

DMP-11, DMR-11,
M8203

M8203 STATIC DIAG
CZDMRFO

AH-E232F-MC
FICHE 1 OF 1

APR 1982
COPYRIGHT © 79-82
MADE IN USA



The main body of the document is a large, dense grid of data. Each cell in the grid contains a small, structured diagram or table, likely representing a static diagram for a specific component or function. The diagrams are arranged in a regular grid pattern across the page.

CZDMRF M8203 STATIC DIAG #1
CZDMRF.P11 03-NOV-81 10:29

MACY11 30A(1052) 03-NOV-81 10:36 PAGE 2
PROGRAM DOCUMENT

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

.REM @

IDENTIFICATION

PRODUCT CODE: AC-E231F-MC
PRODUCT NAME: CZDMRFO M8203 STATIC DIAG #1
PRODUCT DATE: FEBRUARY 1982
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, 1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

36
37
38
39
40
41
42
43
44
45
46
47
48

***** MODIFICATION HISTORY *****

THE MODIFICATION HISTORY BEGINS WITH VERSION F.

FOR THE 11-NOV-81 RELEASE BERT KLEINSCHMIDT MODIFIED VERSION E TO CREATE
VERSION F. THE ONLY CHANGE THAT WAS MADE WAS THE UPDATING OF THE VERSION
LETTER (TO F) IN THE DOCUMENTATION AND HEADER CODE. THIS CHANGE WAS MADE
TO CORRECT THE ACCIDENTAL RELEASE AND DISTRIBUTION OF A BAD COPY OF
CZDMRE.

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

CONTENTS

49	
50	
51	
52	
53	
54	
55	
56	
57	1.0 INTRODUCTION
58	
59	2.0 HARDWARE REQUIREMENTS
60	
61	3.0 PRELIMINARY PROGRAM REQUIREMENTS
62	
63	4.0 GENERAL PROGRAM CONSIDERATIONS
64	4.1 DIAGNOSTIC SUPERVISOR
65	4.2 EXECUTION TIME
66	4.3 XXDP+
67	4.4 ACT/SLIDE
68	4.5 APT
69	4.6 MEMORY MANAGEMENT
70	4.7 MEMORY PARITY OPTION
71	4.8 ERROR LOGGING
72	
73	5.0 PROGRAM LOAD MEDIA
74	
75	6.0 OPERATING INSTRUCTIONS
76	6.1 LOADING AND STARTING PROCEDURES
77	6.1.1 LOADING PROCEDURES
78	6.1.2 STARTING PROCEDURES
79	6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION
80	6.2 INITIAL DIALOGUE
81	6.3 PROGRAM OPTIONS
82	6.3.1 START COMMAND
83	6.3.1.1 TESTS SWITCH
84	6.3.1.2 PASS SWITCH
85	6.3.1.3 FLAGS SWITCH
86	6.3.1.4 END OF PASS SWITCH
87	6.3.1.5 EFFECT OF START COMMAND
88	6.3.2 RESTART COMMAND
89	6.3.2.1 TESTS, PASS, AND FLAG SWITCHES
90	6.3.2.2 UNITS SWITCH
91	6.3.2.3 EFFECT OF RESTART COMMAND
92	6.3.3 CONTINUE COMMAND
93	6.3.3.1 PASS SWITCH
94	6.3.3.2 FLAGS SWITCH
95	6.3.3.3 EFFECT OF CONTINUE COMMAND
96	6.3.4 PROCEED COMMAND
97	6.3.4.1 FLAGS SWITCH
98	6.3.4.2 EFFECT OF PROCEED COMMAND
99	6.3.5 ADD COMMAND
100	6.3.5.1 UNITS SWITCH
101	6.3.5.2 EFFECT OF ADD COMMAND
102	6.3.6 DROP COMMAND
103	6.3.6.1 UNITS SWITCH
104	6.3.6.2 EFFECT OF DROP COMMAND

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125

6.3.7 PRINT COMMAND
6.3.7.1 EFFECT OF PRINT COMMAND
6.3.8 DISPLAY COMMAND
6.3.8.1 UNITS SWITCH
6.3.8.2 EFFECT OF DISPLAY COMMAND
6.3.9 FLAGS COMMAND
6.3.9.1 EFFECT OF FLAGS COMMAND
6.3.10 ZFLAGS COMMAND
6.3.10.1 EFFECT OF ZFLAGS COMMAND
6.3.11 CONTROL CHARACTERS
6.3.12 HARDWARE PARAMETERS
6.3.13 SOFTWARE PARAMETERS
6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE

7.0 DEVICE INFORMATION TABLES

8.0 TEST DESCRIPTIONS

8.1 DATA PATTERNS USED

9.0 ERROR INFORMATION

9.1 ERROR REPORTING

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

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. 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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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.

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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

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>)
- C) ENTER STA<CR>
- D) ANSWER HARDWARE AND SOFTWARE QUESTIONS

CZDMRF.P11 03-NOV-81 10:29

PROGRAM DOCUMENT

E) GET END OF PASS MESSAGES OR ERROR MESSAGES
F) TO END EXECUTION, ENTER CONTROL/C

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-F-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

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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

END OF 6.3.1.5.

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

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

PROGRAM DOCUMENT

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

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 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

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>)

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

PROGRAM DOCUMENT

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

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

6.3.4.2 EFFECT OF PROCEED COMMAND

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.

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

6.3.7 PRINT COMMAND

PRI(NT)

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.

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

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

6.3.12 HARDWARE PARAMETERS

THE FOLLOWING 2 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. M8207 RUN SWITCH (E28 SW7) - 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 ONE-TO ONE CORRESPONDENCE BETWEEN THE HARDWARE PARAMETER QUESTIONS AND THE SLOTS IN THE P-TABLE FORMAT.

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757

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 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).

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

7.0 DEVICE INFORMATION TABLES

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

* MAINTENANCE REGISTER - BSEL1

RUN = BIT7
MCLR = BIT6
STEPLU = BIT4
LULoop = 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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

```

:*****
* 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
OVRR     = BIT3
RAB      = BIT2

```

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

EBLK = BIT1
BCC = BIT0

* 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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

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

ASBC1 = BIT5
ASBC0 = BIT4
ROR = BIT3
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

CZDMRF.P11 03-NOV-81 10:29

PROGRAM DOCUMENT

982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005

I422 = BIT7
XYZ = BIT6
C32BCC = BIT5
V35 = BIT4
INTGRL = BIT3
C32ENB = BIT2
OP = BIT1
TEST = BIT0
AX315U = I422!XYZ!C32BCC!V35!INTGRL!OP

:*****
* AX3-16 - USYRT REG 7
:*****
TXLEN2 = BIT7
TXLEN1 = BIT6
TXLEN0 = BIT5
RXLEN2 = BIT2
RXLEN1 = BIT1
RXLEN0 = BIT0

1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061

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.

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117

```
*****  
; 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  
*****
```


CZDMRF.P11 03-NOV-81 10:29

PROGRAM DOCUMENT

1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173

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.

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-

CZDMRF.P11 03-NOV-81 10:29

PROGRAM DOCUMENT

1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229

```

* 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 M8207 RUN SWITCH (E28 SW7)
* 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 = 000,000,000,000,000,103,000,000
:*****

```

```

:*****
* 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 = 000,000,000,000,000,103,000,000
* 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,
* 000,000,000,000,000,000,376,375,373,367,357,337,277,177,
* 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,

```


PROGRAM DOCUMENT

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
1284
1285

* 004,010,020,040,100,200,377,377,377,377,377,377,377,377,
* 376,375,373,367,357,337,277,177.
:*****

:*****
* 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
:*****

1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341

```

;*****
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.
;*****

```


1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1358
 1359
 1360
 1361
 1362
 1363
 1364
 1365
 1366
 1367
 1368
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376
 1377
 1378
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388
 1389
 1390
 1391
 1392
 1393
 1394
 1395
 1396
 1397

```

;*****
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.
;*****

```

```

;*****
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.

```

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453

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, CRC

PROGRAM DOCUMENT

1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509

*
* 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.
:*****

:*****
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.
:*****

1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1518
 1519
 1520
 1521
 1522
 1523
 1524
 1525
 1526
 1527
 1528
 1529
 1530
 1531
 1532
 1533
 1534
 1535
 1536
 1537
 1538
 1539
 1540
 1541
 1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1550
 1551
 1552
 1553
 1554
 1555
 1556
 1557
 1558
 1559
 1560
 1561
 1562
 1563
 1564
 1565

```

;*****
  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
* 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

```


CZDMRF.P11 03-NOV-81 10:29

PROGRAM DOCUMENT

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621

* 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.
:*****

:*****
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.
:*****

1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677

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.

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 (TEOM SHOULD CAUSE TX ENABLE TO DROP).
* THE ABOVE TEST IS REPEATED FOR EACH OF THE FOLLOWING SYNCH CHAR DATA

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

* PATTERNS : 226,000,125,252,376,177.
;*****

1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733

8.1 DATA PATTERNS USED

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

.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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789

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

PATE:

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

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

PATF:

.BYTE 050
.BYTE 051
.BYTE 050

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

PATG:

.BYTE 000
.BYTE 000
.BYTE 240
.BYTE 120
.BYTE 177
.BYTE 000
.BYTE 000
.BYTE 001

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

PATH:

.BYTE 000
.BYTE 000
.BYTE 377
.BYTE 017
.BYTE 377
.BYTE 377
.BYTE 375
.BYTE 377

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

PATI:

.BYTE 000
.BYTE 000
.BYTE 000
.BYTE 000
.BYTE 000
.BYTE 103
.BYTE 000
.BYTE 000

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

PATJ:

.BYTE 000
.BYTE 000
.BYTE 010
.BYTE 002
.BYTE 004
.BYTE 103

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1790	.BYTE	001
1791	.BYTE	100
1792		
1793		
1794		
1795		
1796		
1797		
1798		
1799		
1800		
1801		
1802		
1803		
1804		
1805		
1806		
1807		
1808		
1809		
1810		
1811		
1812		
1813		
1814		
1815		
1816		
1817		
1818		
1819		
1820		
1821		
1822		
1823		
1824		
1825		
1826		
1827		
1828		
1829		
1830		
1831		
1832		
1833		
1834		
1835		
1836		
1837		
1838		
1839		
1840		
1841		
1842		
1843		
1844		
1845		

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

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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1846	.BYTE	277
1847	.BYTE	377
1848	.BYTE	177
1849	.BYTE	377
1850	.BYTE	377
1851	.BYTE	376
1852	.BYTE	377
1853	.BYTE	375
1854	.BYTE	377
1855	.BYTE	373
1856	.BYTE	377
1857	.BYTE	367
1858	.BYTE	377
1859	.BYTE	357
1860	.BYTE	377
1861	.BYTE	337
1862	.BYTE	377
1863	.BYTE	277
1864	.BYTE	377
1865	.BYTE	177
1866		
1867		
1868		
1869		
1870		
1871		
1872		
1873		
1874		
1875		
1876		
1877		
1878		
1879		
1880		
1881		
1882		
1883		
1884		
1885		
1886		
1887		
1888		
1889		
1890		
1891		
1892		
1893		
1894		
1895		
1896		
1897		
1898		
1899		
1900		
1901		

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

PATL:

.BYTE	000
.BYTE	000
.BYTE	377
.BYTE	377
.BYTE	000
.BYTE	000

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

PATM:

.BYTE	000
.BYTE	020
.BYTE	000
.BYTE	000
.BYTE	200
.BYTE	000
.BYTE	000
.BYTE	051

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

PATN:

.BYTE	000
.BYTE	000
.BYTE	000
.BYTE	125
.BYTE	000
.BYTE	252
.BYTE	000
.BYTE	377
.BYTE	005
.BYTE	000
.BYTE	012
.BYTE	000
.BYTE	017

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

1902
 1903
 1904
 1905
 1906
 1907
 1908
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1918
 1919
 1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1928
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952
 1953
 1954
 1955
 1956
 1957

.BYTE 000

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

PATO:

.BYTE 000
 .BYTE 041
 .BYTE 004
 .BYTE 010
 .BYTE 020
 .BYTE 040
 .BYTE 100
 .BYTE 101
 .BYTE 200
 .BYTE 201
 .BYTE 300
 .BYTE 111
 .BYTE 301
 .BYTE 375

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

PATP:

.BYTE 000
 .BYTE 113
 .BYTE 200
 .BYTE 040
 .BYTE 020
 .BYTE 010
 .BYTE 001
 .BYTE 104
 .BYTE 007
 .BYTE 105
 .BYTE 007
 .BYTE 144
 .BYTE 107
 .BYTE 157

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

PATU:

.BYTE 000
 .BYTE 000
 .BYTE 001
 .BYTE 000
 .BYTE 013
 .BYTE 000
 .BYTE 011
 .BYTE 000
 .BYTE 021
 .BYTE 000
 .BYTE 101
 .BYTE 000
 .BYTE 301
 .BYTE 000
 .BYTE 000
 .BYTE 001
 .BYTE 000
 .BYTE 002
 .BYTE 000

1958	.BYTE	004
1959	.BYTE	000
1960	.BYTE	040
1961	.BYTE	000
1962	.BYTE	100
1963	.BYTE	000
1964	.BYTE	200
1965	.BYTE	000
1966	.BYTE	346
1967	.BYTE	000
1968	.BYTE	345
1969	.BYTE	000
1970	.BYTE	343
1971	.BYTE	000
1972	.BYTE	307
1973	.BYTE	000
1974	.BYTE	247
1975	.BYTE	000
1976	.BYTE	147

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

1978		
1979	PATV:	.BYTE 000
1980		.BYTE 000
1981		.BYTE 333
1982		.BYTE 000
1983		.BYTE 331
1984		.BYTE 000
1985		.BYTE 323
1986		.BYTE 000
1987		.BYTE 313
1988		.BYTE 000
1989		.BYTE 233
1990		.BYTE 000
1991		.BYTE 133
1992		.BYTE 000
1993		.BYTE 000
1994		.BYTE 001
1995		.BYTE 000
1996		.BYTE 002
1997		.BYTE 000
1998		.BYTE 004
1999		.BYTE 000
2000		.BYTE 040
2001		.BYTE 000
2002		.BYTE 100
2003		.BYTE 000
2004		.BYTE 200
2005		.BYTE 000
2006		.BYTE 346
2007		.BYTE 000
2008		.BYTE 345
2009		.BYTE 000
2010		.BYTE 343
2011		.BYTE 000
2012		.BYTE 307
2013		.BYTE 000

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

SEQ 40

2014
2015
2016
2017
2018
2019
2020
2021
2022

.BYTE 247
.BYTE 000
.BYTE 147

2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078

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

CZDMRF.P11

03-NOV-81 10:29

PROGRAM DOCUMENT

2079
2080
2081
2082
2083
2084
2085
2086
2087

a

2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119

002000

.TITLE CZDMRF M8203 STATIC DIAG #1
.=2000

002000

.MCALL SVC
SVC

; INITIALIZE SUPERVISOR MACROS

002000

BGNMOD LU1MOD

000001
000001
000001
000001
000001
000001
000001
000001

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

PROGRAM HEADER

2120		
2121		
2122		
2123		
2124		
2125		
2126	002000	
2127		
2128		
2129		
2130		
2131		
2132		
2133		
2134	002000	
2135	002000	
2136	002000	103
2137	002001	132
2138	002002	104
2139	002003	115
2140	002004	122
2141	002005	000
2142	002006	000
2143	002007	000
2144	002010	
2145	002010	106
2146	002011	
2147	002011	060
2148	002012	
2149	002012	000000
2150	002014	
2151	002014	000055
2152	002016	
2153	002016	035500
2154	002020	
2155	002020	000000
2156	002022	
2157	002022	002252
2158	002024	
2159	002024	000000
2160	002026	
2161	002026	036054
2162	002030	
2163	002030	000000
2164	002032	
2165	002032	000000
2166	002034	
2167	002034	000000
2168	002036	
2169	002036	000000
2170	002040	
2171	002040	002124
2172	002042	
2173	002042	000000
2174	002044	
2175	002044	000000

```

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

```

POINTER BGNAU,BGNDU

```

:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
: 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'.
:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

HEADER CZDMR,F,0,45.,0

```

LSNAME::
        .ASCII /C/
        .ASCII /Z/
        .ASCII /D/
        .ASCII /M/
        .ASCII /R/
        .BYTE 0
        .BYTE 0
        .BYTE 0
LSREV::
        .ASCII /F/
LSDEPO::
        .ASCII /O/
LSUNIT::
        .WORD 0
LSTIML::
        .WORD 45.
LSHPCP::
        .WORD LSHARD
LSSPCP::
        .WORD 0
LSHPTP::
        .WORD LSHW
LSSPTP::
        .WORD 0
LSLADP::
        .WORD LSLAST
LSSTA::
        .WORD 0
LSCO::
        .WORD 0
LSDTYP::
        .WORD 0
LSAPT::
        .WORD 0
LSDTP::
        .WORD LSDISPATCH
LSPRIO::
        .WORD 0
LSENV1::
        .WORD 0

```

```

2176 002046
2177 002046 000000
2178 002050
2179 002050 003
2180 002051 003
2181 002052
2182 002052 000000
2183 002054 000000
2184 002056
2185 002056 000000
2186 002060
2187 002060 003440
2188 002062
2189 002062 000000
2190 002064
2191 002064 000000
2192 002066
2193 002066 000000
2194 002070
2195 002070 021556
2196 002072
2197 002072 021474
2198 002074
2199 002074 000000
2200 002076
2201 002076 003446
2202 002100
2203 002100 104035
2204 002102
2205 002102 000000
2206 002104
2207 002104 021074
2208 002106
2209 002106 021472
2210 002110
2211 002110 021412
2212 002112
2213 002112 021066
2214 002114
2215 002114 000000
2216 002116
2217 002116 000000
2218 002120
2219 002120 000000

```

```

L$EXP1:: .WORD 0
L$MREV:: .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$ETP:: .WORD 0
L$ICP:: .WORD L$INIT
L$CCP:: .WORD L$CLEAN
L$ACP:: .WORD L$AUTO
L$PRT:: .WORD L$PROT
L$TEST:: .WORD 0
L$DLY:: .WORD 0
L$HIME:: .WORD 0

```

```

:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
: CHANGE THE 'HEADER' TO CONTAIN THE PROPER ARGUMENTS.
:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

.EVEN

```

2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231

```


CZDMRF.P11 03-NOV-81 10:29

DISPATCH TABLE

.SBTTL DISPATCH TABLE

```

:////////////////////
:/ THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
:/ IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
:////////////////////

```

DISPATCH 42

```

2232
2233
2234
2235
2236
2237
2238
2239 002122
2240 002122 000052
2241 002124
2242 002124 021560
2243 002126 021642
2244 002130 021726
2245 002132 022052
2246 002134 022154
2247 002136 022302
2248 002140 022422
2249 002142 022576
2250 002144 023032
2251 002146 023150
2252 002150 023266
2253 002152 023404
2254 002154 023522
2255 002156 024152
2256 002160 024504
2257 002162 024774
2258 002164 025204
2259 002166 025410
2260 002170 025614
2261 002172 026052
2262 002174 026304
2263 002176 026556
2264 002200 027234
2265 002202 027354
2266 002204 027464
2267 002206 027660
2268 002210 030224
2269 002212 030434
2270 002214 030764
2271 002216 031064
2272 002220 031434
2273 002222 031776
2274 002224 032246
2275 002226 032610
2276 002230 032740
2277 002232 033652
2278 002234 034134
2279 002236 034334
2280 002240 034634
2281 002242 034724
2282 002244 035066
2283 002246 035234
2284
2285
2286
2287

```

```

.LSDISPATCH::
.WORD 42
.WORD T1
.WORD T2
.WORD T3
.WORD T4
.WORD T5
.WORD T6
.WORD T7
.WORD T8
.WORD T9
.WORD T10
.WORD T11
.WORD T12
.WORD T13
.WORD T14
.WORD T15
.WORD T16
.WORD T17
.WORD T18
.WORD T19
.WORD T20
.WORD T21
.WORD T22
.WORD T23
.WORD T24
.WORD T25
.WORD T26
.WORD T27
.WORD T28
.WORD T29
.WORD T30
.WORD T31
.WORD T32
.WORD T33
.WORD T34
.WORD T35
.WORD T36
.WORD T37
.WORD T38
.WORD T39
.WORD T40
.WORD T41
.WORD T42

```

```

:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
: CHANGE THE ARGUMENT OF 'DISPATCH' TO BE THE
: NUMBER OF HARDWARE TESTS IN YOUR PROGRAM.

```


CZDMRF.P11 03-NOV-81 10:29

DEFAULT HARDWARE P-TABLE

.SBTTL DEFAULT HARDWARE P-TABLE

```

:////////////////////
:/ THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
:/ THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
:/ IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
:////////////////////

```

2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316

002250
002250 000002
002252
002252
002252 160170
002254 000001
002256
002256

BGNHW DFPTBL

.WORD 160170
.WORD 1

ENDHW

.WORD L10000-LSHW/2
LSHW::
DFPTBL::

:M8207 CSR UNIBUS ADDRESS
:M8207 RUN SWITCH (E28 SW7) IS ON

L10000:

CZDMRF.P11 03-NOV-81 10:29

SOFTWARE P-TABLE

2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337

002256
002256 000000
002260
002260

002260
002260

.SBTTL SOFTWARE P-TABLE
:////////////////////
:/ THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
:/ PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
:////////////////////

BGNSW SFPTBL

.WORD L10001-LSSW/2
LSSW::
SFPTBL::

ENDSW

L10001:

2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393

002260

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

.SBTTL GLOBAL EQUATES SECTION

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

EQUALS

: BIT DIFINITIONS

BIT15== 100000
BIT14== 40000
BIT13== 20000
BIT12== 10000
BIT11== 4000
BIT10== 2000
BIT09== 1000
BIT08== 400
BIT07== 200
BIT06== 100
BIT05== 40
BIT04== 20
BIT03== 10
BIT02== 4
BIT01== 2
BIT00== 1

BIT9== BIT09
BIT8== BIT08
BIT7== BIT07
BIT6== BIT06
BIT5== BIT05
BIT4== BIT04
BIT3== BIT03
BIT2== BIT02
BIT1== BIT01
BIT0== BIT00

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

EF.START== 32. : START COMMAND WAS ISSUED
EF.RESTART== 31. : RESTART COMMAND WAS ISSUED

CZDMRF.P11 03-NOV-81 10:29

GLOBAL EQUATES SECTION

: CONTINUE COMMAND WAS ISSUED
: A NEW PASS HAS BEEN STARTED
: A POWER-FAIL/POWER-UP OCCURRED

2394 000036
2395 000035
2396 000034
2397
2398
2399
2400
2401 000340
2402 000300
2403 000240
2404 000200
2405 000140
2406 000100
2407 000040
2408 000000

EF.CONTINUE== 30.
EF.NEW== 29.
EF.PWR== 28.

: PRIORITY LEVEL DEFINITIONS

2409
2410
2411
2412 000004
2413 000010
2414 000020
2415 000040
2416 000100
2417 000200
2418 000400
2419 001000
2420 002000
2421 004000
2422 010000
2423 020000
2424 040000
2425 100000

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

: OPERATOR FLAG BITS

2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441 000200
2442 000100
2443 000020
2444 000010
2445 000004
2446 000002
2447 000001
2448
2449

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

: *****
: * PROGRAM EVENT FLAG DEFINITIONS
: *****

: *****
: * MAINTENANCE REGISTER - BSEL1
: *****

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

: *****

GLOBAL EQUATES SECTION

```

2450      ;* OBUS REG 10 - TRANSMITTER BUFFER
2451      ;*****
2452      000200 TX7      = BIT7
2453      000100 TX6      = BIT6
2454      000040 TX5      = BIT5
2455      000020 TX4      = BIT4
2456      000010 TX3      = BIT3
2457      000004 TX2      = BIT2
2458      000002 TX1      = BIT1
2459      000001 TX0      = BIT0
2460
2461      ;*****
2462      ;* OBUS REG 11
2463      ;*****
2464      000200 OC       = BIT7
2465      000010 GOAH    = BIT3
2466      000004 ABORT   = BIT2
2467      000002 EOM     = BIT1
2468      000001 SOM     = BIT0
2469
2470      ;*****
2471      ;* OBUS REG 12
2472      ;*****
2473      000200 IC       = BIT7
2474      000100 BPOLL   = BIT6
2475      000040 LULP    = BIT5
2476
2477      ;*****
2478      ;* OBUS REG 13
2479      ;*****
2480      000200 POLL    = BIT7
2481      000100 DTR     = BIT6
2482      000040 SELFR   = BIT5
2483      000020 HDX     = BIT4
2484      000010 MAINT1  = BIT3
2485      000004 MAINT2  = BIT2
2486      000002 SELSBY  = BIT1
2487
2488      ;*****
2489      ;* OBUS REG 14
2490      ;*****
2491      000100 TXEN    = BIT6
2492      000040 DISSI   = BIT5
2493      000020 RDAX    = BIT4
2494      000010 WAX     = BIT3
2495      000004 ENAX    = BIT2
2496      000002 AX2     = BIT1
2497      000001 AX1     = BIT0
2498
2499      ;*****
2500      ;* OBUS REG 17
2501      ;*****
2502      000200 CRC2    = BIT7
2503      000100 CRC1    = BIT6
2504      000040 IDLE    = BIT5
2505      000020 SECA    = BIT4

```

CZDMRF.P11

03-NOV-81 10:29

GLOBAL EQUATES SECTION

2506 000010
 2507 000004
 2508 000002
 2509 000001
 2510
 2511
 2512
 2513
 2514 000200
 2515 000100
 2516 000040
 2517 000020
 2518 000010
 2519 000004
 2520 000002
 2521 000001
 2522
 2523
 2524
 2525
 2526 000200
 2527 000100
 2528 000040
 2529 000020
 2530 000010
 2531 000004
 2532 000002
 2533 000001
 2534
 2535
 2536
 2537
 2538 000200
 2539 000100
 2540 000040
 2541 000020
 2542 000010
 2543 000004
 2544 000002
 2545 000001
 2546
 2547
 2548
 2549
 2550 000200
 2551 000100
 2552 000040
 2553 000020
 2554 000010
 2555 000004
 2556 000002
 2557 000001
 2558
 2559
 2560
 2561

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
 OVRR = BIT3
 RAB = BIT2
 EBLK = BIT1
 BCC = BIT0

::*****
 :* IBUS REG 13

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

::*****
 :* IBUS REG 14

::*****

CZDMRF.P11

03-NOV-81 10:29

GLOBAL EQUATES SECTION

2562 000200
 2563 000100
 2564 000040
 2565 000020
 2566 000010
 2567 000004
 2568 000002
 2569 000001
 2570
 2571
 2572
 2573
 2574 000200
 2575 000100
 2576 000040
 2577 000020
 2578 000010
 2579 000004
 2580 000002
 2581 000001
 2582
 2583
 2584
 2585
 2586 000200
 2587 000100
 2588 000040
 2589 000020
 2590 000010
 2591 000004
 2592 000002
 2593 000001
 2594
 2595
 2596
 2597
 2598 000200
 2599 000100
 2600 000040
 2601 000020
 2602 000010
 2603 000004
 2604 000002
 2605 000001
 2606
 2607
 2608
 2609
 2610 000200
 2611 000100
 2612 000040
 2613 000020
 2614 000010
 2615 000004
 2616 000002
 2617 000001

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
 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

CZDMRF.P11

03-NOV-81 10:29

GLOBAL EQUATES SECTION

```

2618
2619
2620
2621
2622      000200
2623      000010
2624      000004
2625      000002
2626      000001
2627
2628
2629
2630
2631      000200
2632      000100
2633      000040
2634      000020
2635      000010
2636      000004
2637      000002
2638      000001
2639      000226
2640
2641
2642
2643
2644      000200
2645      000100
2646      000040
2647      000020
2648      000010
2649      000004
2650      000002
2651      000001
2652
2653
2654
2655
2656      000200
2657      000100
2658      000040
2659      000020
2660      000010
2661      000004
2662      000002
2663      000001
2664      000372
2665
2666
2667
2668
2669      000200
2670      000100
2671      000040
2672      000004
2673      000002

```

```

*****
;* 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
SYNO      = 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
INTGRL   = BIT3
C32ENB   = BIT2
OP       = BIT1
TEST     = BIT0
AX315U   = I422!XYZ!C32BCC!V35!INTGRL!OP

*****
;* AX3-16 - USYRT REG 7
*****
TXLEN2   = BIT7
TXLEN1   = BIT6
TXLEN0   = BIT5
RXLEN2   = BIT2
RXLEN1   = BIT1

```

CZDMRF.P11

03-NOV-81 10:29

GLOBAL EQUATES SECTION

2674 000001

RXLENO = BIT0

2675

2676

2677

2678

2679

2680

2681

2682

2683

2684

2685

2686

2687

2688

2689

2690

2691

2692

2693

2694

2695

2696

2697

2698

2699

2700

2701

2702

2703

2704

2705

2706

2707

2708

2709

2710

2711

2712

2713

2714

2715

2716

2717

2718

2719

2720

2721

2722

2723

2724

2725

2726

2727

2728

2729

004000

002000

001000

000400

004000

002000

001000

000400

002260

002262

002264

002266

002270

002272

002274

002276

002300

002302

002304

002306

002310

002312

002314

002316

100000

100000

:* TX CONTROL BITS DEFINED ON WORD BASIS

TXGOA = BIT11
TXABT = BIT10
TXEOM = BIT9
TXSOM = BIT8

:* RCV CONTROL BITS DEFINED ON WORD BASIS

RXOVR = BIT11
RXABT = BIT10
RXEBL = BIT9
RXBCC = BIT8

:* ADDRESS EQUATES FOR REGISTER STORAGE TABLE (LUREG:)

LUR10 = LUREG+0 ;LINE UNIT IBUS REG 10
LUR11 = LUREG+2 ;LINE UNIT IBUS REG 11
LUR12 = LUREG+4 ;LINE UNIT IBUS REG 12
LUR13 = LUREG+6 ;LINE UNIT IBUS REG 13
LUR14 = LUREG+10 ;LINE UNIT IBUS REG 14
LUR15 = LUREG+12 ;LINE UNIT IBUS REG 15
LUR16 = LUREG+14 ;LINE UNIT IBUS REG 16
LUR17 = LUREG+16 ;LINE UNIT IBUS REG 17
AX0.15 = LUREG+20 ;USYRT REG 0
AX0.16 = LUREG+22 ;USYRT REG 1
AX1.15 = LUREG+24 ;USYRT REG 2
AX1.16 = LUREG+26 ;USYRT REG 3
AX2.15 = LUREG+30 ;USYRT REG 4
AX2.16 = LUREG+32 ;USYRT REG 5
AX3.15 = LUREG+34 ;USYRT REG 6
AX3.16 = LUREG+36 ;USYRT REG 7

CHPCHK = BIT15

BCCCHK = BIT15

CZDMRF.P11

03-NOV-81 10:29

GLOBAL EQUATES SECTION

CRCCHK = BIT15

2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752

100000

021000
122000
121000

000001
000002

```
*****  
;* 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
```


GLOBAL DATA SECTION

2753
 2754
 2755
 2756
 2757
 2758
 2759
 2760
 2761
 2762
 2763 002260 000020
 2764
 2765
 2766
 2767
 2768 002320 000000
 2769 002322 000000
 2770 002324 000000
 2771 002326 000000
 2772 002330 000000
 2773 002332 000000
 2774
 2775 002334 000000
 2776 002336 000000
 2777 002340 000000
 2778 002342 000000
 2779 002344 000000
 2780 002346 000000
 2781 002350 000000
 2782 002352 000000
 2783 002354 000000
 2784 002356 000000
 2785 002360 000000
 2786 002362 000000
 2787 002364 000000
 2788 002366 000000
 2789 002370 000000
 2790 002372 000000
 2791 002374 000000
 2792 002376 000000
 2793 002400 000000
 2794 002402 000000
 2795 002404 000000
 2796 002406 000000
 2797 002410 000000
 2798 002412 000000
 2799 002414 000000
 2800 002416 000000
 2801 002420 000000
 2802 002422 000000
 2803
 2804
 2805 002424 160170
 2806 002426 160171
 2807 002430 160172
 2808 002432

```

.SBTTL GLOBAL DATA SECTION
://////
:/ THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
:/ IN MORE THAN ONE TEST.
://////

:*****
:* STORAGE FOR DEVICE REGISTERS
:*****
LUREG: .BLKW 16.

:*****
:* MISCELLANEOUS STORAGE
:*****
SCRACH: .WORD 0 ;GEN'L PURPOSE SCRATCH WORD
LOGDEV: .WORD 0 ;LOGICAL DEVICE NUMBER
PSTACK: .WORD 0 ;CONTAINS BASE LEVEL PROGRAM STACK POINTER
PRIOR: .WORD 0 ;CPU PRIORITY FOR PRINTOUT
SUBRPC: .WORD 0 ;PC OF SUBR CALL FOR ERROR REPORTS
INTFLG: .WORD 0 ;INTERRUPT RECEIVED FLAGS
; BIT 0 FOR TX, BIT 1 FOR RCV
ERRFLG: .WORD 0 ;SUBROUTINE ERROR FLAG
TIMFLG: .WORD 0 ;EVENT TIME-OUT FLAG
RETADR: .WORD 0 ;SUBR ERROR RETURN ADDRESS
REDBYT: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM LU REG
WRIBYT: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO LU REG
RAX15: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 15
RAX16: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 16
WAX15: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 15
WAX16: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 16
REGNUM: .WORD 0 ;NUMBER (10-17) OF LINE UNIT REG BEING TESTED
AXNUM: .WORD 0 ;NUMBER (0-7) OF EXTENDED REG BYTE BEING TESTED
GOODAT: .WORD 0 ;STORAGE FOR EXPECTED DATA
BADDAT: .WORD 0 ;STORAGE FOR ACTUAL DATA
LOADAT: .WORD 0 ;CONTAINS TEST DATA LOADED INTO REG
FRSTIM: .WORD 0 ;FLAG=0 IF PROGRAM JUST LOADED
SAVE4: .WORD 0 ;SAVE LOC 4 HERE (ERROR TRAP VECTOR)
SAVE6: .WORD 0 ;SAVE LOC 6 HERE (ERROR TRAP VECTOR)
ERROR1: .WORD 0 ;SUBR ERROR BIT FLAGS (DEF'D IN GLOBAL EQUATES)
TXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA TO LOAD INTO TX SILO
RXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA READ FROM RCV SILO
DISILO: .WORD 0 ;CONTAINS CURRENT STATE OF DISSI IN BIT 5
CHPTYP: .WORD 0 ;USYRT CHIP TYPE, =0 FOR SIG, ELSE =1
SAVLEN: .WORD 0 ;SAVED TX AND RCV CHAR LENGTHS
DEVMAP: .WORD 0 ;BIT MAP OF ACTIVE DEVICES
DEVPTR: .WORD 0 ;DEVICE MAP BIT POINTER
UNIT: .WORD 0 ;CONTAINS UNIT NO. (1 TO N)
TSTNUM: .WORD 0 ;CONTAINS TEST NUMBER FOR SOME TESTS
STARES: .WORD 0 ;FLAG=0 IF FIRST PASS AFTER STA OR RES

:***** CURRENT DEVICE PARAMETERS *****
MPCSR: .WORD 160170 ;POINTER TO MICROPROCESSOR CSR'S
BSEL1: .WORD 160171 ;POINTER TO BSEL1
BSEL2: .WORD 160172 ;POINTER TO BSEL2
BSEL4:

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

```

2809 002432 160174 SEL4: .WORD 160174 ;POINTER TO SEL4
2810 002434 160176 SEL6: .WORD 160176 ;POINTER TO SEL6
2811 002436 000300 MPIVEC: .WORD 300 ;MICROPROCESSOR INPUT INTERRUPT VECTOR
2812 002440 000304 MPOVEC: .WORD 304 ;MICROPROCESSOR OUTPUT INTERRUPT VECTOR
2813 002442 000240 MPRIOR: .WORD 240 ;MICROPROCESSOR DEVICE PRIORITY
2814 002444 000000 LUSWI1: .WORD 0 ;LINE UNIT SWITCH PACK #1
2815 002446 000000 LUSWI2: .WORD 0 ;LINE UNIT SWITCH PACK #2
2816 002450 000000 LUSWI3: .WORD 0 ;LINE UNIT SWITCH PACK #3
2817 002452 000000 TSTCON: .WORD 0 ;TEST CONNECTOR INDICATOR
2818 002454 000000 RUNINH: .WORD 0 ;RUN SWITCH INDICATOR
2819
2820 ;***** STORAGE FOR DATA READ IN ADDRESS TESTS *****
2821 002456 000 REDDAT: .BYTE 0
2822 002457 000 .BYTE 0
2823 002460 000 .BYTE 0
2824 002461 000 .BYTE 0
2825 002462 000 .BYTE 0
2826 002463 000 .BYTE 0
2827 002464 000 .BYTE 0
2828 002465 000 .BYTE 0
2829
2830 ;***** GEN'L PURPOSE SCRATCH STORAGE *****
2831 002466 000000 REG0: .WORD 0
2832 002470 000000 REG1: .WORD 0
2833 002472 000000 REG2: .WORD 0
2834 002474 000000 REG3: .WORD 0
2835 002476 000000 REG4: .WORD 0
2836 002500 000000 REG5: .WORD 0
2837 002502 000000 REG6: .WORD 0
2838 002504 000000 REG7: .WORD 0
2839
2840 ;***** SCRATCH STORAGE FOR MESSAGE REPORTING *****
2841 002506 000000 TMP0: .WORD 0
2842 002510 000000 TMP1: .WORD 0
2843 002512 000000 TMP2: .WORD 0
2844 002514 000000 TMP3: .WORD 0
2845 002516 000000 TMP4: .WORD 0
2846 002520 000000 TMP5: .WORD 0
2847 002522 000000 TMP6: .WORD 0
2848 002524 000000 TMP7: .WORD 0
2849
2850 ;***** INBUS LU REG BIT MASKS FOR UNPREDICTABLE BITS *****
2851 002526 UPBITS: .BYTE 377 ;MASK FOR REG 10
2852 002526 377 .BYTE 056 ;MASK FOR REG 11
2853 002527 056 .BYTE 017 ;MASK FOR REG 12
2854 002530 017 .BYTE 257 ;MASK FOR REG 13
2855 002531 257 .BYTE 100 ;MASK FOR REG 14
2856 002532 100 .BYTE 377 ;MASK FOR REG 15
2857 002533 377 .BYTE 377 ;MASK FOR REG 16
2858 002534 377 .BYTE 306 ;MASK FOR REG 17
2859 002535 306
2860
2861 002536 200 R14NRW: .BYTE 200 ;REG 14 NON-R/W BITS
2862
2863 ;***** MASKS FOR EXTENDED REGISTER NON-READ/WRITE BITS *****
2864 002537 ANBITS:

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

2865	002537	377	.BYTE	377	:MASK FOR AX0-15
2866	002540	377	.BYTE	377	:MASK FOR AX0-16
2867	002541	000	.BYTE	000	:MASK FOR AX1-15
2868	002542	360	.BYTE	360	:MASK FOR AX1-16
2869	002543	000	.BYTE	000	:MASK FOR AX2-15
2870	002544	000	.BYTE	000	:MASK FOR AX2-16
2871	002545	004	.BYTE	004	:MASK FOR AX3-15
2872	002546	030	.BYTE	030	:MASK FOR AX3-16

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

PATA:

2875	002547				
2876	002547	125	.BYTE	125	
2877	002550	252	.BYTE	252	
2878	002551	000	.BYTE	000	
2879	002552	377	.BYTE	377	
2880	002553	001	.BYTE	001	
2881	002554	002	.BYTE	002	
2882	002555	004	.BYTE	004	
2883	002556	010	.BYTE	010	
2884	002557	020	.BYTE	020	
2885	002560	040	.BYTE	040	
2886	002561	100	.BYTE	100	
2887	002562	200	.BYTE	200	
2888	002563	376	.BYTE	376	
2889	002564	375	.BYTE	375	
2890	002565	373	.BYTE	373	
2891	002566	367	.BYTE	367	
2892	002567	357	.BYTE	357	
2893	002570	337	.BYTE	337	
2894	002571	277	.BYTE	277	
2895	002572	177	.BYTE	177	

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

PATB:

2898	002573				
2899	002573	000	.BYTE	000	
2900	002574	000	.BYTE	000	
2901	002575	040	.BYTE	040	
2902	002576	100	.BYTE	100	
2903	002577	220	.BYTE	220	
2904	002600	000	.BYTE	000	
2905	002601	000	.BYTE	000	
2906	002602	051	.BYTE	051	

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

PATC:

2909	002603				
2910	002603	020	.BYTE	020	
2911	002604	020	.BYTE	020	
2912	002605	020	.BYTE	020	

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

PATD:

2915	002606				
2916	002606	000	.BYTE	000	
2917	002607	040	.BYTE	040	
2918	002610	000	.BYTE	000	

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

2919
2920

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

2921 002611
 2922 002611 000
 2923 002612 120
 2924 002613 020
 2925 002614 100
 2926 002615 120
 2927 002616 000
 2928
 2929
 2930 002617
 2931 002617 050
 2932 002620 051
 2933 002621 050
 2934
 2935
 2936 002622
 2937 002622 000
 2938 002623 000
 2939 002624 240
 2940 002625 120
 2941 002626 177
 2942 002627 000
 2943 002630 000
 2944 002631 001
 2945
 2946
 2947 002632
 2948 002632 000
 2949 002633 000
 2950 002634 377
 2951 002635 017
 2952 002636 377
 2953 002637 377
 2954 002640 375
 2955 002641 377
 2956
 2957
 2958 002642
 2959 002642 000
 2960 002643 000
 2961 002644 000
 2962 002645 000
 2963 002646 000
 2964 002647 103
 2965 002650 000
 2966 002651 000
 2967
 2968
 2969 002652
 2970 002652 000
 2971 002653 000
 2972 002654 010
 2973 002655 002
 2974 002656 004
 2975 002657 103
 2976 002660 001

PATE:
 .BYTE 000
 .BYTE 120
 .BYTE 020
 .BYTE 100
 .BYTE 120
 .BYTE 000

***** DATA PATTERN F *****
 PATF:
 .BYTE 050
 .BYTE 051
 .BYTE 050

***** DATA PATTERN G *****
 PATG:
 .BYTE 000
 .BYTE 000
 .BYTE 240
 .BYTE 120
 .BYTE 177
 .BYTE 000
 .BYTE 000
 .BYTE 001

***** DATA PATTERN H *****
 PATH:
 .BYTE 000
 .BYTE 000
 .BYTE 377
 .BYTE 017
 .BYTE 377
 .BYTE 377
 .BYTE 375
 .BYTE 377

***** DATA PATTERN I *****
 PATI:
 .BYTE 000
 .BYTE 000
 .BYTE 000
 .BYTE 000
 .BYTE 000
 .BYTE 000
 .BYTE 103
 .BYTE 000
 .BYTE 000

***** DATA PATTERN J *****
 PATJ:
 .BYTE 000
 .BYTE 000
 .BYTE 010
 .BYTE 002
 .BYTE 004
 .BYTE 103
 .BYTE 001

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

2977	002661	100	.BYTE	100
2978				
2979				
2980	002662	000		
2981	002663	000		
2982	002664	377		
2983	002665	377		
2984	002666	125		
2985	002667	125		
2986	002670	252		
2987	002671	252		
2988	002672	001		
2989	002673	000		
2990	002674	002		
2991	002675	000		
2992	002676	004		
2993	002677	000		
2994	002700	010		
2995	002701	000		
2996	002702	020		
2997	002703	000		
2998	002704	040		
2999	002705	000		
3000	002706	100		
3001	002707	000		
3002	002710	200		
3003	002711	000		
3004	002712	000		
3005	002713	001		
3006	002714	000		
3007	002715	002		
3008	002716	000		
3009	002717	004		
3010	002720	000		
3011	002721	010		
3012	002722	000		
3013	002723	020		
3014	002724	000		
3015	002725	040		
3016	002726	000		
3017	002727	100		
3018	002730	000		
3019	002731	200		
3020	002732	376		
3021	002733	377		
3022	002734	375		
3023	002735	377		
3024	002736	373		
3025	002737	377		
3026	002740	367		
3027	002741	377		
3028	002742	357		
3029	002743	377		
3030	002744	337		
3031	002745	377		
3032	002746	277		

