-898	24-898#	24-900	24-900#	24-913	24-915	24-915#	24-920	24-920#	24-924
	24-924#	24-936	24-938	24-938#	24-939	24-939#	24-960	24-960#	24-983	24-984	24-984#	24-993	24-994	24-994#
	24-999	24-999#	24-:00	24-:00#	24-:98	24-:99	24-:99#	24-:21	24-:22	24-:22#	24-:32	24-:32#	24-:81	24-:81#
	24-<27	24-<27#	24-<83	24-<87	24-<87#	24-=26	24-=35	24-=47	24-=66	24-=74	24-=76	24-=76#	24->32	24->32#
	24-?12	24-?12#	24-?47	24-?47#	24-?97	24-@12	24-@27	24-@27#	24-@95	24-A10	24-A10#	24-A60	24-A73	24-A73#
	24-B41	24-B56	24-B56#	24-B85	24-B88	24-B88#	24-C89	24-D34	24-D34#	24-E35	24-E35#	24-E87	24-E87#	24-F55
	24-F55#	24-F87	24-F87#	24-G40	24-G40#	24-G90	24-G90#	24-H60	24-H60#					
M\$TLAB	1-C29#	7-18#	15-272#	15-278#	15-288#	15-294#	15-401#	15-407#	15-565#	15-597#	15-603#	15-613#	15-619#	15-654#
	15-660#	15-694#	15-700#	15-708#	15-714#	15-860#	15-865#	15-884#	15-890#	15-899#	15-905#	15-914#	15-920#	15-929#
	15-935#	15-:67#	15-:73#	16-101#	16-102#	16-107#	16-108#	16-109#	16-110#	16-111#	16-112#	16-113#	16-114#	16-115#
	16-122#	16-123#	16-124#	16-125#	16-126#	16-127#	16-128#	16-129#	16-130#	16-131#	16-132#	16-133#	16-134#	16-141#
	16-142#	16-143#	16-144#	16-145#	16-146#	16-147#	16-148#	16-149#	16-150#	16-151#	16-152#	16-153#	16-160#	16-161#
	16-162#	16-163#	16-164#	16-165#	16-166#	16-167#	16-168#	16-169#	16-170#	16-171#	16-172#	16-173#	16-180#	16-181#
	16-182#	16-183#	16-184#	16-185#	16-186#	16-187#	16-188#	16-189#	16-190#	16-191#	16-192#	16-199#	16-200#	16-201#
	16-202#	16-203#	16-204#	16-205#	16-206#	16-207#	16-208#	16-209#	16-210#	16-217#	16-218#	16-219#	16-220#	16-221#
	16-222#	16-223#	16-224#	16-225#	16-226#	16-227#	16-228#	16-229#	16-230#	16-237#	16-238#	16-239#	16-240#	16-241#
	16-242#	16-243#	16-244#	17-11#	19-25#	19-28#	19-31#	19-34#	19-51#	19-74#	20-9#	20-16#	20-19#	21-11#
	22-10#	22-12#	22-13#	23-10#	24-16#	24-24#	24-27#	24-50#	24-52#	24-72#	24-84#	24-86#	24-90#	24-115#
	24-117#	24-133#	24-148#	24-150#	24-184#	24-186#	24-226#	24-227#	24-232#	24-278#	24-279#	24-286#	24-303#	24-314#
	24-316#	24-320#	24-337#	24-348#	24-350#	24-354#	24-371#	24-382#	24-384#	24-388#	24-405#	24-416#	24-418#	24-422#
	24-456#	24-480#	24-482#	24-499#	24-501#	24-518#	24-520#	24-523#	24-551#	24-552#	24-565#	24-566#	24-577#	24-578#
	24-588#	24-589#	24-594#	24-637#	24-638#	24-647#	24-648#	24-655#	24-680#	24-694#	24-695#	24-704#	24-706#	24-710#
	24-731#	24-745#	24-746#	24-754#	24-756#	24-760#	24-778#	24-792#	24-793#	24-801#	24-803#	24-807#	24-838#	24-855#
	24-856#	24-865#	24-867#	24-872#	24-898#	24-900#	24-913#	24-915#	24-920#	24-924#	24-936#	24-938#	24-939#	24-960#
	24-983#	24-984#	24-993#	24-994#	24-999#	24-:00#	24-:98#	24-:99#	24-:21#	24-:22#	24-:32#	24-:81#	24-<27#	24-<83#
	24-<87#	24-=26#	24-=35#	24-=47#	24-=66#	24-=74#	24-=76#	24->32#	24-?12#	24-?47#	24-?97#	24-@12#	24-@27#	24-@95#
	24-A10#	24-A60#	24-A73#	24-B41#	24-B56#	24-B85#	24-B88#	24-C89#	24-D34#	24-E35#	24-E87#	24-F55#	24-F87#	24-G40#
	24-G90#	24-H60#												
M\$TSTL	1-C21#	7-18#	15-272	15-272#	15-272#	15-278	15-278#	15-278#	15-288	15-288#	15-288#	15-294	15-294#	15-294#