***** DATA PATTERN K *****
 PATK: .BYTE 000

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

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

3033	002747	377	.BYTE	377
3034	002750	177	.BYTE	177
3035	002751	377	.BYTE	377
3036	002752	377	.BYTE	377
3037	002753	376	.BYTE	376
3038	002754	377	.BYTE	377
3039	002755	375	.BYTE	375
3040	002756	377	.BYTE	377
3041	002757	373	.BYTE	373
3042	002760	377	.BYTE	377
3043	002761	367	.BYTE	367
3044	002762	377	.BYTE	377
3045	002763	357	.BYTE	357
3046	002764	377	.BYTE	377
3047	002765	337	.BYTE	337
3048	002766	377	.BYTE	377
3049	002767	277	.BYTE	277
3050	002770	377	.BYTE	377
3051	002771	177	.BYTE	177
3052				
3053				
3054	002772		:***** DATA PATTERN L *****	
3055	002772	000	PATL:	
3056	002773	000	.BYTE	000
3057	002774	377	.BYTE	377
3058	002775	377	.BYTE	377
3059	002776	000	.BYTE	000
3060	002777	000	.BYTE	000
3061				
3062				
3063	003000		:***** DATA PATTERN M *****	
3064	003000	000	PATM:	
3065	003001	020	.BYTE	000
3066	003002	000	.BYTE	020
3067	003003	000	.BYTE	000
3068	003004	200	.BYTE	000
3069	003005	000	.BYTE	200
3070	003006	000	.BYTE	000
3071	003007	051	.BYTE	000
3072			.BYTE	051
3073				
3074	003010		:***** DATA PATTERN N *****	
3075	003010	000	PATN:	
3076	003011	000	.BYTE	000
3077	003012	000	.BYTE	000
3078	003013	125	.BYTE	000
3079	003014	000	.BYTE	125
3080	003015	252	.BYTE	000
3081	003016	000	.BYTE	252
3082	003017	377	.BYTE	000
3083	003020	005	.BYTE	377
3084	003021	000	.BYTE	005
3085	003022	012	.BYTE	000
3086	003023	000	.BYTE	012
3087	003024	017	.BYTE	000
3088	003025	000	.BYTE	017
			.BYTE	000

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

3089			
3090			
3091	003026		
3092	003026	000	
3093	003027	041	
3094	003030	004	
3095	003031	010	
3096	003032	020	
3097	003033	040	
3098	003034	100	
3099	003035	101	
3100	003036	200	
3101	003037	201	
3102	003040	300	
3103	003041	111	
3104	003042	301	
3105	003043	375	
3106			
3107			
3108	003044		
3109	003044	000	
3110	003045	113	
3111	003046	200	
3112	003047	040	
3113	003050	020	
3114	003051	010	
3115	003052	001	
3116	003053	104	
3117	003054	007	
3118	003055	105	
3119	003056	007	
3120	003057	144	
3121	003060	107	
3122	003061	157	
3123			
3124			
3125	003062	000	
3126	003063	000	
3127	003064	001	
3128	003065	000	
3129	003066	013	
3130	003067	000	
3131	003070	011	
3132	003071	000	
3133	003072	021	
3134	003073	000	
3135	003074	101	
3136	003075	000	
3137	003076	301	
3138	003077	000	
3139	003100	000	
3140	003101	001	
3141	003102	000	
3142	003103	002	
3143	003104	000	
3144	003105	004	

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

PATO:

.BYTE	000
.BYTE	041
.BYTE	004
.BYTE	010
.BYTE	020
.BYTE	040
.BYTE	100
.BYTE	101
.BYTE	200
.BYTE	201
.BYTE	300
.BYTE	111
.BYTE	301
.BYTE	375

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

PATP:

.BYTE	000
.BYTE	113
.BYTE	200
.BYTE	040
.BYTE	020
.BYTE	010
.BYTE	001
.BYTE	104
.BYTE	007
.BYTE	105
.BYTE	007
.BYTE	144
.BYTE	107
.BYTE	157

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

PATU:

.BYTE	000
.BYTE	000
.BYTE	001
.BYTE	000
.BYTE	013
.BYTE	000
.BYTE	011
.BYTE	000
.BYTE	021
.BYTE	000
.BYTE	101
.BYTE	000
.BYTE	301
.BYTE	000
.BYTE	000
.BYTE	001
.BYTE	000
.BYTE	002
.BYTE	000
.BYTE	004

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

3145	003106	000	.BYTE	000
3146	003107	040	.BYTE	040
3147	003110	000	.BYTE	000
3148	003111	100	.BYTE	100
3149	003112	000	.BYTE	000
3150	003113	200	.BYTE	200
3151	003114	000	.BYTE	000
3152	003115	346	.BYTE	346
3153	003116	000	.BYTE	000
3154	003117	345	.BYTE	345
3155	003120	000	.BYTE	000
3156	003121	343	.BYTE	343
3157	003122	000	.BYTE	000
3158	003123	307	.BYTE	307
3159	003124	000	.BYTE	000
3160	003125	247	.BYTE	247
3161	003126	000	.BYTE	000
3162	003127	147	.BYTE	147
3163				
3164				
3165	003130	000		
3166	003131	000		
3167	003132	333		
3168	003133	000		
3169	003134	331		
3170	003135	000		
3171	003136	323		
3172	003137	000		
3173	003140	313		
3174	003141	000		
3175	003142	233		
3176	003143	000		
3177	003144	133		
3178	003145	000		
3179	003146	000		
3180	003147	001		
3181	003150	000		
3182	003151	002		
3183	003152	000		
3184	003153	004		
3185	003154	000		
3186	003155	040		
3187	003156	000		
3188	003157	100		
3189	003160	000		
3190	003161	200		
3191	003162	000		
3192	003163	346		
3193	003164	000		
3194	003165	345		
3195	003166	000		
3196	003167	343		
3197	003170	000		
3198	003171	307		
3199	003172	000		
3200	003173	247		

***** PATTERN V *****

PATV:	.BYTE	000
	.BYTE	000
	.BYTE	333
	.BYTE	000
	.BYTE	331
	.BYTE	000
	.BYTE	323
	.BYTE	000
	.BYTE	313
	.BYTE	000
	.BYTE	233
	.BYTE	000
	.BYTE	133
	.BYTE	000
	.BYTE	000
	.BYTE	001
	.BYTE	000
	.BYTE	002
	.BYTE	000
	.BYTE	004
	.BYTE	000
	.BYTE	040
	.BYTE	000
	.BYTE	100
	.BYTE	000
	.BYTE	200
	.BYTE	000
	.BYTE	346
	.BYTE	000
	.BYTE	345
	.BYTE	000
	.BYTE	343
	.BYTE	000
	.BYTE	307
	.BYTE	000
	.BYTE	247

CZDMRF.P11 03-NOV-81 10:29

GLOBAL DATA SECTION

3201 003174 000
 3202 003175 147
 3203
 3204 003176
 3205
 3206
 3207
 3208
 3209
 3210
 3211
 3212
 3213 003176 000400
 3214 003200 000400
 3215 003202 000000
 3216 003204 000125
 3217 003206 000252
 3218 003210 000377
 3219 003212 000000
 3220 003214 001000
 3221 003216 001000
 3222 003220 001000
 3223 003222 001000
 3224
 3225 003224 000377
 3226 003226 000000
 3227 003230 000125
 3228 003232 000252
 3229 003234 001000
 3230 003236 001000
 3231
 3232
 3233
 3234
 3235
 3236
 3237 003240 000100
 3238
 3239
 3240
 3241
 3242
 3243
 3244
 3245
 3246
 3247

.BYTE 000
.BYTE 147

ENDPAT:
.EVEN

;*** 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.

GLOBAL SUBROUTINES

3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347

003516			
003516	152777	000006	176702
003524	017677	000000	176702
003532	152777	000007	176666
003540	142777	000007	176660
003546	062716	000002	
003552	000207		
003554			
003554	010146		
003556	013746	002356	
003562	112777	000100	176636
003570	142777	000300	176630
003576	012701	000024	
003602	000240		
003604	005301		
003606	001375		
003610	152777	000010	176610
003616	012737	000013	002356
003624	005037	002344	
003630	004737	003726	
003634	012637	002356	
003640	012601		
003642	005037	002410	
003646	000207		

.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 LULOOK

```

MSTCLR:
MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE LU REG NO.
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 #LULOOK,@BSEL1 ;SET LU LOOP
MOV #13,REGNUM ;SET LU REG NO. = 13
CLR WRIBYT
JSR PC,WRITLU ;CLEAR REG 13
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
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.

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3348 003650
 3349 003650 010146
 3350 003652 013701 002356
 3351 003656 006301
 3352 003660 006301
 3353 003662 006301
 3354 003664 006301
 3355 003666 052701 000004
 3356 003672 052701 021000
 3357 003676 010137 003706
 3358 003702 004737 003516
 3359 003706 000000
 3360 003710 117737 176516 002342
 3361 003716 105037 002343
 3362 003722 012601
 3363 003724 000207

```

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
ASL R1
BIS #4,R1 ;SET DESTINATION = BSEL4
BIS #MVI0X,R1 ;SET REST OF MOVE INSTRUCTION
MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION
;INSTRUCTION GOES HERE
2$: .WORD 0 ;GET LU REG CONTENTS INTO REDBYT
MOV B @BSEL4,REDBYT ;CLR HI BYTE OF STORAGE
CLRB REDBYT+1 ;RESTORE R1
MOV (SP)+,R1 ;RETURN
RTS PC

```

```

;*****
;* WRITLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO
;* EXECUTE AN INSTRUCTION WHICH LOADS THE BYTE CONTAINED IN WRIBYT
;* INTO THE LU REG WHOSE NUMBER IS PASSED IN REGNUM,
;*****

```

3374 003726
 3375 003726 010146
 3376 003730 013701 002356
 3377 003734 052701 000100
 3378 003740 052701 122000
 3379 003744 010137 003766
 3380 003750 105037 002345
 3381 003754 113777 002344 176450
 3382 003762 004737 003516
 3383 003766 000000
 3384 003770 012601
 3385 003772 000207

```

WRITLU:
MOV R1,-(SP) ;SAVE R1
MOV REGNUM,R1 ;GET LINE UNIT REG NUMBER
BIS #100,R1 ;SET SOURCE = BSEL4
BIS #MVI0X,R1 ;SET REST OF MOVE INSTRUCTION
MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
CLRB WRIBYT+1 ;CLR HI BYTE OF STORAGE
MOV B WRIBYT,@BSEL4 ;LOAD BYTE INTO BSEL4
JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION
2$: .WORD 0
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

```

```

;*****
;* GETREG - THIS SUBROUTINE READS THE LINE UNIT REGISTERS 10-17 INTO THE
;* REGISTER STORAGE TABLE (LUREG:).
;*****

```

3395 003774 010146
 3396 003776 013746 002356
 3397 004002 012701 002260
 3398 004006 012737 000010 002356
 3399 004014 004737 003650
 3400 004020 113721 002342
 3401 004024 105021
 3402 004026 005237 002356
 3403 004032 023727 002356 000020

```

GETREG:
MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE CURRENT REG NO.
MOV #LUR10,R1 ;INIT POINTER TO REG STORAGE TABLE
MOV #10,REGNUM ;INIT LU REG NO. TO 10
3$: JSR PC,READLU ;READ A LINE UNIT REG
MOV B REDBYT,(R1)+ ;PUT BYTE READ INTO TABLE
CLRB (R1)+ ;CLEAR UPPER BYTE OF TABLE ENTRY
INC REGNUM ;INCREMENT REG NO.
CMP REGNUM,#20 ;SEE IF ALL REGS READ YET

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3404 004040 002765
 3405 004042 012637 002356
 3406 004046 012601
 3407 004050 000207
 3408
 3409
 3410
 3411
 3412
 3413

```
BLT 3$ ;BR IF NOT
MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN
```

3414
 3415
 3416
 3417
 3418
 3419
 3420

```
*****
;* 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.
*****
```

3421 004052
 3422 004052 152777 000006 176346
 3423 004060 017677 000000 176346
 3424 004066 152777 000206 176332
 3425 004074 062716 000002
 3426 004100 000207

```
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
```

3427
 3428
 3429
 3430
 3431
 3432
 3433

```
*****
;* 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.
*****
```

3439 004102 010146
 3440 004104 013746 002356
 3441 004110 042737 000001 002376
 3442 004116 012737 000014 002356
 3443 004124 113737 002360 002344
 3444 004132 006237 002344
 3445 004136 152737 000024 002344
 3446 004144 053737 002404 002344
 3447 004152 004737 003726
 3448 004156 005001
 3449 004160 004737 003650
 3450 004164 132737 000200 002342
 3451 004172 001006
 3452 004174 005201
 3453 004176 001370
 3454 004200 052737 000001 002376
 3455 004206 000424
 3456 004210 012737 000015 002356
 3457 004216 004737 003650
 3458 004222 113737 002342 002346
 3459 004230 105037 002347

```
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
```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3460	004234	012737	000016	002356
3461	004242	004737	003650	
3462	004246	113737	002342	002350
3463	004254	105037	002351	
3464	004260	012637	002356	
3465	004264	012601		
3466	004266	000207		

```

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

```

3467
3468
3469
3470
3471
3472
3473
3474
3475
3476

```

*****
;* 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.
*****

```

3477	004270	010146		
3478	004272	013746	002356	
3479	004276	042737	000002	002376
3480	004304	012737	000014	002356
3481	004312	113737	002360	002344
3482	004320	006237	002344	
3483	004324	053737	002404	002344
3484	004332	004737	003726	
3485	004336	012737	000015	002356
3486	004344	105037	002353	
3487	004350	113737	002352	002344
3488	004356	004737	003726	
3489	004362	005237	002356	
3490	004366	105037	002355	
3491	004372	113737	002354	002344
3492	004400	004737	003726	
3493	004404	012737	000014	002356
3494	004412	113737	002360	002344
3495	004420	006237	002344	
3496	004424	152737	000014	002344
3497	004432	053737	002404	002344
3498	004440	004737	003726	
3499	004444	005001		
3500	004446	004737	003650	
3501	004452	132737	000200	002342
3502	004460	001005		
3503	004462	005201		
3504	004464	001370		
3505	004466	052737	000002	002376
3506	004474	012637	002356	
3507	004500	012601		
3508	004502	000207		

```

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

```

3509
3510
3511
3512
3513
3514
3515

```

*****
;* GETALL - THIS SUBROUTINE READS THE LINE UNIT REGS 10-17 AND THE EXTENDED

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3516
3517
3518 004504 010146
3519 004506 013746 002360
3520 004512 012737 014413 002506
3521 004520 032737 000001 002360
3522 004526 001403
3523 004530 012737 014416 002506
3524 004536 004737 003774
3525 004542 142777 000010 175656
3526 004550 012701 002300
3527 004554 005037 002360
3528 004560 004737 004102
3529 004564 113721 002346
3530 004570 105021
3531 004572 113721 002350
3532 004576 105021
3533 004600 062737 000002 002360
3534 004606 023727 002360 000010
3535 004614 002761
3536 004616 012637 002360
3537 004622 012601
3538 004624 013737 002360 002510
3539 004632 006237 002510
3540 004636 000207
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554 004640 013746 002356
3555 004644 013746 002330
3556 004650 005737 002330
3557 004654 001006
3558 004656 016637 000004 002330
3559 004664 162737 000004 002330
3560 004672 012737 000011 002356
3561 004700 004737 003650
3562 004704 032776 000001 000004
3563 004712 001413
3564 004714 132737 000020 002342
3565 004722 001022
3566 004724 004737 004504
3567
3568 004730
3569 004730 104455
3570 004732 000007
3571 004734 012352

```

```

;*   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,TMPO        ;SET AX LO BYTE NO.
        BIT   #BIT0,AXNUM      ;SEE IF LO OR HI BYTE
        BEQ   1$              ;BR IF LO BYTE
        MOV   #DH6,TMPO        ;SET AX HI BYTE NO.
        JSR   PC,GETREG        ;READ AND STORE REGS 10-17
        BICB  #LLOOP,@BSEL1    ;CLEAR LLOOP
        MOV   #AX0.15,R1      ;INIT POINTER TO REG STORAGE TABLE
        CLR   AXNUM            ;INIT AX REG BYTE NO. TO 0
3$:     JSR   PC,READAX        ;READ 2 AX REG BYTES
        MOVB  RAX15,(R1)+      ;PUT LO BYTE READ INTO TABLE
        CLRB  (R1)+           ;CLEAR UPPER BYTE OF TABLE ENTRY
        MOVB  RAX16,(R1)+      ;PUT HI BYTE READ INTO TABLE
        CLRB  (R1)+           ;CLEAR UPPER BYTE OF TABLE ENTRY
        ADD   #2,AXNUM         ;INCR AX REG BYTE NO.
        CMP   AXNUM,#10       ;SEE IF ALL REGS READ YET
        BLT   3$              ;BR IF NOT
        MOV   (SP)+,AXNUM      ;RESTORE CURRENT AX REG BYTE NO.
        MOV   (SP)+,R1        ;RESTORE R1
        MOV   AXNUM,TMP1
        ASR   TMP1             ;GET EXTENDED REG NO. FOR PRINTOUT
        RTS   PC              ;RETURN

```

```

;*****
;* 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.
;*****

```

```

OSIRDY: MOV   REGNUM,-(SP)      ;SAVE LU REG NO.
        MOV   SUBRPC,-(SP)
        TST   SUBRPC          ;SEE IF THIS IS A NESTED CALL
        BNE  1$              ;BR IF YES
        MOV   4(SP),SUBRPC     ;GET PC OF SUBROUTINE CALL
        SUB   #4,SUBRPC
        MOV   #11,REGNUM      ;SET REG NO. TO 11
        JSR   PC,READLU       ;READ REG 11
        BIT   #BIT0,@4(SP)    ;GET EXPECTED STATE OF ORDY
        BEQ   3$              ;BR IF EXPECTED ORDY = 0
        BITB #ORDY,REDBYT     ;SEE IF ORDY = 1
        BNE  9$              ;BR IF ORDY = 1
        JSR   PC,GETALL       ;GET REGS FOR PRINTOUT
;REPORT ORDY NOT SET
ERRDF  7,EM7,ERR4

```

```

TRAP  CSERDF
.WORD 7
.WORD EM7

```


CZDMRF .P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3572 004736 015600                                     .WORD  ERR4
3573 004740 000451
3574 004742 132737 000020 002342 3$:  BR      16$          ;TAKE ERROR RETURN
3575 004750 001407          BITB   #ORDY,REDBYT ;SEE IF ORDY = 0
3576 004752 004737 004504          BEQ    9$          ;BR IF ORDY = 0
3577          JSR    PC,GETALL ;GET REGS FOR PRINTOUT
3578          ;REPORT ORDY NOT CLEARED
3579          ERRDF  8,EM8,ERR4
3580 004756 104455                                     TRAP   C$ERDF
3581 004760 000010                                     .WORD  8
3582 004762 012367                                     .WORD  EM8
3583 004764 015600                                     .WORD  ERR4
3584 004766 000436
3585 004770 012737 000017 002356 9$:  BR      16$          ;TAKE ERROR RETURN
3586 004776 004737 003650          MOV    #17,REGNUM ;SET REG NO. = 17
3587 005002 132776 000002 000004          JSR    PC,READLU  ;READ LU REG 17
3588 005010 001413          BITB   #BIT1,24(SP) ;GET EXPECTED STATE OF OCOR
3589 005012 132737 000020 002342          BEQ    12$          ;BR IF EXPECTED OCOR = 0
3590 005020 001031          BITB   #OCOR,REDBYT ;SEE IF OCOR = 1
3591 005022 004737 004504          BNE    20$          ;BR IF OCOR = 1
3592          JSR    PC,GETALL ;GET REGS FOR PRINTOUT
3593          ;REPORT OCOR NOT SET
3594          ERRDF  9,EM9,ERR4
3595 005026 104455                                     TRAP   C$ERDF
3596 005030 000011                                     .WORD  9
3597 005032 012410                                     .WORD  EM9
3598 005034 015600                                     .WORD  ERR4
3599 005036 000412
3600 005040 132737 000020 002342 12$: BR      16$          ;TAKE ERROR RETURN
3601 005046 001416          BITB   #OCOR,REDBYT ;SEE IF OCOR = 0
3602 005050 004737 004504          BEQ    20$          ;BR IF OCOR = 0
3603          JSR    PC,GETALL ;GET REGS FOR PRINTOUT
3604          ;REPORT OCOR NOT CLEARED
3605          ERRDF  10,EM10,ERR4
3606 005054 104455                                     TRAP   C$ERDF
3607 005056 000012                                     .WORD  10
3608 005060 012425                                     .WORD  EM10
3609 005062 015600                                     .WORD  ERR4
3610 005064 016637 000002 002356 16$: MOV    2(SP),REGNUM ;RESTORE LU REG NO.
3611 005072 013706 002324          MOV    PSTACK,SP  ;RESTORE STACK POINTER TO BASE LEVEL
3612 005076 013746 002340          MOV    RETADR,-(SP) ;FIX ERROR RETURN PC
3613 005102 000407          BR      23$
3614 005104 062766 000002 000004 20$: ADD    #2,4(SP)    ;FIX UP ERROR-FREE RETURN PC
3615 005112 012637 002330          MOV    (SP)+,SUBRPC
3616 005116 012637 002356          MOV    (SP)+,REGNUM ;RESTORE LU REG NO.
3617 005122 000207          RTS     PC         ;RETURN
3618
3619
3620
3621          ;*****
3622          ;* WAIT50 - THIS SUBROUTINE STALLS FOR AT LEAST 50 MICRO-SEC, AND THEN RETURNS.
3623          ;*****
3624 005124 010146 000310 WAIT50: MOV    R1,-(SP) ;SAVE R1
3625 005126 012701          MOV    #200.,R1   ;INIT COUNTER
3626 005132 005301          3$:  DEC    R1     ;DECREMENT COUNTER
3627 005134 001376          BNE    3$         ;BR IF NOT DONE YET
3628 005136 012601          MOV    (SP)+,R1   ;RESTORE R1

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3628 005140 000207 RTS PC ;RETURN

3629

3630

3631

3632

3633

3634

3635

3636

3637 005142 000240

3638 005144 000240

3639 005146 000240

3640 005150 000207

3641

3642

3643

3644

3645

3646

3647

3648

3649

3650 005152 013746 002356

3651 005156 042737 170000 002400

3652 005164 012737 000011 002356

3653 005172 113737 002401 002344

3654 005200 004737 003726

3655 005204 012737 000010 002356

3656 005212 113737 002400 002344

3657 005220 004737 003726

3658 005224 012637 002356

3659

3660

3661

3662

3663

3664

3665

3666

3667

3668

3669

3670

3671 005232 010146 000002

3672 005234 017601 000002

3673 005240 001426 000002

3674 005242 100006 000002

3675 005244 042701 100000

3676 005250 005737 002406

3677 005254 001401 000002

3678 005256 005301 000010 175140 2\$:

3679 005260 152777 000020 175132 3\$:

3680 005266 152777 000020 175132

3681 005274 004737 005142

3682 005300 142777 000020 175120

3683 005306 004737 005142

 ;* STALL - THIS SUBROUTINE STALLS FOR ABOUT A MICRO-SEC.
 ;*
 ;*
 ;*

STALL: NOP
 NOP
 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

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3684 005312 005301          DEC      R1          ;DECREMENT CYCLE COUNTER
3685 005314 001364          BNE      3$          ;BR IF NOT DONE YET
3686 005316 062766 000002 000002 6$:  ADD      #2,2(SP)    ;FIX UP RETURN PC
3687 005324 012601          MOV      (SP)+,R1    ;RESTORE R1
3688 005326 000207          RTS       PC         ;RETURN

```

3689
3690
3691
3692
3693

```

3694 :*****
3695 :* OACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF OACT (REG 11) AND
3696 :* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
3697 :* WORD FOLLOWING THE CALL.
3698 :*****

```

```

3699 005330 013746 002356  OACTIV: MOV      REGNUM,-(SP) ;SAVE LU REG NO.
3700 005334 013746 002330  MOV      SUBRPC,-(SP)
3701 005340 005737 002330  TST      SUBRPC      ;SEE IF THIS IS A NESTED CALL
3702 005344 001006          BNE      1$          ;BR IF YES
3703 005346 016637 000004 002330  MOV      4(SP),SUBRPC
3704 005354 162737 000004 002330  SUB      #4,SUBRPC    ;GET PC OF SUBROUTINE CALL
3705 005362 012737 000011 002356  1$:  MOV      #11,REGNUM   ;SET REG NO. = 11
3706 005370 004737 003650          JSR      PC,READLU   ;READ REG 11
3707 005374 032776 000001 000004  BIT      #BIT0,24(SP) ;GET EXPECTED STATE OF OACT
3708 005402 001413          BEQ      3$          ;BR IF EXPECTED OACT = 0
3709 005404 132737 000100 002342  BITB     #OACT,REDBYT ;SEE IF OACT = 1
3710 005412 001031          BNE      9$          ;BR IF OACT = 1
3711 005414 004737 004504          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
3712 :REPORT OACT NOT SET

```

```

3713 005420          ERRDF  11,EM11,ERR4
3714 005420 104455          TRAP   CSERDF
3715 005422 000013          .WORD  11
3716 005424 012446          .WORD  EM11
3717 005426 015600          .WORD  ERR4

```

```

3718 005430 000412          BR       6$          ;TAKE ERROR RETURN
3719 005432 132737 000100 002342  3$:  BITB     #OACT,REDBYT ;SEE IF OACT = 0
3720 005440 001416          BEQ      9$          ;BR IF OACT = 0
3721 005442 004737 004504          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
3722 :REPORT OACT NOT CLEARED

```

```

3723 005446          ERRDF  12,EM12,ERR4
3724 005446 104455          TRAP   CSERDF
3725 005450 000014          .WORD  12
3726 005452 012463          .WORD  EM12
3727 005454 015600          .WORD  ERR4

```

```

3728 005456 016637 000002 002356  6$:  MOV      2(SP),REGNUM ;RESTORE LU REG NO.
3729 005464 013706 002324          MOV      PSTACK,SP   ;RESTORE PROGRAM STACK TO BASE LEVEL
3730 005470 013746 002340          MOV      RETADR,-(SP) ;FIX UP ERROR RETURN PC
3731 005474 000407          BR       12$         ;
3732 005476 062766 000002 000004  9$:  ADD      #2,4(SP)    ;FIX UP ERROR-FREE RETURN PC
3733 005504 012637 002330          MOV      (SP)+,SUBRPC
3734 005510 012637 002356          MOV      (SP)+,REGNUM ;RESTORE LU REG NO.
3735 005514 000207 12$:  RTS       PC         ;RETURN

```

3736
3737
3738
3739

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795

005516 010146
005520 013746 002356
005524 013746 002360
005530 016637 000006 002330
005536 162737 000004 002330
005544 004737 003554
005550 004737 004640
005554 000001
005556 004737 005330
005562 000000
005564 012737 000004 002360
005572 117637 000006 002352
005600 012737 000400 002400
005606 113737 002352 002400
005614 005037 002354
005620 004737 004270
005624 012737 000017 002356
005632 062766 000002 000006
005640 117637 000006 002344
005646 004737 003726
005652 004737 005152
005656 004737 005152
005662 004737 005124
005666 004737 004640
005672 000003
005674 004737 005330
005700 000000
005702 005001
005704 012737 000011 002356
005712 152777 000010 174506 6\$:
005720 152777 000020 174500
005726 004737 005142
005732 004737 003650
005736 132737 000100 002342
005744 001014
005746 142777 000020 174452
005754 004737 005142
005760 005201
005762 020127 000003
005766 002751
005770 004737 005330
005774 000001
005776 012737 000017 002356 9\$:
006004 005037 002406
006010 004737 003650

```
*****
* INITRN - THIS SUBROUTINE INITIATES TRANSMISSION OF A MESSAGE, BY DOING A
* MASTER CLEAR, LOADING AX2-15 AND REG 17 WITH THE DATA PASSED IN THE 2
* WORDS FOLLOWING THE CALL, LOADING 2 SOM CHARS INTO THE TX SILO, AND
* CLOCKING THE LINE UNIT UNTIL THE FIRST SYNCH OR FLAG HAS BEEN SERIALIZED
* IN THE USYRT. THE PROGRAM MONITORS ORDY,OCOR, AND OACT FOR VALID STATES,
* THROUGHOUT THE PROCESS.
* IF THE SUBROUTINE DETECTS AN ERROR, A RETURN IS MADE TO THE TEST, AT THE
* ADDRESS CONTAINED IN RETADR.
*****
INITRN: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV AXNUM,-(SP) ;SAVE AX BYTE NO.
MOV 6(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBR CALL
JSR PC,MSTCLR ;ISSUE A MASTER CLEAR
JSR PC,OSIRDY ;CHECK ORDY=1, OCOR=0
1
JSR PC,OACTIV ;CHK OACT=0
0
MOV #4,AXNUM ;SET AX BYTE NO. = 4 FOR AX2
MOVB @6(SP),WAX15 ;SET DATA BYTE TO LOAD INTO AX2-15
MOV #TXSOM,TXWORD ;SET TSOM BIT
MOVB WAX15,TXWORD ;SET SYNCH CHAR
CLR WAX16
JSR PC,WRITAX ;LOAD AX2
MOV #17,REGNUM ;SET REG NO. = 17
ADD #2,6(SP) ;INCR POINTER TO NEXT DATA BYTE
MOVB @6(SP),WRIBYT ;SET DATA BYTE TO LOAD INTO REG 17
JSR PC,WRITLU ;LOAD REG 17
JSR PC,LDTXSI ;LOAD THE SILO WITH SOM CHAR
JSR PC,LDTXSI ;LOAD ANOTHER SOM INTO SILO
JSR PC,WAIT50 ;WAIT FOR DATA TO RIPPLE
JSR PC,OSIRDY ;CHK ORDY=1, OCOR=1
3
JSR PC,OACTIV ;CHK FOR OACT = 0
0
CLR R1 ;INIT CYCLE COUNTER
MOV #11,REGNUM ;SET LU REG NO. = 11
BISB #LULUOOP,@BSEL1 ;SET LINE UNIT LOOP BIT
BISB #STEPLU,@BSEL1 ;SET CLOCK BIT
JSR PC,STALL ;STALL FOR MICRO-SEC
JSR PC,READLU ;READ REG 11
BITB #OACT,REDBYT ;SEE IF OACT = 1 YET
BNE 9$ ;BR IF OACT = 1
BICB #STEPLU,@BSEL1 ;CLEAR CLOCK BIT
JSR PC,STALL ;STALL FOR A MICRO-SEC
INC R1 ;INCR CYCLE COUNT
CMP R1,#3 ;SEE IF 3 CYCLES DONE YET
BLT 6$ ;BR IF NOT
JSR PC,OACTIV ;CHK FOR OACT = 1
1
MOV #17,REGNUM ;SET REG NO. = 17
CLR CHPTYP ;CLEAR USYRT CHIP INDICATOR
JSR PC,READLU ;READ REG 17
```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3796 006014 132737 000020 002342 BITB #OCOR,REDBYT ;CHK FOR OCOR CLEARED YET
3797 006022 001403 BEQ 12$ ;BR IF YES - IT IS SIG CHIP
3798 006024 012737 000001 002406 MOV #1,CHPTYP ;SET INDICATOR FOR OTHER CHIP TYPE
3799 006032 142777 000020 174366 12$: BICB #STPLU,@BSEL1 ;CLEAR CLOCK BIT
3800 006040 004737 005142 JSR PC,STALL ;STALL FOR MICRO-SEC
3801 006044 004737 004640 JSR PC,OSIRDY ;CHK FOR ORDY = 1, OCOR = 0
3802 006050 000001 1
3803 006052 062766 000002 000006 ADD #2,6(SP) ;FIX UP RETURN PC
3804 006060 012637 002360 MOV (SP)+,AXNUM ;RESTORE AX BYTE NO.
3805 006064 012637 002356 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
3806 006070 012601 MOV (SP)+,R1 ;RESTORE R1
3807 006072 005037 002330 CLR SUBRPC ;CLEAR SUBR CALL PC
3808 006076 000207 RTS PC ;RETURN
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824

```

```

*****
;* 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.
*****

```

```

3825 006100 010146 TXCHAR: MOV R1,-(SP) ;SAVE R1
3826 006102 010246 MOV R2,-(SP) ;SAVE R2
3827 006104 016637 000004 002330 MOV 4(SP),SUBRPC
3828 006112 162737 000004 002330 SUB #4,SUBRPC ;GET PC OF SUBR CALL
3829 006120 017637 000004 002400 MOV @4(SP),TXWORD ;GET DATA TO BE TRANSMITTED
3830 006126 004737 005152 JSR PC,LDTXSI ;LOAD THE TX SILO WITH THE DATA
3831 006132 004737 005124 JSR PC,WAIT50 ;WAIT FOR DATA TO RIPPLE DOWN SILO
3832 006136 062766 000002 000004 ADD #2,4(SP) ;INCR POINTER
3833 006144 005001 CLR R1 ;INIT CYCLE COUNT
3834 006146 017602 000004 MOV @4(SP),R2 ;GET DESIRED NO. OF CYCLES
3835 006152 005702 TST R2 ;SEE IF CHIP TYPE CHK SHOULD BE MADE
3836 006154 100006 BPL 9$ ;BR IF NOT
3837 006156 042702 100000 BIC #BIT15,R2 ;CLEAR FLAG BIT
3838 006162 005737 002406 TST CHPTYP ;SEE IF SIG USYRT
3839 006166 001401 BEQ 9$ ;BR IF YES
3840 006170 005302 DEC R2 ;DECREMENT NO. OF CYCLES
3841 006172 004737 005330 9$: JSR PC,OACTIV ;CHK OACT = 1
3842 006176 000001 1
3843 006200 020102 CMP R1,R2 ;SEE IF REQUIRED CYCLES DONE YET
3844 006202 001410 BEQ 12$ ;BR IF YES
3845 006204 004737 004640 JSR PC,OSIRDY ;CHK ORDY=1, OCOR=1
3846 006210 000003 3
3847 006212 004737 005232 JSR PC,STPLU ;STEP LU ONE CYCLE
3848 006216 000001 1
3849 006220 005201 INC R1 ;INCR CYCLE COUNT
3850 006222 000763 BR 9$
3851 006224 004737 004640 12$: JSR PC,OSIRDY ;CHK ORDY=1, OCOR=0

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

3852 006230 000001
 3853 006232 062766 000002 000004
 3854 006240 005037 002330
 3855 006244 012602
 3856 006246 012601
 3857 006250 000207

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

3858
 3859
 3860
 3861
 3862
 3863

 ;* 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.

3867 006252 010146
 3868 006254 013746 002356
 3869 006260 016637 000004 002330
 3870 006266 162737 000004 002330
 3871 006274 012737 000011 002356
 3872 006302 012737 000200 002344
 3873 006310 004737 003726
 3874 006314 004737 005124
 3875 006320 004737 004640
 3876 006324 000001
 3877 006326 004737 005330
 3878 006332 000000
 3879 006334 012737 000013 002356
 3880 006342 004737 003650
 3881 006346 032737 000040 002342
 3882 006354 001406
 3883 006356 004737 004504

ENDTRN: MOV R1,-(SP) ;SAVE R1
 MOV REGNUM,-(SP) ;SAVE LU REG NO.
 MOV 4(SP),SUBRPC
 SUB #4,SUBRPC ;GET PC OF SUBROUTINE CALL
 MOV #11,REGNUM ;SET LU REG NO. = 11
 MOV #OC,WRIBYT ;SET OC IN DATA
 JSR PC,WRITLU ;SET OC IN REG 11
 JSR PC,WAIT50 ;STALL FOR >50 US.
 JSR PC,OSIRDY ;CHK ORDY=1, OCOR=0
 1
 JSR PC,OACTIV ;CHK OACT = 0
 0
 MOV #13,REGNUM ;SET REG NO. = 13
 JSR PC,READLU ;READ REG 13
 BIT #RTS,REDBYT ;CHK FOR RTS = 0
 BEQ 3\$;BR IF RTS = 0
 JSR PC,GETALL ;GET REGS FOR PRINTOUT
 ;REPORT RTS NOT CLEARED
 ERRDF 65,EM65,ERR4

3884
 3885 006362
 3886 006362 104455
 3887 006364 000101
 3888 006366 014222
 3889 006370 015600
 3890 006372 005037 002330
 3891 006376 012637 002356
 3892 006402 012601
 3893 006404 000207

3\$: CLR SUBRPC ;CLEAR SUBR CALL PC
 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
 MOV (SP)+,R1 ;RESTORE R1
 RTS PC ;RETURN

TRAP C\$ERDF
 .WORD 65
 .WORD EM65
 .WORD ERR4

3894
 3895
 3896
 3897
 3898
 3899

 ;* ISIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ICIR (REG 17)
 ;* AND IRDY (REG 12) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET
 ;* AS PASSED IN BIT 0 (ICIR) AND BIT 1 (IRDY) OF THE WORD FOLLOWING THE
 ;* CALL.
 ;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS
 ;* IN RETADR.

3900
 3901
 3902
 3903
 3904
 3905
 3906
 3907 006406 013746 002356

ISIRDY: MOV REGNUM,-(SP) ;SAVE LU REG NO.

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3908 006412 013746 002330      MOV      SUBRPC, -(SP)
3909 006416 005737 002330      TST      SUBRPC          ;SEE IF THIS IS A NESTED CALL
3910 006422 001006                BNE      1$              ;BR IF YES
3911 006424 016637 000004 002330  MOV      4(SP), SUBRPC
3912 006432 162737 000004 002330  SUB      #4, SUBRPC      ;GET PC OF SUBR CALL
3913 006440 012737 000012 002356 1$:  MOV      #12, REGNUM     ;SET REG NO. TO 12
3914 006446 004737 003650        JSR      PC, READLU      ;READ REG 12
3915 006452 032776 000002 000004  BIT      #BIT1, @4(SP)   ;GET EXPECTED STATE OF IRDY
3916 006460 001413                BEQ      3$              ;BR IF EXPECTED IRDY = 0
3917 006462 132737 000020 002342  BITB     #IRDY, REDBYT   ;SEE IF IRDY = 1
3918 006470 001022                BNE      9$              ;BR IF IRDY = 1
3919 006472 004737 004504        JSR      PC, GETALL      ;GET REGS FOR PRINTOUT
3920                                ;REPORT IRDY NOT SET
3921 006476                                ERRDF    17, EM17, ERR4
3922 006476 104455                                TRAP    C$ERDF
3923 006500 000021                                .WORD  17
3924 006502 012621                                .WORD  EM17
3925 006504 015600                                .WORD  ERR4
3926 006506 000451
3927 006510 132737 000020 002342 3$:  BR       16$            ;TAKE ERROR EXIT
3928 006516 001407                BITB     #IRDY, REDBYT   ;SEE IF IRDY = 0
3929 006520 004737 004504        BEQ      9$              ;BR IF IRDY = 0
3930                                JSR      PC, GETALL      ;GET REGS FOR PRINTOUT
3931                                ;REPORT IRDY NOT CLEARED
3932 006524                                ERRDF    18, EM18, ERR4
3933 006524 104455                                TRAP    C$ERDF
3934 006526 000022                                .WORD  18
3935 006530 012636                                .WORD  EM18
3936 006532 015600                                .WORD  ERR4
3937 006534 000436
3938 006536 012737 000017 002356 9$:  BR       16$            ;TAKE ERROR RETURN
3939 006544 004737 003650        MOV      #17, REGNUM     ;SET REG NO. = 17
3940 006550 132776 000001 000004  JSR      PC, READLU      ;READ REG 17
3941 006556 001413                BITB     #BIT0, @4(SP)   ;GET EXPECTED STATE OF ICIR
3942 006560 132737 000010 002342  BEQ      12$            ;BR IF EXPECTED ICIR = 0
3943 006566 001031                BITB     #ICIR, REDBYT   ;SEE IF ICIR = 1
3944 006570 004737 004504        BNE      20$            ;BR IF ICIR = 1
3945                                JSR      PC, GETALL      ;GET REGS FOR PRINTOUT
3946                                ;REPORT ICIR NOT SET
3947 006574                                ERRDF    19, EM19, ERR4
3948 006574 104455                                TRAP    C$ERDF
3949 006576 000023                                .WORD  19
3950 006600 012657                                .WORD  EM19
3951 006602 015600                                .WORD  ERR4
3952 006604 000412
3953 006606 132737 000010 002342 12$: BR       16$            ;TAKE ERROR RETURN
3954 006614 001416                BITB     #ICIR, REDBYT   ;SEE IF ICIR = 0
3955 006616 004737 004504        BEQ      20$            ;BR IF ICIR = 0
3956                                JSR      PC, GETALL      ;GET REGS FOR PRINTOUT
3957                                ;REPORT ICIR NOT CLEARED
3958 006622                                ERRDF    20, EM20, ERR4
3959 006622 104455                                TRAP    C$ERDF
3960 006624 000024                                .WORD  20
3961 006626 012674                                .WORD  EM20
3962 006630 015600                                .WORD  ERR4
3963 006632 016637 000002 002356 16$: MOV      2(SP), REGNUM     ;RESTORE LU REG NO.
3964 006640 013706 002324        MOV      PSTACK, SP      ;RESTORE STACK POINTER TO BASE LEVEL
3965 006644 013746 002340        MOV      RETADR, -(SP)   ;FIX ERROR RETURN PC
3966 006650 000407                BR       23$

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

3964 006652 062766 000002 000004 20$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
3965 006660 012637 002330 MOV (SP)+,SUBRPC
3966 006664 012637 002356 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
3967 006670 000207 23$: RTS PC ;RETURN
3968
3969
3970
3971
3972
3973

```

```

:*****
:* 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.
:*****

```

```

3980 006672 013746 002356 IACTIV: MOV REGNUM,-(SP) ;SAVE LU REG NO.
3981 006676 013746 002330 MOV SUBRPC,-(SP)
3982 006702 005737 002330 TST SUBRPC ;SEE IF THIS IS A NESTED CALL
3983 006706 001006 BNE 1$ ;BR IF YES
3984 006710 016637 000004 002330 MOV 4(SP),SUBRPC
3985 006716 162737 000004 002330 SUB #4,SUBRPC ;GET PC OF SUBR CALL
3986 006724 012737 000012 002356 1$: MOV #12,REGNUM ;SET REG NO. = 12
3987 006732 004737 003650 JSR PC,READLU ;READ REG 12
3988 006736 032776 000001 000004 BIT #BIT0,@4(SP) ;GET EXPECTED STATE OF IACT
3989 006744 001413 BEQ 3$ ;BR IF EXPECTED IACT = 0
3990 006746 132737 000100 002342 BITB #IACT,REDBYT ;SEE IF IACT = 1
3991 006754 001031 BNE 9$ ;BR IF IACT = 1
3992 006756 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT
3993

```

```

:REPORT IACT NOT SET
ERRDF 21,EM21,ERR4

```

```

TRAP C$ERDF
.WORD 21
.WORD EM21
.WORD ERR4

```

```

3994 006762 104455
3995 006762 000025
3996 006764 012715
3997 006770 015600
3998 006772 000412
3999 006774 132737 000100 002342 3$: BR 6$ ;TAKE ERROR EXIT
4000 007002 001416 BITB #IACT,REDBYT ;SEE IF IACT = 0
4001 007004 004737 004504 BEQ 9$ ;BR IF IACT = 0
4002 JSR PC,GETALL ;GET REGS FOR PRINTOUT
4003 :REPORT IACT NOT CLEARED
4004 ERRDF 22,EM22,ERR4

```

```

TRAP C$ERDF
.WORD 22
.WORD EM22
.WORD ERR4

```

```

4005 007010 104455
4006 007012 000026
4007 007014 012732
4008 007016 015600
4009 007020 016637 000002 002356 6$: MOV 2(SP),REGNUM ;RESTORE LU REG NO.
4010 007026 013706 002324 MOV PSTACK,SP ;RESTORE PROGRAM STACK TO BASE LEVEL
4011 007032 013746 002340 MOV RETADR,-(SP) ;FIX UP ERROR RETURN PC
4012 007036 000407 BR 12$
4013 007040 062766 000002 000004 9$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
4014 007046 012637 002330 MOV (SP)+,SUBRPC
4015 007052 012637 002356 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
4016 007056 000207 12$: RTS PC ;RETURN
4017
4018
4019

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075

007060 013746 002360
007064 013746 002330
007070 005737 002330
007074 001006
007076 016637 000004 002330
007104 162737 000004 002330
007112 012737 000001 002360
007120 004737 004102
007124 032776 000001 000004
007132 001413
007134 132737 000001 002350
007142 001022
007144 004737 004504

007150
007150 104455
007152 000035
007154 013151
007156 016770
007160 000444
007162 132737 000001 002350
007170 001407
007172 004737 004504

007176
007176 104455
007200 000034
007202 013130
007204 016770
007206 000431
007210 132776 000002 000004
007216 001413
007220 132737 000002 002350
007226 001031
007230 004737 004504

007234
007234 104455
007236 000037
007240 013207
007242 016770
007244 000412
007246 132737 000002 002350
007254 001416
007256 004737 004504

007262
007262 104455

```
*****  
;* RSEOM - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF RSOM AND REOM IN  
;* AX0-16, AND REPORTS AN ERROR IF EITHER IS NOT SET TO THE STATE PASSED IN BITS  
;* 0,1, RESPECTIVELY, OF THE WORD FOLLOWING THE CALL.  
;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN RETADR.  
*****  
RSEOM:  MOV    AXNUM,-(SP)      ;SAVE AX BYTE NO.  
        MOV    SUBRPC,-(SP)  
        TST    SUBRPC          ;SEE IF THIS IS A NESTED CALL  
        BNE    1$              ;BR IF YES  
        MOV    4(SP),SUBRPC    ;GET PC OF SUBR CALL  
        SUB    #4,SUBRPC       ;SET AX BYTE NO. FOR AX0-16  
1$:     MOV    #1,AXNUM        ;READ AX0  
        JSR    PC,READAX       ;GET EXPECTED STATE OF RSOM  
        BIT    #BIT0,@4(SP)    ;BR IF EXPECTED RSOM = 0  
        BEQ    3$              ;SEE IF RSOM = 1  
        BITB   #RSOM,RAX16     ;BR IF RSOM = 1  
        BNE    9$              ;GET REGS FOR PRINTOUT  
        JSR    PC,GETALL  
;REPORT RSOM NOT SET  
        ERRDF  29,EM29,ERR6  
  
        BR    16$              ;TAKE ERROR EXIT  
3$:     BITB   #RSOM,RAX16     ;SEE IF RSOM = 0  
        BEQ    9$              ;BR IF RSOM = 0  
        JSR    PC,GETALL       ;GET REGS FOR PRINTOUT  
;REPORT RSOM NOT CLEARED  
        ERRDF  28,EM28,ERR6  
  
        BR    16$              ;TAKE ERROR RETURN  
9$:     BITB   #BIT1,@4(SP)    ;GET EXPECTED STATE OF REOM  
        BEQ    12$             ;BR IF EXPECTED REOM = 0  
        BITB   #REOM,RAX16     ;SEE IF REOM = 1  
        BNE    20$             ;BR IF REOM = 1  
        JSR    PC,GETALL       ;GET REGS FOR PRINTOUT  
;REPORT REOM NOT SET  
        ERRDF  31,EM31,ERR6  
  
        BR    16$              ;TAKE ERROR RETURN  
12$:    BITB   #REOM,RAX16     ;SEE IF REOM = 0  
        BEQ    20$             ;BR IF REOM = 0  
        JSR    PC,GETALL       ;GET REGS FOR PRINTOUT  
;REPORT REOM NOT CLEARED  
        ERRDF  30,EM30,ERR6
```

TRAP C\$ERDF
.WORD 29
.WORD EM29
.WORD ERR6

TRAP C\$ERDF
.WORD 28
.WORD EM28
.WORD ERR6

TRAP C\$ERDF
.WORD 31
.WORD EM31
.WORD ERR6

TRAP C\$ERDF

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

.WORD 30
.WORD EM30
.WORD ERR6

4076 007264 000036
 4077 007266 013166
 4078 007270 016770
 4079 007272 016637 000002 002360 16\$:
 4080 007300 013706 002324
 4081 007304 013746 002340
 4082 007310 000407
 4083 007312 062766 000002 000004 20\$:
 4084 007320 012637 002330
 4085 007324 012637 002360
 4086 007330 000207 23\$:

```

MOV 2(SP),AXNUM ;RESTORE AX BYTE NO.
MOV PSTACK,SP ;RESTORE STACK POINTER TO BASE LEVEL
MOV RETADR,-(SP) ;FIX ERROR RETURN PC
BR 23$
ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
MOV (SP)+,SUBRPC
MOV (SP)+,AXNUM ;RESTORE AX BYTE NO.
RTS PC ;RETURN

```

4087
4088
4089
4090
4091
4092

```

*****
* RDRXSI - THIS SUBROUTINE READS THE RCV SILO (REGS 10,12) AND RETURNS THE
* SILO ENTRY IN BITS 0-11 OF RXWORD.
*****

```

4093
4094
4095
4096 007332 013746 002356
4097 007336 012737 000012 002356
4098 007344 004737 003650
4099 007350 113737 002342 002403
4100 007356 042737 170000 002402
4101 007364 012737 000010 002356
4102 007372 004737 003650
4103 007376 113737 002342 002402
4104 007404 012637 002356
4105 007410 000207

```

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

```

4106
4107
4108
4109
4110
4111

```

*****
* 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.
*****

```

4112
4113
4114
4115
4116
4117
4118
4119
4120 007412 010146
4121 007414 010346
4122 007416 013746 002356
4123 007422 016637 000006 002330
4124 007430 162737 000004 002330
4125 007436 012737 000012 002356
4126 007444 005001
4127 007446 017603 000006
4128 007452 062703 000003
4129 007456 005776 000006
4130 007462 001414
4131 007464 004737 006672

```

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

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

4132 007470 000000      0
4133 007472 004737 006406    JSR    PC,ISIRDY      ;CHK FOR ICIR = 1, IRDY = 0
4134 007476 000001      1
4135 007500 004737 007060    JSR    PC,RSEOM      ;CHK RSOM = 0, REOM = 0 IN AX0-16
4136 007504 000000      0
4137 007506 004737 005232    6$:   JSR    PC,STPLU      ;CLOCK LU FOR 1 CYCLE
4138 007512 000001      1
4139 007514 004737 005124    8$:   JSR    PC,WAIT50     ;ALLOW SILO DATA TO RIPPLE
4140 007520 005201      INC    R1              ;INCREMENT CYCLE COUNT
4141 007522 004737 003650    JSR    PC,READLU     ;READ REG 12
4142 007526 132737 000020 002342    BITB  #IRDY,REDBYT    ;SEE IF IRDY = 1 YET
4143 007534 001005      BNE   9$              ;BR IF IRDY = 1
4144 007536 020103      CMP   R1,R3          ;SEE IF LIMIT EXCEEDED
4145 007540 002762      BLT   6$              ;BR IF NOT YET
4146 007542 004737 006406    JSR    PC,ISIRDY     ;CHK FOR ICIR = 1, IRDY = 1
4147 007546 000003      3
4148 007550 020176 000006    9$:   CMP   R1,@6(SP)    ;SEE IF LESS THAN REQUIRED CYCLES
4149 007554 002003      BGE   12$            ;BR IF NOT
4150 007556 004737 006406    JSR    PC,ISIRDY     ;CHK FOR ICIR = 1, IRDY = 0
4151 007562 000001      1
4152 007564 004737 006672    12$:  JSR    PC,IACTIV     ;CHK FOR IACT = 1
4153 007570 000001      1
4154 007572 004737 006406    JSR    PC,ISIRDY     ;CHK FOR ICIR = 1, IRDY = 1
4155 007576 000003      3
4156 007600 062766 000002 000006    ADD   #2,6(SP)        ;FIX UP RETURN PC
4157 007606 012637 002356    MOV   (SP)+,REGNUM    ;RESTORE LU REG NO.
4158 007612 012603      MOV   (SP)+,R3        ;RESTORE R3
4159 007614 012601      MOV   (SP)+,R1        ;RESTORE R1
4160 007616 005037 002330    CLR   SUBRPC          ;CLEAR SUBR CALL PC
4161 007622 000207      RTS   PC              ;RETURN

```

4162

4163

4164

4165

4166

4167

4168

4169

4170

4171

4172

4173 007624 013746 002356

4174 007630 012737 000017 002356

4175 007636 017637 000002 002344

4176 007644 152737 000002 002344

4177 007652 004737 003726

4178 007656 062766 000002 000002

4179 007664 017637 000002 007676

4180 007672 004737 005232

4181 007676 000000 3\$:

4182 007700 142737 000002 002344

4183 007706 004737 003726

4184 007712 062766 000002 000002

4185 007720 012637 002356

4186 007724 000207

4187

```

*****
;* 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.
*****

```

```

STPERR: MOV   REGNUM,-(SP)      ;SAVE LU REG NO.
        MOV   #17,REGNUM     ;SET LU REG NO. = 17
        MOV   @2(SP),WRIBYT
        BISB  #IERR,WRIBYT
        JSR   PC,WRITLU      ;SET IERR BIT IN REG 17
        ADD  #2,2(SP)        ;INCREMENT SUBR ARGUMENT POINTER
        MOV   @2(SP),3$      ;GET DESIRED NO. OF CYCLES
        JSR   PC,STPLU      ;CLOCK LU FOR DESIRED NO. OF CYCLES
        .WORD 0              ;NO. OF CYCLES GOES HERE
        BICB  #IERR,WRIBYT
        JSR   PC,WRITLU      ;CLEAR IERR BIT IN REG 17
        ADD  #2,2(SP)        ;FIX UP RETURN PC
        MOV   (SP)+,REGNUM    ;RESTORE LU REG NO.
        RTS   PC            ;RETURN

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243

007726 010146
007730 013746 002356
007734 016637 000004 002330
007742 162737 000004 002330
007750 017601 000004
007754 042701 170000
007760 004737 007332
007764 023727 002410 000347
007772 001005
007774 042701 000200
010000 042737 000200 002402
010006 120137 002402
010012 001445
010014 005037 002362
010020 110137 002362
010024 005037 002364
010030 113737 002402 002364
010036 012737 000011 002356
010044 004737 003650
010050 132737 000001 002342
010056 001410
010060 004737 004504

010064
010064 104455
010066 000066
010070 014164
010072 015600
010074 000137 010556
010100 012737 000010 002356
010106 004737 004504

010112
010112 104455
010114 000042
010116 013276
010120 020100
010122 000137 010556
010126 000301
010130 012737 000012 002356
010136 120137 002403
010142 001002
010144 000137 010532

```

*****
* 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.
*****
CKDATA: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV 4(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBR CALL
MOV @4(SP),R1 ;GET EXPECTED SILO ENTRY
BIC #170000,R1 ;CLEAR UNUSED BITS FOR COMPARE
JSR PC,RDRXSI ;READ RCV SILO
CMP SAVLEN,#TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO
BNE 4$ ;BR IF CHAR LENGTH NOT = 7
BIC #BIT7,R1 ;MASK OFF BIT 8TH BIT
BIC #BIT7,RXWORD
4$: CMPB R1,RXWORD ;COMPARE EXPECTED BITS 0-7 TO ACTUAL
BEQ 6$ ;BR IF MATCH
CLR GOODAT
MOVB R1,GOODAT ;GET EXPECTED DATA
CLR BADDAT
MOVB RXWORD,BADDAT ;GET ACTUAL DATA
MOV #11,REGNUM ;SET REG NO. = 11
JSR PC,READLU ;READ REG 11
BITB #UNRR,REDBYT ;SEE IF TX UNDERRUN ERROR
BEQ 5$ ;BR IF NOT
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT TX UNDERRUN ERROR
ERRDF 54,EM54,ERR4
TRAP C$ERDF
.WORD 54
.WORD EM54
.WORD ERR4

5$: JMP 36$ ;TAKE ERROR EXIT
MOV #10,REGNUM ;SET REG NO. = 10
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT RCV'D DATA MISCOMPARE
ERRDF 34,EM34,ERR8
TRAP C$ERDF
.WORD 34
.WORD EM34
.WORD ERR8

6$: JMP 36$ ;TAKE ERROR EXIT
SWAB R1
MOV #12,REGNUM ;SET LU REG NO. FOR ERROR REPORTS
CMPB R1,RXWORD+1 ;COMPARE EXPECTED SILO BITS 8-11 TO ACTUAL
BNE 7$ ;BR IF MISMATCH
JMP 22$ ;CONTINUE

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

4244 010150 005037 002362      7$:  CLR      GOODAT
4245 010154 110137 002362      MOVB     R1,GOODAT      ;SET EXPECTED DATA
4246 010160 005037 002364      CLR      BADDAT
4247 010164 113737 002403 002364  MOVB     RXWORD+1,BADDAT ;SET ACTUAL DATA
4248 010172 032776 100000 000004  BIT      #BCCCHK,@4(SP) ;SEE IF BCC SHOULD BE IGNORED
4249 010200 001433          BEQ      10$           ;BR IF YES
4250 010202 132701 000001          BITB     #BCC,R1       ;SEE IF EXPECTED BIT = 1
4251 010206 001014          BNE      8$           ;BR IF YES
4252 010210 132737 000001 002403  BITB     #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 0
4253 010216 001424          BEQ      10$           ;BR IF YES
4254 010220 004737 004504          JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
4255          ;REPORT BCC NOT CLEARED
4256 010224          ERRDF 35,EM35,ERR8
4257 010224 104455          TRAP   C$ERDF
4258 010226 000043          .WORD 35
4259 010230 013324          .WORD EM35
4260 010232 020100          .WORD ERR8
4261 010234 000137 010556          JMP      36$           ;TAKE ERROR EXIT
4262 010240 132737 000001 002403  8$:  BITB     #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 1
4263 010246 001010          BNE      10$           ;BR IF YES
4264 010250 004737 004504          JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
4265          ;REPORT BCC NOT SET
4266 010254          ERRDF 36,EM36,ERR8
4267 010254 104455          TRAP   C$ERDF
4268 010256 000044          .WORD 36
4269 010260 013344          .WORD EM36
4270 010262 020100          .WORD ERR8
4271 010264 000137 010556          JMP      36$           ;TAKE ERROR EXIT
4272 010270          10$:  BITB     #EBLK,R1       ;SEE IF EXPECTED BIT = 1
4273 010270 132701 000002          BNE      12$           ;BR IF YES
4274 010274 001014          BITB     #EBLK,RXWORD+1 ;SEE IF ACTUAL BIT = 0
4275 010276 132737 000002 002403  BEQ      14$           ;BR IF YES
4276 010304 001424          JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
4277 010306 004737 004504          ;REPORT EBLK NOT CLEARED
4278          ERRDF 37,EM37,ERR8
4279 010312          TRAP   C$ERDF
4280 010312 104455          .WORD 37
4281 010314 000045          .WORD EM37
4282 010316 013360          .WORD ERR8
4283 010320 020100          JMP      36$           ;TAKE ERROR EXIT
4284 010322 000137 010556          12$:  BITB     #EBLK,RXWORD+1 ;SEE IF ACTUAL BIT = 1
4285 010326 132737 000002 002403  BNE      14$           ;BR IF YES
4286 010334 001010          JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
4287 010336 004737 004504          ;REPORT EBLK NOT SET
4288          ERRDF 38,EM38,ERR8
4289 010342          TRAP   C$ERDF
4290 010342 104455          .WORD 38
4291 010344 000046          .WORD EM38
4292 010346 013401          .WORD ERR8
4293 010350 020100          JMP      36$           ;TAKE ERROR EXIT
4294 010352 000137 010556          14$:  BITB     #RAB,R1       ;SEE IF EXPECTED BIT = 1
4295 010356          BNE      16$           ;BR IF YES
4296 010356 132701 000004          BITB     #RAB,RXWORD+1 ;SEE IF ACTUAL BIT = 0
4297 010362 001014          BEQ      18$           ;BR IF YES
4298 010364 132737 000004 002403
4299 010372 001424

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

4300 010374 004737 004504      ;REPORT JSR PC,GETALL      ;GET REGS FOR PRINTOUT
4301                                RAB NOT CLEARED
4302 010400                                ERRDF 39,EM39,ERR8
4303 010400 104455                                TRAP CSERDF
4304 010402 000047                                .WORD 39
4305 010404 013416                                .WORD EM39
4306 010406 020100                                .WORD ERR8
4307 010410 000137 010556      JMP 36$      ;TAKE ERROR EXIT
4308 010414 132737 000004 002403 16$: BITB #RAB,RXWORD+1 ;SEE IF ACTUAL BIT = 1
4309 010422 001010                                BNE 18$      ;BR IF YES
4310 010424 004737 004504      JSR PC,GETALL ;GET REGS FOR PRINTOUT
4311                                RAB NOT SET
4312 010430                                ERRDF 40,EM40,ERR8
4313 010430 104455                                TRAP CSERDF
4314 010432 000050                                .WORD 40
4315 010434 013436                                .WORD EM40
4316 010436 020100                                .WORD ERR8
4317 010440 000137 010556      JMP 36$      ;TAKE ERROR EXIT
4318 010444      18$:                                ;SEE IF EXPECTED BIT = 1
4319 010444 132701 000010      BITB #OVRR,R1 ;SEE IF ACTUAL BIT = 1
4320 010450 001014                                BNE 20$      ;BR IF YES
4321 010452 132737 000010 002403 BITB #OVRR,RXWORD+1 ;SEE IF ACTUAL BIT = 0
4322 010460 001424                                BEQ 22$      ;BR IF YES
4323 010462 004737 004504      JSR PC,GETALL ;GET REGS FOR PRINTOUT
4324                                OVRR NOT CLEARED
4325 010466                                ERRDF 41,EM41,ERR8
4326 010466 104455                                TRAP CSERDF
4327 010470 000051                                .WORD 41
4328 010472 013452                                .WORD EM41
4329 010474 020100                                .WORD ERR8
4330 010476 000137 010556      JMP 36$      ;TAKE ERROR EXIT
4331 010502 132737 000010 002403 20$: BITB #OVRR,RXWORD+1 ;SEE IF ACTUAL BIT = 1
4332 010510 001010                                BNE 22$      ;BR IF YES
4333 010512 004737 004504      JSR PC,GETALL ;GET REGS FOR PRINTOUT
4334                                OVRR NOT SET
4335 010516                                ERRDF 42,EM42,ERR8
4336 010516 104455                                TRAP CSERDF
4337 010520 000052                                .WORD 42
4338 010522 013473                                .WORD EM42
4339 010524 020100                                .WORD ERR8
4340 010526 000137 010556      JMP 36$      ;TAKE ERROR EXIT
4341 010532      22$:                                ;INCR SUBROUTINE ARGUMENT POINTER
4342 010532 062766 000002 000004 ADD #2,4(SP) ;GET DESIRED CYCLE COUNT
4343 010540 017637 000004 010552 MOV @4(SP),24$ ;CLOCK LU FOR DESIRED CYCLES
4344 010546 004737 005232      JSR PC,STPLU
4345 010552 000000      24$: .WORD 0
4346 010554 000407      BR 38$      ;TAKE ERROR-FREE EXIT
4347 010556 011637 002356      36$: MOV (SP),REGNUM ;RESTORE LU REG NO.
4348 010562 013706 002324      MOV PSTACK,SP ;RESTORE PROGRAM STACK TO BASE LEVEL
4349 010566 013746 002340      MOV RETADR,-(SP) ;FIX UP ERROR RETURN PC
4350 010572 000406      BR 40$
4351 010574 062766 000002 000004 38$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
4352 010602 012637 002356      MOV (SP)+,REGNUM ;RESTORE LU REG NO.
4353 010606 012601      MOV (SP)+,R1 ;RESTORE R1
4354 010610 005037 002330      40$: CLR SUBRPC ;CLEAR SUBROUTINE PC
4355 010614 000207      RTS PC ;RETURN

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366 010616 013746 002360
4367 010622 013746 002356
4368 010626 012737 000004 002360
4369 010634 017637 000004 002352
4370 010642 005037 002354
4371 010646 004737 004270
4372 010652 012737 000017 002356
4373 010660 062766 000002 000004
4374 010666 017637 000004 002344
4375 010674 004737 003726
4376 010700 012737 000006 002360
4377 010706 062766 000002 000004
4378 010714 017637 000004 002352
4379 010722 062766 000002 000004
4380 010730 017637 000004 002354
4381 010736 013737 002354 002410
4382 010744 004737 004270
4383 010750 062766 000002 000004
4384 010756 012637 002356
4385 010762 012637 002360
4386 010766 005037 002330
4387 010772 000207
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398 010774 010146
4399 010776 010246
4400 011000 017601 000004
4401 011004 062766 000002 000004
4402 011012 017602 000004
4403 011016 062766 000002 000004
4404 011024 012137 002400
4405 011030 004737 005152
4406 011034 005302
4407 011036 001372
4408 011040 004737 005124
4409 011044 012602
4410 011046 012601
4411 011050 000207

```

*****
;* 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.
*****
SETUP: MOV     AXNUM, -(SP)      ;SAVE AX BYTE NO.
      MOV     REGNUM, -(SP)   ;SAVE LU REG NO.
      MOV     #4, AXNUM       ;SET AX BYTE NO. FOR AX2
      MOV     @4(SP), WAX15
      CLR     WAX16
      JSR     PC, WRITAX      ;SET SYNCH CHAR IN AX2-15, CLEAR AX2-16
      MOV     #17, REGNUM     ;SET LU REG NO. = 17
      ADD     #2, 4(SP)       ;INCREMENT ARGUMENT POINTER
      MOV     @4(SP), WRIBYT
      JSR     PC, WRITLU     ;LOAD REG 17
      MOV     #6, AXNUM       ;SET AX BYTE NO. FOR AX3
      ADD     #2, 4(SP)       ;INCREMENT ARGUMENT POINTER
      MOV     @4(SP), WAX15
      ADD     #2, 4(SP)       ;INCR ARGUMENT POINTER
      MOV     @4(SP), WAX16
      MOV     WAX16, SAVLEN   ;STORE TX AND RCV CHAR LENGTH
      JSR     PC, WRITAX     ;LOAD AX3-15, AX3-16
      ADD     #2, 4(SP)       ;FIX RETURN PC
      MOV     (SP)+, REGNUM   ;RESTORE LU REG NO.
      MOV     (SP)+, AXNUM    ;RESTORE AX BYTE NO.
      CLR     SUBRPC         ;CLEAR SUBROUTINE PC STORAGE
      RTS     PC             ;RETURN

```

```

*****
;* 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.
*****
LODMSG: MOV     R1, -(SP)     ;SAVE R1
      MOV     R2, -(SP)     ;SAVE R2
      MOV     @4(SP), R1     ;GET MSG POINTER INTO R1
      ADD     #2, 4(SP)     ;INCR ARG POINTER
      MOV     @4(SP), R2     ;GET WORD COUNT INTO R2
      ADD     #2, 4(SP)     ;FIX UP RETURN PC
6$:     MOV     (R1)+, TXWORD ;GET NEXT MSG WORD
      JSR     PC, LDTXSI    ;LOAD A WORD INTO TX SILO
      DEC     R2            ;DECR COUNT
      BNE    6$            ;BR IF NOT DONE YET
      JSR     PC, WAIT50    ;WAIT FOR SILO TO RIPPLE
      MOV     (SP)+, R2     ;RESTORE R2
      MOV     (SP)+, R1     ;RESTORE R1
      RTS     PC           ;RETURN

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467

011052	010146		
011054	016637	000002	002330
011062	162737	000004	002330
011070	004737	007332	
011074	004737	005124	
011100	005001		
011102	020176	000002	
011106	001410		
011110	004737	006406	
011114	000001		
011116	004737	005232	
011122	000001		
011124	005201		
011126	000765		
011130	004737	006406	
011134	000003		
011136	062766	000002	000002
011144	005037	002330	
011150	012601		
011152	000207		

```

*****
;* 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.
*****
    
```

```

RXCHAR: MOV     R1,-(SP)           ;SAVE R1
        MOV     2(SP),SUBRPC
        SUB     #4,SUBRPC       ;GET PC OF SUBR CALL
        JSR    PC,RDRXSI       ;READ RCV SILO
        JSR    PC,WAIT50       ;ALLOW SILO TO RIPPLE
        CLR     R1              ;INIT CYCLE COUNT
9$:     CMP     R1,@2(SP)       ;SEE IF REQUIRED CYCLES DONE YET
        BEQ    12$             ;BR IF YES
        JSR    PC,ISIRDY       ;CHK ICIR = 1, IRDY = 0
        1
        JSR    PC,STPLU        ;CLK LU 1 CYCLE
        1
        INC     R1              ;INCR CYCLE COUNT
        BR     9$
12$:    JSR    PC,ISIRDY       ;CHK ICIR = 1 IRDY = 1
        3
        ADD     #2,2(SP)       ;FIX RETURN PC
        CLR     SUBRPC         ;CLEAR SUBR CALL PC
        MOV     (SP)+,R1       ;RESTORE R1
        RTS     PC             ;RETURN
    
```

```

*****
;* CKTBIT - THIS SUBROUTINE CLOCKS THE LINE UNIT FOR 8 CYCLES, AND AFTER EACH
;* CYCLE, THE TXDATA BIT IN REG 17 IS EXAMINED, AND COMPARED TO A BIT OF
;* THE CHAR PASSED IN THE WORD FOLLOWING THE CALL.
;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN
;* RETADR.
*****
    
```

```

CKTBIT: MOV     REGNUM,-(SP)     ;SAVE LU REG NO.
        MOV     R1,-(SP)       ;SAVE R1
        MOV     4(SP),SUBRPC
        SUB     #4,SUBRPC       ;GET PC OF SUBROUTINE CALL
        MOV     #BIT0,R1       ;INIT BIT POINTER
        MOV     #17,REGNUM      ;SET REG NO. = 17
4$:     JSR    PC,STPLU        ;CLOCK LINE UNIT 1 CYCLE
        1
        JSR    PC,READLU       ;READ REG 17
        BIT     R1,@4(SP)      ;SEE IF EXPECTED BIT = 1
        BNE    9$              ;BR IF YES
        BITB   #TXDATA,REDBYT  ;SEE IF ACTUAL BIT = 0
    
```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

```

4468 011236 001422
4469 011240 004737 004504
4470
4471 011244
4472 011244 104455
4473 011246 000040
4474 011250 013224
4475 011252 015600
4476 011254 000420
4477 011256 132737 000040 002342 9$:
4478 011264 001007
4479 011266 004737 004504
4480
4481 011272
4482 011272 104455
4483 011274 000041
4484 011276 013253
4485 011300 015600
4486 011302 000405
4487 011304 006301 12$:
4488 011306 020127 000400
4489 011312 001336
4490 011314 000405
4491 011316 013706 002324
4492 011322 013746 002340
4493 011326 000406
4494 011330 062766 000002 000004 22$:
4495 011336 012601
4496 011340 012637 002356
4497 011344 005037 002330
4498 011350 000207 40$:
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509 011352 010146
4510 011354 010246
4511 011356 004737 010774
4512 011362 003176
4513 011364 000011
4514 011366 012701 003202
4515 011372 012702 003240
4516 011376 012122 3$:
4517 011400 020127 003214
4518 011404 103774
4519 011406 052762 100400 177776
4520 011414 012726 000160
4521 011420 012726 000034
4522 011424 012602
4523 011426 012601

```

```

BEQ 12$ ;BR IF YES
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT TXDATA NOT CLEARED
ERRDF 32,EM32,ERR4
BR 20$ ;TAKE ERROR RETURN
BITB #TXDATA,REDBYT ;SEE IF ACTUAL BIT = 1
BNE 12$ ;BR IF YES
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT TXDATA BIT NOT SET
ERRDF 33,EM33,ERR4
BR 20$ ;TAKE ERROR EXIT
ASL R1 ;SHIFT BIT POINTER
CMP R1,#400 ;SEE IF 8 BITS SCANNED YET
BNE 4$ ;BR IF NO
BR 22$
MOV PSTACK,SP ;RESTORE PROGRAM STACK POINTER TO BASE LEVEL
MOV RETADR,-(SP) ;FIX UP RETURN PC
BR 40$
ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
MOV (SP)+,R1 ;RESTORE R1
MOV (SP)+,REGNUM ;RESTORE REG NO.
CLR SUBRPC ;CLEAR SUBR CALL PC
RTS PC ;RETURN

```

```

TRAP C$ERDF
.WORD 32
.WORD EM32
.WORD ERR4

TRAP C$ERDF
.WORD 33
.WORD EM33
.WORD ERR4

```

```

*****
;* 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.
*****

```

```

LDMSG1: MOV R1,-(SP) ;SAVE R1
MOV R2,-(SP) ;SAVE R2
JSR PC,LODMSG ;LOAD MSG1 INTO TX SILO
MSG1
9.
MOV #MSG1+4,R1 ;GET POINTER TO MSG1
MOV #RCVBUF,R2 ;GET POINTER TO MSG BUF
3$: MOV (R1)+,(R2)+ ;LOAD A CHAR INTO MSG BUF
CMP R1,#MSG1+14. ;SEE IF DID LAST DATA CHAR YET
BLO 3$ ;BR IF NOT
BIS #CRCCHK!RXBCC,-2(R2) ;SET EXPECTED BCC
MOV #160,(SP)+ ;LOAD HI CRC BYTE
MOV #034,(SP)+ ;LOAD LO CRC BYTE
MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL SUBROUTINES

4524	011430	000207	RTS	PC	;RETURN
4525					
4526					
4527					
4528					
4529					

GLOBAL ERROR REPORT SECTION

.SBTTL GLOBAL ERROR REPORT SECTION

```

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

```

4530					
4531					
4532					
4533					
4534					
4535					
4536					
4537					
4538	011432	052045	047445	022466	FMT1: .ASCIZ /%T%06%N/
4539	011440	000116			
4540	011442	047045	040445	040506	FMT2: .ASCIZ /%N%AFAILING REG: /
4541	011450	046111	047111	020107	
4542	011456	042522	035107	000040	
4543	011464	040445	054105	042520	FMT3: .ASCIZ /%AEXPECTED: %03%S5%AACTUAL: %03%N/
4544	011472	052103	042105	020072	
4545	011500	047445	022463	032523	
4546	011506	040445	041501	052524	
4547	011514	046101	020072	047445	
4548	011522	022463	000116		
4549	011526	047045	052045	047045	FMT4: .ASCIZ /%N%T%N%T%N/
4550	011534	052045	047045	000	
4551	011541	045	031517	051445	FMT5: .ASCIZ /%03%S5%03%S5%03%S5%03%N/
4552	011546	022465	031517	051445	
4553	011554	022465	031517	051445	
4554	011562	022465	031517	047045	
4555	011570	000			
4556	011571	045	032123	047445	FMT6: .ASCIZ /%S4%03%S5%03%S5%03%S5%03%N/
4557	011576	022463	032523	047445	
4558	011604	022463	032523	047445	
4559	011612	022463	032523	047445	
4560	011620	022463	000116		
4561	011624	052045	047445	022462	FMT7: .ASCIZ /%T%02%N/
4562	011632	000116			
4563	011634	040445	054105	042524	FMT8: .ASCIZ /%AEXTENDED REG AX%01%A-%T%N/
4564	011642	042116	042105	051040	
4565	011650	043505	040440	022530	
4566	011656	030517	040445	022455	
4567	011664	022524	000116		
4568	011670	052045	047045	000	FMT9: .ASCIZ /%T%N/
4569	011675	045	050101	020103	FMT10: .ASCIZ /%APC OF SUBR CALL: %06%N/
4570	011702	043117	051440	041125	
4571	011710	020122	040503	046114	
4572	011716	020072	047445	022466	
4573	011724	000116			
4574	011726	040445	042522	020107	FMT11: .ASCIZ /%AREG %02%A LOADED WITH: %03%N/
4575	011734	047445	022462	020101	
4576	011742	047514	042101	042105	
4577	011750	053440	052111	035110	
4578	011756	022440	031517	047045	
4579	011764	000			
4580	011765	045	022516	052101	FMT19: .ASCIZ /%N%ATEST %D2%A NOT RUN%N/
4581	011772	051505	020124	042045	
4582	012000	022462	020101	047516	
4583	012006	020124	052522	022516	
4584	012014	000116			
4585	012016	047045	040445	046120	FMT24: .ASCIZ /%N%PLEASE INSURE M8207 RUN SWITCH (E28 SW7 IS ON%N/

CZDMRF.P11

03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4586	012024	040505	042523	044440	
4587	012032	051516	051125	020105	
4588	012040	034115	030062	020067	
4589	012046	052522	020116	053523	
4590	012054	052111	044103	024040	
4591	012062	031105	020070	053523	
4592	012070	024467	044440	020123	
4593	012076	047117	047045	000	
4594					
4595					
4596					
4597	012103	103	051123	040440	EM1: .ASCIZ /CSR ADDRESS TIME-OUT (SELO)/
4598	012110	042104	042522	051523	
4599	012116	052040	046511	026505	
4600	012124	052517	020124	051450	
4601	012132	046105	024460	000	
4602	012137	122	043505	047040	EM2: .ASCIZ /REG NOT INITIALIZED BY MST CLR/
4603	012144	052117	044440	044516	
4604	012152	044524	046101	055111	
4605	012160	042105	041040	020131	
4606	012166	051515	020124	046103	
4607	012174	000122			
4608	012176	042522	020107	044515	EM3: .ASCIZ /REG MISCOMPARE/
4609	012204	041523	046517	040520	
4610	012212	042522	000		
4611	012215	122	043505	047040	EM4: .ASCIZ /REG NOT INITIALIZED BY UNIBUS RESET (INIT)/
4612	012222	052117	044440	044516	
4613	012230	044524	046101	055111	
4614	012236	042105	041040	020131	
4615	012244	047125	041111	051525	
4616	012252	051040	051505	052105	
4617	012260	024040	047111	052111	
4618	012266	000051			
4619	012270	040515	047111	020124	EM5: .ASCIZ /MAINT CLK BIT STUCK AT 0/
4620	012276	046103	020113	044502	
4621	012304	020124	052123	041525	
4622	012312	020113	052101	030040	
4623	012320	000			
4624	012321	115	044501	052116	EM6: .ASCIZ /MAINT CLK BIT STUCK AT 1/
4625	012326	041440	045514	041040	
4626	012334	052111	051440	052524	
4627	012342	045503	040440	020124	
4628	012350	000061			
4629	012352	051117	054504	047040	EM7: .ASCIZ /ORDY NOT SET/
4630	012360	052117	051440	052105	
4631	012366	000			
4632	012367	117	042122	020131	EM8: .ASCIZ /ORDY NOT CLEARED/
4633	012374	047516	020124	046103	
4634	012402	040505	042522	000104	
4635	012410	041517	051117	047040	EM9: .ASCIZ /OCOR NOT SET/
4636	012416	052117	051440	052105	
4637	012424	000			
4638	012425	117	047503	020122	EM10: .ASCIZ /OCOR NOT CLEARED/
4639	012432	047516	020124	046103	
4640	012440	040505	042522	000104	
4641	012446	040517	052103	047040	EM11: .ASCIZ /OACT NOT SET/

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4642	012454	052117	051440	052105	
4643	012462	000			
4644	012463	117	041501	020124	EM12: .ASCIZ /OACT NOT CLEARED/
4645	012470	047516	020124	046103	
4646	012476	040505	042522	000104	
4647	012504	047125	051122	047040	EM13: .ASCIZ /UNRR NOT CLEARED BY SOM/
4648	012512	052117	041440	042514	
4649	012520	051101	042105	041040	
4650	012526	020131	047523	000115	
4651	012534	047125	051122	047040	EM14: .ASCIZ /UNRR NOT SET/
4652	012542	052117	051440	052105	
4653	012550	000			
4654	012551	125	051116	020122	EM15: .ASCIZ /UNRR NOT CLEARED BY OC/
4655	012556	047516	020124	046103	
4656	012564	040505	042522	020104	
4657	012572	054502	047440	000103	
4658	012600	047125	051122	047040	EM16: .ASCIZ /UNRR NOT CLEARED/
4659	012606	052117	041440	042514	
4660	012614	051101	042105	000	
4661	012621	111	042122	020131	EM17: .ASCIZ /IRDY NOT SET/
4662	012626	047516	020124	042523	
4663	012634	000124			
4664	012636	051111	054504	047040	EM18: .ASCIZ /IRDY NOT CLEARED/
4665	012644	052117	041440	042514	
4666	012652	051101	042105	000	
4667	012657	111	044503	020122	EM19: .ASCIZ /ICIR NOT SET/
4668	012664	047516	020124	042523	
4669	012672	000124			
4670	012674	041511	051111	047040	EM20: .ASCIZ /ICIR NOT CLEARED/
4671	012702	052117	041440	042514	
4672	012710	051101	042105	000	
4673	012715	111	041501	020124	EM21: .ASCIZ /IACT NOT SET/
4674	012722	047516	020124	042523	
4675	012730	000124			
4676	012732	040511	052103	047040	EM22: .ASCIZ /IACT NOT CLEARED/
4677	012740	052117	041440	042514	
4678	012746	051101	042105	000	
4679	012753	104	051523	020111	EM23: .ASCIZ /DSSI NOT CLEARED/
4680	012760	047516	020124	046103	
4681	012766	040505	042522	000104	
4682	012774	051504	044523	047040	EM24: .ASCIZ /DSSI NOT SET/
4683	013002	052117	051440	052105	
4684	013010	000			
4685	013011	104	051523	020111	EM25: .ASCIZ /DSSI NOT CLEARED BY MST CLR/
4686	013016	047516	020124	046103	
4687	013024	040505	042522	020104	
4688	013032	054502	046440	052123	
4689	013040	041440	051114	000	
4690	013045	111	041516	051117	EM26: .ASCIZ /INCORRECT DATA CHAR RCV'D/
4691	013052	042522	052103	042040	
4692	013060	052101	020101	044103	
4693	013066	051101	051040	053103	
4694	013074	042047	000		
4695	013077	111	041516	051117	EM27: .ASCIZ /INCORRECT CRC BYTE RCV'D/
4696	013104	042522	052103	041440	
4697	013112	041522	041040	052131	

CZDMRF .P11

03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4698	013120	020105	041522	023526	
4699	013126	000104			
4700	013130	051522	046517	047040	EM28: .ASCIZ /RSOM NOT CLEARED/
4701	013136	052117	041440	042514	
4702	013144	051101	042105	000	
4703	013151	122	047523	020115	EM29: .ASCIZ /RSOM NOT SET/
4704	013156	047516	020124	042523	
4705	013164	000124			
4706	013166	042522	046517	047040	EM30: .ASCIZ /REOM NOT CLEARED/
4707	013174	052117	041440	042514	
4708	013202	051101	042105	000	
4709	013207	122	047505	020115	EM31: .ASCIZ /REOM NOT SET/
4710	013214	047516	020124	042523	
4711	013222	000124			
4712	013224	054124	040504	040524	EM32: .ASCIZ /TXDATA BIT NOT CLEARED/
4713	013232	041040	052111	047040	
4714	013240	052117	041440	042514	
4715	013246	051101	042105	000	
4716	013253	124	042130	052101	EM33: .ASCIZ /TXDATA BIT NOT SET/
4717	013260	020101	044502	020124	
4718	013266	047516	020124	042523	
4719	013274	000124			
4720	013276	041522	023526	020104	EM34: .ASCIZ /RCV'D DATA MISCOMPARE/
4721	013304	040504	040524	046440	
4722	013312	051511	047503	050115	
4723	013320	051101	000105		
4724	013324	041502	020103	047516	EM35: .ASCIZ /BCC NOT CLEARED/
4725	013332	020124	046103	040505	
4726	013340	042522	000104		
4727	013344	041502	020103	047516	EM36: .ASCIZ /BCC NOT SET/
4728	013352	020124	042523	000124	
4729	013360	041105	045514	047040	EM37: .ASCIZ /EBLK NOT CLEARED/
4730	013366	052117	041440	042514	
4731	013374	051101	042105	000	
4732	013401	105	046102	020113	EM38: .ASCIZ /EBLK NOT SET/
4733	013406	047516	020124	042523	
4734	013414	000124			
4735	013416	040522	020102	047516	FM39: .ASCIZ /RAB NOT CLEARED/
4736	013424	020124	046103	040505	
4737	013432	042522	000104		
4738	013436	040522	020102	047516	EM40: .ASCIZ /RAB NOT SET/
4739	013444	020124	042523	000124	
4740	013452	053117	051122	047040	EM41: .ASCIZ /OVRR NOT CLEARED/
4741	013460	052117	041440	042514	
4742	013466	051101	042105	000	
4743	013473	117	051126	020122	EM42: .ASCIZ /OVRR NOT SET/
4744	013500	047516	020124	042523	
4745	013506	000124			
4746	013510	053523	050040	041501	EM43: .ASCIZ /SW PACK #1 INCORRECT/
4747	013516	020113	030443	044440	
4748	013524	041516	051117	042522	
4749	013532	052103	000		
4750	013535	123	020127	040520	EM44: .ASCIZ /SW PACK #2 INCORRECT/
4751	013542	045503	021440	020062	
4752	013550	047111	047503	051122	
4753	013556	041505	000124		

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4754	013562	053523	050040	041501	EM45:	.ASCIZ /SW PACK #3 INCORRECT/
4755	013570	020113	031443	044440		
4756	013576	041516	051117	042522		
4757	013604	052103	000			
4758	013607	122	053103	051440	EM46:	.ASCIZ /RCV SILO NOT CLEARED BY IC/
4759	013614	046111	020117	047516		
4760	013622	020124	046103	040505		
4761	013630	042522	020104	054502		
4762	013636	044440	000103			
4763	013642	051501	042523	041115	EM47:	.ASCIZ /ASSEMB BIT COUNT INCORRECT/
4764	013650	041040	052111	041440		
4765	013656	052517	052116	044440		
4766	013664	041516	051117	042522		
4767	013672	052103	000			
4768	013675	117	042104	053040	EM48:	.ASCIZ /ODD VRC PARITY BIT NOT SET/
4769	013702	041522	050040	051101		
4770	013710	052111	020131	044502		
4771	013716	020124	047516	020124		
4772	013724	042523	000124			
4773	013730	042117	020104	051126	EM49:	.ASCIZ /ODD VRC PARITY BIT NOT CLEARED/
4774	013736	020103	040520	044522		
4775	013744	054524	041040	052111		
4776	013752	047040	052117	041440		
4777	013760	042514	051101	042105		
4778	013766	000				
4779	013767	105	042526	020116	EM50:	.ASCIZ /EVEN VRC PARITY BIT NOT SET/
4780	013774	051126	020103	040520		
4781	014002	044522	054524	041040		
4782	014010	052111	047040	052117		
4783	014016	051440	052105	000		
4784	014023	105	042526	020116	EM51:	.ASCIZ /EVEN VRC PARITY BIT NOT CLEARED/
4785	014030	051126	020103	040520		
4786	014036	044522	054524	041040		
4787	014044	052111	047040	052117		
4788	014052	041440	042514	051101		
4789	014060	042105	000			
4790	014063	122	040505	054504	EM52:	.ASCIZ /READY NOT SET AFTER AX REG WRITE/
4791	014070	047040	052117	051440		
4792	014076	052105	040440	052106		
4793	014104	051105	040440	020130		
4794	014112	042522	020107	051127		
4795	014120	052111	000105			
4796	014124	042522	042101	020131	EM53:	.ASCIZ /READY NOT SET AFTER AX REG READ/
4797	014132	047516	020124	042523		
4798	014140	020124	043101	042524		
4799	014146	020122	054101	051040		
4800	014154	043505	051040	040505		
4801	014162	000104				
4802	014164	054124	052440	042116	EM54:	.ASCIZ /TX UNDERRUN ERROR/
4803	014172	051105	052522	020116		
4804	014200	051105	047522	000122		
4805	014206	052122	020123	047516	EM60:	.ASCIZ /RTS NOT SET/
4806	014214	020124	042523	000124		
4807	014222	052122	020123	047516	EM65:	.ASCIZ /RTS NOT CLEARED/
4808	014230	020124	046103	040505		
4809	014236	042522	000104			

CZDMRF .P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