	15-401	15-401#	15-401#	15-407	15-407#	15-407#	15-565	15-565#	15-565#	15-597	15-597#	15-597#	15-603	15-603#
	15-603#	15-613	15-613#	15-613#	15-619	15-619#	15-619#	15-654	15-654#	15-654#	15-660	15-660#	15-660#	15-694
	15-694#	15-694#	15-700	15-700#	15-700#	15-708	15-708#	15-708#	15-714	15-714#	15-714#	15-860	15-860#	15-860#
	15-865	15-865#	15-865#	15-884	15-884#	15-884#	15-890	15-890#	15-890#	15-899	15-899#	15-899#	15-905	15-905#
	15-905#	15-914	15-914#	15-914#	15-920	15-920#	15-920#	15-929	15-929#	15-929#	15-935	15-935#	15-935#	15-:67
	15-:67#	15-:67#	15-:73	15-:73#	15-:73#	16-101	16-101#	16-102	16-102#	16-107	16-107#	16-108	16-108#	16-109

16-109#	16-110	16-110#	16-111	16-111#	16-112	16-112#	16-113	16-113#	16-114	16-114#	16-115	16-115#	16-122
16-122#	16-123	16-123#	16-124	16-124#	16-125	16-125#	16-126	16-126#	16-127	16-127#	16-128	16-128#	16-129
16-129#	16-130	16-130#	16-131	16-131#	16-132	16-132#	16-133	16-133#	16-134	16-134#	16-141	16-141#	16-142
16-142#	16-143	16-143#	16-144	16-144#	16-145	16-145#	16-146	16-146#	16-147	16-147#	16-148	16-148#	16-149
16-149#	16-150	16-150#	16-151	16-151#	16-152	16-152#	16-153	16-153#	16-160	16-160#	16-161	16-161#	16-162
16-162#	16-163	16-163#	16-164	16-164#	16-165	16-165#	16-166	16-166#	16-167	16-167#	16-168	16-168#	16-169
16-169#	16-170	16-170#	16-171	16-171#	16-172	16-172#	16-173	16-173#	16-180	16-180#	16-181	16-181#	16-182
16-182#	16-183	16-183#	16-184	16-184#	16-185	16-185#	16-186	16-186#	16-187	16-187#	16-188	16-188#	16-189
16-189#	16-190	16-190#	16-191	16-191#	16-192	16-192#	16-199	16-199#	16-200	16-200#	16-201	16-201#	16-202
16-202#	16-203	16-203#	16-204	16-204#	16-205	16-205#	16-206	16-206#	16-207	16-207#	16-208	16-208#	16-209
16-209#	16-210	16-210#	16-217	16-217#	16-218	16-218#	16-219	16-219#	16-220	16-220#	16-221	16-221#	16-222
16-222#	16-223	16-223#	16-224	16-224#	16-225	16-225#	16-226	16-226#	16-227	16-227#	16-228	16-228#	16-229
16-229#	16-230	16-230#	16-237	16-237#	16-238	16-238#	16-239	16-239#	16-240	16-240#	16-241	16-241#	16-242
16-242#	16-243	16-243#	16-244	16-244#	17-11	17-11#	19-25	19-25#	19-28	19-28#	19-31	19-31#	19-34
19-34#	19-51	19-51#	19-74	19-74#	20-9	20-9#	20-16	20-16#	20-19	20-19#	21-11	21-11#	22-10
22-10#	22-12	22-12#	22-13	22-13#	23-10	23-10#	24-16	24-16#	24-24	24-24#	24-24#	24-27	24-27#
24-50	24-50#	24-50#	24-52	24-52#	24-72	24-72#	24-84	24-84#	24-84#	24-86	24-86#	24-90	24-90#
24-115	24-115#	24-115#	24-117	24-117#	24-133	24-133#	24-148	24-148#	24-148#	24-150	24-150#	24-184	24-184#
24-184#	24-186	24-186#	24-226	24-226#	24-226#	24-227	24-227#	24-232	24-232#	24-278	24-278#	24-278#	24-279
24-279#	24-286	24-286#	24-303	24-303#	24-314	24-314#	24-314#	24-316	24-316#	24-320	24-320#	24-337	24-337#
24-348	24-348#	24-348#	24-350	24-350#	24-354	24-354#	24-371	24-371#	24-382	24-382#	24-382#	24-384	24-384#
24-388	24-388#	24-405	24-405#	24-416	24-416#	24-416#	24-418	24-418#	24-422	24-422#	24-456	24-456#	24-480
24-480#	24-480#	24-482	24-482#	24-499	24-499#	24-499#	24-501	24-501#	24-518	24-518#	24-518#	24-520	24-520#
24-523	24-523#	24-551	24-551#	24-551#	24-552	24-552#	24-565	24-565#	24-565#	24-566	24-566#	24-577	24-577#