```

4810
4811
4812
4813 014242 047111 052502 027523 DH1: .ASCIZ &INBUS/OUTBUS REG &
4814 014250 052517 041124 051525
4815 014256 051040 043505 000040
4816 014264 044514 042516 052440 DH2: .ASCIZ /LINE UNIT INBUS REGS :/
4817 014272 044516 020124 047111
4818 014300 052502 020123 042522
4819 014306 051507 035040 000
4820 014313 122 043505 030061 DH3: .ASCIZ /REG10 REG11 REG12 REG13/
4821 014320 020040 051040 043505
4822 014326 030461 020040 051040
4823 014334 043505 031061 020040
4824 014342 051040 043505 031461
4825 014350 000
4826 014351 040 020040 051040 DH4: .ASCIZ / REG14 REG15 REG16 REG17/
4827 014356 043505 032061 020040
4828 014364 051040 043505 032461
4829 014372 020040 051040 043505
4830 014400 033061 020040 051040
4831 014406 043505 033461 000
4832 014413 061 000065 DH5: .ASCIZ /15/
4833 014416 033061 000 DH6: .ASCIZ /16/
4834 014421 114 047111 020105 DH7: .ASCIZ /LINE UNIT EXTENDED REGS :/
4835 014426 047125 052111 042440
4836 014434 052130 047105 042504
4837 014442 020104 042522 051507
4838 014450 035040 000
4839 014453 101 030130 030455 DH8: .ASCIZ /AX0-15 AX0-16 AX1-15 AX1-16/
4840 014460 020065 040440 030130
4841 014466 030455 020066 040440
4842 014474 030530 030455 020065
4843 014502 040440 030530 030455
4844 014510 000066
4845 014512 020040 020040 054101 DH9: .ASCIZ / AX2-15 AX2-16 AX3-15 AX3-16/
4846 014520 026462 032461 020040
4847 014526 054101 026462 033061
4848 014534 020040 054101 026463
4849 014542 032461 020040 054101
4850 014550 026463 033061 000

```

```

4851
4852 014556 .EVEN
4853
4854
4855
4856
4857

```

```

4858 014556 BGNMSG ERR1
4859 014556 PRINTB #FM11,#ADDRES,MPCSR
4860 014556
4861 014556 013746 002424
4862 014562 012746 035522
4863 014566 012746 011432
4864 014572 012746 000003
4865 014576 010600

```

ERR1::

```

MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0

```


CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

```

4866 014600 104414
4867 014602 062706 000010
4868 014606
4869 014606
4870 014606 104423
4871
4872
4873
4874 014610
4875 014610
4876 014610
4877 014610 013746 002424
4878 014614 012746 035522
4879 014620 012746 011432
4880 014624 012746 000003
4881 014630 010600
4882 014632 104414
4883 014634 062706 000010
4884 014640
4885 014640 012746 011442
4886 014644 012746 000001
4887 014650 010600
4888 014652 104414
4889 014654 062706 000004
4890 014660
4891 014660 013746 002356
4892 014664 012746 014242
4893 014670 012746 011624
4894 014674 012746 000003
4895 014700 010600
4896 014702 104414
4897 014704 062706 000010
4898 014710
4899 014710 013746 002364
4900 014714 013746 002362
4901 014720 012746 011464
4902 014724 012746 000003
4903 014730 010600
4904 014732 104414
4905 014734 062706 000010
4906 014740
4907 014740 012746 014313
4908 014744 012746 014264
4909 014750 012746 011526
4910 014754 012746 000003
4911 014760 010600
4912 014762 104415
4913 014764 062706 000010
4914 014770
4915 014770 013746 002266
4916 014774 013746 002264
4917 015000 013746 002262
4918 015004 013746 002260
4919 015010 012746 011541
4920 015014 012746 000005
4921 015020 010600

```

ENDMSG

BGNMSG ERR2

PRINTB #FMT1,#ADDRES,MPCSR

PRINTB #FMT2

PRINTB #FMT7,#DH1,REGNUM

PRINTB #FMT3,GOODAT,BADDAT

PRINTX #FMT4,#DH2,#DH3

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

```

TRAP ADD C$PNTB #10,SP
L10002: TRAP C$MSG
ERR2::
MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP ADD C$PNTB #10,SP
MOV #FMT2,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP ADD C$PNTB #4,SP
MOV REGNUM,-(SP)
MOV #DH1,-(SP)
MOV #FMT7,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP ADD C$PNTB #10,SP
MOV BADDAT,-(SP)
MOV GOODAT,-(SP)
MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP ADD C$PNTB #10,SP
MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP ADD C$PNTX #10,SP
MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0

```

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4922	015022	104415			TRAP	C\$PNTX
4923	015024	062706	000014		ADD	#14,SP
4924	015030			PRINTX #FMT9,#DH4		
4925	015030	012746	014351		MOV	#DH4,-(SP)
4926	015034	012746	011670		MOV	#FMT9,-(SP)
4927	015040	012746	000002		MOV	#2,-(SP)
4928	015044	010600			MOV	SP,R0
4929	015046	104415			TRAP	C\$PNTX
4930	015050	062706	000006		ADD	#6,SP
4931	015054			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17		
4932	015054	013746	002276		MOV	LUR17,-(SP)
4933	015060	013746	002274		MOV	LUR16,-(SP)
4934	015064	013746	002272		MOV	LUR15,-(SP)
4935	015070	013746	002270		MOV	LUR14,-(SP)
4936	015074	012746	011571		MOV	#FMT6,-(SP)
4937	015100	012746	000005		MOV	#5,-(SP)
4938	015104	010600			MOV	SP,R0
4939	015106	104415			TRAP	C\$PNTX
4940	015110	062706	000014		ADD	#14,SP
4941	015114			ENDMSG		
4942	015114					
4943	015114	104423			L10003: TRAP	C\$MSG
4944						
4945						
4946						
4947						
4948						
4949	015116			BGNMSG ERR3		
4950	015116					
4951	015116			PRINTB #FMT1,#ADDRES,MPCSR	ERR3::	
4952	015116	013746	002424		MOV	MPCSR,-(SP)
4953	015122	012746	035522		MOV	#ADDRES,-(SP)
4954	015126	012746	011432		MOV	#FMT1,-(SP)
4955	015132	012746	000003		MOV	#3,-(SP)
4956	015136	010600			MOV	SP,R0
4957	015140	104414			TRAP	C\$PNTB
4958	015142	062706	000010		ADD	#10,SP
4959	015146			PRINTB #FMT2		
4960	015146	012746	011442		MOV	#FMT2,-(SP)
4961	015152	012746	000001		MOV	#1,-(SP)
4962	015156	010600			MOV	SP,R0
4963	015160	104414			TRAP	C\$PNTB
4964	015162	062706	000004		ADD	#4,SP
4965	015166			PRINTB #FMT8,TMP1,TMP0		
4966	015166	013746	002506		MOV	TMP0,-(SP)
4967	015172	013746	002510		MOV	TMP1,-(SP)
4968	015176	012746	011634		MOV	#FMT8,-(SP)
4969	015202	012746	000003		MOV	#3,-(SP)
4970	015206	010600			MOV	SP,R0
4971	015210	104414			TRAP	C\$PNTB
4972	015212	062706	000010		ADD	#10,SP
4973	015216			PRINTB #FMT3,GOODAT,BADDAT		
4974	015216	013746	002364		MOV	BADDAT,-(SP)
4975	015222	013746	002362		MOV	GOODAT,-(SP)
4976	015226	012746	011464		MOV	#FMT3,-(SP)
4977	015232	012746	000003		MOV	#3,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

4978	015236	010600			MOV	SP,R0
4979	015240	104414			TRAP	C\$PNTB
4980	015242	062706	000010		ADD	#10,SP
4981	015246			PRINTX	#FMT4,#DH2,#DH3	
4982	015246	012746	014313		MOV	#DH3,-(SP)
4983	015252	012746	014264		MOV	#DH2,-(SP)
4984	015256	012746	011526		MOV	#FMT4,-(SP)
4985	015262	012746	000003		MOV	#3,-(SP)
4986	015266	010600			MOV	SP,R0
4987	015270	104415			TRAP	C\$PNTX
4988	015272	062706	000010		ADD	#10,SP
4989	015276			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
4990	015276	013746	002266		MOV	LUR13,-(SP)
4991	015302	013746	002264		MOV	LUR12,-(SP)
4992	015306	013746	002262		MOV	LUR11,-(SP)
4993	015312	013746	002260		MOV	LUR10,-(SP)
4994	015316	012746	011541		MOV	#FMT5,-(SP)
4995	015322	012746	000005		MOV	#5,-(SP)
4996	015326	010600			MOV	SP,R0
4997	015330	104415			TRAP	C\$PNTX
4998	015332	062706	000014		ADD	#14,SP
4999	015336			PRINTX	#FMT9,#DH4	
5000	015336	012746	014351		MOV	#DH4,-(SP)
5001	015342	012746	011670		MOV	#FMT9,-(SP)
5002	015346	012746	000002		MOV	#2,-(SP)
5003	015352	010600			MOV	SP,R0
5004	015354	104415			TRAP	C\$PNTX
5005	015356	062706	000006		ADD	#6,SP
5006	015362			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17	
5007	015362	013746	002276		MOV	LUR17,-(SP)
5008	015366	013746	002274		MOV	LUR16,-(SP)
5009	015372	013746	002272		MOV	LUR15,-(SP)
5010	015376	013746	002270		MOV	LUR14,-(SP)
5011	015402	012746	011571		MOV	#FMT6,-(SP)
5012	015406	012746	000005		MOV	#5,-(SP)
5013	015412	010600			MOV	SP,R0
5014	015414	104415			TRAP	C\$PNTX
5015	015416	062706	000014		ADD	#14,SP
5016	015422			PRINTX	#FMT4,#DH7,#DH8	
5017	015422	012746	014453		MOV	#DH8,-(SP)
5018	015426	012746	014421		MOV	#DH7,-(SP)
5019	015432	012746	011526		MOV	#FMT4,-(SP)
5020	015436	012746	000003		MOV	#3,-(SP)
5021	015442	010600			MOV	SP,R0
5022	015444	104415			TRAP	C\$PNTX
5023	015446	062706	000010		ADD	#10,SP
5024	015452			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16	
5025	015452	013746	002306		MOV	AX1.16,-(SP)
5026	015456	013746	002304		MOV	AX1.15,-(SP)
5027	015462	013746	002302		MOV	AX0.16,-(SP)
5028	015466	013746	002300		MOV	AX0.15,-(SP)
5029	015472	012746	011541		MOV	#FMT5,-(SP)
5030	015476	012746	000005		MOV	#5,-(SP)
5031	015502	010600			MOV	SP,R0
5032	015504	104415			TRAP	C\$PNTX
5033	015506	062706	000014		ADD	#14,SP

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5034	015512			PRINTX #FMT9,#DH9	
5035	015512	012746	014512		MOV #DH9,-(SP)
5036	015516	012746	011670		MOV #FMT9,-(SP)
5037	015522	012746	000002		MOV #2,-(SP)
5038	015526	010600			MOV SP,R0
5039	015530	104415			TRAP C\$PNTX
5040	015532	062706	000006		ADD #6,SP
5041	015536			PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
5042	015536	013746	002316		MOV AX3.16,-(SP)
5043	015542	013746	002314		MOV AX3.15,-(SP)
5044	015546	013746	002312		MOV AX2.16,-(SP)
5045	015552	013746	002310		MOV AX2.15,-(SP)
5046	015556	012746	011571		MOV #FMT6,-(SP)
5047	015562	012746	000005		MOV #5,-(SP)
5048	015566	010600			MOV SP,R0
5049	015570	104415			TRAP C\$PNTX
5050	015572	062706	000014		ADD #14,SP
5051	015576			ENDMSG	
5052	015576				L10004:
5053	015576	104423			TRAP C\$MSG
5054					
5055					
5056					
5057					
5058					
5059	015600			BGNMSG ERR4	
5060	015600				ERR4::
5061	015600			PRINTB #FMT10,SUBRPC	
5062	015600	013746	002330		MOV SUBRPC,-(SP)
5063	015604	012746	011675		MOV #FMT10,-(SP)
5064	015610	012746	000002		MOV #2,-(SP)
5065	015614	010600			MOV SP,R0
5066	015616	104414			TRAP C\$PNTB
5067	015620	062706	000006		ADD #6,SP
5068	015624			PRINTB #FMT1,#ADDRES,MPCSR	
5069	015624	013746	002424		MOV MPCSR,-(SP)
5070	015630	012746	035522		MOV #ADDRES,-(SP)
5071	015634	012746	011432		MOV #FMT1,-(SP)
5072	015640	012746	000003		MOV #3,-(SP)
5073	015644	010600			MOV SP,R0
5074	015646	104414			TRAP C\$PNTB
5075	015650	062706	000010		ADD #10,SP
5076	015654			PRINTB #FMT2	
5077	015654	012746	011442		MOV #FMT2,-(SP)
5078	015660	012746	000001		MOV #1,-(SP)
5079	015664	010600			MOV SP,R0
5080	015666	104414			TRAP C\$PNTB
5081	015670	062706	000004		ADD #4,SP
5082	015674			PRINTB #FMT7,#DH1,REGNUM	
5083	015674	013746	002356		MOV REGNUM,-(SP)
5084	015700	012746	014242		MOV #DH1,-(SP)
5085	015704	012746	011624		MOV #FMT7,-(SP)
5086	015710	012746	000003		MOV #3,-(SP)
5087	015714	010600			MOV SP,R0
5088	015716	104414			TRAP C\$PNTB
5089	015720	062706	000010		ADD #10,SP

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5090	015724			PRINTX #FMT4,#DH2,#DH3	
5091	015724	012746	014313		MOV #DH3,-(SP)
5092	015730	012746	014264		MOV #DH2,-(SP)
5093	015734	012746	011526		MOV #FMT4,-(SP)
5094	015740	012746	000003		MOV #3,-(SP)
5095	015744	010600			MOV SP,R0
5096	015746	104415			TRAP C\$PNTX
5097	015750	062706	000010		ADD #10,SP
5098	015754			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13	
5099	015754	013746	002266		MOV LUR13,-(SP)
5100	015760	013746	002264		MOV LUR12,-(SP)
5101	015764	013746	002262		MOV LUR11,-(SP)
5102	015770	013746	002260		MOV LUR10,-(SP)
5103	015774	012746	011541		MOV #FMT5,-(SP)
5104	016000	012746	000005		MOV #5,-(SP)
5105	016004	010600			MOV SP,R0
5106	016006	104415			TRAP C\$PNTX
5107	016010	062706	000014		ADD #14,SP
5108	016014			PRINTX #FMT9,#DH4	
5109	016014	012746	014351		MOV #DH4,-(SP)
5110	016020	012746	011670		MOV #FMT9,-(SP)
5111	016024	012746	000002		MOV #2,-(SP)
5112	016030	010600			MOV SP,R0
5113	016032	104415			TRAP C\$PNTX
5114	016034	062706	000006		ADD #6,SP
5115	016040			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17	
5116	016040	013746	002276		MOV LUR17,-(SP)
5117	016044	013746	002274		MOV LUR16,-(SP)
5118	016050	013746	002272		MOV LUR15,-(SP)
5119	016054	013746	002270		MOV LUR14,-(SP)
5120	016060	012746	011571		MOV #FMT6,-(SP)
5121	016064	012746	000005		MOV #5,-(SP)
5122	016070	010600			MOV SP,R0
5123	016072	104415			TRAP C\$PNTX
5124	016074	062706	000014		ADD #14,SP
5125	016100			PRINTX #FMT4,#DH7,#DH8	
5126	016100	012746	014453		MOV #DH8,-(SP)
5127	016104	012746	014421		MOV #DH7,-(SP)
5128	016110	012746	011526		MOV #FMT4,-(SP)
5129	016114	012746	000003		MOV #3,-(SP)
5130	016120	010600			MOV SP,R0
5131	016122	104415			TRAP C\$PNTX
5132	016124	062706	000010		ADD #10,SP
5133	016130			PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16	
5134	016130	013746	002306		MOV AX1.16,-(SP)
5135	016134	013746	002304		MOV AX1.15,-(SP)
5136	016140	013746	002302		MOV AX0.16,-(SP)
5137	016144	013746	002300		MOV AX0.15,-(SP)
5138	016150	012746	011541		MOV #FMT5,-(SP)
5139	016154	012746	000005		MOV #5,-(SP)
5140	016160	010600			MOV SP,R0
5141	016162	104415			TRAP C\$PNTX
5142	016164	062706	000014		ADD #14,SP
5143	016170			PRINTX #FMT9,#DH9	
5144	016170	012746	014512		MOV #DH9,-(SP)
5145	016174	012746	011670		MOV #FMT9,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5146	016200	012746	000002					MOV	#2,-(SP)
5147	016204	010600						MOV	SP,R0
5148	016206	104415						TRAP	C\$PNTX
5149	016210	062706	000006					ADD	#6,SP
5150	016214			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16				
5151	016214	013746	002316					MOV	AX3.16,-(SP)
5152	016220	013746	002314					MOV	AX3.15,-(SP)
5153	016224	013746	002312					MOV	AX2.16,-(SP)
5154	016230	013746	002310					MOV	AX2.15,-(SP)
5155	016234	012746	011571					MOV	#FMT6,-(SP)
5156	016240	012746	000005					MOV	#5,-(SP)
5157	016244	010600						MOV	SP,R0
5158	016246	104415						TRAP	C\$PNTX
5159	016250	062706	000014					ADD	#14,SP
5160	016254			ENDMSG					
5161	016254								
5162	016254	104423					L10005:	TRAP	C\$MSG
5163									
5164									
5165									
5166									
5167									
5168	016256			BGNMSG	ERR5				
5169	016256						ERR5::		
5170	016256			PRINTB	#FMT1,#ADDRES,MPCSR				
5171	016256	013746	002424					MOV	MPCSR,-(SP)
5172	016262	012746	035522					MOV	#ADDRES,-(SP)
5173	016266	012746	011432					MOV	#FMT1,-(SP)
5174	016272	012746	000003					MOV	#3,-(SP)
5175	016276	010600						MOV	SP,R0
5176	016300	104414						TRAP	C\$PNTB
5177	016302	062706	000010					ADD	#10,SP
5178	016306			PRINTB	#FMT11,REGNUM,LOADAT				
5179	016306	013746	002366					MOV	LOADAT,-(SP)
5180	016312	013746	002356					MOV	REGNUM,-(SP)
5181	016316	012746	011726					MOV	#FMT11,-(SP)
5182	016322	012746	000003					MOV	#3,-(SP)
5183	016326	010600						MOV	SP,R0
5184	016330	104414						TRAP	C\$PNTB
5185	016332	062706	000010					ADD	#10,SP
5186	016336			PRINTB	#FMT2				
5187	016336	012746	011442					MOV	#FMT2,-(SP)
5188	016342	012746	000001					MOV	#1,-(SP)
5189	016346	010600						MOV	SP,R0
5190	016350	104414						TRAP	C\$PNTB
5191	016352	062706	000004					ADD	#4,SP
5192	016356			PRINTB	#FMT8,TMP1,TMP0				
5193	016356	013746	002506					MOV	TMP0,-(SP)
5194	016362	013746	002510					MOV	TMP1,-(SP)
5195	016366	012746	011634					MOV	#FMT8,-(SP)
5196	016372	012746	000003					MOV	#3,-(SP)
5197	016376	010600						MOV	SP,R0
5198	016400	104414						TRAP	C\$PNTB
5199	016402	062706	000010					ADD	#10,SP
5200	016406			PRINTB	#FMT3,GOODAT,BADDAT				
5201	016406	013746	002364					MOV	BADDAT,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5202	016412	013746	002362		MOV	GOODAT,-(SP)
5203	016416	012746	011464		MOV	#FMT3,-(SP)
5204	016422	012746	000003		MOV	#3,-(SP)
5205	016426	010600			MOV	SP,R0
5206	016430	104414			TRAP	CSPNTB
5207	016432	062706	000010		ADD	#10,SP
5208	016436			PRINTX		
5209	016436	012746	014313		MOV	#DH3,-(SP)
5210	016442	012746	014264		MOV	#DH2,-(SP)
5211	016446	012746	011526		MOV	#FMT4,-(SP)
5212	016452	012746	000003		MOV	#3,-(SP)
5213	016456	010600			MOV	SP,R0
5214	016460	104415			TRAP	CSPNTX
5215	016462	062706	000010		ADD	#10,SP
5216	016466			PRINTX		
5217	016466	013746	002266		MOV	LUR13,-(SP)
5218	016472	013746	002264		MOV	LUR12,-(SP)
5219	016476	013746	002262		MOV	LUR11,-(SP)
5220	016502	013746	002260		MOV	LUR10,-(SP)
5221	016506	012746	011541		MOV	#FMT5,-(SP)
5222	016512	012746	000005		MOV	#5,-(SP)
5223	016516	010600			MOV	SP,R0
5224	016520	104415			TRAP	CSPNTX
5225	016522	062706	000014		ADD	#14,SP
5226	016526			PRINTX		
5227	016526	012746	014351		MOV	#DH4,-(SP)
5228	016532	012746	011670		MOV	#FMT9,-(SP)
5229	016536	012746	000002		MOV	#2,-(SP)
5230	016542	010600			MOV	SP,R0
5231	016544	104415			TRAP	CSPNTX
5232	016546	062706	000006		ADD	#6,SP
5233	016552			PRINTX		
5234	016552	013746	002276		MOV	LUR17,-(SP)
5235	016556	013746	002274		MOV	LUR16,-(SP)
5236	016562	013746	002272		MOV	LUR15,-(SP)
5237	016566	013746	002270		MOV	LUR14,-(SP)
5238	016572	012746	011571		MOV	#FMT6,-(SP)
5239	016576	012746	000005		MOV	#5,-(SP)
5240	016602	010600			MOV	SP,R0
5241	016604	104415			TRAP	CSPNTX
5242	016606	062706	000014		ADD	#14,SP
5243	016612			PRINTX		
5244	016612	012746	014453		MOV	#DH8,-(SP)
5245	016616	012746	014421		MOV	#DH7,-(SP)
5246	016622	012746	011526		MOV	#FMT4,-(SP)
5247	016626	012746	000003		MOV	#3,-(SP)
5248	016632	010600			MOV	SP,R0
5249	016634	104415			TRAP	CSPNTX
5250	016636	062706	000010		ADD	#10,SP
5251	016642			PRINTX		
5252	016642	013746	002306		MOV	AX1.16,-(SP)
5253	016646	013746	002304		MOV	AX1.15,-(SP)
5254	016652	013746	002302		MOV	AX0.16,-(SP)
5255	016656	013746	002300		MOV	AX0.15,-(SP)
5256	016662	012746	011541		MOV	#FMT5,-(SP)
5257	016666	012746	000005		MOV	#5,-(SP)

GLOBAL ERROR REPORT SECTION

5258	016672	010600			MOV	SP,R0
5259	016674	104415			TRAP	C\$PNTX
5260	016676	062706	000014		ADD	#14,SP
5261	016702			PRINTX	#FMT9,#DH9	
5262	016702	012746	014512		MOV	#DH9,-(SP)
5263	016706	012746	011670		MOV	#FMT9,-(SP)
5264	016712	012746	000002		MOV	#2,-(SP)
5265	016716	010600			MOV	SP,R0
5266	016720	104415			TRAP	C\$PNTX
5267	016722	062706	000006		ADD	#6,SP
5268	016726			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
5269	016726	013746	002316		MOV	AX3.16,-(SP)
5270	016732	013746	002314		MOV	AX3.15,-(SP)
5271	016736	013746	002312		MOV	AX2.16,-(SP)
5272	016742	013746	002310		MOV	AX2.15,-(SP)
5273	016746	012746	011571		MOV	#FMT6,-(SP)
5274	016752	012746	000005		MOV	#5,-(SP)
5275	016756	010600			MOV	SP,R0
5276	016760	104415			TRAP	C\$PNTX
5277	016762	062706	000014		ADD	#14,SP
5278	016766			ENDMSG		
5279	016766					
5280	016766	104423			L10006:	TRAP
5281						C\$MSG
5282						
5283						
5284						
5285						
5286	016770			BGNMSG	ERR6	
5287	016770					
5288	016770			PRINTB	#FMT10,SUBRPC	ERR6::
5289	016770	013746	002330			MOV
5290	016774	012746	011675			MOV
5291	017000	012746	000002			MOV
5292	017004	010600				MOV
5293	017006	104414				MOV
5294	017010	062706	000006			TRAP
5295	017014			PRINTB	#FMT1,#ADDRES,MPCSR	ADD
5296	017014	013746	002424			MOV
5297	017020	012746	035522			MOV
5298	017024	012746	011432			MOV
5299	017030	012746	000003			MOV
5300	017034	010600				MOV
5301	017036	104414				MOV
5302	017040	062706	000010			TRAP
5303	017044			PRINTB	#FMT2	ADD
5304	017044	012746	011442			MOV
5305	017050	012746	000001			MOV
5306	017054	010600				MOV
5307	017056	104414				MOV
5308	017060	062706	000004			TRAP
5309	017064			PRINTB	#FMT8,TMP1,TMP0	ADD
5310	017064	013746	002506			MOV
5311	017070	013746	002510			MOV
5312	017074	012746	011634			MOV
5313	017100	012746	000003			MOV

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5314 017104 010600
 5315 017106 104414
 5316 017110 062706 000010
 5317 017114
 5318 017114 012746 014313
 5319 017120 012746 014264
 5320 017124 012746 011526
 5321 017130 012746 000003
 5322 017134 010600
 5323 017136 104415
 5324 017140 062706 000010
 5325 017144
 5326 017144 013746 002266
 5327 017150 013746 002264
 5328 017154 013746 002262
 5329 017160 013746 002260
 5330 017164 012746 011541
 5331 017170 012746 000005
 5332 017174 010600
 5333 017176 104415
 5334 017200 062706 000014
 5335 017204
 5336 017204 012746 014351
 5337 017210 012746 011670
 5338 017214 012746 000002
 5339 017220 010600
 5340 017222 104415
 5341 017224 062706 000006
 5342 017230
 5343 017230 013746 002276
 5344 017234 013746 002274
 5345 017240 013746 002272
 5346 017244 013746 002270
 5347 017250 012746 011571
 5348 017254 012746 000005
 5349 017260 010600
 5350 017262 104415
 5351 017264 062706 000014
 5352 017270
 5353 017270 012746 014453
 5354 017274 012746 014421
 5355 017300 012746 011526
 5356 017304 012746 000003
 5357 017310 010600
 5358 017312 104415
 5359 017314 062706 000010
 5360 017320
 5361 017320 013746 002306
 5362 017324 013746 002304
 5363 017330 013746 002302
 5364 017334 013746 002300
 5365 017340 012746 011541
 5366 017344 012746 000005
 5367 017350 010600
 5368 017352 104415
 5369 017354 062706 000014

PRINTX #FMT4,#DH2,#DH3

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

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

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
 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

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5370	017360			PRINTX #FMT9,#DH9	
5371	017360	012746	014512		MOV #DH9,-(SP)
5372	017364	012746	011670		MOV #FMT9,-(SP)
5373	017370	012746	000002		MOV #2,-(SP)
5374	017374	010600			MOV SP,R0
5375	017376	104415			TRAP C\$PNTX
5376	017400	062706	000006		ADD #6,SP
5377	017404			PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
5378	017404	013746	002316		MOV AX3.16,-(SP)
5379	017410	013746	002314		MOV AX3.15,-(SP)
5380	017414	013746	002312		MOV AX2.16,-(SP)
5381	017420	013746	002310		MOV AX2.15,-(SP)
5382	017424	012746	011571		MOV #FMT6,-(SP)
5383	017430	012746	000005		MOV #5,-(SP)
5384	017434	010600			MOV SP,R0
5385	017436	104415			TRAP C\$PNTX
5386	017440	062706	000014		ADD #14,SP
5387	017444			ENDMSG	
5388	017444				
5389	017444	104423			L10007: TRAP C\$MSG
5390					
5391					
5392					
5393					
5394					
5395	017446			BGNMSG ERR7	
5396	017446				ERR7::
5397	017446			PRINTB #FMT1,#ADDRES,MPCSR	
5398	017446	013746	002424		MOV MPCSR,-(SP)
5399	017452	012746	035522		MOV #ADDRES,-(SP)
5400	017456	012746	011432		MOV #FMT1,-(SP)
5401	017462	012746	000003		MOV #3,-(SP)
5402	017466	010600			MOV SP,R0
5403	017470	104414			TRAP C\$PNTB
5404	017472	062706	000010		ADD #10,SP
5405	017476			PRINTB #FMT2	
5406	017476	012746	011442		MOV #FMT2,-(SP)
5407	017502	012746	000001		MOV #1,-(SP)
5408	017506	010600			MOV SP,R0
5409	017510	104414			TRAP C\$PNTB
5410	017512	062706	000004		ADD #4,SP
5411	017516			PRINTB #FMT7,#DH1,REGNUM	
5412	017516	013746	002356		MOV REGNUM,-(SP)
5413	017522	012746	014242		MOV #DH1,-(SP)
5414	017526	012746	011624		MOV #FMT7,-(SP)
5415	017532	012746	000003		MOV #3,-(SP)
5416	017536	010600			MOV SP,R0
5417	017540	104414			TRAP C\$PNTB
5418	017542	062706	000010		ADD #10,SP
5419	017546			PRINTX #FMT4,#DH2,#DH3	
5420	017546	012746	014313		MOV #DH3,-(SP)
5421	017552	012746	014264		MOV #DH2,-(SP)
5422	017556	012746	011526		MOV #FMT4,-(SP)
5423	017562	012746	000003		MOV #3,-(SP)
5424	017566	010600			MOV SP,R0
5425	017570	104415			TRAP C\$PNTX

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5426	017572	062706	000010		ADD	#10,SP
5427	017576			PRINTX		
5428	017576	013746	002266		MOV	LUR13,-(SP)
5429	017602	013746	002264		MOV	LUR12,-(SP)
5430	017606	013746	002262		MOV	LUR11,-(SP)
5431	017612	013746	002260		MOV	LUR10,-(SP)
5432	017616	012746	011541		MOV	#FMT5,-(SP)
5433	017622	012746	000005		MOV	#5,-(SP)
5434	017626	010600			MOV	SP,R0
5435	017630	104415			TRAP	C\$PNTX
5436	017632	062706	000014		ADD	#14,SP
5437	017636			PRINTX		
5438	017636	012746	014351		MOV	#DH4,-(SP)
5439	017642	012746	011670		MOV	#FMT9,-(SP)
5440	017646	012746	000002		MOV	#2,-(SP)
5441	017652	010600			MOV	SP,R0
5442	017654	104415			TRAP	C\$PNTX
5443	017656	062706	000006		ADD	#6,SP
5444	017662			PRINTX		
5445	017662	013746	002276		MOV	LUR17,-(SP)
5446	017666	013746	002274		MOV	LUR16,-(SP)
5447	017672	013746	002272		MOV	LUR15,-(SP)
5448	017676	013746	002270		MOV	LUR14,-(SP)
5449	017702	012746	011571		MOV	#FMT6,-(SP)
5450	017706	012746	000005		MOV	#5,-(SP)
5451	017712	010600			MOV	SP,R0
5452	017714	104415			TRAP	C\$PNTX
5453	017716	062706	000014		ADD	#14,SP
5454	017722			PRINTX		
5455	017722	012746	014453		MOV	#DH8,-(SP)
5456	017726	012746	014421		MOV	#DH7,-(SP)
5457	017732	012746	011526		MOV	#FMT4,-(SP)
5458	017736	012746	000003		MOV	#3,-(SP)
5459	017742	010600			MOV	SP,R0
5460	017744	104415			TRAP	C\$PNTX
5461	017746	062706	000010		ADD	#10,SP
5462	017752			PRINTX		
5463	017752	013746	002306		MOV	AX1.16,-(SP)
5464	017756	013746	002304		MOV	AX1.15,-(SP)
5465	017762	013746	002302		MOV	AX0.16,-(SP)
5466	017766	013746	002300		MOV	AX0.15,-(SP)
5467	017772	012746	011541		MOV	#FMT5,-(SP)
5468	017776	012746	000005		MOV	#5,-(SP)
5469	020002	010600			MOV	SP,R0
5470	020004	104415			TRAP	C\$PNTX
5471	020006	062706	000014		ADD	#14,SP
5472	020012			PRINTX		
5473	020012	012746	014512		MOV	#DH9,-(SP)
5474	020016	012746	011670		MOV	#FMT9,-(SP)
5475	020022	012746	000002		MOV	#2,-(SP)
5476	020026	010600			MOV	SP,R0
5477	020030	104415			TRAP	C\$PNTX
5478	020032	062706	000006		ADD	#6,SP
5479	020036			PRINTX		
5480	020036	013746	002316		MOV	AX3.16,-(SP)
5481	020042	013746	002314		MOV	AX3.15,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5482 020046 013746 002312
5483 020052 013746 002310
5484 020056 012746 011571
5485 020062 012746 000005
5486 020066 010600
5487 020070 104415
5488 020072 062706 000014
5489 020076
5490 020076
5491 020076 104423
5492
5493
5494
5495
5496
5497 020100
5498 020100
5499 020100
5500 020100 013746 002330
5501 020104 012746 011675
5502 020110 012746 000002
5503 020114 010600
5504 020116 104414
5505 020120 062706 000006
5506 020124
5507 020124 013746 002424
5508 020130 012746 035522
5509 020134 012746 011432
5510 020140 012746 000003
5511 020144 010600
5512 020146 104414
5513 020150 062706 000010
5514 020154
5515 020154 012746 011442
5516 020160 012746 000001
5517 020164 010600
5518 020166 104414
5519 020170 062706 000004
5520 020174
5521 020174 013746 002356
5522 020200 012746 014242
5523 020204 012746 011624
5524 020210 012746 000003
5525 020214 010600
5526 020216 104414
5527 020220 062706 000010
5528 020224
5529 020224 013746 002364
5530 020230 013746 002362
5531 020234 012746 011464
5532 020240 012746 000003
5533 020244 010600
5534 020246 104414
5535 020250 062706 000010
5536 020254
5537 020254 012746 014313

ENDMSG

BGNMSG ERR8

PRINTB #FMT10,SUBRPC

PRINTB #FMT1,#ADDRES,MPCSR

PRINTB #FMT2

PRINTB #FMT7,#DH1,REGNUM

PRINTB #FMT3,GOODAT,BADDAT

PRINTX #FMT4,#DH2,#DH3

MOV AX2.16,-(SP)
MOV AX2.15,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

L10010: TRAP C\$MSG

ERR8::

MOV SUBRPC,-(SP)
MOV #FMT10,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP

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

MOV #FMT2,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #4,SP

MOV REGNUM,-(SP)
MOV #DH1,-(SP)
MOV #FMT7,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV BADDAT,-(SP)
MOV GOODAT,-(SP)
MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5538	020260	012746	014264		MOV	#DH2,-(SP)
5539	020264	012746	011526		MOV	#FMT4,-(SP)
5540	020270	012746	000003		MOV	#3,-(SP)
5541	020274	010600			MOV	SP,R0
5542	020276	104415			TRAP	C\$PNTX
5543	020300	062706	000010		ADD	#10,SP
5544	020304			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
5545	020304	013746	002266		MOV	LUR13,-(SP)
5546	020310	013746	002264		MOV	LUR12,-(SP)
5547	020314	013746	002262		MOV	LUR11,-(SP)
5548	020320	013746	002260		MOV	LUR10,-(SP)
5549	020324	012746	011541		MOV	#FMT5,-(SP)
5550	020330	012746	000005		MOV	#5,-(SP)
5551	020334	010600			MOV	SP,R0
5552	020336	104415			TRAP	C\$PNTX
5553	020340	062706	000014		ADD	#14,SP
5554	020344			PRINTX	#FMT9,#DH4	
5555	020344	012746	014351		MOV	#DH4,-(SP)
5556	020350	012746	011670		MOV	#FMT9,-(SP)
5557	020354	012746	000002		MOV	#2,-(SP)
5558	020360	010600			MOV	SP,R0
5559	020362	104415			TRAP	C\$PNTX
5560	020364	062706	000006		ADD	#6,SP
5561	020370			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17	
5562	020370	013746	002276		MOV	LUR17,-(SP)
5563	020374	013746	002274		MOV	LUR16,-(SP)
5564	020400	013746	002272		MOV	LUR15,-(SP)
5565	020404	013746	002270		MOV	LUR14,-(SP)
5566	020410	012746	011571		MOV	#FMT6,-(SP)
5567	020414	012746	000005		MOV	#5,-(SP)
5568	020420	010600			MOV	SP,R0
5569	020422	104415			TRAP	C\$PNTX
5570	020424	062706	000014		ADD	#14,SP
5571	020430			PRINTX	#FMT4,#DH7,#DH8	
5572	020430	012746	014453		MOV	#DH8,-(SP)
5573	020434	012746	014421		MOV	#DH7,-(SP)
5574	020440	012746	011526		MOV	#FMT4,-(SP)
5575	020444	012746	000003		MOV	#3,-(SP)
5576	020450	010600			MOV	SP,R0
5577	020452	104415			TRAP	C\$PNTX
5578	020454	062706	000010		ADD	#10,SP
5579	020460			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16	
5580	020460	013746	002306		MOV	AX1.16,-(SP)
5581	020464	013746	002304		MOV	AX1.15,-(SP)
5582	020470	013746	002302		MOV	AX0.16,-(SP)
5583	020474	013746	002300		MOV	AX0.15,-(SP)
5584	020500	012746	011541		MOV	#FMT5,-(SP)
5585	020504	012746	000005		MOV	#5,-(SP)
5586	020510	010600			MOV	SP,R0
5587	020512	104415			TRAP	C\$PNTX
5588	020514	062706	000014		ADD	#14,SP
5589	020520			PRINTX	#FMT9,#DH9	
5590	020520	012746	014512		MOV	#DH9,-(SP)
5591	020524	012746	011670		MOV	#FMT9,-(SP)
5592	020530	012746	000002		MOV	#2,-(SP)
5593	020534	010600			MOV	SP,R0

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5594	020536	104415			TRAP	C\$PNTX
5595	020540	062706	000006		ADD	#6,SP
5596	020544			PRINTX		#FMT6,AX2.15,AX2.16,AX3.15,AX3.16
5597	020544	013746	002316		MOV	AX3.16,-(SP)
5598	020550	013746	002314		MOV	AX3.15,-(SP)
5599	020554	013746	002312		MOV	AX2.16,-(SP)
5600	020560	013746	002310		MOV	AX2.15,-(SP)
5601	020564	012746	011571		MOV	#FMT6,-(SP)
5602	020570	012746	000005		MOV	#5,-(SP)
5603	020574	010600			MOV	SP,R0
5604	020576	104415			TRAP	C\$PNTX
5605	020600	062706	000014		ADD	#14,SP
5606	020604			ENDMSG		
5607	020604				L10011:	
5608	020604	104423			TRAP	C\$MSG
5609						
5610						
5611						
5612						
5613						
5614	020606			BGNMSG	ERR9	
5615	020606					ERR9::
5616	020606			PRINTB	#FMT1,#ADDRES,MPCSR	
5617	020606	013746	002424		MOV	MPCSR,-(SP)
5618	020612	012746	035522		MOV	#ADDRES,-(SP)
5619	020616	012746	011432		MOV	#FMT1,-(SP)
5620	020622	012746	000003		MOV	#3,-(SP)
5621	020626	010600			MOV	SP,R0
5622	020630	104414			TRAP	C\$PNTB
5623	020632	062706	000010		ADD	#10,SP
5624	020636			PRINTB	#FMT2	
5625	020636	012746	011442		MOV	#FMT2,-(SP)
5626	020642	012746	000001		MOV	#1,-(SP)
5627	020646	010600			MOV	SP,R0
5628	020650	104414			TRAP	C\$PNTB
5629	020652	062706	000004		ADD	#4,SP
5630	020656			PRINTB	#FMT7,#DH1,REGNUM	
5631	020656	013746	002356		MOV	REGNUM,-(SP)
5632	020662	012746	014242		MOV	#DH1,-(SP)
5633	020666	012746	011624		MOV	#FMT7,-(SP)
5634	020672	012746	000003		MOV	#3,-(SP)
5635	020676	010600			MOV	SP,R0
5636	020700	104414			TRAP	C\$PNTB
5637	020702	062706	000010		ADD	#10,SP
5638	020706			PRINTX	#FMT4,#DH2,#DH3	
5639	020706	012746	014313		MOV	#DH3,-(SP)
5640	020712	012746	014264		MOV	#DH2,-(SP)
5641	020716	012746	011526		MOV	#FMT4,-(SP)
5642	020722	012746	000003		MOV	#3,-(SP)
5643	020726	010600			MOV	SP,R0
5644	020730	104415			TRAP	C\$PNTX
5645	020732	062706	000010		ADD	#10,SP
5646	020736			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13	
5647	020736	013746	002266		MOV	LUR13,-(SP)
5648	020742	013746	002264		MOV	LUR12,-(SP)
5649	020746	013746	002262		MOV	LUR11,-(SP)

CZDMRF.P11 03-NOV-81 10:29

GLOBAL ERROR REPORT SECTION

5650 020752 013746 002260
 5651 020756 012746 011541
 5652 020762 012746 000005
 5653 020766 010600
 5654 020770 104415
 5655 020772 062706 000014
 5656 020776
 5657 020776 012746 014351
 5658 021002 012746 011670
 5659 021006 012746 000002
 5660 021012 010600
 5661 021014 104415
 5662 021016 062706 000006
 5663 021022
 5664 021022 013746 002276
 5665 021026 013746 002274
 5666 021032 013746 002272
 5667 021036 013746 002270
 5668 021042 012746 011571
 5669 021046 012746 000005
 5670 021052 010600
 5671 021054 104415
 5672 021056 062706 000014
 5673 021062
 5674 021062
 5675 021062 104423
 5676
 5677
 5678
 5679
 5680

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

ENDMSG

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

CZDMRF.P11 03-NOV-81 10:29

LOAD DEVICE PROTECTION TABLE

.SBTTL LOAD DEVICE PROTECTION TABLE

```

:////////////////////
:/ THIS TABLE IDENTIFIES THE LOAD DEVICE TO THE SUPERVISOR, SO THAT IT CAN BE
:/ PROTECTED FROM TESTING, IF DESIRED.
:////////////////////

```

```

5700
5701
5702
5703
5704
5705
5706
5707 021066
5708 021066
5709 021066 177777
5710 021070 177777
5711 021072 177777
5712 021074
5713
5714
5715
5716
5717

```

BGNPROT

```

.WORD -1
.WORD -1
.WORD -1
ENDPROT

```

```

:DON'T CHK CSR ADRS
:DON'T CHK MASSBUS UNIT NO.
:DON'T CHK DRIVE NO.

```

L\$PROT::

CZDMRF.P11 03-NOV-81 10:29

INITIALIZE SECTION

.SBTTL INITIALIZE SECTION

:/ THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
:/ AT THE BEGINNING OF THE TEST SEQUENCE ON THE NEXT UNIT.

5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773

021074
021074

021074

021100

021104

021110

021114

021120

021124

021130

021132

021140

021146

021150

021156

021164

021172

021172

021176

021200

021200

021202

021202

021206

021210

021210

021212

021212

021216

021220

021220

021222

021222

021226

021230

021230

021232

021234

021234

021240

021244

021244

021252

010637 002324
005037 002330
005037 002404
005037 002406
005037 002376
005037 002410
005737 002370
001007
013737 000004 002372
013737 000006 002374
000406
013737 002372 000004 6\$:
013737 002374 000006
012737 000001 002370 9\$:
021172
012700 000040
104447
103415
021202
012700 000037
104447
021210
103411
021212
012700 000035
104447
021220
103411
021222
012700 000036
104447
021230
103467
021232 000414
005037 002422
005037 002412
021244
012737 177777 002322
005237 002422

BGNINIT

LS\$INIT::

MOV SP,PSTACK ;SAVE BASE-LEVEL STACK POINTER
CLR SUBRPC ;CLEAR SUBR CALL PC
CLR DISILO ;CLEAR CURRENT STATE OF DISSI
CLR CHPTYP ;CLEAR USYRT CHIP TYPE INDICATOR
CLR ERROR1 ;CLEAR ERROR FLAG
CLR SAVLEN ;CLEAR CHAR LENGTH FROM SETUP
TST FRSTIM ;SEE IF FIRST TIME THROUGH AFTER LOAD
BNE 6\$;BR IF NOT
MOV @#4,SAVE4 ;SAVE ERROR TRAP VECTOR
MOV @#6,SAVE6
BR 9\$
6\$: MOV SAVE4,@#4 ;RESTORE ERROR TRAP VECTOR
MOV SAVE6,@#6
9\$: MOV #1,FRSTIM ;MARK FLAG FOR NEXT TIME THROUGH
;SEE IF PROGRAM JUST STARTED, BR IF YES
READEF #EF.START
BCOMplete STARST
;SEE IF PROGRAM JUST RESTARTED, BR IF YES
READEF #EF.RESTART
BCOMplete STARST
;SEE IF THIS IS A NEW PASS, BR IF YES
READEF #EF.NEW
BCOMplete NEWST
;SEE IF PROGRAM WAS JUST CONTINUED
READEF #EF.CONTINUE
BCOMplete ENDIT
STARST: BR GETPRM
;CLEAR DEVICE MAP
CLR STARES ;CLEAR FLAG TO SHOW JUST HAD STA OR RES
CLR DEVMAP
NEWST: MOV #-1,LOGDEV ;RESET LOGICAL DEVICE TO -1
INC STARES ;INCR NO. OF PASSES SINCE STA OR RES

MOV #EF.START,RO
TRAP C\$REFG
BCS STARST
MOV #EF.RESTART,RO
TRAP C\$REFG
BCS STARST
MOV #EF.NEW,RO
TRAP C\$REFG
BCS NEWST
MOV #EF.CONTINUE,RO
TRAP C\$REFG
BCS ENDIT

CZDMRF.P11 03-NOV-81 10:29

INITIALIZE SECTION