24-577#	24-578	24-578#	24-588	24-588#	24-588#	24-589	24-589#	24-594	24-594#	24-637	24-637#	24-637#	24-638
24-638#	24-647	24-647#	24-647#	24-648	24-648#	24-655	24-655#	24-680	24-680#	24-694	24-694#	24-694#	24-695
24-695#	24-704	24-704#	24-704#	24-706	24-706#	24-710	24-710#	24-731	24-731#	24-745	24-745#	24-745#	24-746
24-746#	24-754	24-754#	24-754#	24-756	24-756#	24-760	24-760#	24-778	24-778#	24-792	24-792#	24-792#	24-793
24-793#	24-801	24-801#	24-801#	24-803	24-803#	24-807	24-807#	24-838	24-838#	24-855	24-855#	24-855#	24-856
24-856#	24-865	24-865#	24-865#	24-867	24-867#	24-872	24-872#	24-898	24-898#	24-900	24-900#	24-913	24-913#
24-913#	24-915	24-915#	24-920	24-920#	24-924	24-924#	24-936	24-936#	24-936#	24-938	24-938#	24-939	24-939#
24-960	24-960#	24-983	24-983#	24-983#	24-984	24-984#	24-993	24-993#	24-993#	24-994	24-994#	24-999	24-999#
24-:00	24-:00#	24-:98	24-:98#	24-:98#	24-:99	24-:99#	24-:21	24-:21#	24-:21#	24-:22	24-:22#	24-:32	24-:32#
24-:81	24-:81#	24-<27	24-<27#	24-<83	24-<83#	24-<83#	24-<87	24-<87#	24-26	24-26#	24-26#	24-35	24-35#
24-35#	24-47	24-47#	24-47#	24-66	24-66#	24-66#	24-74	24-74#	24-74#	24-76	24-76#	24-76#	24->32
24-?12	24-?12#	24-?47	24-?47#	24-?97	24-?97#	24-?97#	24-a12	24-a12#	24-a12#	24-a27	24-a27#	24-a95	24-a95#
24-a95#	24-A10	24-A10#	24-A60	24-A60#	24-A60#	24-A73	24-A73#	24-B41	24-B41#	24-B41#	24-B56	24-B56#	24-B85
24-B85#	24-B85#	24-B88	24-B88#	24-C89	24-C89#	24-C89#	24-D34	24-D34#	24-E35	24-E35#	24-E87	24-E87#	24-F55
24-F55#	24-F87	24-F87#	24-G40	24-G40#	24-G90	24-G90#	24-H60	24-H60#					
MSWORD	1-C94#	7-18#	8-17	8-17#	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8	9-8
	9-8	9-8	9-8	9-8	9-8	9-8#	15-272	15-272	15-272	15-272#	15-278	15-278	15-278#
15-288	15-288	15-288	15-288#	15-294	15-294	15-294	15-294#	15-401	15-401	15-401	15-401#	15-407	15-407#
15-407	15-407#	15-565	15-565	15-565	15-565#	15-597	15-597	15-597	15-597#	15-603	15-603	15-603	15-603#
15-613	15-613	15-613	15-613#	15-619	15-619	15-619	15-619#	15-654	15-654	15-654	15-654#	15-660	15-660#
15-660	15-660#	15-694	15-694	15-694	15-694#	15-700	15-700	15-700	15-700#	15-708	15-708	15-708	15-708#
15-714	15-714	15-714	15-714#	15-860	15-860	15-860	15-860#	15-865	15-865	15-865	15-865#	15-884	15-884#
15-884	15-884#	15-890	15-890	15-890	15-890#	15-899	15-899	15-899	15-899#	15-905	15-905	15-905	15-905#
15-914	15-914	15-914	15-914#	15-920	15-920	15-920	15-920#	15-929	15-929	15-929	15-929#	15-935	15-935#
15-935	15-935#	15-:67	15-:67	15-:67	15-:67#	15-:73	15-:73	15-:73	15-:73#	24-24	24-24	24-24	24-24#
24-50	24-50	24-50	24-50#	24-84	24-84	24-84	24-84#	24-115	24-115	24-115	24-115#	24-148	24-148#
24-148	24-148#	24-184	24-184	24-184	24-184#	24-226	24-226	24-226	24-226#	24-278	24-278	24-278	24-278#
24-314	24-314	24-314	24-314#	24-348	24-348	24-348	24-348#	24-382	24-382	24-382	24-382#	24-416	24-416#
24-416	24-416#	24-480	24-480	24-480	24-480#	24-499	24-499	24-499	24-499#	24-518	24-518	24-518	24-518#
24-551	24-551	24-551	24-551#	24-565	24-565	24-565	24-565#	24-577	24-577	24-577	24-577#	24-588	24-588#
24-588	24-588#	24-637	24-637	24-637	24-637#	24-647	24-647	24-647	24-647#	24-694	24-694	24-694	24-694#