```

5774 021256 012737 000001 002414      MOV    #BIT0,DEVPTR      ;INIT DEVICE MAP BIT POINTER
5775                                     ; GET UNIBUS ADDRESS, VECTOR, PRIORITY LEVEL, SWITCH PACKS, TEST
5776                                     ; CONNECTOR INFORMATION FOR THIS LOGICAL DEVICE
5777 021264                                     GETPRM:
5778 021264 005237 002322      INC    LOGDEV           ;INCREMENT LOGICAL DEVICE NUMBER
5779 021270 023737 002322 002012      CMP    LOGDEV,LSUNIT    ;SEE IF MAXIMUM UNIT NO. EXCEEDED
5780 021276 002362                                     BGE    NEWST           ;BR IF YES
5781 021300                                     GPHARD LOGDEV,R1      ;GET P-TABLE POINTER INTO R1
5782 021300 013700 002322                                     MOV    LOGDEV,R0
5783 021304 104442                                     TRAP   CS$GPHRD
5784 021306 010001                                     MOV    R0,R1
5785 021310      BCOMPLETE      10$      ;BR IF DEVICE AVAILABLE
5786 021310 103403                                     BCS   10$
5787 021312 006337 002414      ASL    DEVPTR          ;SHIFT DEVICE POINTER
5788 021316 000762      BR     GETPRM          ;SKIP THIS DEVICE
5789 021320 053737 002414 002412 10$:  BIS    DEVPTR,DEVMAP    ;SET BIT FOR THIS DEVICE
5790 021326 006337 002414      ASL    DEVPTR          ;SHIFT BIT POINTER
5791 021332 011137 002424      MOV    (R1),MPCSR      ;STORE POINTER TO MICROPROCESSOR CSR'S
5792 021336 011137 002426      MOV    (R1),BSEL1
5793 021342 005237 002426      INC    BSEL1           ;GET POINTER TO BSEL1 (MAINTENANCE REGISTER)
5794 021346 013737 002426 002430      MOV    BSEL1,BSEL2
5795 021354 005237 002430      INC    BSEL2           ;GET POINTER TO BSEL2
5796 021360 011137 002432      MOV    (R1),SEL4
5797 021364 062737 000004 002432      ADD    #4,SEL4         ;GET POINTER TO SEL4
5798 021372 012137 002434      MOV    (R1)+,SEL6
5799 021376 062737 000006 002434      ADD    #6,SEL6         ;STORE POINTER TO SEL6
5800 021404 011137 002454      MOV    (R1),RUNINH     ;GET STATE OF MICROPROCESSOR RUN SWITCH
5801 021410      ENDIT:
5802 021410      ENDINIT
5803 021410
5804 021410 104411      L10015:  TRAP   CS$INIT
5805
5806
5807
5808
5809
5810

```

CZDMRF.P11 03-NOV-81 10:29

AUTO DROP UNIT SECTION

.SBTTL AUTO DROP UNIT SECTION

:/ THE AUTO DROP CODING DETERMINES WHETHER OR NOT THE DEVICE WHOSE P-TABLE WAS JUST OBTAINED IS READY FOR TESTING, AND IT IS DROPPED IF NOT READY.

BGNAUTO

LSAUTO::

:ESTABLISH CPU PRIORITY = 7
SETPRI #PRI07

MOV #PRI07,R0
TRAP C\$SPRI

:SET UP NON-EXISTENT MEMORY ERROR TRAP VECTOR

MOV #6\$,@#4
MOV #PRI07,@#6
TST @MPCSR
BR 9\$

:ADDRESS SELO
:TAKE THIS BR IF DEVICE RESPONDS

:COME HERE IF DEVICE CSR IS NON-EXISTENT

6\$: ADD #4,SP
DODU LOGDEV

:CLEAN UP THE STACK POINTER
:DROP THIS UNIT FROM TESTING

MOV LOGDEV,R0
TRAP C\$DODU

9\$: MOV SAVE4,@#4
MOV SAVE6,@#6
ENDAUTO

:RESTORE ERROR TRAP VECTOR

L10016:

TRAP C\$AUTO

5811
5812
5813
5814
5815
5816
5817 021412
5818 021412
5819
5820 021412
5821 021412 012700 000340
5822 021416 104441
5823 021420 012737 021442 000004
5824 021426 012737 000340 000006
5825 021434 005777 160764
5826 021440 000405
5827
5828 021442 062706 000004
5829 021446
5830 021446 013700 002322
5831 021452 104451
5832 021454 013737 002372 000004
5833 021462 013737 002374 000006
5834 021470
5835 021470
5836 021470 104461
5837
5838
5839
5840
5841

CLEANUP CODING SECTION

.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.
:////////////////////

```

5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860

021472
021472

021472
021472
021472 104412

BGNCLN

ENDCLN

L\$CLEAN::

L10017: TRAP C\$CLEAN

CZDMRF.P11 03-NOV-81 10:29

DROP UNIT SECTION

5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895

021474
021474
021474 104433
021476 013746 002322
021502 012746 021524
021506 012746 000002
021512 010600
021514 104417
021516 062706 000006
021522
021522 104453
021524 047045 040445 047125
021532 052111 022440 031104
021540 040445 042040 047522
021546 050120 042105 047045
021554 000
021556

.SBTTL DROP UNIT SECTION
:////
:/ THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
:/ TO NO LONGER BE TESTED.
:////

BGNDU
:ISSUE UNIBUS RESET TO CLEAN UP
BRESET
:PRINT 'UNIT XX DROPPED'
PRINTF #FMT27,LOGDEV

L\$DU::

TRAP C\$RESET
MOV LOGDEV,-(SP)
MOV #FMT27,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PRINTF
ADD #6,SP

ENDDU

L10020:

TRAP C\$DU

FMT27: .ASCIZ /%N%AUNIT %D2%A DROPPED%N/

.EVEN

CZDMRF.P11 03-NOV-81 10:29

ADD UNIT SECTION

5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914

021556
021556
021556
021556
021556 104452

.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.

BGNAU
ENDAU

LSAU::
L10021: TRAP CSAU

CZDMRF.P11 03-NOV-81 10:29

TEST 2 - INBUS/OUTBUS REG 14 INITIALIZATION TEST

5971 021702 013737 002342 002364
 5972 021710 004737 003774
 5973
 5974 021714
 5975 021714 104455
 5976 021716 000002
 5977 021720 012137
 5978 021722 014610
 5979 021724
 5980 021724
 5981 021724
 5982 021724 104401
 5983
 5984
 5985
 5986
 5987
 5988
 5989
 5990
 5991
 5992
 5993
 5994
 5995
 5996
 5997 021726
 5998 021726
 5999 021726 004737 003554
 6000 021732 012737 000014 002356
 6001 021740 012701 002547
 6002 021744
 6003 021744
 6004 021744 104404
 6005 021746 111137 002344
 6006 021752 143737 002536 002344
 6007 021760 004737 003726
 6008 021764 004737 003650
 6009 021770 143737 002536 002342
 6010 021776 123737 002342 002344
 6011 022004 001414
 6012 022006 013737 002344 002362
 6013 022014 013737 002342 002364
 6014 022022 004737 003774
 6015
 6016 022026
 6017 022026 104455
 6018 022030 000003
 6019 022032 012176
 6020 022034 014610
 6021 022036
 6022 022036
 6023 022036
 6024 022036 104405
 6025 022040 005201
 6026 022042 020127 002573

```

MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
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

L10023:
TRAP C\$ETST

```

:*****
.SBTTL 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.
:*****
BGNTST

```

T3::

```

JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #14,REGNUM ;SET LU REG NO. = 14
MOV #PATA,R1 ;GET POINTER TO DATA PAT IN R1

```

3\$:

BGNSEG

TRAP C\$BSEG

```

MOVB (R1),WRIBYT ;GET A BYTE OF PAT A
BICB R14NRW,WRIBYT ;MASK OFF NON-READ/WRITE BITS
JSR PC,WRITLU ;WRITE DATA BYTE INTO REG 14
JSR PC,READLU ;READ DATA BYTE FROM REG 14
BICB R14NRW,REDBYT ;MASK OFF NON-READ/WRITE BITS
CMPB REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
BEQ 6$ ;BR IF BYTES MATCH
MOV WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
JSR PC,GETREG ;GET REGS FOR PRINTOUT

```

```

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

```

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

6\$:

ENDSEG

10000\$:

```

INC R1 ;INCREMENT DATA PATTERN POINTER
CMP R1,#PATB ;SEE IF ALL WORDS OF PATTERN A USED YET

```

TRAP C\$ESEG

CZDMRF.P11 03-NOV-81 10:29

TEST 3 - INBUS/OUTBUS REG 14 READ/WRITE BIT TEST

6027 022046 103736
6028 022050
6029 022050
6030 022050 104401

ENDTST BLO 3\$;BR IF NOT DONE YET

L10024: TRAP C\$ETST

6031
6032
6033
6034
6035
6036
6037
6038
6039

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

6040

BGNTST

T4::

6041 022052
6042 022052
6043 022052 004737 003554
6044 022056 012737 000014 002356
6045 022064 112737 000377 002344
6046 022072 004737 003726
6047 022076 004737 003554
6048 022102 004737 003650
6049 022106 123737 002342 003004
6050 022114 001416
6051 022116 005037 002362
6052 022122 113737 003004 002362
6053 022130 013737 002342 002364
6054 022136 004737 003774

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

6055
6056 022142
6057 022142 104455
6058 022144 000002
6059 022146 012137
6060 022150 014610

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

6061 022152
6062 022152
6063 022152

6\$:
ENDTST

L10025: TRAP C\$ETST

6064 022152 104401
6065
6066
6067
6068
6069

6070
6071
6072
6073
6074

: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.

6075

BGNTST

T5::

6076 022154
6077 022154 004737 003554
6078 022160 012737 000014 002356
6079 022166 112737 000377 002344
6080 022174 004737 003726
6081 022200
6082 022200 104433

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)

TRAP C\$RESET

CZDMRF.P11 03-NOV-81 10:29

TEST 5 - REG 14 UNIBUS RESET (INIT) TEST

6083 022202 142777 000200 160216
 6084 022210 012701 000024
 6085 022214 000240
 6086 022216 005301
 6087 022220 001375
 6088 022222 004737 003650
 6089 022226 142737 000100 002342
 6090 022234 123737 002342 003004
 6091 022242 001416
 6092 022244 005037 002362
 6093 022250 113737 003004 002362
 6094 022256 013737 002342 002364
 6095 022264 004737 003774
 6096
 6097 022270
 6098 022270 104455
 6099 022272 000004
 6100 022274 012215
 6101 022276 014610
 6102 022300
 6103 022300
 6104 022300
 6105 022300 104401
 6106
 6107
 6108
 6109
 6110
 6111
 6112
 6113
 6114
 6115
 6116
 6117
 6118
 6119
 6120
 6121 022302
 6122 022302
 6123 022302 004737 003554
 6124 022306 012737 000014 002356
 6125 022314 013701 002356
 6126 022320 052701 000100
 6127 022324 052701 121000
 6128 022330 010137 022346
 6129 022334 112777 000041 160070
 6130 022342 004737 003516
 6131 022346 000000
 6132 022350 004737 003650
 6133 022354 123737 002342 003004
 6134 022362 001416
 6135 022364 005037 002362
 6136 022370 113737 003004 002362
 6137 022376 013737 002342 002364
 6138 022404 004737 003774

```

BICB #RUN,@BSEL1 ;CLEAR RUN BIT
MOV #20.,R1 ;INIT LOOP COUNTER
2$: NOP
DEC R1 ;DECR COUNTER
BNE 2$ ;BR TO STALL
JSR PC,READLU ;READ REG 14
BICB #TXEN,REDBYT ;CLEAR UNPREDICTABLE BIT
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 UNIBUS RESET (INIT)
ERRDF 4,EM4,ERR2

```

```

TRAP C$ERDF
.WORD 4
.WORD EM4
.WORD ERR2

```

6\$: ENDTST

```

L10026: TRAP C$SETST

```

```

:*****
: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.
:*****
BGNTST

```

```

T6::
JSR PC,MSTCLR ;ISSUE A MASTER CLEAR
MOV #14,REGNUM ;SET LU REG NO. = 14
MOV REGNUM,R1 ;SET DESTINATION = OBUS* REG 14
BIS #100,R1 ;SET SOURCE = BSEL4
BIS #MVIXOX,R1 ;SET REST OF MOVE INSTRUCTION
MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
MOVB #041,@BSEL4 ;SET DATA BYTE = 041
JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION
.WORD 0 ;INSTRUCTION GOES HERE
JSR PC,READLU ;READ LU REG 14
CMPB REDBYT,PATM+4 ;CHECK FOR LU REG 14 UNCHANGED
BEQ 4$ ;BR IF LU REG 14 UNCHANGED
CLR GOODAT ;SET EXPECTED DATA
MOVB PATM+4,GOODAT
MOV REDBYT,BADDAT ;SET ACTUAL DATA
JSR PC,GETREG ;GET REGS FOR PRINTOUT

```


CZDMRF.P11 03-NOV-81 10:29

TEST 6 - LINE UNIT FALSE SELECTION TEST

:REPORT REGISTER MISCOMPARE
ERRDF 3,EM3,ERR2

TRAP CSERDF
.WORD 3
.WORD EM3
.WORD ERR2

6139
6140 022410
6141 022410 104455
6142 022412 000003
6143 022414 012176
6144 022416 014610
6145 022420
6146 022420
6147 022420
6148 022420 104401
6149
6150
6151
6152
6153
6154

4\$:
ENDTST

L10027:
TRAP CSETST

: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

6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165 022422
6166 022422
6167 022422 004737 003554
6168 022426 012737 000010 002356
6169 022434 012701 002622
6170 022440 112137 002344
6171 022444 004737 003726
6172 022450 005237 002356
6173 022454 020127 002632
6174 022460 103767
6175 022462 004737 003554
6176 022466 012737 000010 002356
6177 022474 012702 003000
6178 022500 012701 002526
6179 022504
6180 022504 004737 003650
6181 022510 142137 002342
6182 022514 123712 002342
6183 022520 001417
6184 022522 005037 002362
6185 022526 111237 002362
6186 022532 013737 002342 002364
6187 022540 004737 003774
6188
6189 022544
6190 022544 104455
6191 022546 000002
6192 022550 012137
6193 022552 014610
6194 022554

BGNTST

T7::

JSR PC,MSTCLR :ISSUE MASTER CLEAR
MOV #10,REGNUM :INIT REG NO. TO 10
MOV #PATG,R1 :INIT DATA PATTERN POINTER
2\$: MOVB (R1)+,WRIBYT :SET DATA PATTERN BYTE TO BE WRITTEN
JSR PC,WRITLU :WRITE BYTE INTO REG
INC REGNUM :INCREMENT REG NO. FOR WRITING
CMP R1,#PATH :SEE IF ALL BYTES WRITTEN YET
BLO 2\$:BR IF NOT DONE WRITING YET
JSR PC,MSTCLR :ISSUE MASTER CLEAR
MOV #10,REGNUM :INIT LU REG NO. TO 10
MOV #PATM,R2 :INIT DATA PATTERN POINTER
MOV #UPBITS,R1 :INIT POINTER TO UNPREDICTABLE BITS
3\$: JSR PC,READLU :READ A LINE UNIT REG
BICB (R1)+,REDBYT :MASK OUT UNPREDICTABLE BITS FOR THIS REG
CMPB REDBYT,(R2) :COMPARE MASKED DATA TO EXPECTED
BEQ 6\$:BR IF DATA READ IS OK
CLR GOODAT
MOVB (R2),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 CSERDF
.WORD 2
.WORD EM2
.WORD ERR2

ESCAPE TST

CZDMRF.P11 03-NOV-81 10:29

TEST 7 - INBUS REG MASTER CLEAR TEST

TRAP C\$ESCAPE
.WORD L10030-

6195 022554 104410
6196 022556 000016
6197 022560 005237 002356
6198 022564 005202
6199 022566 020227 003010
6200 022572 103744
6201 022574
6202 022574
6203 022574 104401
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218 022576
6219 022576
6220 022576 004737 003554
6221 022602 012701 002456
6222 022606 012702 003000
6223 022612 012703 000010
6224 022616 112221
6225 022620 005303
6226 022622 001375
6227 022624 005001
6228 022626 010137 002356
6229 022632 062737 000010 002356
6230 022640 116137 002573 002344
6231 022646 113761 002344 002456
6232 022654 146161 002526 002456
6233 022662 004737 003726
6234 022666 005003
6235 022670 010337 002356
6236 022674 062737 000010 002356
6237 022702 004737 003650
6238 022706 146337 002526 002342
6239 022714 023727 002356 000011
6240 022722 001006
6241 022724 142737 000020 002342
6242 022732 142763 000020 002456
6243 022740 123763 002342 002456
6244 022746 001420
6245 022750 005037 002362
6246 022754 116337 002456 002362
6247 022762 013737 002342 002364
6248 022770 004737 003774
6249
6250 022774

6\$: INC REGNUM :INCREMENT REG NO.
INC R2 :INCR DATA PATTERN POINTER
CMP R2,#PATN :SEE IF ALL DONE YET
BLO 3\$:BR IF NOT DONE YET

ENDTST

L10030: TRAP C\$ETST

:SBTTL 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

:BGNTST

T8::

JSR PC,MSTCLR :ISSUE MASTER CLEAR
MOV #REDDAT,R1 :INIT POINTER TO EXPECTED DATA AREA
MOV #PATM,R2 :GET POINTER TO PATTERN M
MOV #8,R3 :SET COUNTER
3\$: MOV (R2)+,(R1)+ :LOAD BYTE OF PATRN INTO EXPECTED DATA AREA
DEC R3 :DECR COUNTER
BNE 3\$:BR IF NOT DONE LOADING YET
CLR R1 :INIT DATA PATTERN INDEX FOR WRITING
6\$: MOV R1,REGNUM :GET REG NO. FOR WRITING
ADD #10,REGNUM :SET DATA BYTE TO BE WRITTEN
MOVB PATB(R1),WRIBYT :SET EXPECTED DATA FOR READ
MOVB WRIBYT,REDDAT(R1) :MASK OUT UNPREDICTABLE BITS
BICB UPBITS(R1),REDDAT(R1) :WRITE DATA BYTE INTO REG
JSR PC,WRITLU :INIT DATA PAT INDEX FOR READS
CLR R3
9\$: MOV R3,REGNUM :GET REG NO. FOR READING
ADD #10,REGNUM :READ A LINE UNIT REG
JSR PC,READLU :MASK OUT UNPREDICTABLE BITS
BICB UPBITS(R3),REDBYT :SEE IF READING REG 11
CMP REGNUM,#11 :BR IF NOT
BNE 10\$:MASK ORDY BIT IN ACTUAL BYTE
BICB #ORDY,REDBYT :MASK ORDY BIT IN EXPECTED BYTE
BICB #ORDY,REDDAT(R3) :COMPARE BYTE READ TO EXPECTED
10\$: CMPB REDBYT,REDDAT(R3) :BR IF DATA MATCHES
BEQ 12\$
CLR GOODAT :SET EXPECTED DATA
MOVB REDDAT(R3),GOODAT :SET ACTUAL DATA
MOV REDBYT,BADDAT :READ AND STORE REGS 10-17 FOR PRINTOUT
JSR PC,GETREG
:REPORT REG MISCOMPARE
ERRDF 3,EM3,ERR2

CZDMRF.P11 03-NOV-81 10:29

TEST 9 - REG 11 READ/WRITE BIT TEST

6307 023144 103741
6308 023146
6309 023146
6310 023146 104401

ENDTST BLO 3\$;BR IF NOT DONE YET

L10032: TRAP C\$ETST

6311
6312
6313
6314
6315

: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.
:*****

6316
6317
6318
6319
6320
6321

BGNTST

6322 023150

6323 023150

6324 023150 004737 003554

6325 023154 012737 000012 002356

6326 023162 012701 002606

6327 023166

6328 023166

6329 023166 104404

6330 023170 111137 002344

6331 023174 004737 003726

6332 023200 004737 003650

6333 023204 143737 002530 002342

6334 023212 123737 002342 002344

6335 023220 001414

6336 023222 013737 002344 002362

6337 023230 013737 002342 002364

6338 023236 004737 003774

6339

6340 023242

6341 023242 104455

6342 023244 000003

6343 023246 012176

6344 023250 014610

6345 023252

6346 023252

6347 023252

6348 023252 104405

6349 023254 005201

6350 023256 020127 002611

6351 023262 103741

6352 023264

6353 023264

6354 023264 104401

6355

6356

6357

6358

6359

6360

6361

6362

T10::
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #PATD,R1 ;GET POINTER TO DATA PAT IN R1

3\$:

BGNSEG

MOVB (R1),WRIBYT ;GET A BYTE OF PAT D TRAP C\$BSEG
JSR PC,WRITLU ;WRITE DATA BYTE INTO REG 12
JSR PC,READLU ;READ DATA BYTE FROM REG 12
BICB UPBITS+2,REDBYT ;MASK OUT UNPREDICTABLE BITS
CMPB REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
BEQ 6\$;BR IF BYTES MATCH
MOV WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
JSR PC,GETREG ;GET REGS FOR PRINTOUT

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

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

6\$:

ENDSEG

10000\$: TRAP C\$ESEG

INC R1 ;INCREMENT DATA PATTERN POINTER
CMP R1,#PATE ;SEE IF ALL WORDS OF PATTERN D USED YET
BLO 3\$;BR IF NOT DONE YET

ENDTST

L10033: TRAP C\$ETST

:SBTTL TEST 11 - REG 13 READ/WRITE BIT TEST
:*

CZDMRF.P11 03-NOV-81 10:29

TEST 11 - REG 13 READ/WRITE BIT TEST

6363
6364
6365
6366 023266
6367 023266
6368 023266 004737 003554
6369 023272 012737 000013 002356
6370 023300 012701 002611
6371 023304
6372 023304
6373 023304 104404
6374 023306 111137 002344
6375 023312 004737 003726
6376 023316 004737 003650
6377 023322 143737 002531 002342
6378 023330 123737 002342 002344
6379 023336 001414
6380 023340 013737 002344 002362
6381 023346 013737 002342 002364
6382 023354 004737 003774
6383
6384 023360
6385 023360 104455
6386 023362 000003
6387 023364 012176
6388 023366 014610
6389 023370
6390 023370
6391 023370
6392 023370 104405
6393 023372 005201
6394 023374 020127 002617
6395 023400 103741
6396 023402
6397 023402
6398 023402 104401
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410 023404
6411 023404
6412 023404 004737 003554
6413 023410 012737 000017 002356
6414 023416 012701 002617
6415 023422
6416 023422
6417 023422 104404
6418 023424 111137 002344

```

;* WRITE, READ, AND COMPARE EACH WORD OF DATA PATTERN E INTO REG 13 :
;* DATA PATTERN E = 000,120,020,100,120,000.
*****
BGNTST
T11::
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #13,REGNUM ;SET LU REG NO. = 13
MOV #PATE,R1 ;GET POINTER TO DATA PAT IN R1
3$:
BGNSEG
TRAP CSBSEG
MOVB (R1),WRIBYT ;GET A BYTE OF PAT E
JSR PC,WRITLU ;WRITE DATA BYTE INTO REG 13
JSR PC,READLU ;READ DATA BYTE FROM REG 13
BICB UPBITS+3,REDBYT ;MASK OUT UNPREDICTABLE BITS
CMPB REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
BEQ 6$ ;BR IF BYTES MATCH
MOV WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
JSR PC,GETREG ;GET REGS FOR PRINTOUT
;REPORT LINE UNIT REG MISCOMPARE
ERRDF 3,EM3,ERR2
TRAP CSERDF
.WORD 3
.WORD EM3
.WORD ERR2
6$:
ENDSEG
10000$:
TRAP CSESEG
INC R1 ;INCREMENT DATA PATTERN POINTER
CMP R1,#PATF ;SEE IF ALL WORDS OF PATTERN E USED YET
BLO 3$ ;BR IF NOT DONE YET
ENDTST
L10034:
TRAP CSETST

```

```

*****
.SBTTL 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.
*****
BGNTST

```

6410 023404
6411 023404
6412 023404 004737 003554
6413 023410 012737 000017 002356
6414 023416 012701 002617
6415 023422
6416 023422
6417 023422 104404
6418 023424 111137 002344

```

T12::
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #17,REGNUM ;SET LU REG NO. = 17
MOV #PATF,R1 ;GET POINTER TO DATA PAT IN R1
3$:
BGNSEG
TRAP CSBSEG
MOVB (R1),WRIBYT ;GET A BYTE OF PAT F

```


CZDMRF.P11 03-NOV-81 10:29

TEST 12 - REG 17 READ/WRITE BIT TEST

6419 023430 004737 003726
6420 023434 004737 003650
6421 023440 143737 002535 002342
6422 023446 123737 002342 002344
6423 023454 001414
6424 023456 013737 002344 002362
6425 023464 013737 002342 002364
6426 023472 004737 003774
6427
6428 023476
6429 02347E 104455
6430 023500 000003
6431 023502 012176
6432 023504 014610
6433 023506
6434 023506
6435 023506
6436 023506 104405
6437 023510 005201
6438 023512 020127 002622
6439 023516 103741
6440 023520
6441 023520
6442 023520 104401
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469 023522
6470 023522
6471 023522 004737 003554
6472 023526 012737 000015 002420
6473 023534 005737 002454
6474 023540 001020

```

JSR PC,WRITLU ;WRITE DATA BYTE INTO REG 17
JSR PC,READLU ;READ DATA BYTE FROM REG 17
BICB UPBITS+7,REDBYT ;MASK OUT UNPREDICTABLE BITS
CMPB REDBYT,WRIBYT ;COMPARE BYTE READ TO BYTE WRITTEN
BEQ 6$ ;BR IF BYTES MATCH
MOV WRIBYT,GOODAT ;SET EXPECTED REG CONTENTS
MOV REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
JSR PC,GETREG ;GET REGS FOR PRINTOUT
:REPORT LINE UNIT REG MISCOMPARE
ERRDF 3,EM3,ERR2
TRAP C$ERDF
.WORD 3
.WORD EM3
.WORD ERR2
6$:
ENDSEG
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
ENDTST
L10035:
TRAP C$ETST

```

```

:*****
: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 M8207 RUN SWITCH (E28 SW7)
:* IS OFF, THE TEST WILL BE SKIPPED.
:*****
BGNTST

```

```

T13::
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

```


CZDMRF .P11 03-NOV-81 10:29

TEST 13 - MAINTENANCE CLOCK BIT TEST

```

6475 023542 023727 002422 000001      CMP      STARES,#1      ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
6476 023550 001012                      BNE      40$            ;BR IF NOT, TO SKIP PRINTING
6477 023552                      PRINTF   #FMT19,TSTNUM  ;PRINT MSG TO SAY TEST NOT RUN
6478 023552 013746 002420                      MOV      TSTNUM,-(SP)
6479 023556 012746 011765                      MOV      #FMT19,-(SP)
6480 023562 012746 000002                      MOV      #2,-(SP)
6481 023566 010600                      MOV      SP,R0
6482 023570 104417                      TRAP    C$PNTF
6483 023572 062706 000006                      ADD      #6,SP
6484 023576 000137 024144
6485 023602 012737 000017 002356 40$:  JMP      A1            ;GO TO SKIP TEST
6486 023610 012701 000360 1$:  MOV      #17,REGNUM   ;SET REG NO. = 17
6487 023614 052701 000002      MOV      #360,R1      ;SET INSTRUCTION SOURCE = INBUS REG 17
6488 023620 052701 021000      BIS      #2,R1        ;SET INSTRUCTION DESTINATION = BSEL2
6489 023624 010137 023634      BIS      #MVIOX,R1    ;SET REST OF MOVE INSTRUCTION
6490 023630 004737 004052      MOV      R1,2$        ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
6491 023634 000000 2$:  JSR      PC,LOOPIN    ;GET MICROPROCESSOR LOOPING ON MOVE INSTRUCTION
6492                      .WORD    0            ;INSTRUCTION GOES HERE
6493
6494 :-----:
6495 : WAIT FOR MCLK BIT TO BE SET TO 1
6496 :-----:
6496 023636 005037 002466
6497 023642 117737 156562 002342 3$:  CLR      REG0          ;INIT PROGRAM TIMER
6498 023650 132737 000002 002342  MOVB     @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
6499 023656 001031      BITB     #MCLK,REDBYT ;SEE IF MCLK BIT = 1 YET
6500 023660 005237 002466      BNE      6$           ;BR IF MCLK = 1
6501 023664 001366      INC      REG0          ;INCREMENT TIMER
6502                      BNE      3$           ;BR IF PROGRAM TIMER DID NOT TIME OUT YET
6503 023666 012737 000002 002362      (TIME OUT = SEVERAL HUNDRED MILLI-SEC)
6504 023674 013737 002342 002364      MOV      #MCLK,GOODAT ;SET EXPECTED REG CONTENTS
6505 023702 004737 003774      MOV      REDBYT,BADAT ;SET ACTUAL REG CONTENTS
6506                      JSR      PC,GETREG  ;GET REGS FOR PRINTOUT
6507 :REPORT MCLK BIT STUCK AT 0
6508 023706 104455      ERRDF   5,EM5,ERR2
6509 023710 000005      TRAP    C$ERDF
6510 023712 012270      .WORD   5
6511 023714 014610      .WORD   EM5
6512                      .WORD   ERR2
6513 :TYPE 'PLEASE INSURE M8207 RUN SWITCH (E28 SW7) IS ON'
6514 023716 012746 012016      PRINTF  #FMT24
6515 023722 012746 000001      MOV      #FMT24,-(SP)
6516 023726 010600      MOV      #1,-(SP)
6517 023730 104417      MOV      SP,R0
6518 023732 062706 000004      TRAP    C$PNTF
6519 023736 000137 024144      ADD      #4,SP
6520                      JMP      A1            ;ESCAPE TO END OF TEST
6521
6522 :-----:
6523 : WAIT FOR MCLK BIT TO BE CLEARED TO 0
6524 :-----:
6524 023742
6525 023742 005037 002466 6$:  CLR      REG0          ;INIT PROGRAM TIMER
6526 023746 117737 156456 002342 8$:  MOVB     @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
6527 023754 132737 000002 002342  BITB     #MCLK,REDBYT ;SEE IF MCLK BIT = 0 YET
6528 023762 001430      BEQ      10$          ;BR IF MCLK = 0
6529 023764 005237 002466      INC      REG0          ;INCREMENT TIMER
6530 023770 001366      BNE      8$           ;BR IF TIMER DID NOT TIME OUT YET

```

CZDMRF.P11 03-NOV-81 10:29

TEST 13 - MAINTENANCE CLOCK BIT TEST

```

6531 023772 005037 002362          CLR    GOODAT      ;SET EXPECTED REG CONTENTS
6532 023776 013737 002342 002364  MOV    REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
6533 024004 004737 003774          JSR    PC,GETREG    ;GET REGS FOR PRINTOUT
6534                                     ;REPORT MCLK BIT STUCK AT 1
6535 024010                                     ERRDF  6,EM6,ERR2
6536 024010 104455                                     TRAP  C$ERDF
6537 024012 000006                                     .WORD 6
6538 024014 012321                                     .WORD EM6
6539 024016 014610                                     .WORD ERR2
6540                                     ;TYPE 'PLEASE INSURE M8207 RUN SWITCH (E28 SW7) IS ON'
6541 024020                                     PRINTF #FMT24
6542 024020 012746 012016          MOV    #FMT24,-(SP)
6543 024024 012746 000001          MOV    #1,-(SP)
6544 024030 010600          MOV    SP,R0
6545 024032 104417          TRAP  C$PNTF
6546 024034 062706 000004          ADD   #4,SP
6547 024040 000137 024144          JMP   A1          ;ESCAPE TO END OF TEST
6548
6549                                     -----
6550                                     ; WAIT FOR MCLK BIT TO BE SET TO 1 AGAIN
6551                                     -----
6552 024044 10$:
6553 024044 005037 002466          CLR    REGO        ;INIT PROGRAM TIMER
6554 024050 117737 156354 002342 12$: MOVB  @BSEL2,REDBYT ;GET REG 17 INTO REDBYT
6555 024056 132737 000002 002342  BITB  #MCLK,REDBYT ;SEE IF MCLK BIT = 1 YET
6556 024064 001027          BNE   A1          ;BE IF MCLK = 1
6557 024066 005237 002466          INC   REGO        ;INCREMENT TIMER
6558 024072 001366          BNE   12$        ;BR IF TIMER DID NOT TIME OUT YET
6559 024074 012737 000002 002362  MOV    #MCLK,GOODAT ;SET EXPECTED REG CONTENTS
6560 024102 013737 002342 002364  MOV    REDBYT,BADDAT ;SET ACTUAL REG CONTENTS
6561 024110 004737 003774          JSR    PC,GETREG    ;GET REGS FOR PRINTOUT
6562                                     ;REPORT MCLK BIT STUCK AT 0
6563 024114                                     ERRDF  5,EM5,ERR2
6564 024114 104455                                     TRAP  C$ERDF
6565 024116 000005                                     .WORD 5
6566 024120 012270                                     .WORD EM5
6567 024122 014610                                     .WORD ERR2
6568                                     ;TYPE 'PLEASE INSURE M8207 RUN SWITCH (E28 SW7) IS ON'
6569 024124                                     PRINTF #FMT24
6570 024124 012746 012016          MOV    #FMT24,-(SP)
6571 024130 012746 000001          MOV    #1,-(SP)
6572 024134 010600          MOV    SP,R0
6573 024136 104417          TRAP  C$PNTF
6574 024140 062706 000004          ADD   #4,SP
6575 024144          A1:
6576 024144 004737 003554          JSR   PC,MSTCLR    ;ISSUE MASTER CLEAR
6577 024150          ENDTST
6578 024150          L10036:
6579 024150 104401          TRAP  C$ETST
6580
6581
6582
6583
6584
6585                                     ;*****
6586 .SBTTL      TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST

```


CZDMRF.P11 03-NOV-81 10:29

TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST

```

6587
6588
6589
6590
6591
6592
6593
6594
6595 024152
6596 024152
6597 024152 004737 003554
6598 024156 005037 002360
6599 024162 012701 002632
6600 024166 112137 002352
6601 024172 112137 002354
6602 024176 004737 004270
6603 024202 032737 000002 002376
6604 024210 001413
6605 024212 012737 000014 002356
6606 024220 004737 003774
6607
6608 024224
6609 024224 104455
6610 024226 000064
6611 024230 014063
6612 024232 020606
6613 024234
6614 024234 104410
6615 024236 000244
6616 024240 062737 000002 002360
6617 024246 020127 002642
6618 024252 103745
6619 024254 004737 003554
6620 024260 005037 002360
6621 024264 012701 002642
6622 024270 004737 004102
6623 024274 032737 000001 002376
6624 024302 001413
6625 024304 012737 000014 002356
6626 024312 004737 003774
6627
6628 024316
6629 024316 104455
6630 024320 000065
6631 024322 014124
6632 024324 020606
6633 024326
6634 024326 104410
6635 024330 000152
6636 024332 023727 002360 000006
6637 024340 001003
6638 024342 142737 000372 002346
6639 024350 123711 002346
6640 024354 001417
6641 024356 005037 002362
6642 024362 111137 002362

```

```

:*
:* 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
:*****
BGNTST

```

```

T14::
JSR PC,MSTCLR ;ISSUE AN INITIAL MASTER CLEAR
CLR AXNUM ;INIT AX REG BYTE NO. TO 0
MOV #PATH,R1 ;INIT DATA PATTERN POINTER
1$: MOV (R1)+,WAX15 ;SET BITS TO LOAD INTO LO BYTE
MOVB (R1)+,WAX16 ;SET BITS TO LOAD INTO HI BYTE
JSR PC,WRITAX ;WRITE EXTENDED REG
BIT #WRDYTO,ERROR1 ;SEE IF READY FAILED TO SET
BEQ 8$ ;BR IF NOT
MOV #14,REGNUM ;SET LU REG NO = 14
JSR PC,GETREG ;GET REGS FOR PRINTOUT
;REPORT READY NOT SET AFTER AX REG WRITE
ERRDF 52,EM52,ERR9

```

```

TRAP C$ERDF
.WORD 52
.WORD EM52
.WORD ERR9

```

```

ESCAPE TST
TRAP C$ESCAPE
.WORD L10037-.

```

```

8$: ADD #2,AXNUM ;INCR REG BYTE NO.
CMP R1,#PATI ;SEE IF ALL REGS WRITTEN YET
BLO 1$ ;BR IF NOT DONE WRITING YET
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
CLR AXNUM ;INIT EXTENDED REG BYTE NO. TO 0
MOV #PATI,R1 ;INIT DATA PAT POINTER FOR READS
2$: JSR PC,READAX ;READ AN EXTENDED REG
BIT #RRDYTO,ERROR1 ;SEE IF READY FAILED TO SET
BEQ 10$ ;BR IF NOT
MOV #14,REGNUM ;SET LU REG NO. = 14
JSR PC,GETREG ;GET REGS FOR PRINTOUT
;REPORT READY NOT SET AFTER AX REG READ
ERRDF 53,EM53,ERR9

```

```

TRAP C$ERDF
.WORD 53
.WORD EM53
.WORD ERR9

```

```

ESCAPE TST
TRAP C$ESCAPE
.WORD L10037-.

```

```

10$: CMP AXNUM,#6 ;SEE IF AX3-15
BNE 3$ ;BR IF NOT
BICB #AX315U,RAX15 ;MASK OFF UNPREDICTABLE BITS
3$: CMPB RAX15,(R1) ;COMPARE LO BYTE TO EXPECTED VALUE
BEQ 4$ ;BR IF DATA MATCHES
CLR GOODAT
MOVB (R1),GOODAT ;GET EXPECTED DATA BYTE

```


CZDMRF.P11 03-NOV-81 10:29

TEST 14 - EXTENDED REGISTER MASTER CLEAR TEST

6643 024366 013737 002346 002364
 6644 024374 004737 004504
 6645
 6646 024400
 6647 024400 104455
 6648 024402 000002
 6649 024404 012137
 6650 024406 015116
 6651 024410
 6652 024410 104410
 6653 024412 000070
 6654 024414 005237 002360
 6655 024420 005201
 6656 024422 123711 002350
 6657 024426 001417
 6658 024430 005037 002362
 6659 024434 111137 002362
 6660 024440 013737 002350 002364
 6661 024446 004737 004504

MOV RAX15,BADDAT ;GET ACTUAL DATA BYTE
 JSR PC,GETALL ;GET REGS FOR PRINTOUT
 ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
 ERRDF 2,EM2,ERR3

TRAP C\$ERDF
 .WORD 2
 .WORD EM2
 .WORD ERR3

ESCAPE TST

TRAP C\$ESCAPE
 .WORD L10037-

4\$: INC AXNUM ;INCREMENT AX BYTE NO.
 INC R1 ;INCREMENT PAT POINTER
 CMPB RAX16,(R1) ;COMPARE HI BYTE TO EXPECTED VALUE
 BEQ 6\$;BR IF DATA MATCHES

CLR GOODAT
 MOVB (R1),GOODAT ;GET EXPECTED DATA BYTE
 MOV RAX16,BADDAT ;GET ACTUAL DATA BYTE
 JSR PC,GETALL ;GET REGS FOR PRINTOUT
 ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
 ERRDF 2,EM2,ERR3

TRAP C\$ERDF
 .WORD 2
 .WORD EM2
 .WORD ERR3

ESCAPE TST

TRAP C\$ESCAPE
 .WORD L10037-

6\$: INC AXNUM ;INCR AX BYTE NO.
 INC R1 ;INCR PAT POINTER
 CMP R1,#PATJ ;SEE IF ALL REGS READ YET
 BLO 2\$;BR IF NOT DONE READING YET

ENDTST

L10037: TRAP C\$ETST

6662 024452
 6663 024452 104455
 6664 024454 000002
 6665 024456 012137
 6666 024460 015116
 6667 024462
 6668 024462 104410
 6669 024464 000016
 6670 024466 005237 002360
 6671 024472 005201
 6672 024474 020127 002652
 6673 024500 103673
 6674 024502
 6675 024502
 6676 024502
 6677 024502 104401

 .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

T15::

6693 024504
 6694 024504
 6695 024504 004737 003554
 6696 024510 012701 002642
 6697 024514 012702 002456
 6698 024520 112122

JSR PC,MSTCLR ;ISSUE MASTER CLEAR
 MOV #PATI,R1 ;INIT POINTER TO PAT I
 MOV #REDDAT,R2 ;INIT POINTER TO EXPECTED DATA AREA
 3\$: MOVB (R1)+,(R2)+ ;MOVE PAT I INTO REDDAT TABLE

CZDMRF.P11

03-NOV-81 10:29

TEST 15 - EXTENDED REGISTER ADDRESSING TEST

6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773

```

:*****
.SBTTL      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,
:*               376,375,373,367,357,337,277,177.
:*****
BGNTST

```

6774 024774
6775 024774
6776 024774 004737 003554
6777 025000 012701 002662
6778 025004
6779 025004
6780 025004 104404
6781 025006 012737 000004 002360
6782 025014 111137 002352
6783 025020 116137 000001 002354
6784 025026 143737 002543 002352
6785 025034 143737 002544 002354
6786 025042 004737 004270
6787 025046 004737 004102
6788 025052 123737 002346 002352
6789 025060 001416
6790 025062 013737 002352 002362
6791 025070 013737 002346 002364
6792 025076 004737 004504
6793
6794 025102
6795 025102 104455
6796 025104 000003
6797 025106 012176
6798 025110 015116
6799 025112
6800 025112 104410
6801 025114 000052
6802 025116 005237 002360
6803 025122 123737 002350 002647
6804 025130 001416
6805 025132 005037 002362
6806 025136 113737 002647 002362
6807 025144 013737 002350 002364
6808 025152 004737 004504
6809
6810 025156

```

T16::
JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
MOV      #PATK,R1      ;INIT DATA PATTERN POINTER
3$:      BGNSEG
MOV      #4,AXNUM      ;SET BYTE NO. = 4
MOVB     1(R1),WAX15    ;SET DATA TO WRITE INTO LO BYTE
MOVB     1(R1),WAX16    ;SET DATA TO WRITE INTO HI BYTE
BICB     ANBITS+4,WAX15 ;MASK OFF NON-READ/WRITE BITS IN LO BYTE
BICB     ANBITS+5,WAX16 ;MASK OFF NON-READ/WRITE BITS IN HI BYTE
JSR      PC,WRITAX     ;LOAD DATA INTO AX2-15,AX2-16
JSR      PC,READAX     ;READ AX2-15 AND AX2-16
CMPB     RAX15,WAX15   ;COMPARE LO BYTE DATA READ
BEQ      6$            ;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 MISCMPARE
ERRDF   3,EM3,ERR3
TRAP    C$BSEG
TRAP    C$SERDF
.WORD   3
.WORD   EM3
.WORD   ERR3
ESCAPE  SEG
TRAP    C$ESCAPE
.WORD   10000$-
6$:      INC      AXNUM      ;SET AX BYTE NO. = 5
CMPB     RAX16,PATI+5  ;COMPARE HI BYTE DATA READ
BEQ      9$            ;BR IF DATA MATCHES
CLR      GOODAT
MOVB     PATI+5,GOODAT ;SET EXPECTED DATA
MOV      RAX16,BADDAT ;SET ACTUAL DATA
JSR      PC,GETALL     ;GET REGS FOR PRINTOUT
:REPORT  REG MISCMPARE
ERRDF   3,EM3,ERR3

```


CZDMRF.P11 03-NOV-81 10:29

TEST 16 - REGS 15,16 / AX2-15,AX2-16 READ/WRITE BIT TEST

6811 025156 104455
 6812 025160 000003
 6813 025162 012176
 6814 025164 015116
 6815 025166
 6816 025166
 6817 025166
 6818 025166 104405
 6819 025170 062701 000002
 6820 025174 020127 002772
 6821 025200 103701
 6822 025202
 6823 025202
 6824 025202 104401
 6825
 6826
 6827
 6828
 6829
 6830
 6831
 6832
 6833
 6834
 6835
 6836
 6837
 6838
 6839
 6840
 6841 025204
 6842 025204
 6843 025204 004737 003554
 6844 025210 012701 002772
 6845 025214
 6846 025214
 6847 025214 104404
 6848 025216 012737 000000 002360
 6849 025224 111137 002352
 6850 025230 116137 000001 002354
 6851 025236 143737 002537 002352
 6852 025244 143737 002540 002354
 6853 025252 004737 004270
 6854 025256 004737 004102
 6855 025262 123737 002346 002352
 6856 025270 001416
 6857 025272 013737 002352 002362
 6858 025300 013737 002346 002364
 6859 025306 004737 004504
 6860
 6861 025312
 6862 025312 104455
 6863 025314 000003
 6864 025316 012176
 6865 025320 015116
 6866 025322

9\$:
 ENDSEG
 10000\$:
 TRAP C\$ERDF
 .WORD 3
 .WORD EM3
 .WORD ERR3
 TRAP C\$ESEG
 ADD #2,R1 ;INCR PATTERN POINTER
 CMP R1,#PATL ;SEE IF ALL DATA WRITTEN YET
 BLO 3\$;BR IF NOT DONE YET
 ENDTST
 L10041:
 TRAP C\$ETST

 .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 MISCMPARE
 ERRDF 3,EM3,ERR3
 TRAP C\$ERDF
 .WORD 3
 .WORD EM3
 .WORD ERR3
 ESCAPE SEG

CZDMRF.P11 03-NOV-81 10:29

TEST 17 - AX0-15,AX0-16 READ/WRITE BIT TEST

```

6867 025322 104410
6868 025324 000046
6869 025326 005237 002360
6870 025332 123737 002350 002354 6$: INC AXNUM ;SET AX BYTE NO. = 1
        CMPB RAX16,WAX16 ;COMPARE HI BYTE DATA READ
6871 025340 001414 BEQ 9$ ;BR IF DATA MATCHES
6872 025342 013737 002354 002362 MOV WAX16,GOODAT ;SET EXPECTED DATA
6873 025350 013737 002350 002364 MOV RAX16,BADDAT ;SET ACTUAL DATA
6874 025356 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6875 ;REPORT REG MISCMPARE
        ERRDF 3,EM3,ERR3
6876 025362
6877 025362 104455
6878 025364 000003 TRAP C$ERDF
6879 025366 012176 .WORD 3
6880 025370 015116 .WORD EM3
6881 025372 .WORD ERR3
6882 025372
6883 025372
6884 025372 104405 10000$: TRAP C$SEEG
6885 025374 062701 000002 ADD #2,R1 ;INCR PATTERN POINTER
6886 025400 020127 003000 CMP R1,#PATM ;SEE IF ALL DATA WRITTEN YET
6887 025404 103703 BLO 3$ ;BR IF NOT DONE YET
6888 025406
6889 025406
6890 025406 104401 L10042: TRAP C$ETST
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904 025410
6905 025410
6906 025410 004737 003554 JSR PC,MSTCLR ;ISSUE MASTER CLEAR T18::
6907 025414 012701 002662 MOV #PATK,R1 ;INIT DATA PATTERN POINTER
6908 025420
6909 025420
6910 025420 104404 BGNSEG TRAP C$BSEG
6911 025422 012737 000002 002360 MOV #2,AXNUM ;SET BYTE NO. = 2
6912 025430 111137 002352 MOVB (R1),WAX15 ;SET DATA TO WRITE INTO LO BYTE
6913 025434 116137 000001 002354 MOVB 1(R1),WAX16 ;SET DATA TO WRITE INTO HI BYTE
6914 025442 143737 002541 002352 BICB ANBITS+2,WAX15 ;MASK OFF NON-READ/WRITE BITS IN LO BYTE
6915 025450 143737 002542 002354 BICB ANBITS+3,WAX16 ;MASK OFF NON-READ/WRITE BITS IN HI BYTE
6916 025456 004737 004270 JSR PC,WRITAX ;LOAD DATA INTO AX1-15,AX1-16
6917 025462 004737 004102 JSR PC,READAX ;READ AX1-15 AND AX1-16
6918 025466 123737 002346 002352 CMPB RAX15,WAX15 ;COMPARE LO BYTE DATA READ
6919 025474 001416 BEQ 6$ ;BR IF DATA MATCHES
6920 025476 013737 002352 002362 MOV WAX15,GOODAT ;SET EXPECTED DATA
6921 025504 013737 002346 002364 MOV RAX15,BADDAT ;SET ACTUAL DATA
6922 025512 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT

```

```

*****
:SBTTL 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.
*****
BGNSTST

```

```

6904 025410
6905 025410
6906 025410 004737 003554 JSR PC,MSTCLR ;ISSUE MASTER CLEAR T18::
6907 025414 012701 002662 MOV #PATK,R1 ;INIT DATA PATTERN POINTER
6908 025420
6909 025420
6910 025420 104404 BGNSEG TRAP C$BSEG
6911 025422 012737 000002 002360 MOV #2,AXNUM ;SET BYTE NO. = 2
6912 025430 111137 002352 MOVB (R1),WAX15 ;SET DATA TO WRITE INTO LO BYTE
6913 025434 116137 000001 002354 MOVB 1(R1),WAX16 ;SET DATA TO WRITE INTO HI BYTE
6914 025442 143737 002541 002352 BICB ANBITS+2,WAX15 ;MASK OFF NON-READ/WRITE BITS IN LO BYTE
6915 025450 143737 002542 002354 BICB ANBITS+3,WAX16 ;MASK OFF NON-READ/WRITE BITS IN HI BYTE
6916 025456 004737 004270 JSR PC,WRITAX ;LOAD DATA INTO AX1-15,AX1-16
6917 025462 004737 004102 JSR PC,READAX ;READ AX1-15 AND AX1-16
6918 025466 123737 002346 002352 CMPB RAX15,WAX15 ;COMPARE LO BYTE DATA READ
6919 025474 001416 BEQ 6$ ;BR IF DATA MATCHES
6920 025476 013737 002352 002362 MOV WAX15,GOODAT ;SET EXPECTED DATA
6921 025504 013737 002346 002364 MOV RAX15,BADDAT ;SET ACTUAL DATA
6922 025512 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT

```

CZDMRF.P11 03-NOV-81 10:29

TEST 18 - AX1-15,AX1-16 READ/WRITE BIT TEST

```

6923      ;REPORT REG MISCOMPARE
6924 025516 ERRDF 3,EM3,ERR3
6925 025516 104455 TRAP C$ERDF
6926 025520 000003 .WORD 3
6927 025522 012176 .WORD EM3
6928 025524 015116 .WORD ERR3
6929 025526 ESCAPE SEG
6930 025526 104410 TRAP C$ESCAPE
6931 025530 000046 .WORD 10000$-.
6932 025532 005237 002360 6$: INC AXNUM ;SET AX BYTE NO. = 3
6933 025536 123737 002350 002354 CMPB RAX16,WAX16 ;COMPARE HI BYTE DATA READ
6934 025544 001414 BEQ 9$ ;BR IF DATA MATCHES
6935 025546 013737 002354 002362 MOV WAX16,GOODAT ;SET EXPECTED DATA
6936 025554 013737 002350 002364 MOV RAX16,BADDAT ;SET ACTUAL DATA
6937 025562 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6938      ;REPORT REG MISCOMPARE
6939 025566 ERRDF 3,EM3,ERR3
6940 025566 104455 TRAP C$ERDF
6941 025570 000003 .WORD 3
6942 025572 012176 .WORD EM3
6943 025574 015116 .WORD ERR3
6944 025576 9$:
6945 025576 ENDSEG
6946 025576 10000$:
6947 025576 104405 TRAP C$ESEG
6948 025600 062701 000002 ADD #2,R1 ;INCR PATTERN POINTER
6949 025604 020127 002772 CMP R1,#PATL ;SEE IF ALL DATA WRITTEN YET
6950 025610 103703 BLO 3$ ;BR IF NOT DONE YET
6951 025612
6952 025612
6953 025612 104401 L10043:
6954 TRAP C$ETST
6955
6956
6957
6958
6959

```

```

6960      ;*****
6961      ;SBTTL TEST 19 - AX3-15,AX3-16 READ/WRITE BIT TEST
6962      ;*
6963      ;* IN THIS TEST A MASTER CLEAR IS DONE AND THEN A WRITE, READ, AND COMPARE ARE
6964      ;* PERFORMED IN REGS AX3-15,AX3-16 USING EACH WORD OF PATTERN V FOR WRITING,
6965      ;* AND PATTERN U FOR COMPARING.
6966      ;* ANY BITS IN AX3-15,AX3-16 WHICH ARE NOT READ/WRITE ARE MASKED OFF (TO 0)
6967      ;* IN THE EXPECTED VALUE BEFORE COMPARISON.
6968      ;* PATTERN V =
6969      ;* FOR REG 15 : 000,333,331,323,313,233,133,000,000,000,000,
6970      ;* 000,000,000,000,000,000,000,000
6971      ;* FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,
6972      ;* 100,200,346,345,343,307,247,147
6973      ;* PATTERN U =
6974      ;* FOR REG 15 : 000,001,013,011,021,101,301,000,000,000,000,
6975      ;* 000,000,000,000,000,000,000,000
6976      ;* FOR REG 16 : 000,000,000,000,000,000,000,001,002,004,040,
6977      ;* 100,200,346,345,343,307,247,147
6978      ;*****
6978 025614 BGNTST

```


CZDMRF.P11 03-NOV-81 10:29

TEST 19 - AX3-15,AX3-16 READ/WRITE BIT TEST

```

6979 025614
6980 025614 004737 003554
6981 025620 142777 000010 154600
6982 025626 012702 003130
6983 025632 012701 003062
6984 025636
6985 025636
6986 025636 104404
6987 025640 012737 000006 002360
6988 025646 111237 002352
6989 025652 116237 000001 002354
6990 025660 004737 004270
6991 025664 004737 004102
6992 025670 132737 000001 002352
6993 025676 001003
6994 025700 142737 000372 002346
6995 025706 142737 000040 002346
6996 025714 123711 002346
6997 025720 001417
6998 025722 005037 002362
6999 025726 111137 002362
7000 025732 013737 002346 002364
7001 025740 004737 004504
7002
7003 025744
7004 025744 104455
7005 025746 000003
7006 025750 012176
7007 025752 015116
7008 025754
7009 025754 104410
7010 025756 000052
7011 025760 005237 002360
7012 025764 123761 002350 000001
7013 025772 001416
7014 025774 005037 002362
7015 026000 116137 000001 002362
7016 026006 013737 002350 002364
7017 026014 004737 004504
7018
7019 026020
7020 026020 104455
7021 026022 000003
7022 026024 012176
7023 026026 015116
7024 026030
7025 026030
7026 026030
7027 026030 104405
7028 026032 062702 000002
7029 026036 062701 000002
7030 026042 020127 003130
7031 026046 103673
7032 026050
7033 026050
7034 026050 104401

```

```

T19::
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
BICB #LULoop,@BSEL1 ;CLEAR LULoop
MOV #PATV,R2 ;INIT PATTERN V POINTER
MOV #PATU,R1 ;INIT PATTERN U POINTER
3$:
BGNSEG
TRAP CSBSEG
MOV #6,AXNUM ;SET BYTE NO. = 6
MOVB (R2),WAX15 ;SET DATA TO WRITE INTO LO BYTE
MOVB 1(R2),WAX16 ;SET DATA TO WRITE INTO HI BYTE
JSR PC,WRITAX ;LOAD DATA INTO AX3-15,AX3-16
JSR PC,READAX ;READ AX3-15 AND AX3-16
BITB #TEST,WAX15 ;SEE IF AN INTERFACE IS SELECTED
BNE 4$ ;BR IF YES
BICB #AX315U,RAX15 ;MASK OFF UNPREDICTABLE BITS
BICB #C32BCC,RAX15 ;CLEAR CRC32 BCC BIT
CMPB RAX15,(R1) ;COMPARE LO BYTE DATA READ
BEQ 6$ ;BR IF DATA MATCHES
CLR GOODAT
MOVB (R1),GOODAT ;SET EXPECTED DATA
MOV RAX15,BADDAT ;SET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT REG MISC COMPARE
ERRDF 3,EM3,ERR3
TRAP CSERDF
.WORD 3
.WORD EM3
.WORD ERR3
ESCAPE SEG
TRAP CSESCAPE
.WORD 10000$-
6$:
INC AXNUM ;SET AX BYTE NO. = 7
CMPB RAX16,1(R1) ;COMPARE HI BYTE DATA READ
BEQ 9$ ;BR IF DATA MATCHES
CLR GOODAT
MOVB 1(R1),GOODAT ;SET EXPECTED DATA
MOV RAX16,BADDAT ;SET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT REG MISC COMPARE
ERRDF 3,EM3,ERR3
TRAP CSERDF
.WORD 3
.WORD EM3
.WORD ERR3
9$:
ENDSEG
10000$:
TRAP CSESEG
ADD #2,R2 ;INCR PATTERN V POINTER
ADD #2,R1 ;INCR PATTERN U POINTER
CMP R1,#PATV ;SEE IF ALL DATA WRITTEN YET
BLO 3$ ;BR IF NOT DONE YET
ENDTST
L10044:
TRAP CSETST

```

CZDMRF.P11 03-NOV-81 10:29

TEST 19 - AX3-15,AX3-16 READ/WRITE BIT TEST

7035
7036
7037
7038
7039
7040
7041
7042
7043
7044
7045
7046
7047
7048
7049
7050
7051
7052
7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064
7065
7066
7067
7068
7069
7070
7071
7072
7073
7074
7075
7076
7077
7078
7079
7080
7081
7082
7083
7084
7085
7086
7087
7088
7089
7090

026052
026052
026052 004737 003554
026056 012701 003026
026062 012702 003044
026066 012737 000017 002356
026074 012737 000005 002360

026102
026102
026102 104402
026104
026104 104404
026106 111137 002344
026112 004737 003726
026116 004737 004102
026122 123712 002350
026126 001421
026130 005037 002366
026134 111137 002366
026140 005037 002362
026144 111237 002362
026150 013737 002350 002364
026156 004737 004504

026162
026162 104455
026164 000003
026166 012176
026170 016256
026172
026172
026172 104405
026174 005201
026176 005202
026200 020127 003044

```

:*****
: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).
:*****
BGNTST
                                           T20::
      JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
      MOV      #PATO,R1      ;INIT PAT O POINTER
      MOV      #PATP,R2      ;INIT PAT P POINTER
      MOV      #17,REGNUM    ;SET LU REG NO. = 17
      MOV      #5,AXNUM      ;SET AX BYTE NO. = 5 FOR AX2-16
-----
: WRITE REG 17, READ AND COMPARE AX2-16
-----
      BGNSUB
                                           T20.1:
3$:
      BGNSEG
                                           TRAP      C$BSUB
      MOVB     (R1),WRIBYT    ;SET BYTE TO BE WRITTEN
      JSR      PC,WRITLU     ;WRITE DATA BYTE INTO REG 17
      JSR      PC,READAX     ;READ AX2
      CMPB     RAX16,(R2)    ;COMPARE AX2-16 TO EXPECTED DATA
      BEQ      6$            ;BR IF DATA MATCHES
      CLR      LOADAT
      MOVB     (R1),LOADAT   ;SET DATA WHICH WAS WRITTEN
      CLR      GOODAT
      MOVB     (R2),GOODAT   ;SET EXPECTED DATA READ
      MOV      RAX16,BADDAT ;SET ACTUAL DATA READ
      JSR      PC,GETALL     ;GET REGS FOR PRINTOUT
:REPORT REG MISCMPARE
      ERRDF   3,EM3,ERR5
                                           TRAP      C$ERDF
                                           .WORD    3
                                           .WORD    EM3
                                           .WORD    ERR5
6$:
      ENDSEG
                                           10000$:
                                           TRAP      C$ESEG
      INC      R1            ;INCR PAT O POINTER
      INC      R2            ;INCR PAT P POINTER
      CMP      R1,#PATP     ;SEE IF ALL BYTES LOADED YET

```


CZDMRF.P11 03-NOV-81 10:29

TEST 20 - REG 17 - AX2-16 READ/WRITE, MASTER CLEAR TEST

```

7091 026204 103737          BLO      3$          ;BR IF NOT
7092 026206          ENDSUB
7093 026206
7094 026206 104403          L10046:          TRAP      C$ESUB
7095
7096 -----
7097 : LOAD REG 17, DO MASTER CLEAR, READ AND COMPARE AX2-16
7098 -----
7098 026210          BGNSUB
7099 026210
7100 026210 104402          T20.2:          TRAP      C$BSUB
7101 026212 112737 000375 002344      MOVB    #375,WRIBYT      ;SET DATA TO BE LOADED
7102 026220 004737 003726          JSR    PC,WRITLU        ;LOAD DATA INTO REG 17
7103 026224 004737 003554          JSR    PC,MSTCLR        ;PERFORM MASTER CLEAR
7104 026230 004737 004102          JSR    PC,READAX        ;READ AX2-15,AX2-16
7105 026234 123737 002350 002647      CMPB   RAX16,PATI+5     ;SEE IF AX2-16 WAS INIT'D CORRECTLY
7106 026242 001416          BEQ    6$              ;BR IF YES
7107 026244 005037 002362          CLR    GOODAT
7108 026250 113737 002647 002362      MOVB   PATI+5,GOODAT    ;SET EXPECTED DATA
7109 026256 013737 002350 002364      MOV    RAX16,BADDAT     ;SET ACTUAL DATA
7110 026264 004737 004504          JSR    PC,GETALL        ;GET REGS FOR PRINTOUT
7111          ;REPORT REG NOT INITIALIZED BY MASTER CLEAR
7112          ERRDF 2,EM2,ERR3
7113 026270 104455          TRAP   C$ERDF
7114 026272 000002          .WORD 2
7115 026274 012137          .WORD EM2
7116 026276 015116          .WORD ERR3
7117 026300          6$:
7118 026300          ENDSUB
7119 026300
7120 026300 104403          L10047:          TRAP      C$ESUB
7121 026302          ENDTST
7122 026302
7123 026302 104401          L10045:          TRAP      C$ETST
7124
7125
7126
7127
7128
7129
7130 :*****
7131 :SBTTL      TEST 21 - TRANSMITTER BUFFER DATA TEST
7132 :* A MASTER CLEAR IS DONE FIRST, AND THEN A BYTE OF PATTERN N IS LOADED INTO
7133 :* REG 11 AND THE NEXT BYTE IS LOADED TWICE INTO REG 10. THE PROGRAM THEN WAITS
7134 :* AT LEAST 50 MICRO-SEC, AND THEN IT READS AND COMPARES AX1-15 TO THE BYTE
7135 :* WHICH WAS LOADED INTO REG 10, AND IT READS AND COMPARES AX1-16 TO THE BYTE
7136 :* WHICH WAS LOADED INTO REG 11. THIS PROCESS IS REPEATED (INCLUDING THE MASTER
7137 :* CLEAR) FOR EACH PAIR OF BYTES IN PATTERN N.
7138 :* PATTERN N =
7139 :*   FOR REG 10: 000,125,252,377,000,000,000
7140 :*   FOR REG 11: 000,000,000,000,005,012,017
7141 :*****
7141 026304          BGNTST
7142 026304
7143 026304 012701 003010          MOV    #PATN,R1        ;INIT PATTERN POINTER
7144 026310 012737 026552 002340      MOV    #A2,RETADR      ;SET SUBROUTINE ERROR RETURN ADDRESS
7145 026316          BGNSUB
7146 026316          T21.1:

```


7203 026554 104401

TRAP CSETST

7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229
7230
7231
7232
7233
7234
7235
7236
7237
7238
7239
7240
7241
7242
7243
7244
7245
7246
7247
7248
7249
7250
7251
7252
7253
7254
7255
7256
7257
7258

026556
026556
026556 012737 027232 002340

026564 004737 003554
026570 004737 004640
026574 000001

026576 012737 000400 002400
026604 004737 005152
026610 004737 005152
026614 004737 005124
026620 004737 004640
026624 000003

026626 005001
026630 012737 000017 002356
026636 004737 005232
026642 000001
026644 004737 003650
026650 132737 000020 002342
026656 001404
026660 005201
026662 020127 000003
026666 002763
026670 004737 004640
026674 000001

```

*****
SBTTL      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.
*****
BGNTST
                                T22::
                                MOV      #A3,RETADR      ;SET SUBR ERROR RETURN ADDR
-----
; SET MASTER CLEAR, CHECK FOR ORDY=1, OCOR=0
-----
                                JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
                                JSR      PC,OSIRDY      ;CHK ORDY=1, OCOR=0
                                1
-----
; LOAD 2 SOM CHARS, ALLOW SILO TO RIPPLE, CHK ORDY=1, OCOR=1
-----
                                MOV      #TXSOM,TXWORD   ;SET DATA TO WRITE INTO SILO
                                JSR      PC,LDTXSI      ;LOAD THE SILO WITH SOM
                                JSR      PC,LDTXSI      ;LOAD ANOTHER SOM
                                JSR      PC,WAIT50      ;WAIT FOR DATA TO RIPPLE
                                JSR      PC,OSIRDY      ;CHK ORDY=1, OCOR=1
                                3
-----
; CLOCK LINE UNIT, CHK FOR OCOR = 0 WITHIN 3 CYCLES
-----
                                CLR      R1             ;INIT CYCLE COUNTER TO 0
                                MOV      #17,REGNUM     ;SET REG NO. = 17
3$:                               JSR      PC,STPLU      ;STEP LU 1 CYCLE
                                1
                                JSR      PC,READLU      ;READ REG 17
                                BITB     #OCOR,REDBYT   ;SEE IF OCOR = 0 YET
                                BEQ      6$             ;BR IF OCOR = 0
                                INC      R1             ;INCR CYCLE COUNT
                                CMP      R1,#3         ;SEE IF 3 CYCLES DONE YET
                                BLT      3$             ;BR IF NO
6$:                               JSR      PC,OSIRDY      ;CHK ORDY=1, OCOR=0
                                1
-----

```


CZDMRF.P11 03-NOV-81 10:29

TEST 22 - TRANSMITTER BUFFER SEQUENCING TEST

```

7259 ; LOAD 64 BINARY COUNT CHARS INTO SILO, CHK ORDY=0
7260 -----
7261 026676 005003          CLR      R3          ;INIT PATTERN FOR WRITING
7262 026700 010337 002400 8$:  MOV      R3,TXWORD ;SET DATA TO BE WRITTEN
7263 026704 004737 005152   JSR      PC,LDTXSI  ;LOAD DATA CHAR INTO TX SILO
7264 026710 004737 005124   JSR      PC,WAIT50  ;WAIT FOR DATA TO RIPPLE IN SILO
7265 026714 020327 000077   CMP      R3,#63.    ;SEE IF 64TH CHAR JUST LOADED
7266 026720 001004          BNE      9$         ;BR IF NO
7267 026722 012737 000002 026744  MOV      #2,14$    ;SET UP TO CHK ORDY=0,OCOR=1
7268 026730 000403          BR       12$
7269 026732 012737 000003 026744 9$:  MOV      #3,14$    ;SET UP TO CHK ORDY=1,OCOR=1
7270 026740 004737 004640 12$: JSR      PC,OSIRDY ;CHK ORDY, OCOR
7271 026744 000000 14$:  .WORD   0
7272 026746 005203          INC      R3          ;INCR PATTERN FOR WRITES
7273 026750 020327 000100   CMP      R3,#64.    ;SEE IF 64 CHARS LOADED YET
7274 026754 002751          BLT      8$         ;BR IF NO
7275 -----
7276 ; CLOCK LINE UNIT, CHECK ORDY = 1 WITHIN 8 CYCLES
7277 -----
7278 026756 012737 000011 002356  MOV      #11,REGNUM ;SET REG NO. = 11
7279 026764 005001          CLR      R1          ;INIT CYCLE COUNT
7280 026766 004737 005232 16$: JSR      PC,STPLU  ;CLOCK LU FOR 1 CYCLE
7281 026772 000001          1
7282 026774 004737 003650   JSR      PC,READLU  ;READ REG 11
7283 027000 132737 000020 002342 BITB    #ORDY,REDBYT ;SEE IF ORDY = 1 YET
7284 027006 001004          BNE      19$        ;BR IF YES
7285 027010 005201          INC      R1          ;INCR CYCLE COUNT
7286 027012 020127 000010   CMP      R1,#8.     ;SEE IF 8 CYCLES YET
7287 027016 002763          BLT      16$        ;BR IF NOT YET
7288 027020 004737 004640 19$: JSR      PC,OSIRDY ;CHK ORDY = 1, OCOR = 1
7289 027024 000003          3
7290 -----
7291 ; READ AND COMPARE FIRST CHARACTER IN AX1-15
7292 -----
7293 027026 005004          CLR      R4          ;INIT PATTERN FOR READING
7294 027030 012737 000002 002360  MOV      #2,AXNUM   ;SET AX BYTE NO. FOR AX1-15
7295 027036 004737 004102   JSR      PC,READAX  ;READ AX1-15
7296 027042 123704 002346   CMPB    RAX15,R4    ;COMPARE AX1-15 TO EXPECTED
7297 027046 001415          BEQ      20$        ;BR IF MATCH
7298 027050 010437 002362   MOV      R4,GOODAT  ;SET EXPECTED DATA
7299 027054 013737 002346 002364  MOV      RAX15,BADDAT ;SET ACTUAL DATA
7300 027062 004737 004504   JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
7301 ;REPORT REG MISCMPARE
7302 027066 104455          ERRDF   3,EM3,ERR3
7303 027066 000003          TRAP    C$ERDF
7304 027070 012176          .WORD   3
7305 027072 015116          .WORD   EM3
7306 027074 010410          .WORD   ERR3
7307 027076 000132          ESCAPE  TST
7308 027076 000132          TRAP    C$ESCAPE
7309 027100 000132          .WORD   L10052-.
7310 027102
7311 -----
7312 ; LOAD AND COMPARE REST OF CHARS, MONITOR ORDY, OCOR
7313 -----
7314 027102 020327 000377 24$:  CMP      R3,#255.  ;SEE IF ALL CHARS LOADED YET

```


CZDMRF.P11 03-NOV-81 10:29

TEST 22 - TRANSMITTER BUFFER SEQUENCING TEST

```

7315 027106 003010          BGT      26$          ;BR IF YES
7316 027110 010337 002400    MOV      R3,TXWORD    ;SET DATA TO BE WRITTEN
7317 027114 004737 005152    JSR      PC,LDTXSI    ;LOAD DATA CHAR INTO TX SILO
7318 027120 005203          INC      R3           ;INCR DATA TO BE WRITTEN
7319 027122 004737 004640    JSR      PC,OSIRDY    ;CHK ORDY=0, OCOR=1
7320 027126 000002          2
7321 027130 005204          26$: INC      R4           ;INCR PAT FOR READING
7322 027132 004737 005232    JSR      PC,STPLU     ;CLOCK LINE UNIT FOR 8 CYCLES
7323 027136 000010          8.
7324 027140 004737 004102    JSR      PC,READAX    ;READ AX1-15
7325 027144 123704 002346    CMPB    RAX15,R4     ;COMPARE AX1-15 TO EXPECTED
7326 027150 001415          BEQ     27$          ;BR IF MATCH
7327 027152 010437 002362    MOV      R4,GOODAT    ;SET EXPECTED DATA
7328 027156 013737 002346 002364 MOV      RAX15,BADDAT ;SET ACTUAL DATA
7329 027164 004737 004504    JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
7330          ;REPORT REG MISCMPARE
7331 027170          ERRDF  3,EM3,ERR3
7332 027170 104455          TRAP    C$ERDF
7333 027172 000003          .WORD  3
7334 027174 012176          .WORD  EM3
7335 027176 015116          .WORD  ERR3
7336 027200          ESCAPE  TST
7337 027200 104410          TRAP    C$ESCAPE
7338 027202 000030          .WORD  L10052-.
7339 027204 020427 000377 27$:  CMP      R4,#255.    ;SEE IF WE READ LAST CHAR YET
7340 027210 001004          BNE     29$          ;BR IF NOT
7341 027212 004737 004640    JSR      PC,OSIRDY    ;CHK ORDY=1, OCOR=0
7342 027216 000001          1
7343 027220 000404          BR      A3
7344 027222 004737 004640 29$:  JSR      PC,OSIRDY    ;CHK ORDY=1, OCOR=1
7345 027226 000003          3
7346 027230 000724          BR      24$          ;CONTINUE
7347 027232          A3:
7348 027232          ENDTST
7349 027232
7350 027232 104401          L10052: TRAP    C$ETST
7351
7352
7353
7354
7355

```

```

:*****
: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

```

7370 027234

CZDMRF.P11 03-NOV-81 10:29

TEST 23 - TX MSG TIMING TEST, CHAR MODE, WITH CRC

```

7371 027234
7372 027234 012737 027346 002340
7373 027242 004737 005516
7374 027246 000226
7375 027250 000011
7376 027252 004737 006100
7377 027256 000000
7378 027260 100010
7379 027262 004737 006100
7380 027266 000000
7381 027270 000010
7382 027272 004737 006100
7383 027276 001000
7384 027300 000010
7385 027302 004737 006100
7386 027306 001000
7387 027310 000010
7388 027312 004737 006100
7389 027316 001000
7390 027320 000020
7391 027322 004737 006100
7392 027326 001000
7393 027330 000010
7394 027332 004737 006100
7395 027336 001000
7396 027340 000010
7397 027342 004737 006252
7398 027346
7399 027346 004737 003554
7400 027352
7401 027352
7402 027352 104401
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418
7419
7420
7421
7422 027354
7423 027354
7424 027354 012737 027456 002340
7425 027362 004737 005516
7426 027366 000000

```

```

MOV #A4,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS T23::
JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'2
SYNCH
STRIP!DDCMP
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 CRC-16 CHAR
TXEOM
16.
JSR PC, TXCHAR ;LOAD EOM, TX FIRST TERMINATING SYNCH
TXEOM
8.
JSR PC, TXCHAR ;LOAD EOM, TX 2ND TERMINATING SYNCH
TXEOM
8.
JSR PC,ENDTRN ;SET OC, MONITOR OCOR
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP

```

A4:
ENDTST

L10053: TRAP C\$ETST

```

:*****
: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

```

```

MOV #A5,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS T24::
JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'2
000

```


CZDMRF.P11 03-NOV-81 10:29

TEST 24 - TX MSG TIMING TEST, BIT MODE, WITH CRC

```

7427 027370 000000      000
7428 027372 004737 006100 JSR    PC,TXCHAR      ;LOAD A 000 CHAR, TX FIRST FLAG
7429 027376 000000      000
7430 027400 100010      CHPCHK!8.
7431 027402 004737 006100 JSR    PC,TXCHAR      ;LOAD 2ND 000 CHAR, TX 2ND FLAG
7432 027406 C00000      000
7433 027410 000010      8.
7434 027412 004737 006100 JSR    PC,TXCHAR      ;LOAD EOM CHAR, TX FIRST 000 CHAR
7435 027416 001000      TXEOM
7436 027420 000010      8.
7437 027422 004737 006100 JSR    PC,TXCHAR      ;LOAD EOM, TX 2ND 000 CHAR AND CRC-CCITT-1 CHAR
7438 027426 001000      TXEOM
7439 027430 000030      24.
7440 027432 004737 006100 JSR    PC,TXCHAR      ;LOAD EOM, TX FIRST TERMINATING FLAG
7441 027436 001000      TXEOM
7442 027440 000010      8.
7443 027442 004737 006100 JSR    PC,TXCHAR      ;LOAD EOM, TX 2ND TERMINATING FLAG
7444 027446 001000      TXEOM
7445 027450 000010      8.
7446 027452 004737 006252 JSR    PC,ENDTRN      ;SET OC, MONITOR OCOR
7447 027456 004737 003554 JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR TO CLEAN UP
7448 027456 004737 003554
7449 027462
7450 027462
7451 027462 104401
7452
7453
7454
7455
7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473 027464
7474 027464
7475 027464 012737 027652 002340
7476 027472 004737 005516
7477 027476 000226
7478 027500 000311
7479 027502 012737 000014 002356
7480 027510 012737 000100 002344
7481 027516 004737 003726
7482 027522 004737 006100

```

A5:

ENDTST

L10054: TRAP C\$ETST

```

:*****
:SBTTL      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)

```


CZDMRF.P11 03-NOV-81 10:29

TEST 25 - TX MSG TIMING TEST, CHAR MODE, WITH NO CRC

```

7483 027526 000000      000
7484 027530 100010      CHPCHK!8.
7485 027532 004737 006100 JSR      PC,TXCHAR      ;LOAD 2ND 000 CHAR, TX 2ND SYNCH
7486 027536 000000      000
7487 027540 000010      8.
7488 027542 004737 006100 JSR      PC,TXCHAR      ;LOAD EOM CHAR, TX FIRST 000 CHAR
7489 027546 001000      TXEOM
7490 027550 000010      8.
7491 027552 004737 006100 JSR      PC,TXCHAR      ;LOAD EOM CHAR, TX 2ND 000 CHAR
7492 027556 001000      TXEOM
7493 027560 000010      8.
7494 027562 004737 006100 JSR      PC,TXCHAR      ;LOAD EOM, TX FIRST TERMINATING SYNCH
7495 027566 001000      TXEOM
7496 027570 000010      8.
7497 027572 004737 006100 JSR      PC,TXCHAR      ;LOAD EOM, TX 2ND TERMINATING SYNCH
7498 027576 001000      TXEOM
7499 027600 000010      8.
7500 027602 004737 005232 JSR      PC,STPLU      ;CLK PAST END OF MSG
7501 027606 000030      24.
7502 027610 012737 000013 002356 MOV      #13,REGNUM    ;SET REG NO. = 13
7503 027616 004737 003650 JSR      PC,READLU    ;READ REG 13
7504 027622 032737 000040 002342 BIT      #RTS,REDBYT  ;CHK FOR RTS STILL SET
7505 027630 001006      3$ BNE      3$          ;BR IF RTS SET
7506 027632 004737 004504 JSR      PC,GETALL    ;GET REGS FOR PRINTOUT
7507                                     ;REPORT RTS NOT SET
7508 027636                                     ERRDF 60,EM60,ERR7
7509 027636 104455
7510 027640 000074 TRAP    C$ERDF
7511 027642 014206 .WORD 60
7512 027644 017446 .WORD EM60
7513 027646 004737 006252 3$: JSR      PC,ENDTRN ;SET OC, MONITOR OCOR .WORD ERR7
7514 027652 A6:
7515 027652 004737 003554 JSR      PC,MSTCLR    ;ISSUE MASTER CLEAR TO CLEAN UP
7516 027656
7517 027656
7518 027656 104401 L10055: TRAP    C$SETST
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538 027660

```

```

:*****
: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

```

CZDMRF.P11 03-NOV-81 10:29

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

T26::

```

-----
: CAUSE TX UNDERRUN, CHK UNRR = 1; SET SOM, CHK UNRR = 0
-----
7539 027660
7540
7541
7542
7543 027660 012737 030216 002340      MOV    #A7,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
7544 027666 004737 005516                JSR    PC,INITRN      ;DO MASTER CLEAR, LOAD 2 SOM'S
7545 027672 000226
7546 027674 000011
7547 027676 004737 006100      JSR    PC,TXCHAR      ;LOAD A 000 CHAR, TX FIRST SYNCH
7548 027702 000000
7549 027704 100010
7550 027706 004737 005232      JSR    PC,STPLU      ;CLOCK THE TRANSMITTER UNTIL 7 BITS OF
7551 027712 000016 14.                ; THE 000 CHAR HAVE BEEN TRANSMITTED
7552 027714 012737 000011 002356      MOV    #11,REGNUM     ;SET LU REG NO. = 11
7553 027722 004737 003650                JSR    PC,READLU     ;READ REG 11
7554 027726 132737 000001 002342      BITB  #UNRR,REDBYT   ;CHK FOR UNRR = 0
7555 027734 001410                BEQ    6$            ;BR IF UNRR = 0
7556 027736 004737 004504      JSR    PC,GETALL     ;GET REGS FOR PRINTOUT
7557
:REPORT UNRR NOT CLEARED
7558 027742      ERRDF 16,EM16,ERR7
7559 027742 104455
7560 027744 000020
7561 027746 012600
7562 027750 017446
7563 027752 000137 030216
7564 027756 004737 005232      6$:   JMP    A7          ;SKIP TO END OF TEST
7565 027762 000003
7566 027764 004737 003650                JSR    PC,STPLU     ;CLOCK LAST BIT OF 000 CHAR
7567 027770 132737 000001 002342      JSR    PC,READLU     ;READ REG 11
7568 027776 001010                BITB  #UNRR,REDBYT   ;CHK FOR UNRR = 1
7569 030000 004737 004504      BNE   9$            ;BR IF UNRR = 1
7570
:REPORT UNRR NOT SET
7571 030004      ERRDF 14,EM14,ERR7
7572 030004 104455
7573 030006 000016
7574 030010 012534
7575 030012 017446
7576 030014 000137 030216
7577 030020 012737 000400 002400      9$:   JMP    A7          ;SKIP TO END OF TEST
7578 030026 004737 005152                MOV    #TXSOM,TXWORD ;SET SOM CHAR TO BE WRITTEN
7579 030032 004737 005124                JSR    PC,LDTXSI    ;LOAD SOM CHAR INTO TX SILO
7580 030036 004737 005232                JSR    PC,WAIT50    ;WAIT FOR SILO DATA TO RIPPLE
7581 030042 000002
7582 030044 004737 003650                JSR    PC,STPLU     ;CLOCK LU FOR 2 CYCLES
7583 030050 132737 000001 002342      JSR    PC,READLU     ;READ REG 11
7584 030056 001410                BITB  #UNRR,REDBYT   ;CHK FOR UNRR = 0
7585 030060 004737 004504      BEQ    12$          ;BR IF UNRR = 0
7586
:REPORT UNRR NOT CLEARED BY SOM
7587 030064      ERRDF 13,EM13,ERR7
7588 030064 104455
7589 030066 000015
7590 030070 012504
7591 030072 017446
7592 030074 000137 030216
7593 030100
7594
:-----
12$:   JMP    A7          ;SKIP TO END OF TEST
:-----

```

```

TRAP  CSERDF
.WORD 16
.WORD EM16
.WORD ERR7

```

```

TRAP  CSERDF
.WORD 14
.WORD EM14
.WORD ERR7

```

```

TRAP  CSERDF
.WORD 13
.WORD EM13
.WORD ERR7

```


CZDMRF.P11 03-NOV-81 10:29

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

:CAUSE TX UNDERRUN, CHK UNRR = 1; DO MASTER CLR, CHK UNRR = 0

```

7595
7596
7597 030100 004737 005516      JSR      PC,INITRN      ;DO MASTER CLEAR, LOAD 2 SOM'S
7598 030104 000226              SYNCH
7599 030106 000051              IDLE!STRIP!DDCMP
7600 030110 004737 006100      JSR      PC,TXCHAR      ;LOAD A 000 CHAR, TX FIRST SYNCH
7601 030114 000000              000
7602 030116 100010              CHPCHK!8.
7603 030120 004737 005232      JSR      PC,STPLU       ;STEP THE LU UNTIL 000 HAS BEN TRANSMITTED
7604 030124 000021              17.
7605 030126 004737 003650      JSR      PC,READLU      ;READ REG 11
7606 030132 132737 000001 002342 BITB     #UNRR,REDBYT    ;CHK FOR UNR = 1
7607 030140 001010              BNE     16$             ;BR IF UNRR = 1
7608 030142 004737 004504      JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
7609
7610 030146              :REPORT UNRR NOT SET
7611 030146 104455              ERRDF   14,EM14,ERR7
7612 030150 000016
7613 030152 012534
7614 030154 017446
7615 030156 000137 030216      JMP      A7             ;SKIP TO END OF TEST
7616 030162 004737 003554 16$:    JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
7617 030166 004737 003650      JSR      PC,READLU      ;READ REG 11
7618 030172 132737 000001 002342 BITB     #UNRR,REDBYT    ;CHK FOR UNRR = 0
7619 030200 001406              BEQ     A7             ;BR IF UNRR = 0
7620 030202 004737 004504      JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
7621
7622              :REPORT UNRR NOT CLEARED BY OC
7623 030206 104455              ERRDF   15,EM15,ERR7
7624 030210 000017
7625 030212 012551
7626 030214 017446
7627 030216 004737 003554  A7:    JSR      PC,MSTCLR      ;ISSUE CLEAN-UP MASTER CLEAR
7628 030222              ENDTST
7629 030222
7630 030222 104401              L10056: TRAP     CSETST
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642
7643
7644
7645 030224
7646 030224
7647 030224 012737 030432 002340      MOV     #A8,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
7648 030232 004737 005516      JSR      PC,INITRN      ;DO MASTER CLR, LOAD 2 SOM'S
7649 030236 000000
7650 030240 000041              000
                          IDLE!DDCMP

```

```

:*****
:SBTTL      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.
:*****
BGNTST

```

```

7645 030224
7646 030224
7647 030224 012737 030432 002340      MOV     #A8,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
7648 030232 004737 005516      JSR      PC,INITRN      ;DO MASTER CLR, LOAD 2 SOM'S
7649 030236 000000
7650 030240 000041              000
                          IDLE!DDCMP

```


CZDMRF.P11 03-NOV-81 10:29

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

```

7651 030242 012737 000006 002360      MOV      #6,AXNUM          ;SET BYTE NO. = 6 FOR AX3
7652 030250 012737 000000 002352      MOV      #000,WAX15       ;SET DATA FOR AX3-15 = 0
7653 030256 012737 000240 002354      MOV      #TXLEN2!TXLENO,WAX16 ;SET TX LENGTH = 5 FOR AX3-16
7654 030264 004737 004270          JSR      PC,WRITAX        ;LOAD AX3-15,AX3-16
7655 030270 004737 006100          JSR      PC,TXCHAR        ;LOAD 5-BIT 000 CHAR, TX 8-BIT SYNCH
7656 030274 000000          000
7657 030276 100010          CHPCHK!8.
7658 030300 004737 006100          JSR      PC,TXCHAR        ;LOAD 6-BIT 000 CHAR, TX 5-BIT SYNCH
7659 030304 000000          000
7660 030306 000005          5
7661 030310 012737 000300 002354      MOV      #TXLEN2!TXLEN1,WAX16
7662 030316 004737 004270          JSR      PC,WRITAX        ;SET TX CHAR LENGTH = 6
7663 030322 004737 006100          JSR      PC,TXCHAR        ;LOAD 7-BIT 000 CHAR, TX 5-BIT 000 CHAR
7664 030326 000000          000
7665 030330 000005          5
7666 030332 012737 000340 002354      MOV      #TXLEN2!TXLEN1!TXLENO,WAX16
7667 030340 004737 004270          JSR      PC,WRITAX        ;SET TX CHAR LENGTH = 7
7668 030344 004737 006100          JSR      PC,TXCHAR        ;LOAD 8-BIT 000 CHAR, TX 6-BIT 000 CHAR
7669 030350 000000          000
7670 030352 000006          6
7671 030354 012737 000000 002354      MOV      #000,WAX16
7672 030362 004737 004270          JSR      PC,WRITAX        ;SET TX CHAR LENGTH = 8
7673 030366 004737 006100          JSR      PC,TXCHAR        ;LOAD EOM, TX 7-BIT 000 CHAR
7674 030372 001000          TXEOM
7675 030374 000007          7
7676 030376 004737 006100          JSR      PC,TXCHAR        ;LOAD EOM, TX 8-BIT 000 CHAR
7677 030402 001000          TXEOM
7678 030404 000010          8.
7679 030406 004737 006100          JSR      PC,TXCHAR        ;LOAD EOM, TX CRC-16 CHAR
7680 030412 001000          TXEOM
7681 030414 000020          16.
7682 030416 004737 006100          JSR      PC,TXCHAR        ;LOAD EOM, TX FIRST TERMINATING SYNCH
7683 030422 001000          TXEOM
7684 030424 000010          8.
7685 030426 004737 006252          JSR      PC,ENDTRN        ;CLEAR TRANSMITTER

```

A8:
ENDTST

L10057: TRAP CSETST

```

7690
7691
7692
7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705 030434
7706 030434

```

```

:*****
: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

```

T28::

CZDMRF.P11 03-NOV-81 10:29

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

7707	030434	012737	030762	002340	MOV	#A9,RETADR	;SET TEST EXIT ADDRESS FOR ERRORS
7708	030442	004737	005516		JSR	PC,INITRN	;DO MASTER CLR, LOAD 2 SOM'S
7709	030446	000000			000		
7710	030450	000000			000		
7711	030452	004737	006100		JSR	PC,TXCHAR	;LOAD FIRST 8-BIT 000 CHAR, TX FIRST FLAG
7712	030456	000000			000		
7713	030460	100010			CHPCHK!8.		
7714	030462	004737	006100		JSR	PC,TXCHAR	;LOAD 2ND 8-BIT 000 CHAR, TX 2ND FLAG
7715	030466	000000			000		
7716	030470	000010			8.		
7717	030472	004737	006100		JSR	PC,TXCHAR	;LOAD 1-BIT 000 CHAR, TX FIRST 8-BIT 000 CHAR
7718	030476	000000			000		
7719	030500	000010			8.		
7720	030502	012737	000006	002360	MOV	#6,AXNUM	;SET BYTE NO. = 6 FOR AX3
7721	030510	012737	000000	002352	MOV	#000,WAX15	;SET DATA FOR AX3-15 = 0
7722	030516	012737	000040	002354	MOV	#TXLENO,WAX16	;SET TX CHAR LENGTH = 1 FOR AX3-16
7723	030524	004737	004270		JSR	PC,WRITAX	;LOAD AX3-15,AX3-16
7724	030530	004737	006100		JSR	PC,TXCHAR	;LOAD 2-BIT 000 CHAR, TX 2ND 8-BIT 000 CHAR
7725	030534	000000			000		
7726	030536	000010			8.		
7727	030540	012737	000100	002354	MOV	#TXLEN1,WAX16	
7728	030546	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 2
7729	030552	004737	006100		JSR	PC,TXCHAR	;LOAD 3-BIT 000 CHAR, TX 1-BIT 000 CHAR
7730	030556	000000			000		
7731	030560	000001			1		
7732	030562	012737	000140	002354	MOV	#TXLEN1!TXLENO,WAX16	
7733	030570	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 3
7734	030574	004737	006100		JSR	PC,TXCHAR	;LOAD 4-BIT 000 CHAR, TX 2-BIT 000 CHAR
7735	030600	000000			000		
7736	030602	000002			2		
7737	030604	012737	000200	002354	MOV	#TXLEN2,WAX16	
7738	030612	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 4
7739	030616	004737	006100		JSR	PC,TXCHAR	;LOAD 5-BIT 000 CHAR, TX 3-BIT 000 CHAR
7740	030622	000000			000		
7741	030624	000003			3		
7742	030626	012737	000240	002354	MOV	#TXLEN2!TXLENO,WAX16	
7743	030634	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 5
7744	030640	004737	006100		JSR	PC,TXCHAR	;LOAD 6-BIT 000 CHAR, TX 4-BIT 000 CHAR
7745	030644	000000			000		
7746	030646	000004			4		
7747	030650	012737	000300	002354	MOV	#TXLEN2!TXLEN1,WAX16	
7748	030656	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 6
7749	030662	004737	006100		JSR	PC,TXCHAR	;LOAD 7-BIT 000 CHAR, TX 5-BIT 000 CHAR
7750	030666	000000			000		
7751	030670	000005			5		
7752	030672	012737	000340	002354	MOV	#TXLEN2!TXLEN1!TXLENO,WAX16	
7753	030700	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 7
7754	030704	004737	006100		JSR	PC,TXCHAR	;LOAD 8-BIT 000 CHAR, TX 6-BIT 000 CHAR
7755	030710	000000			000		
7756	030712	000006			6		
7757	030714	012737	000000	002354	MOV	#000,WAX16	
7758	030722	004737	004270		JSR	PC,WRITAX	;SET TX CHAR LENGTH = 8
7759	030726	004737	006100		JSR	PC,TXCHAR	;LOAD EOM, TX 7-BIT 000 CHAR
7760	030732	001000			TXEOM		
7761	030734	000007			7		
7762	030736	004737	006100		JSR	PC,TXCHAR	;LOAD EOM, TX 8-BIT 000 CHAR, CRC-CCITT-1 CHAR

CZDMRF.P11 03-NOV-81 10:29

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

7763 030742 001000
 7764 030744 000031
 7765 030746 004737 006100
 7766 030752 001000
 7767 030754 000010
 7768 030756 004737 006252
 7769 030762
 7770 030762
 7771 030762
 7772 030762 104401

```

TXEOM
25.
JSR PC,TXCHAR ;LOAD EOM, TX FIRST TERMINATING FLAG
TXEOM
8.
JSR PC,ENDTRN ;CLEAR TRANSMITTER
A9:
ENDTST
L10060: TRAP CSETST

```

7773
 7774
 7775
 7776
 7777

```

*****
.SBTTL TEST 29 - TXDATA BIT TEST - CHAR MODE, 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

```

7778
 7779
 7780
 7781
 7782
 7783
 7784
 7785
 7786
 7787 030764
 7788 030764
 7789 030764 004737 005516
 7790 030770 000226
 7791 030772 000011
 7792 030774 012701 003202
 7793 031000 010103
 7794 031002 012137 002400
 7795 031006 004737 005152
 7796 031012 020127 003220
 7797 031016 103771
 7798 031020 004737 005124
 7799 031024 004737 005232
 7800 031030 100020
 7801 031032 011337 031042
 7802 031036 004737 011154
 7803 031042 000000
 7804 031044 062703 000002
 7805 031050 020327 003214
 7806 031054 103766
 7807 031056 004737 003554
 7808 031062
 7809 031062
 7810 031062 104401

```

T29::
JSR PC,INITRN ;DO MASTER CLR, LOAD 2 SOM'S
SYNCH
STRIP!DDCMP
MOV #MSG1+4,R1 ;GET POINTER TO DATA
MOV R1,R3
3$: MOV (R1)+,TXWORD
JSR PC,LDTXSI ;LOAD A DATA CHAR INTO TX SILO
CMP R1,#MSG1+18. ;SEE IF ALL CHARS LOADED YET
BLO 3$ ;BR IF NOT YET
JSR PC,WAIT50 ;WAIT FOR SILO TO RIPPLE
JSR PC,STPLU ;CLOCK LU UNTIL SYNCHS ARE TX'D
CHPCHK!16.
6$: MOV (R3),8$ ;GET EXPECTED DATA CHAR
JSR PC,CKTBIT ;CHECK TXDATA FOR CHAR BITS
8$: .WORD 0 ;EXPECTED CHAR GOES HERE
ADD #2,R3 ;INCR PATTERN POINTER
CMP R3,#MSG1+14. ;SEE IF ALL CHARS CHECKED YET
BLO 6$ ;BR IF NOT YET
16$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10061: TRAP CSETST

```

7811
 7812
 7813
 7814
 7815
 7816
 7817
 7818

```

*****
.SBTTL TEST 30 - USYRT RECEIVER MSG TEST - CHAR MODE, CRC
*

```


CZDMRF.P11 03-NOV-81 10:29

TEST 30 - USYRT RECEIVER MSG TEST - CHAR MODE, CRC

```

7819
7820
7821
7822
7823
7824
7825
7826 031064
7827 031064
7828 031064 012737 031426 002340
7829 031072 004737 003554
7830 031076 004737 010616
7831 031102 000226
7832 031104 000013
7833 031106 000000
7834 031110 000000
7835 031112 012737 000012 002356
7836 031120 005037 002360
7837 031124 112737 000040 002344
7838 031132 004737 003726
7839 031136 012701 003176
7840 031142 012137 002400
7841 031146 004737 005152
7842 031152 020127 003224
7843 031156 103771
7844 031160 004737 005124
7845 031164 004737 005232
7846 031170 000050
7847 031172 004737 006672
7848 031176 000000
7849 031200 004737 005232
7850 031204 000006
7851 031206 012701 003202
7852 031212 004737 006672
7853 031216 000001
7854 031220 004737 004102
7855 031224 023721 002346
7856 031230 001415
7857 031232 016137 177776 002362
7858 031240 013737 002346 002364
7859 031246 004737 004504
7860
7861 031252
7862 031252 104455
7863 031254 000032
7864 031256 013045
7865 031260 015116
7866 031262 000461
7867 031264 004737 005232
7868 031270 000010
7869 031272 020127 003214
7870 031276 103745
7871 031300 004737 006672
7872 031304 000001
7873 031306 004737 004102
7874 031312 123727 002346 000160

```

```

:* 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

T30::

```

MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM THE USYRT
SYNCH
STRIP! IERR!DDCMP
000
000
MOV #12,REGNUM ;SET LU REG NO. = 12
CLR AXNUM ;SET AX BYTE NO. = 0 FOR AX0
MOVB #LULP,WRIBYT
JSR PC,WRITLU ;SET LULP IN REG 12
MOV #MSG1,R1 ;GET POINTER TO MSG DATA TABLE
3$: MOV (R1)+,TXWORD ;GET CHAR TO BE LOADED
JSR PC,LDTXSI ;LOAD CHAR INTO TX SILO
CMP R1,#MSG1+22. ;SEE IF ALL MSG CHARS LOADED YET
BLO 3$ ;BR IF NOT YET
JSR PC,WAIT50 ;ALLOW DATA TO RIPPLE IN SILO
JSR PC,STPLU ;CLOCK LU FOR 40 CYCLES (UNTIL FIRST
; DATA CHAR IS ABOUT TO BE RECEIVED)
40.
JSR PC,IACTIV ;CHK IACT = 0
0
JSR PC,STPLU ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
6
MOV #MSG1+4,R1
10$: JSR PC,IACTIV ;CHK IACT = 1
1
JSR PC,READAX ;READ AX0
CMP RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
BEQ 12$ ;BR IF RCV'D DATA OK
MOV -2(R1),GOODAT ;GET EXPECTED DATA
MOV RAX15,BADDAT ;GET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT INCORRECT DATA CHAR RCV'D
ERRDF 26,EM26,ERR3

```

```

TRAP C$ERDF
.WORD 26
.WORD EM26
.WORD ERR3

```

```

12$: BR 24$
JSR PC,STPLU ;CLOCK LU 8 CYCLES
8.
CMP R1,#MSG1+14. ;SEE IF CHECKING HI CRC BYTE YET
BLO 10$ ;BR IF NOT YET
JSR PC,IACTIV ;CHK IACT = 1
1
JSR PC,READAX ;READ AX0
CMPB RAX15,#160 ;CMP RCV'D CHAR TO EXPECTED HI CRC BYTE

```

CZDMRF.P11 03-NOV-81 10:29

TEST 30 - USYRT RECEIVER MSG TEST - CHAR MODE, CRC

```

7875 031320 001415          BEQ      16$          ;BR IF HI CRC BYTE RCV'D OK
7876 031322 012737 000160 002362  MOV     #160,GOODAT ;GET EXPECTED DATA
7877 031330 013737 002346 002364 14$:  MOV     RAX15,BADAT  ;SET ACTUAL DATA
7878 031336 004737 004504          JSR     PC,GETALL   ;GET REGS FOR PRINTOUT
7879          ;REPORT INCORRECT CRC BYTE RCV'D
7880          ERRDF   27,EM27,ERR3
7881 031342          TRAP   C$ERDF
7882 031342 104455          .WORD  27
7883 031344 000033          .WORD  EM27
7884 031346 013077          .WORD  ERR3
7885 031350 015116
7886 031352 000425
7887 031354 004737 005232 16$:  BR      24$
7888 031360 000010          JSR     PC,STPLU   ;CLOCK LU FOR 8 CYCLES
7889 031362 004737 006672          JSR     PC,IACTIV ;CHK IACT = 1
7890 031366 000001
7891 031370 012737 000034 002362  MOV     #034,GOODAT ;GET EXPECTED LO CRC BYTE
7892 031376 004737 004102          JSR     PC,READAX ;READ AX0
7893 031402 123727 002346 000034  CMPB   RAX15,#034  ;CMP RCV'D CHAR TO EXPECTED LO CRC BYTE
7894 031410 001347          BNE     14$        ;BR IF LO CRC INCORRECT
7895 031412 004737 005232          JSR     PC,STPLU   ;CLOCK LU 8 CYCLES
7896 031416 000010          JSR     PC,IACTIV ;CHK IACT STILL = 1
7897 031420 004737 006672
7898 031424 000001
7899 031426 004737 003554 24$:  JSR     PC,MSTCLR  ;ISSUE CLEAN-UP MASTER CLEAR
7900 031432          ENDTST
7901 031432 104401          L10062: TRAP   C$SETST

```

7902
7903
7904
7905
7906
7907

```

:*****
:SBTTL      TEST 31 - USYRT RECEIVER MSG TEST - BIT MODE, CRC
:

```

```

7910 :* THE LINE UNIT IS INITIALIZED AND A MESSAGE IS INITIATED (USING STEPLU) WITH
7911 :* LULP (REG 12) SET TO LOOP THE DATA INTERNALLY IN THE USYRT, AND WITH CRC-
7912 :* CCITT-1. TWO FLAGS, 000,125,252,377,000, AND TWO TERMINATING FLAGS ARE THEN
7913 :* SENT. THE PROGRAM MONITORS IACT, RSOM, AND THE RCV'D CHARS ARE READ
7914 :* FROM AX0-15 AND COMPARED TO EXPECTED VALUES. THE PROGRAM THEN CHECKS FOR
7915 :* IACT = 0, SETS IC TO CLEAR THE RECEIVER, AND CHECKS FOR IACT STILL = 0.
7916 :*****

```

```

7917 031434          BGNST
7918 031434
7919 031434 012737 031770 002340          MOV     #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
7920 031442 004737 003554          JSR     PC,MSTCLR  ;ISSUE MASTER CLEAR
7921 031446 004737 010616          JSR     PC,SETUP   ;PROGRAM THE USYRT
7922 031452 000000
7923 031454 000002          000
7924 031456 000000          IERR
7925 031460 000000          000
7926 031462 012737 000012 002356  MOV     #12,REGNUM  ;SET LU REG NO. = 12
7927 031470 005037 002360          CLR     AXNUM      ;SET AX BYTE NO. = 0 FOR AX0
7928 031474 112737 000040 002344  MOVB   #LULP,WRIBYT
7929 031502 004737 003726          JSR     PC,WRITLU  ;SET LULP IN REG 12
7930 031506 012701 003176          MOV     #MSG1,R1   ;GET POINTER TO MSG DATA TABLE

```


CZDMRF.P11 03-NOV-81 10:29

TEST 31 - USYRT RECEIVER MSG TEST - BIT MODE, CRC

7931	031512	012137	002400	3\$:	MOV	(R1)+,TXWORD	:GET CHAR TO BE LOADED	
7932	031516	004737	005152		JSR	PC,LDTXSI	:LOAD CHAR INTO TX SILO	
7933	031522	020127	003220		CMP	R1,#MSG1+18.	:SEE IF ALL MSG CHARS LOADED YET	
7934	031526	103771			BLO	3\$:BR IF NOT YET	
7935	031530	004737	005124		JSR	PC,WAIT50	:ALLOW DATA TO RIPPLE IN SILO	
7936	031534	004737	005232		JSR	PC,STPLU	:CLOCK LU FOR 50 CYCLES (UNTIL FIRST	
7937	031540	000062			50.		: DATA CHAR IS ABOUT TO BE RECEIVED)	
7938	031542	004737	006672		JSR	PC,IACTIV	:CHK IACT = 0	
7939	031546	000000			0			
7940	031550	004737	007060		JSR	PC,RSEOM	:CHK RSOM = 0, REOM = 0	
7941	031554	000000			0			
7942	031556	004737	005232		JSR	PC,STPLU	:CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D	
7943	031562	000006			6			
7944	031564	012701	003202		MOV	#MSG1+4,R1		
7945	031570	020127	003202	5\$:	CMP	R1,#MSG1+4	:SEE IF 1ST CHAR RCV'D	
7946	031574	001007			BNE	6\$:BR IF NO	
7947	031576	004737	006672		JSR	PC,IACTIV	:CHK IACT = 1	
7948	031602	000001			1			
7949	031604	004737	007060		JSR	PC,RSEOM	:CHK RSOM = 1, REOM = 0	
7950	031610	000001			1			
7951	031612	000420			BR	9\$		
7952	031614	020127	003212	6\$:	CMP	R1,#MSG1+12.	:SEE IF LAST CHAR RCV'D	
7953	031620	001007			BNE	8\$:BR IF NO	
7954	031622	004737	006672		JSR	PC,IACTIV	:CHK FOR IACT = 0	
7955	031626	000000			0			
7956	031630	004737	007060		JSR	PC,RSEOM	:CHK RSOM = 0, REOM = 0	
7957	031634	000000			0			
7958	031636	000406			BR	9\$		
7959	031640	004737	006672	8\$:	JSR	PC,IACTIV	:CHK FOR IACT = 1	
7960	031644	000001			1			
7961	031646	004737	007060		JSR	PC,RSEOM	:CHK RSOM = 0, REOM = 0	
7962	031652	000000			0			
7963	031654	004737	004102	9\$:	JSR	PC,READAX	:READ AX0	
7964	031660	023721	002346		CMP	RAX15,(R1)+	:COMPARE RCV'D CHAR TO EXPECTED	
7965	031664	001415			BEQ	12\$:BR IF RCV'D DATA OK	
7966	031666	016137	177776	002362	MOV	-2(R1),GOODAT	:GET EXPECTED DATA	
7967	031674	013737	002346	002364	MOV	RAX15,BADDAT	:GET ACTUAL DATA	
7968	031702	004737	004504		JSR	PC,GETALL	:GET REGS FOR PRINTOUT	
7969								
7970	031706				:REPORT	INCORRECT DATA CHAR RCV'D		
7971	031706	104455			ERRDF	26,EM26,ERR3		
7972	031710	000032					TRAP	C\$ERDF
7973	031712	013045					.WORD	26
7974	031714	015116					.WORD	EM26
7975	031716	000424					.WORD	ERR3
7976	031720	004737	005232	12\$:	BR	24\$		
7977	031724	000010			JSR	PC,STPLU	:CLOCK LU 8 CYCLES	
7978	031726	020127	003214		8.			
7979	031732	103716			CMP	R1,#MSG1+14.	:SEE IF ALL DATA CHARS CHECKED YET	
7980	031734	004737	006672		BLO	5\$:BR IF NOT YET	
7981	031740	000000			JSR	PC,IACTIV	:CHK IACT = 0	
7982	031742	004737	007060		0			
7983	031746	000000			JSR	PC,RSEOM	:CHK RSOM = 0, REOM = 0	
7984	031750	012737	000200	002344	0			
7985	031756	004737	003726		MOV	#IC,WRIBYT	:SET IC (INPUT CLEAR) IN REG 12	
7986	031762	004737	006672		JSR	PC,WRITLU	:CHK IACT = 0	
					JSR	PC,IACTIV		

CZDMRF.P11 03-NOV-81 10:29

TEST 31 - USYRT RECEIVER MSG TEST - BIT MODE, CRC

7987 031766 000000
 7988 031770 004737 003554
 7989 031774
 7990 031774
 7991 031774 104401
 7992
 7993
 7994
 7995
 7996
 7997

0
 24\$: JSR PC,MSTCLR ;ISSUE CLEAN-UP MASTER CLEAR
 ENDTST
 L10063: TRAP CSETST

7998
 7999
 8000
 8001
 8002
 8003
 8004
 8005
 8006

 .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.

8007 031776
 8008 031776
 8009 031776 012737 032240 002340
 8010 032004 004737 003554
 8011 032010 004737 010616
 8012 032014 000226
 8013 032016 000313
 8014 032020 000000
 8015 032022 000000
 8016 032024 012737 000012 002356
 8017 032032 005037 002360
 8018 032036 112737 000040 002344
 8019 032044 004737 003726
 8020 032050 012701 003176
 8021 032054 012137 002400
 8022 032060 004737 005152
 8023 032064 020127 003220
 8024 032070 103771
 8025 032072 004737 005124
 8026 032076 004737 005232
 8027 032102 000030
 8028 032104 004737 006672
 8029 032110 000000
 8030 032112 004737 005232
 8031 032116 000006
 8032 032120 012701 003202
 8033 032124 004737 006672
 8034 032130 000001
 8035 032132 004737 004102
 8036 032136 023721 002346
 8037 032142 001415
 8038 032144 016137 177776 002362
 8039 032152 013737 002346 002364
 8040 032160 004737 004504
 8041
 8042 032164

BGNTST
 T32::
 MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
 JSR PC,SETUP ;PROGRAM THE USYRT
 SYNCH
 CRC2!CRC1!STRIP!IERR!DDCMP
 000
 000
 MOV #12,REGNUM ;SET LU REG NO. = 12
 CLR AXNUM ;SET AX BYTE NO. = 0 FOR AX0
 MOVB #LULP,WRIBYT
 JSR PC,WRITLU ;SET LULP IN REG 12
 MOV #MSG1,R1 ;GET POINTER TO MSG DATA TABLE
 3\$: MOV (R1)+,TXWORD ;GET CHAR TO BE LOADED
 JSR PC,LDTXSI ;LOAD CHAR INTO TX SILO
 CMP R1,#MSG1+18. ;SEE IF ALL MSG CHARS LOADED YET
 BLO 3\$;BR IF NOT YET
 JSR PC,WAIT50 ;ALLOW DATA TO RIPPLE IN SILO
 JSR PC,STPLU ;CLOCK LU FOR 24 CYCLES (UNTIL FIRST
 24. ; DATA CHAR IS ABOUT TO BE RECEIVED)
 JSR PC,IACTIV ;CHK IACT = 0
 0
 JSR PC,STPLU ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
 6
 MOV #MSG1+4,R1
 10\$: JSR PC,IACTIV ;CHK IACT = 1
 1
 JSR PC,READAX ;READ AX0
 CMP RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
 BEQ 12\$;BR IF RCV'D DATA OK
 MOV -2(R1),GOODAT ;GET EXPECTED DATA
 MOV RAX15,BADDAT ;GET ACTUAL DATA
 JSR PC,GETALL ;GET REGS FOR PRINTOUT
 ;REPORT INCORRECT DATA CHAR RCV'D
 ERRDF 26,EM26,ERR3

CZDMRF.P11 03-NOV-81 10:29

TEST 32 - USYRT RECEIVER MSG TEST - CHAR MODE, NO CRC

TRAP C\$ERDF
.WORD 26
.WORD EM26
.WORD ERR3

8043 032164 104455
8044 032166 000032
8045 032170 013045
8046 032172 015116
8047 032174 000421
8048 032176 004737 005232
8049 032202 000010
8050 032204 020127 003214
8051 032210 103745
8052 032212 004737 006672
8053 032216 000001
8054 032220 012737 000200 002344
8055 032226 004737 003726
8056 032232 004737 006672
8057 032236 000000
8058 032240 004737 003554
8059 032244
8060 032244
8061 032244 104401
8062
8063
8064
8065
8066
8067

12\$: BR 24\$
JSR PC,STPLU ;CLOCK LU 8 CYCLES
8.
CMP R1,#MSG1+14. ;SEE IF ALL 5 DATA CHARS RCV'D YET
BLO 10\$;BR IF NOT YET
JSR PC,IACTIV ;CHK FOR IACT STILL = 1
1
MOV #IC,WRIBYT
JSR PC,WRITLU ;SET IC (INPUT CLEAR) IN REG 12
JSR PC,IACTIV ;CHK IACT = 0
0
24\$: JSR PC,MSTCLR ;ISSUE CLEAN-UP MASTER CLEAR
ENDTST

L10064: TRAP C\$SETST

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

8077 032246
8078 032246
8079 032246 012737 032602 002340
8080 032254 004737 003554
8081 032260 004737 010616
8082 032264 000000
8083 032266 000302
8084 032270 000000
8085 032272 000000
8086 032274 012737 000012 002356
8087 032302 005037 002360
8088 032306 112737 000040 002344
8089 032314 004737 003726
8090 032320 012701 003176
8091 032324 012137 002400
8092 032330 004737 005152
8093 032334 020127 003220
8094 032340 103771
8095 032342 004737 005124
8096 032346 004737 005232
8097 032352 000041
8098 032354 004737 006672

T33::
MOV #24\$,RETADR
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM THE USYRT
000
CRC2!CRC1!IERR
000
000
MOV #12,REGNUM ;SET LU REG NO. = 12
CLR AXNUM ;SET AX BYTE NO. = 0 FOR AX0
MOVB #LULP,WRIBYT
JSR PC,WRITLU ;SET LULP IN REG 12
MOV #MSG1,R1 ;GET POINTER TO MSG DATA TABLE
3\$: MOV (R1)+,TXWORD ;GET CHAR TO BE LOADED
JSR PC,LDTXSI ;LOAD CHAR INTO TX SILO
CMP R1,#MSG1+18. ;SEE IF ALL MSG CHARS LOADED YET
BLO 3\$;BR IF NOT YET
JSR PC,WAIT50 ;ALLOW DATA TO RIPPLE IN SILO
JSR PC,STPLU ;CLOCK LU FOR 33 CYCLES (UNTIL FIRST
: DATA CHAR IS ABOUT TO BE RECEIVED)
: ;
JSR PC,IACTIV ;CHK IACT = 0

CZDMRF.P11 03-NOV-81 10:29

TEST 33 - USYRT RECEIVER MSG TEST - BIT MODE, NO CRC

```

8099 032360 000000 0
8100 032362 004737 007060 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
8101 032366 000000 0
8102 032370 004737 005232 JSR PC,STPLU ;CLOCK LU UNTIL 1ST DATA CHAR IS RCV'D
8103 032374 000006 6
8104 032376 012701 003202 MOV #MSG1+4,R1
8105 032402 020127 003202 5$: CMP R1,#MSG1+4 ;SEE IF 1ST CHAR RCV'D
8106 032406 001007 BNE 6$ ;BR IF NO
8107 032410 004737 006672 JSR PC,IACTIV ;CHK IACT = 1
8108 032414 000001 1
8109 032416 004737 007060 JSR PC,RSEOM ;CHK RSOM = 1, REOM = 0
8110 032422 000001 1
8111 032424 000420 BR 9$
8112 032426 020127 003212 6$: CMP R1,#MSG1+12. ;SEE IF LAST CHAR RCV'D
8113 032432 001007 BNE 8$ ;BR IF NO
8114 032434 004737 006672 JSR PC,IACTIV ;CHK FOR IACT = 0
8115 032440 000000 0
8116 032442 004737 007060 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
8117 032446 000000 0
8118 032450 000406 BR 9$
8119 032452 004737 006672 8$: JSR PC,IACTIV ;CHK FOR IACT = 1
8120 032456 000001 1
8121 032460 004737 007060 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
8122 032464 000000 0
8123 032466 004737 004102 9$: JSR PC,READAX ;READ AX0
8124 032472 023721 002346 CMP RAX15,(R1)+ ;COMPARE RCV'D CHAR TO EXPECTED
8125 032476 001415 BEQ 12$ ;BR IF RCV'D DATA OK
8126 032500 016137 177776 002362 MOV -2(R1),GOODAT ;GET EXPECTED DATA
8127 032506 013737 002346 002364 MOV RAX15,BADDAT ;GET ACTUAL DATA
8128 032514 004737 004504 JSR PC,GETALL ;GET REGS FOR PRINTOUT
8129 ;REPORT INCORRECT DATA CHAR RCV'D
ERRDF 26,EM26,ERR3
8130 032520 104455
8131 032520 000032 TRAP C$ERDF
8132 032522 013045 .WORD 26
8133 032524 015116 .WORD EM26
8134 032526 000424 .WORD ERR3
8135 032530 004737 005232 12$: BR 24$
8136 032532 000010 JSR PC,STPLU ;CLOCK LU 8 CYCLES
8137 032536 020127 003214 8. CMP R1,#MSG1+14. ;SEE IF ALL DATA CHARS CHECKED YET
8138 032544 103716 BLO 5$ ;BR IF NOT YET
8139 032546 004737 006672 JSR PC,IACTIV ;CHK IACT = 0
8140 032552 000000 0
8141 032554 004737 007060 JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0
8142 032560 000000 0
8143 032562 012737 000200 002344 MOV #IC,WRIBYT
8144 032570 004737 003726 JSR PC,WRITLU ;SET IC (INPUT CLEAR) IN REG 12
8145 032574 004737 006672 JSR PC,IACTIV ;CHK IACT = 0
8146 032600 000000 0
8147 032602 004737 003554 24$: JSR PC,MSTCLR ;ISSUE CLEAN-UP MASTER CLEAR
8148 032606
8149 104401
8150 L10065:
8151 TRAP C$ETST
8152
8153
8154

```


CZDMRF.P11 03-NOV-81 10:29

TEST 33 - USYRT RECEIVER MSG TEST - BIT MODE, NO CRC

8155
8156
8157
8158
8159
8160
8161
8162
8163
8164
8165
8166
8167
8168
8169
8170
8171
8172
8173
8174
8175
8176
8177
8178
8179
8180
8181
8182
8183
8184
8185
8186
8187
8188
8189
8190
8191
8192
8193
8194
8195
8196
8197
8198
8199
8200
8201
8202
8203
8204
8205
8206
8207
8208
8209
8210

032610
032610
032610 004737 003554
032614 012737 000014 002356
032622 012737 000040 002344
032630 004737 003726
032634 012737 000040 002404
032642 012737 000125 002400
032650 004737 005152
032654 012737 000002 002360
032662 004737 004102
032666 123727 002346 000000
032674 001414
032676 012737 000000 002362
032704 013737 002346 002364
032712 004737 004504
032716
032716 104455
032720 000003
032722 012176
032724 015116
032726 005037 002404
032732 004737 003554
032736
032736 104401

:SBTTL 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.
:*****
BGNTST

T34::
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #14,REGNUM ;SET REG NO. = 14
MOV #DISSI,WRIBYT ;SET DISSI BIT
JSR PC,WRITLU
MOV #DISSI,DISILO ;SET DISABLE SILO FLAG
MOV #125,TXWORD ;LOAD 125 INTO TX SILO
JSR PC,LDTXSI
MOV #2,AXNUM ;SET REG NO. FOR AX1
JSR PC,READAX ;READ AX1-15, AX1-16
CMPB RAX15,#000 ;CHECK FOR AX1-15 UNCHANGED
BEQ 3\$;BR IF UNCHANGED
MOV #000,GOODAT ;GET EXPECTED DATA
MOV RAX15,BADDAT ;GET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT REG MISCMPARE
ERRDF 3,EM3,ERR3

TRAP C\$ERDF
.WORD 3
.WORD EM3
.WORD ERR3

3\$: CLR DISILO ;CLEAR DISABLE SILO FLAG
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST

L10066: TRAP C\$ETST

: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

CZDMRF.P11 03-NOV-81 10:29

TEST 35 - SILO-DISABLED MESSAGE TEST - BIT MODE, NO CRC

8211	032752	000000			0	
8212	032754	000000			0	
8213	032756	004737	003554		JSR	PC,MSTCLR ;ISSUE MASTER CLEAR
8214	032762	004737	010616		JSR	PC,SETUP ;PROGRAM THE USYRT
8215	032766	000000			000	
8216	032770	000302			CRC2!CRC1!IERR	
8217	032772	000000			000	
8218	032774	000000			000	
8219	032776	012737	000014	002356	MOV	#14,REGNUM ;SET REG NO. = 14
8220	033004	012737	000140	002344	MOV	#TXEN!DISSI,WRIBYT
8221	033012	004737	003726		JSR	PC,WRITLU ;SET TXEN AND DISSI IN REG 14
8222	033016	012737	000140	002404	MOV	#TXEN!DISSI,DISILO ;SET DISABLE SILO FLAG
8223	033024	012737	000012	002356	MOV	#12,REGNUM ;SET LU REG NO. = 12
8224	033032	112737	000040	002344	MOVB	#LULP,WRIBYT
8225	033040	004737	003726		JSR	PC,WRITLU ;SET LULP IN REG 12
8226	033044	012701	003176		MOV	#MSG1,R1 ;GET POINTER TO MSG
8227	033050	004737	004640		JSR	PC,OSIRDY ;CHK ORDY = 1
8228	033054	000001			1	
8229	033056	012737	000002	002360	MOV	#2,AXNUM ;SET AX BYTE NO. FOR AX1
8230	033064	112137	002352		MOVB	(R1)+,WAX15 ;GET A CHAR
8231	033070	112137	002354		MOVB	(R1)+,WAX16
8232	033074	004737	004270		JSR	PC,WRITAX ;LOAD CHAR INTO USYRT TX BUFFER
8233	033100	004737	004640		JSR	PC,OSIRDY ;CHK ORDY = 0
8234	033104	000000			0	
8235	033106	004737	005330		JSR	PC,OACTIV ;CHK OACT = 0
8236	033112	000000			0	
8237	033114	004737	005232		JSR	PC,STPLU ;CLOCK LU FOR 3 CYCLES
8238	033120	000003			3	
8239	033122	004737	005330		JSR	PC,OACTIV ;CHK OACT = 1
8240	033126	000001			1	
8241	033130	012703	000004		MOV	#4,R3 ;INIT COUNTER
8242	033134	112137	002352	4\$:	MOVB	(R1)+,WAX15 ;GET ANOTHER CHAR
8243	033140	112137	002354		MOVB	(R1)+,WAX16
8244	033144	020327	000001		CMP	R3,#1 ;SEE IF LOADING LAST DATA CHAR YET
8245	033150	001006			BNE	5\$;BR IF NOT
8246	033152	005737	002406		TST	CHPTYP ;SEE IF SIG USYRT
8247	033156	001403			BEQ	5\$;BR IF YES
8248	033160	112737	000002	002354	MOVB	#TEOM,WAX16 ;SET TEOM WITH LAST DATA CHAR
8249	033166	004737	004640	5\$:	JSR	PC,OSIRDY ;CHK ORDY = 1
8250	033172	000001			1	
8251	033174	004737	004270		JSR	PC,WRITAX ;LOAD ANOTHER CHAR INTO USYRT TX BUFFER
8252	033200	004737	004640		JSR	PC,OSIRDY ;CHK ORDY = 0
8253	033204	000000			0	
8254	033206	004737	006672		JSR	PC,IACTIV ;CHK IACT = 0
8255	033212	000000			0	
8256	033214	004737	007060		JSR	PC,RSEOM ;CHK RSOM = 0, REOM = 0
8257	033220	000000			0	
8258	033222	004737	005232		JSR	PC,STPLU ;CLOCK LU FOR 8 CYCLES
8259	033226	000010			8	
8260	033230	004737	005330		JSR	PC,OACTIV ;CHK OACT = 1
8261	033234	000001			1	
8262	033236	004737	004640		JSR	PC,OSIRDY ;CHK ORDY = 1
8263	033242	000001			1	
8264	033244	005303			DEC	R3 ;DECR COUNTER
8265	033246	001332			BNE	4\$;BR IF NOT DONE YET
8266	033250	004737	006406		JSR	PC,ISIRDY ;CHK IRDY = 0

CZDMRF.P11 03-NOV-81 10:29

TEST 35 - SILO-DISABLED MESSAGE TEST - BIT MODE, NO CRC

8323 033536 000002
8324 033540 004737 006406
8325 033544 000000
8326 033546 123727 002346 000252
8327 033554 001404
8328 033556 012737 000252 002362
8329 033564 000702
8330 033566 012737 000014 002356 14\$:
8331 033574 012737 000040 002344
8332 033602 004737 003726
8333 033606 012737 000011 002356
8334 033614 012737 000200 002344
8335 033622 004737 003726
8336 033626 004737 005124
8337 033632 004737 005330
8338 033636 000000
8339 033640 005037 002404 18\$:
8340 033644 004737 003554
8341 033650
8342 033650
8343 033650 104401
8344
8345
8346
8347
8348
8349
8350
8351
8352
8353
8354
8355
8356
8357
8358
8359
8360
8361
8362
8363
8364
8365
8366
8367 033652
8368 033652
8369 033652 012737 034126 002340
8370
8371
8372
8373 033660 004737 003554
8374 033664 004737 006406
8375 033670 000001
8376
8377
8378

```

2
JSR PC,ISIRDY ;CHK IRDY = 0
0
CMPB RAX15,#252 ;COMPARE 3RD RCV'D CHAR TO 252
BEQ 14$ ;BR IF MATCH
MOV #252,GOODAT ;SET EXPECTED DATA
BR 6$ ;BR TO REPORT ERROR
MOV #14,REGNUM ;SET REG NO. = 14
MOV #DISSI,WRIBYT
JSR PC,WRITLU ;CLEAR TX ENABLE
MOV #11,REGNUM ;SET REG NO. = 11
MOV #OC,WRIBYT
JSR PC,WRITLU ;SET OC TO SHUT DOWN TRANSMITTER
JSR PC,WAIT50 ;WAIT FOR SHUTDOWN
JSR PC,OACTIV ;CHK FOR OACT = 0
0
CLR DISILO ;CLEAR DISABLE SILO FLAG
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10067: TRAP CSETST

```

```

:*****
: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 USVRT, 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.
:*****

```

```

BGNTST
MOV #A10,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
-----
: DO MASTER CLR, CHK FOR ICIR = 1, IRDY = 0
-----
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,ISIRDY ;CHK ICIR = 1, IRDY = 0
1
-----
: LOAD AND CLOCK 2 SOM'S, LOAD 64 BYTES OF BINARY COUNT PATTERN INTO TX SILO,
: CLOCK LINE UNIT, CHK FOR IRDY = 1 WITHIN 40-43 CYCLES

```

CZDMRF.P11 03-NOV-81 10:29

TEST 36 - RECEIVER BUFFER TEST - CHAR MODE, CRC

```

8379
8380 033672 004737 005516      ;-----
8381 033676 000226      JSR    PC,INITRN      ;LOAD 2 SOM'S, CLOCK THEM INTO USYRT
8382 033700 000011      SYNCH
8383 033702 005003      STRIP!DDCMP
8384 033704 010337 002400      CLR    R3              ;INIT BINARY COUNT DATA FOR WRITING
8385 033710 004737 005152      2$:  MOV    R3,TXWORD
8386 033714 005203      JSR    PC,LDTXSI      ;LOAD A DATA BYTE INTO TX SILO
8387 033716 020327 000100      INC    R3              ;INCR DATA
8388 033722 002770      CMP    R3,#64.        ;SEE IF 64 BYTES LOADED YET
8389 033724 004737 007412      BLT    2$              ;BR IF NOT YET
8390 033730 000050      JSR    PC,RCV1ST      ;RECEIVE AND TIME FIRST CHARACTER
8391
8392      ;-----
8393      ; READ RCV SILO, COMPARE FIRST CHAR TO 000, CHK FOR ICIR = 1, IRDY = 0
8394 033732 005004      ;-----
8395 033734 004737 007726      9$:  CLR    R4              ;INIT PATTERN FOR READING
8396 033740 000000      JSR    PC,CKDATA      ;READ RCV SILO, COMPARE DATA
8397 033742 000000      0              ;EXPECTED DATA = 000
8398 033744 005204      16$: INC    R4              ;DON'T CLOCK LINE UNIT
8399 033746 004737 006406      JSR    PC,ISIRDY      ;INCR DATA FOR READING
8400 033752 000001      1              ;CHK FOR ICIR = 1, IRDY = 0
8401
8402      ;-----
8403      ; CLOCK 63 CHARS INTO RCV SILO, CHK ICIR = 1, IRDY = 1
8404 033754 004737 005232      18$: JSR    PC,STPLU      ;CLOCK LU FOR 8 CYCLES
8405 033760 000010      8.
8406 033762 010337 002400      MOV    R3,TXWORD
8407 033766 004737 005152      JSR    PC,LDTXSI      ;LOAD ANOTHER WORD INTO TX SILO
8408 033772 005203      INC    R3              ;INCR PATTERN FOR WRITING
8409 033774 004737 006406      JSR    PC,ISIRDY      ;CHK ICIR = 1, IRDY = 1
8410 034000 000003      3
8411 034002 020327 000177      CMP    R3,#127.       ;SEE IF 63 MORE CHARS CLOCKED YET
8412 034006 002762      BLT    18$             ;BR IF NOT YET
8413
8414      ;-----
8415      ; CLOCK 1 MORE CHAR INTO RCV SILO, CHK ICIR = 0, IRDY = 1
8416 034010 004737 005232      ;-----
8417 034014 000010      JSR    PC,STPLU      ;CLOCK LU FOR 8 CYCLES
8418 034016 004737 006406      8.
8419 034022 000002      JSR    PC,ISIRDY      ;CHK ICIR = 0, IRDY = 1
8420      2
8421
8422      ;-----
8423 034024 010437 034034      ;READ, COMPARE, CLOCK REST OF DATA CHARS
8424 034030 004737 007726      20$: MOV    R4,21$       ;SET EXPECTED DATA
8425 034034 000000      JSR    PC,CKDATA      ;READ AND COMPARE DATA
8426 034036 000000      21$: 0              ;EXPECTED SILO ENTRY GOES HERE
8427 034040 005204      0              ;DON'T CLOCK LINE UNIT
8428 034042 020427 000400      22$: INC    R4              ;INCR DATA PATTERN FOR READS
8429 034046 001427      CMP    R4,#400        ;SEE IF ALL DONE READING YET
8430 034050 004737 006406      BEQ    32$             ;BR IF DONE READING
8431 034054 000003      JSR    PC,ISIRDY      ;CHK ICIR = 1, IRDY = 1
8432 034056 004737 005232      3
8433 034062 000010      JSR    PC,STPLU      ;CLOCK LU FOR 8 CYCLES
8434 034064 020327 000377      8.
      CMP    R3,#377     ;SEE IF ALL CHARS LOADED INTO TX SILO YET

```


CZDMRF.P11 03-NOV-81 10:29

TEST 36 - RECEIVER BUFFER TEST - CHAR MODE, CRC

```

8435 034070 003007          BGT 24$          ;BR IF YES
8436 034072 010337 002400    MOV R3,TXWORD
8437 034076 004737 005152    JSR PC,LDTXSI    ;LOAD ANOTHER CHAR INTO TX SILO
8438 034102 005203          INC R3          ;INCR DATA PATTERN FOR WRITING
8439 034104 000137 034024    JMP 20$
8440 034110 012737 001000 002400 24$: MOV #TXEOM,TXWORD
8441 034116 004737 005152    JSR PC,LDTXSI    ;LOAD EOM INTO TX SILO
8442 034122 000137 034024    JMP 20$
8443 034126          32$:
8444 034126 004737 003554    A10: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
8445 034132          ENDTST
8446 034132
8447 034132 104401          L10070:
8448          TRAP C$ETST
8449
8450
8451
8452
8453
8454
8455
8456
8457
8458
8459
8460
8461
8462
8463
8464
8465
8466
8467
8468
8469
8470
8471
8472
8473
8474
8475
8476
8477
8478
8479
8480
8481
8482
8483
8484
8485
8486
8487
8488
8489
8490

```

```

:*****
.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.
:*****
BGNTST

```

```

8465 034134
8466 034134
8467 034134 012737 034326 002340    MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
8468 034142 004737 005516    JSR PC,INTRN    ;DO MASTER CLR, LOAD 2 SOM'S
8469 034146 000000
8470 034150 000341
8471 034152 012701 000017    CRC2!CRC1!IDLE!DDCMP
8472 034156 005037 002400    MOV #15.,R1    ;INIT COUNTER
8473 034162 004737 005152    CLR TXWORD
8474 034166 005301          3$: JSR PC,LDTXSI    ;LOAD A 000 CHAR INTO TX SILO
8475 034170 001374          DEC R1        ;DECR COUNTER
8476 034172 004737 005124    BNE 3$        ;BR IF NOT DONE LOADING YET
8477 034176 012737 000006 002360    JSR PC,WAIT50  ;WAIT FOR SILO TO RIPPLE
8478 034204 012737 000000 002352    MOV #6,AXNUM   ;SET BYTE NO. = 6 FOR AX3
8479 034212 012737 000005 002354    MOV #000,WAX15 ;SET DATA FOR AX3-15 = 0
8480 034220 004737 004270          JSR #RXLEN2!RXLENO,WAX16 ;SET RCV LEN = 5
8481 034224 004737 005232          JSR PC,WRITAX  ;LOAD AX3
8482 034230 100012          JSR PC,STPLU   ;CLK LU UNTIL TX'ING 1ST DATA CHAR
8483 034232 012737 000006 002354    CHPCHK!10.
8484 034240 004737 004270          MOV #RXLEN2!RXLEN1,WAX16 ;SET RCV LEN = 6
8485 034244 004737 007412          JSR PC,WRITAX  ;LOAD AX3
8486 034250 000005          JSR PC,RCV1ST ;CLOCK 5-BIT DATA CHAR
8487 034252 012737 000007 002354    MOV #RXLEN2!RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 7
8488 034260 004737 004270          JSR PC,WRITAX  ;LOAD AX3
8489 034264 004737 011052          JSR PC,RXCHAR  ;RCV 5-BIT DATA CHAR, CLK 6-BIT
8490 034270 000006          6

```


CZDMRF.P11 03-NOV-81 10:29

TEST 37 - RECEIVER CHAR LENGTH TIMING TEST - CHAR MODE, NO CRC

```

8491 034272 012737 000000 002354      MOV      #0,WAX16      ;SET RCV LEN = 8
8492 034300 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8493 034304 004737 011052              JSR      PC,RXCHAR     ;RCV 6-BIT DATA CHAR, CLK 7-BIT
8494 034310 000007                          7
8495 034312 004737 011052              JSR      PC,RXCHAR     ;RCV 7-BIT DATA CHAR, CLK 8-BIT
8496 034316 000010                          8.
8497 034320 004737 011052              JSR      PC,RXCHAR     ;RCV 8-BIT DATA CHAR
8498 034324 000010                          8.
8499 034326 004737 003554      24$:    JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR TO CLEAN UP
8500 034332      ENDTST
8501 034332
8502 034332 104401
8503
8504
8505
8506
8507
8508
8509
8510
8511
8512
8513
8514
8515
8516
8517
8518
8519
8520 034334
8521 034334
8522 034334 012737 034626 002340      MOV      #24$,RETADR   ;SET TEST EXIT ADRS FOR ERRORS
8523 034342 004737 005516              JSR      PC,INTRN     ;DO MASTER CLR, LOAD 2 SOM'S
8524 034346 000000
8525 034350 000300
8526 034352 012701 000017      CRC2!CRC1
8527 034356 005037 002400              MOV      #15,R1       ;INIT COUNTER
8528 034362 004737 005152      3$:    CLR      TXWORD
8529 034366 005301              JSR      PC,LDTXSI    ;LOAD A 000 CHAR INTO TX SILO
8530 034370 001374              DEC      R1           ;DECR COUNTER
8531 034372 004737 005124              BNE     3$           ;BR IF NOT DONE LOADING YET
8532 034376 012737 000006 002360      JSR      PC,WAIT50    ;WAIT FOR SILO TO RIPPLE
8533 034404 012737 000000 002352      MOV      #6,AXNUM     ;SET BYTE NO. = 6 FOR AX3
8534 034412 004737 007412              MOV      #000,WAX15   ;SET DATA FOR AX3-15 = 0
8535 034416 000040              JSR      PC,RCV1ST    ;CLOCK FIRST 8-BIT DATA CHAR
8536 034420 012737 000000 002354      32.    MOV      #0,WAX16     ;SET RCV LEN = 8
8537 034426 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8538 034432 004737 011052              JSR      PC,RXCHAR     ;RCV FIRST 8-BIT DATA CHAR, CLK SECOND 8-BIT
8539 034436 000010                          8.
8540 034440 012737 000007 002354      MOV      #RXLEN2!RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 7
8541 034446 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8542 034452 004737 011052              JSR      PC,RXCHAR     ;RCV SECOND 8-BIT DATA CHAR, CLK 3RD 8-BIT
8543 034456 000010                          8.
8544 034460 012737 000006 002354      MOV      #RXLEN2!RXLEN1,WAX16 ;SET RCV LEN = 6
8545 034466 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8546 034472 004737 011052              JSR      PC,RXCHAR     ;RCV 3RD 8-BIT DATA CHAR, CLK 7-BIT

```

L10071: TRAP C\$ETST

```

:*****
.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.
:*****
BGNTST

```

T38::

```

8522 034334 012737 034626 002340      MOV      #24$,RETADR   ;SET TEST EXIT ADRS FOR ERRORS
8523 034342 004737 005516              JSR      PC,INTRN     ;DO MASTER CLR, LOAD 2 SOM'S
8524 034346 000000
8525 034350 000300
8526 034352 012701 000017      CRC2!CRC1
8527 034356 005037 002400              MOV      #15,R1       ;INIT COUNTER
8528 034362 004737 005152      3$:    CLR      TXWORD
8529 034366 005301              JSR      PC,LDTXSI    ;LOAD A 000 CHAR INTO TX SILO
8530 034370 001374              DEC      R1           ;DECR COUNTER
8531 034372 004737 005124              BNE     3$           ;BR IF NOT DONE LOADING YET
8532 034376 012737 000006 002360      JSR      PC,WAIT50    ;WAIT FOR SILO TO RIPPLE
8533 034404 012737 000000 002352      MOV      #6,AXNUM     ;SET BYTE NO. = 6 FOR AX3
8534 034412 004737 007412              MOV      #000,WAX15   ;SET DATA FOR AX3-15 = 0
8535 034416 000040              JSR      PC,RCV1ST    ;CLOCK FIRST 8-BIT DATA CHAR
8536 034420 012737 000000 002354      32.    MOV      #0,WAX16     ;SET RCV LEN = 8
8537 034426 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8538 034432 004737 011052              JSR      PC,RXCHAR     ;RCV FIRST 8-BIT DATA CHAR, CLK SECOND 8-BIT
8539 034436 000010                          8.
8540 034440 012737 000007 002354      MOV      #RXLEN2!RXLEN1!RXLENO,WAX16 ;SET RCV LEN = 7
8541 034446 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8542 034452 004737 011052              JSR      PC,RXCHAR     ;RCV SECOND 8-BIT DATA CHAR, CLK 3RD 8-BIT
8543 034456 000010                          8.
8544 034460 012737 000006 002354      MOV      #RXLEN2!RXLEN1,WAX16 ;SET RCV LEN = 6
8545 034466 004737 004270              JSR      PC,WRITAX     ;LOAD AX3
8546 034472 004737 011052              JSR      PC,RXCHAR     ;RCV 3RD 8-BIT DATA CHAR, CLK 7-BIT

```

CZDMRF.P11 03-NOV-81 10:29

TEST 38 - RECEIVER CHAR LENGTH TIMING TEST - BIT MODE, NO CRC

```

8547 034476 000007          7
8548 034500 012737 000005 002354  MOV    #RXLEN2!RXLENO,WAX16    ;SET RCV LEN = 5
8549 034506 004737 004270          JSR    PC,WRITAX                ;LOAD AX3
8550 034512 004737 011052          JSR    PC,RXCHAR                ;RCV 7-BIT DATA CHAR, CLK 6-BIT
8551 034516 000006          6
8552 034520 012737 000004 002354  MOV    #RXLEN2,WAX16           ;SET RCV LEN = 4
8553 034526 004737 004270          JSR    PC,WRITAX                ;LOAD AX3
8554 034532 004737 011052          JSR    PC,RXCHAR                ;RCV 6-BIT DATA CHAR, CLK 5-BIT
8555 034536 000005          5
8556 034540 012737 000003 002354  MOV    #RXLEN1!RXLENO,WAX16    ;SET RCV LEN = 3
8557 034546 004737 004270          JSR    PC,WRITAX                ;LOAD AX3
8558 034552 004737 011052          JSR    PC,RXCHAR                ;RCV 5-BIT DATA CHAR, CLR 4-BIT
8559 034556 000004          4
8560 034560 012737 000002 002354  MOV    #RXLEN1,WAX16           ;SET RCV LEN = 2
8561 034566 004737 004270          JSR    PC,WRITAX                ;LOAD AX3
8562 034572 004737 011052          JSR    PC,RXCHAR                ;RCV 4-BIT DATA CHAR, CLK 3-BIT
8563 034576 000003          3
8564 034600 012737 000001 002354  MOV    #RXLENO,WAX16           ;SET RCV LEN = 1
8565 034606 004737 004270          JSR    PC,WRITAX                ;LOAD AX3
8566 034612 004737 011052          JSR    PC,RXCHAR                ;RCV 3-BIT DATA CHAR, CLK 2-BIT
8567 034616 000002          2
8568 034620 004737 011052          JSR    PC,RXCHAR                ;RCV 2-BIT DATA CHAR, CLK 1-BIT
8569 034624 000001          1
8570 034626 004737 003554          JSR    PC,MSTCLR                ;ISSUE MASTER CLEAR TO CLEAN UP
8571 034632
8572 034632
8573 034632 104401
8574
8575
8576
8577
8578
8579
8580
8581
8582
8583
8584
8585
8586
8587
8588 034634
8589 034634
8590 034634 012737 034716 002340  MOV    #24$,RETADR              T39::
8591 034642 004737 005516          JSR    PC,INITRN                ;MST CLR, LOAD 2 SOM'S
8592 034646 000226          SYNCH
8593 034650 000341          CRC2!CRC1!IDLE!DDCMP
8594 034652 012737 000000 002400  MOV    #000,TXWORD
8595 034660 004737 005152          JSR    PC,LDTXSI                ;LOAD 000 CHAR INTO TX SILO
8596 034664 004737 005232          JSR    PC,STPLU                 ;CLOCK LINE UNIT UNTIL LINE GOES MARKING
8597 034670 000063          51.
8598 034672 004737 007332          JSR    PC,RDRXSI                ;READ 000 CHAR
8599 034676 004737 007726          JSR    PC,CKDATA                ;READ AND CHECK FOR MARK CHAR (377)
8600 034702 000377          377
8601 034704 000000          000
8602 034706 004737 007726          JSR    PC,CKDATA                ;READ AND CHECK FOR ANOTHER MARK CHAR

```

24\$:
ENDTST

L10072:

TRAP C\$ETST

```

:*****
: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

```

T39::

CZDMRF.P11 03-NOV-81 10:29

TEST 39 - TRANSMITTER UNDERRUN ERROR, IDLE MARKING, CHAR MODE,NO CRC

8603	034712	000377	
8604	034714	000000	
8605	034716	004737	003554
8606	034722		
8607	034722		
8608	034722	104401	

```

377
000
24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10073: TRAP C$ETST

```

8609
8610
8611
8612
8613
8614
8615
8616
8617
8618
8619
8620
8621
8622
8623
8624
8625
8626
8627
8628

```

:*****
: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.
:*****

```

8629	034724		
8630	034724		
8631	034724	012737	035060 002340
8632	034732	004737	005516
8633	034736	000000	
8634	034740	000310	
8635	034742	004737	010774
8636	034746	003202	
8637	034750	000005	
8638	034752	012737	005000 002400
8639	034760	004737	005152
8640	034764	004737	005152
8641	034770	004737	005124
8642	034774	004737	005232
8643	035000	100071	
8644	035002	004737	011154
8645	035006	000376	
8646	035010	004737	007726
8647	035014	000000	
8648	035016	000000	
8649	035020	004737	007726
8650	035024	000125	
8651	035026	000000	
8652	035030	004737	007726
8653	035034	000252	
8654	035036	000000	
8655	035040	004737	007726
8656	035044	000377	
8657	035046	000010	
8658	035050	004737	007726

```

BGNTST
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS T40::
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
CRC2!CRC1!STRIP
JSR PC,LODMSG ;LOAD DATA CHARS INTO TX SILO
MSG1+4
5
MOV #TXGOA!TXEOM,TXWORD
JSR PC,LDTXSI ;LOAD A GA CHAR INTO TX SILO
JSR PC,LDTXSI ;LOAD ANOTHER GA
JSR PC,WAIT50 ;ALLOW SILO TO RIPPLE
JSR PC,STPLU ;CLOCK LU UNTIL GA CHAR IS TX'ING
CHPCHK!57.
JSR PC,CKTBIT ;SCAN TXDATA BIT FOR GA CHAR
376
JSR PC,CKDATA ;RCV 000 CHAR, CLK 125
000
0
JSR PC,CKDATA ;RCV 125 CHAR, CLK 252
125
0
JSR PC,CKDATA ;RCV 252 CHAR, CLK 377
252
0
JSR PC,CKDATA ;RCV 377 CHAR
377
8.
JSR PC,CKDATA ;RCV 000 CHAR, CHK RAB = 1, EBLK = 1

```


8659 035054 003000
 8660 035056 000010
 8661 035060 004737 003554
 8662 035064
 8663 035064
 8664 035064 104401
 8665
 8666
 8667
 8668
 8669
 8670
 8671
 8672
 8673
 8674
 8675
 8676
 8677
 8678
 8679
 8680
 8681
 8682

3000
 8.
 24\$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR
 ENDTST

L10074: TRAP CSETST

```

:*****
: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

```

8683 035066
 8684 035066
 8685 035066 012737 035226 002340
 8686 035074 004737 005516
 8687 035100 000226
 8688 035102 000341
 8689 035104 012701 000026
 8690 035110 012737 000226 002400
 8691 035116 004737 005152
 8692 035122 005301
 8693 035124 001374
 8694 035126 004737 005124
 8695 035132 004737 007412
 8696 035136 000030
 8697 035140 012701 000025
 8698 035144 004737 007726
 8699 035150 000226
 8700 035152 000010
 8701 035154 005301
 8702 035156 001372
 8703 035160 004737 007726
 8704 035164 000226
 8705 035166 000004
 8706 035170 012737 000011 002356
 8707 035176 112737 000200 002344
 8708 035204 004737 003726
 8709 035210 004737 005232
 8710 035214 000014
 8711 035216 004737 007726
 8712 035222 000377
 8713 035224 000000
 8714 035226 004737 003554

```

T41::
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
SYNCH
CRC2!CRC1!IDLE!DDCMP
MOV #22.,R1 ;INIT COUNTER
MOV #SYNCH,TXWORD
6$: JSR PC,LDTXSI ;LOAD AN SOM INTO TX SILO
DEC R1 ;DECR COUNTER
BNE 6$ ;BR IF MORE TO LOAD
JSR PC,WAIT50 ;ALLOW SILO TO RIPPLE
JSR PC,RCV1ST ;CLK LU UNTIL 3RD SYNCH RCV'D (RCVR IS ACTIVE)
24.
MOV #21.,R1 ;INIT COUNTER
8$: JSR PC,CKDATA ;READ A SYNCH, CLK NEXT ONE
SYNCH
8.
DEC R1 ;DECR COUNTER
BNE 8$ ;BR IF NOT ALL CHECKED
JSR PC,CKDATA ;CHECK LAST SYNCH
4
MOV #11,REGNUM ;SET REG NO. = 11
MOVB #OC,WRIBYT
JSR PC,WRITLU ;SET OC IN REG 11
JSR PC,STPLU ;FINISH CLOCKING CHAR, AND THEN SOME
12.
JSR PC,CKDATA ;RCV A MARK CHAR, CHK IT
377
0
24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP

```

CZDMRF.P11 03-NOV-81 10:29

TEST 41 - IDLE SYNCHS TEST - CHAR MODE

L10075: TRAP C\$ETST

8715 035232
8716 035232
8717 035232 104401

ENDTST

8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737
8738
8739

```

:*****
: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 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 (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.
:*****

```

```

8740 035234
8741 035234
8742 035234 012737 035454 002340
8743 035242 012701 035462
8744 035246 011137 035256
8745 035252 004737 005516
8746 035256 000000
8747 035260 000311
8748 035262 012737 000400 002400
8749 035270 012702 000026
8750 035274 004737 005152
8751 035300 005302
8752 035302 001374
8753 035304 004737 010774
8754 035310 003224
8755 035312 000006
8756 035314 012737 001000 002400
8757 035322 004737 005152
8758 035326 004737 005152
8759 035332 004737 005124
8760 035336 011137 035360
8761 035342 012702 000027
8762 035346 004737 005232
8763 035352 100010
8764 035354 004737 011154
8765 035360 000000
8766 035362 005302
8767 035364 001373
8768 035366 004737 007412
8769 035372 000010
8770 035374 004737 007726

```

```

BGNTST
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
MOV #SYNPAT,R1 ;GET POINTER TO DATA
2$: MOV (R1),3$ ;GET A SYNCH PATTERN
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
3$: .WORD 0 ;SYNCH PATTERN GOES HERE
CRC2!CRC1!STRIP!DDCMP
MOV #TXSOM,TXWORD
MOV #22.,R2 ;LOAD 22 SOM'S INTO TX SILO
6$: JSR PC,LDTXSI
DEC R2
BNE 6$
JSR PC,LODMSG ;LOAD DATA CHARS INTO TX SILO
MSG4
6
MOV #TXEOM,TXWORD
JSR PC,LDTXSI ;LOAD A TEOM
JSR PC,LDTXSI ;LOAD ANOTHER TEOM
JSR PC,WAIT50 ;ALLOW SILO TO RIPPLE
MOV (R1),16$ ;GET CURRENT SYNCH PATTERN
MOV #23.,R2 ;INIT COUNTER
JSR PC,STPLU ;CLOCK OUT FIRST SYNCH
CHPCHK!8.
14$: JSR PC,CKTBIT ;CHECK TX'D SYNCH
16$: .WORD 0 ;SYNCH PATTERN GOES HERE
DEC R2 ;DECR COUNTER
BNE 14$ ;BR IF NOT DONE CHECKING LAST 23 SYNCHS
JSR PC,RCV1ST ;CLOCK UNTIL 000 CHAR RCV'D
8.
JSR PC,CKDATA ;RCV 377, CLOCK 000

```

CZDMRF.P11 03-NOV-81 10:29

TEST 42 - STRIP SYNCH TEST

```

8771 035400 000377 377
8772 035402 000010 8.
8773 035404 004737 007726 JSR
8774 035410 000000 000 PC,CKDATA ;RCV 000, CLK 125
8775 035412 000010 8.
8776 035414 004737 007726 JSR
8777 035420 000125 125 PC,CKDATA ;RCV 125, CLK 252
8778 035422 000010 8.
8779 035424 004737 007726 JSR
8780 035430 000252 252 PC,CKDATA ;RCV 252, CLK END OF MSG
8781 035432 000040 32.
8782 035434 004737 005330 JSR
8783 035440 000000 0 PC,OACTIV ;CHK FOR OACT = 0
8784 035442 062701 000002 ADD #2,R1 ;INIT SYNCH PATTERN POINTER
8785 035446 020127 035476 CMP R1,#SYNPAT+12. ;SEE IF ALL PATTERNS CHECKED YET
8786 035452 103675 BLO 2$ ;BR IF NOT YET
8787 035454 004737 003554 JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
8788 035460 24$:
8789 035460 ENDTST
8790 035460 104401
8791
8792 035462 000226 SYNPAT: 226
8793 035464 000000 000
8794 035466 000125 125
8795 035470 000252 252
8796 035472 000376 376
8797 035474 000177 177
8798
8799
8800
8801
8802

```

L10076: TRAP C\$ETST

CZDMRF.P11 03-NOV-81 10:29

HARDWARE PARAMETER CODING SECTION

.SBTTL HARDWARE PARAMETER CODING SECTION

8803
8804
8805
8806
8807
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829
8830
8831
8832
8833
8834
8835
8836
8837
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847
8848
8849
8850
8851
8852
8853
8854
8855
8856

035476
035476 000011
035500
035500
035500 000031
035502 035522
035504 160000
035506 177776
035510
035510 001032
035512 035550
035514 000007
035516 000000
035520 000007
035522
035522
035522 042504 044526 042503
035530 041440 051123 040440
035536 042104 042522 051523
035544 035040 000040
035550 034115 030062 020067
035556 052522 020116 053523
035564 052111 044103 024040
035572 031105 020070 053523
035600 024467 026440 052040
035606 050131 020105 020060
035614 043111 047440 043106
035622 020054 020061 043111
035630 047440 020116 020072
035636 000
035640

:/ 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.

BGNHRD

.WORD L10077-L\$HARD/2
L\$HARD::

GPRMA ADDRES,0,0,160000,177776,YES

.WORD T\$CODE
.WORD ADDRES
.WORD T\$LOLIM
.WORD T\$HILIM

GPRMD ISRUN,2,0,7,0,7,YES

.WORD T\$CODE
.WORD ISRUN
.WORD 7
.WORD T\$LOLIM
.WORD T\$HILIM

ENDHRD

.EVEN
L10077:

ADDRES: .ASCIZ /DEVICE CSR ADDRESS : /

ISRUN: .ASCIZ /M8207 RUN SWITCH (E28 SW7) - TYPE 0 IF OFF, 1 IF ON : /

.EVEN

CZDMRF.P11 03-NOV-81 10:29

SOFTWARE PARAMETER CODING SECTION

.SBTTL SOFTWARE PARAMETER CODING SECTION

8857
8858
8859
8860
8861
8862
8863
8864
8865
8866
8867
8868
8869
8870
8871
8872
8873
8874
8875
8876
8877
8878
8879
8880
8881
8882
8883
8884
8885
8886
8887
8888
8889
8890
8891
8892
8893
8894
8895
8896
8897
8898
8899
8900
8901
8902
8903
8904
8905

035640
035640 000000
035642

035642
035642

035642 036042
036042 000240
036044 000240
036046 000240

036050
036050
036052 000000
036054 000000

000001

:/ 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

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

ABORT = 000004	2466#													
ADDRESS = 035522	4862	4878	4953	5070	5172	5297	5399	5508	5618	8822	8836#			
ADR = 000020 G	2414#													
ANBITS = 002537	2864#	6784	6785	6851	6852	6914	6915							
APA = 000200	2644#													
ASBCO = 000020	2601#													
ASBC1 = 000040	2600#													
ASBC2 = 000100	2599#													
ASSEMB = 000010	2098													
AXNUM = 002360	2785#	3443	3481	3494	3519	3521	3527*	3533*	3534	3536*	3538	3753	3761*	
	3804*	4028	4034*	4079*	4085*	4366	4368*	4376*	4385*	6598*	6616*	6620*	6636	
	6654*	6671*	6702*	6709*	6711	6729*	6781*	6802*	6848*	6869*	6911*	6932*	6987*	
	7011*	7057*	7161*	7178*	7294*	7651*	7720*	7836*	7927*	8017*	8087*	8173*	8229*	
	8285*	8477*	8532*											
AX0.15 = 002300	2714#	3526	5028	5137	5255	5364	5466	5583						
AX0.16 = 002302	2715#	5027	5136	5254	5363	5465	5582							
AX1 = 000001	2497#	2569#												
AX1.15 = 002304	2716#	5026	5135	5253	5362	5464	5581							
AX1.16 = 002306	2717#	5025	5134	5252	5361	5463	5580							
AX2 = 000002	2496#	2568#												
AX2.15 = 002310	2718#	5045	5154	5272	5381	5483	5600							
AX2.16 = 002312	2719#	5044	5153	5271	5380	5482	5599							
AX3.15 = 002314	2720#	5043	5152	5270	5379	5481	5598							
AX3.16 = 002316	2721#	5042	5151	5269	5378	5480	5597							
AX315U = 000372	2664#	6638	6713	6994										
A1 = 024144	6484	6519	6547	6556	6575#									
A10 = 034126	8369	8444#												
A2 = 026552	7144	7197#												
A3 = 027232	7227	7343	7347#											
A4 = 027346	7372	7398#												
A5 = 027456	7424	7447#												
A6 = 027652	7475	7514#												
A7 = 030216	7543	7563	7576	7592	7615	7619	7627#							
A8 = 030432	7647	7686#												
A9 = 030762	7707	7769#												
BADDAT = 002364	2787#	4216*	4217*	4246*	4247*	4899	4974	5201	5529	5971*	6013*	6053*	6094*	
	6137*	6186*	6247*	6293*	6337*	6381*	6425*	6504*	6532*	6560*	6643*	6660*	6718*	
	6734*	6791*	6807*	6858*	6873*	6921*	6936*	7000*	7016*	7076*	7109*	7167*	7183*	
	7299*	7328*	7858*	7877*	7967*	8039*	8127*	8178*	8289*					
BCC = 000001	2545#	4250	4252	4262										
BCCCHK = 100000	2729#	4248												
BIT0 = 000001 G	2387#	2447	2459	2468	2497	2509	2521	2533	2545	2557	2569	2581	2593	
	2605	2617	2626	2638	2651	2663	2674	2746	3521	3562	3707	3939	3988	
	4036	4460	5774											
BIT00 = 000001 G	2376#	2387												
BIT01 = 000002 G	2375#	2386												
BIT02 = 000004 G	2374#	2385												
BIT03 = 000010 G	2373#	2384												
BIT04 = 000020 G	2372#	2383												
BIT05 = 000040 G	2371#	2382												
BIT06 = 000100 G	2370#	2381												
BIT07 = 000200 G	2369#	2380												
BIT08 = 000400 G	2368#	2379												
BIT09 = 001000 G	2367#	2378												
BIT1 = 000002 G	2386#	2446	2458	2467	2486	2496	2508	2520	2532	2544	2556	2568	2580	
	2592	2604	2616	2625	2637	2650	2662	2673	2747	3586	3915	4058		

CZDMRF .P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

BIT10 = 002000 G	2366#	2684	2696											
BIT11 = 004000 G	2365#	2683	2695											
BIT12 = 010000 G	2364#													
BIT13 = 020000 G	2363#													
BIT14 = 040000 G	2362#													
BIT15 = 100000 G	2361#	2727	2729	2730	3675	3837								
BIT2 = 000004 G	2385#	2445	2457	2466	2485	2495	2507	2519	2531	2543	2555	2567	2579	
	2591	2603	2615	2624	2636	2649	2661	2672						
BIT3 = 000010 G	2384#	2444	2456	2465	2484	2494	2506	2518	2530	2542	2554	2566	2578	
	2590	2602	2614	2623	2635	2648	2660							
BIT4 = 000020 G	2383#	2443	2455	2483	2493	2505	2517	2529	2541	2553	2565	2577	2589	
	2601	2613	2634	2647	2659									
BIT5 = 000040 G	2382#	2454	2475	2482	2492	2504	2516	2528	2540	2552	2564	2576	2588	
	2600	2612	2633	2646	2658	2671								
BIT6 = 000100 G	2381#	2442	2453	2474	2481	2491	2503	2515	2527	2539	2551	2563	2575	
	2587	2599	2611	2632	2645	2657	2670							
BIT7 = 000200 G	2380#	2441	2452	2464	2473	2480	2502	2514	2526	2538	2550	2562	2574	
	2586	2598	2610	2622	2631	2644	2656	2669	4210	4211				
BIT8 = 000400 G	2379#	2686	2698											
BIT9 = 001000 G	2378#	2685	2697											
BOE = 000400 G	2418#													
BPOLL = 000100	2474#													
BSEL1 002426	2806#	3307*	3309*	3310*	3324*	3325*	3330*	3422*	3424*	3525*	3679*	3680*	3682*	
	3780*	3781*	3786*	3799*	5792*	5793*	5794	6083*	6981*					
BSEL2 002430	2807#	5794*	5795*	6497	6526	6554								
BSEL4 002432	2808#	3360	3381*	6129*										
CARR = 000001	2557#													
CHPCHK= 100000	2727#	7378	7430	7484	7549	7602	7657	7713	7800	8482	8643	8763		
CHPTYP 002406	2796#	3676	3794*	3798*	3838	5731*	8246	8268						
CKDATA 007726	4201#	8395	8424	8599	8602	8646	8649	8652	8655	8658	8698	8703	8711	
	8770	8773	8776	8779										
CKTBIT 011154	4456#	7802	8644	8764										
CRCCHK= 100000	2730#	4519												
CRCTY0= 000001	2651#													
CRCTY1= 000002	2650#													
CRCTY2= 000004	2649#													
CRC1 = 000100	2503#	7478	8013	8083	8216	8470	8525	8593	8634	8688	8747			
CRC2 = 000200	2502#	7478	8013	8083	8216	8470	8525	8593	8634	8688	8747			
CS = 000004	2555#													
C\$AU = 000052	2098#	5908												
C\$AUTO= 000061	2098#	5836												
C\$BRK = 000022	2098#													
C\$BSEG= 000004	2098#	6004	6285	6329	6373	6417	6780	6847	6910	6986	7066			
C\$BSUB= 000002	2098#	7063	7100	7147										
C\$CEFG= 000045	2098#													
C\$CLCK= 000062	2098#													
C\$CLEA= 000012	2098#	5855												
C\$CLOS= 000035	2098#													
C\$CLP1= 000006	2098#													
C\$CVEC= 000036	2098#													
C\$DCLN= 000044	2098#													
C\$DODU= 000051	2098#	5831												
C\$DRPT= 000024	2098#													
C\$DU = 000053	2098#	5883												
C\$EDIT= 000003	2098#	2180												
C\$ERDF= 000055	2098#	3569	3579	3593	3603	3714	3724	3886	3922	3932	3946	3956	3995	

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

FMT6	011571	5584	5651	5011	5046	5120	5155	5238	5273	5347	5382	5449	5484	5566
		4556#	4936											
		5601	5668											
FMT7	011624	4561#	4893	5085	5414	5523	5633							
FMT8	011634	4563#	4968	5195	5312									
FMT9	011670	4568#	4926	5001	5036	5110	5145	5228	5263	5337	5372	5439	5474	5556
		5591	5658											
FRSTIM	002370	2789#	5734	5741*										
F\$AU =	000015	2098#	5905	5907										
F\$AUTO=	000020	2098#	5818	5835										
F\$BGN =	000040	2098#	2104	4859	4875	4950	5060	5169	5287	5396	5498	5615	5690	5708
		5726	5818	5850	5869	5905	5929	5949	5963	5981	5998	6004	6029	6042
		6063	6076	6104	6122	6147	6166	6195	6202	6219	6256	6265	6279	6285
		6309	6323	6329	6353	6367	6373	6397	6411	6417	6441	6470	6578	6596
		6614	6634	6652	6669	6676	6694	6727	6743	6752	6775	6780	6800	6823
		6842	6847	6867	6889	6905	6910	6930	6952	6979	6986	7009	7033	7052
		7062	7066	7093	7099	7119	7122	7142	7146	7176	7192	7199	7202	7226
		7308	7337	7349	7371	7401	7423	7450	7474	7517	7539	7629	7646	7688
		7706	7771	7788	7809	7827	7900	7918	7990	8008	8060	8078	8150	8165
		8189	8208	8342	8368	8446	8466	8501	8521	8572	8589	8607	8630	8663
		8684	8716	8741	8789	8817	8870	8898						
		2098#	5850	5854										
F\$CLEA=	000007	2098#	5869	5882										
F\$DU =	000016	2098#	2104	4871	4944	5054	5163	5281	5390	5492	5609	5676	5695	5805
F\$END =	000041	2098#	5837	5856	5884	5909	5929	5949	5951	5963	5981	5983	5998	6025
		6031	6042	6063	6065	6076	6104	6106	6122	6147	6149	6166	6195	6202
		6204	6219	6256	6265	6267	6279	6305	6309	6311	6323	6349	6353	6355
		6367	6393	6397	6399	6411	6437	6441	6443	6470	6578	6580	6596	6614
		6634	6652	6669	6676	6678	6694	6727	6743	6752	6754	6775	6800	6819
		6823	6825	6842	6867	6885	6889	6891	6905	6930	6948	6952	6954	6979
		7009	7028	7033	7035	7052	7062	7088	7093	7095	7099	7119	7121	7122
		7124	7142	7146	7176	7192	7199	7201	7202	7204	7226	7308	7337	7349
		7351	7371	7401	7403	7423	7450	7452	7474	7517	7519	7539	7629	7631
		7646	7688	7690	7706	7771	7773	7788	7809	7811	7827	7900	7902	7918
		7990	7992	8008	8060	8062	8078	8150	8152	8165	8189	8191	8208	8342
		8344	8368	8446	8448	8466	8501	8503	8521	8572	8574	8589	8607	8609
		8630	8663	8665	8684	8716	8718	8741	8789	8791	8835	8877	8898	
		2098#	8817	8833										
F\$HARD=	000004	2098#	2303	2311										
F\$HW =	000013	2098#	5726	5803										
F\$INIT=	000006	2098#												
F\$JMP =	000050	2098#												
F\$MOD =	000000	2098#	2104	8898										
F\$MSG =	000011	2098#	4859	4869	4875	4942	4950	5052	5060	5161	5169	5279	5287	5388
		5396	5490	5498	5607	5615	5674							
		2098#	5708	5713										
F\$PROT=	000021	2098#												
F\$PWR =	000017	2098#												
F\$RPT =	000012	2098#	5690	5693										
F\$SEG =	000003	2098#	6004	6023	6285	6303	6329	6347	6373	6391	6417	6435	6780	6817
		6847	6883	6910	6946	6986	7026	7066	7086					
		2098#	8870	8875										
F\$SOFT=	000005	2098#												
F\$SRV =	000010	2098#												
F\$SUB =	000002	2098#	7063	7093	7100	7119	7147	7199						
F\$SW =	000014	2098#	2325	2331										
F\$TEST=	000001	2098#	5930	5949	5964	5981	5999	6029	6043	6063	6077	6104	6123	6147
		6167	6202	6220	6265	6280	6309	6324	6353	6368	6397	6412	6441	6471
		6578	6597	6676	6695	6752	6776	6823	6843	6889	6906	6952	6980	7033

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

L\$PROT	021066	G	2213	5708#				
L\$PRT	002112	G	2212#					
L\$REPP	002062	G	2188#					
L\$REV	002010	G	2144#					
L\$RPT	021064	G	5690#					
L\$SOFT	035642	G	8870	8871#				
L\$SPC	002056	G	2184#					
L\$SPCP	002020	G	2154#					
L\$SPTP	002024	G	2158#					
L\$STA	002030	G	2162#					
L\$SW	002260	G	2325	2326#				
L\$TEST	002114	G	2214#					
L\$TIML	002014	G	2150#					
L\$UNIT	002012	G	2148#	5779				
L10000	002256		2303	2311#				
L10001	002260		2325	2331#				
L10002	014606		4869#					
L10003	015114		4942#					
L10004	015576		5052#					
L10005	016254		5161#					
L10006	016766		5279#					
L10007	017444		5388#					
L10010	020076		5490#					
L10011	020604		5607#					
L10012	021062		5674#					
L10013	021064		5693#					
L10015	021410		5803#					
L10016	021470		5835#					
L10017	021472		5854#					
L10020	021522		5882#					
L10021	021556		5907#					
L10022	021640		5949#					
L10023	021724		5981#					
L10024	022050		6029#					
L10025	022152		6063#					
L10026	022300		6104#					
L10027	022420		6147#					
L10030	022574		6196	6202#				
L10031	023030		6257	6265#				
L10032	023146		6309#					
L10033	023264		6353#					
L10034	023402		6397#					
L10035	023520		6441#					
L10036	024150		6578#					
L10037	024502		6615	6635	6653	6670	6676#	
L10040	024772		6728	6744	6752#			
L10041	025202		6823#					
L10042	025406		6889#					
L10043	025612		6952#					
L10044	026050		7033#					
L10045	026302		7122#					
L10046	026206		7093#					
L10047	026300		7119#					
L10050	026554		7202#					
L10051	026552		7177	7193	7199#			
L10052	027232		7309	7338	7349#			

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

OSBGNR=	000000	2098#	2188																		
OSBGNS=	000000	2098#	2154																		
OSDU =	000001	2098#	2127#	2196																	
OSERRT=	000000	2098#	2204																		
OSGNSW=	000000	2098#	2158																		
OSPOIN=	000001	2098#	2127#	2220																	
OSSETU=	000000	2098#	2148	8901																	
PATA	002547	2875#	6001																		
PATB	002573	2898#	6026	6230																	
PATC	002603	2909#	6282																		
PATCH	035642	8888#																			
PATD	002606	2915#	6306	6326																	
PATE	002611	2921#	6350	6370																	
PATF	002617	2930#	6394	6414																	
PATG	002622	2936#	6169	6438																	
PATH	002632	2947#	6173	6599																	
PATI	002642	2958#	6617	6621	6696	6803	6806	7105	7108												
PATJ	002652	2969#	6673	6699	6703	6704															
PATK	002662	2980#	6777	6907																	
PATL	002772	3054#	6820	6844	6949																
PATM	003000	3063#	5967	5970	6049	6052	6090	6093	6133	6136	6177	6222	6886								
PATN	003010	3074#	6199	7143																	
PATO	003026	3091#	7054	7195																	
PATP	003044	3108#	7055	7090																	
PATU	003062	3125#	6983																		
PATV	003130	3165#	6982	7030																	
PNT =	001000	G	2419#																		
POLL =	000200		2480#																		
PRI =	002000	G	2420#																		
PRIOR	002326		2771#																		
PRI00 =	000000	G	2408#																		
PRI01 =	000040	G	2407#																		
PRI02 =	000100	G	2406#																		
PRI03 =	00C140	G	2405#																		
PRI04 =	000200	G	2404#																		
PRI05 =	000240	G	2403#																		
PRI06 =	000300	G	2402#																		
PRI07 =	000340	G	2401#	5821	5824	5932	5935														
PSTACK	002324		2770#	3608	3729	3961	4010	4080	4348	4491	5728*										
RAB =	000004		2543#	4296	4298	4308															
RABT =	000004		2603#																		
RAX15	002346		2780#	3458*	3459*	3529	6638*	6639	6643	6713*	6714	6718	6788	6791	6855						
			6858	6918	6921	6994*	6995*	6996	7000	7163	7167	7296	7299	7325	7328						
			7855	7858	7874	7877	7892	7964	7967	8036	8039	8124	8127	8175	8178						
			8286	8289	8310	8326															
RAX16	002350		2781#	3462*	3463*	3531	4038	4048	4060	4070	6656	6660	6730	6734	6803						
			6807	6870	6873	6933	6936	7012	7016	7070	7076	7105	7109	7179	7183						
			3237#	4515																	
RCVBUF	003240		4120#	8389	8485	8534	8695	8768													
RCV1ST	007412		2507#																		
RDALL =	000004		2493#	2565#	3445																
RDAX =	000020		4096#	4207	4427	8598															
RDRXSI	007332		3439#	3528	4035	6622	6710	6787	6854	6917	6991	7069	7104	7162	7295						
READAX	004102		7324	7854	7873	7891	7963	8035	8123	8174											
READLU	003650		3348#	3399	3449	3457	3461	3500	3561	3585	3706	3783	3795	3880	3914						
			3938	3987	4098	4102	4141	4219	4464	5966	6008	6048	6088	6132	6180						

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

RX1 = 000002	2520#	2592#											
RX2 = 000004	2519#	2591#											
RX3 = 000010	2518#	2590#											
RX4 = 000020	2517#	2589#											
RX5 = 000040	2516#	2588#											
RX6 = 000100	2515#	2587#											
RX7 = 000200	2514#	2586#											
R14NRW 002536	2861#	6006	6009										
SAVE4 002372	2790#	5736*	5739	5832	5946								
SAVE6 002374	2791#	5737*	5740	5833	5947								
SAVLEN 002410	2797#	3336*	4208	4381*	5733*								
SCRACH 002320	2768#												
SEC = 000020	2647#												
SECA = 000020	2505#												
SEIFR = 000040	2482#												
SELSBY= 000002	2486#												
SEL4 002432	2809#	5796*	5797*										
SEL6 002434	2810#	3308*	3423*	5798*	5799*								
SETUP 010616	4366#	7830	7921	8011	8081	8214							
SFPTBL 002260	2327#												
SIGQ = 000100	2575#												
SIGR = 000200	2574#												
SOM = 000001	2468#												
STALL 005142	3637#	3681	3683	3782	3787	3800							
STARES 002422	2802#	5768*	5773*	6475									
STARST 021234	5747	5753	5767#										
STBY = 000002	2556#												
STEPLU= 000020	2443#	3680	3682	3781	3786	3799							
STEPMP= 000001	2447#	3309	3310										
STPCLK 003516	3306#	3358	3382	6130									
STPERR 007624	4173#												
STPLU 005232	3671#	3847	4137	4180	4344	4434	4462	7248	7280	7322	7500	7550	7564
	7580	7603	7799	7845	7849	7867	7886	7894	7936	7942	7976	8026	8030
	8048	8096	8102	8136	8237	8258	8273	8298	8314	8404	8416	8432	8'81
	8596	8642	8709	8762									
STR = 000040	2646#												
STRIP = 000010	2506#	7375	7478	7546	7599	7791	7832	8013	8382	8634	8747		
SUBRPC 002330	2772#	3555	3556	3558*	3559*	3612*	3700	3701	3703*	3704*	3733*	3754*	3755*
	3807*	3827*	3828*	3854*	3869*	3870*	3890*	3908	3909	3911*	3912*	3965*	3981
	3982	3984*	3985*	4014*	4029	4030	4032*	4033*	4084*	4123*	4124*	4160*	4203*
	4204*	4354*	4386*	4425*	4426*	4441*	4458*	4459*	4497*	5062	5289	5500	5729*
SVCGBL= 000000	2098#	2104	2111#	2135	2144	2146	2148	2150	2152	2154	2156	2158	2160
	2162	2164	2166	2168	2170	2172	2174	2176	2178	2181	2184	2186	2188
	2190	2192	2194	2196	2198	2200	2202	2204	2206	2208	2210	2212	2214
	2216	2218	2241	2304	2305	2326	2327	3260	3268	4859	4875	4950	5060
	5169	5287	5396	5498	5615	5690	5708	5726	5818	5850	5869	5905	8818
	8871	8903#	8904										
SVCINS= 000001	2098#	2108#	2136	2137	2138	2139	2140	2141	2142	2143	2145	2147	2149
	2151	2153	2155	2157	2159	2161	2163	2165	2167	2169	2171	2173	2175
	2177	2179	2180	2182	2183	2185	2187	2189	2191	2193	2195	2197	2199
	2201	2203	2205	2207	2209	2211	2213	2215	2217	2219	2240	2242	2243
	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255	2256
	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269
	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282
	2283	2303	2325	3261	3262	3269	3276	3569	3570	3571	3572	3579	3580
	3581	3582	3593	3594	3595	3596	3603	3604	3605	3606	3714	3715	3716

CZDMRF.P11

03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

3717	3724	3725	3726	3727	3886	3887	3888	3889	3922	3923	3924	3925
3932	3933	3934	3935	3946	3947	3948	3949	3956	3957	3958	3959	3995
3996	3997	3998	4005	4006	4007	4008	4043	4044	4045	4046	4053	4054
4055	4056	4065	4066	4067	4068	4075	4076	4077	4078	4225	4226	4227
4228	4234	4235	4236	4237	4257	4258	4259	4260	4267	4268	4269	4270
4280	4281	4282	4283	4290	4291	4292	4293	4303	4304	4305	4306	4313
4314	4315	4316	4326	4327	4328	4329	4336	4337	4338	4339	4472	4473
4474	4475	4482	4483	4484	4485	4861	4862	4863	4864	4865	4866	4867
4870	4877	4878	4879	4880	4881	4882	4883	4885	4886	4887	4888	4889
4891	4892	4893	4894	4895	4896	4897	4899	4900	4901	4902	4903	4904
4905	4907	4908	4909	4910	4911	4912	4913	4915	4916	4917	4918	4919
4920	4921	4922	4923	4925	4926	4927	4928	4929	4930	4932	4933	4934
4935	4936	4937	4938	4939	4940	4943	4952	4953	4954	4955	4956	4957
4958	4960	4961	4962	4963	4964	4966	4967	4968	4969	4970	4971	4972
4974	4975	4976	4977	4978	4979	4980	4982	4983	4984	4985	4986	4987
4988	4990	4991	4992	4993	4994	4995	4996	4997	4998	5000	5001	5002
5003	5004	5005	5007	5008	5009	5010	5011	5012	5013	5014	5015	5017
5018	5019	5020	5021	5022	5023	5025	5026	5027	5028	5029	5030	5031
5032	5033	5035	5036	5037	5038	5039	5040	5042	5043	5044	5045	5046
5047	5048	5049	5050	5053	5062	5063	5064	5065	5066	5067	5069	5070
5071	5072	5073	5074	5075	5077	5078	5079	5080	5081	5083	5084	5085
5086	5087	5088	5089	5091	5092	5093	5094	5095	5096	5097	5099	5100
5101	5102	5103	5104	5105	5106	5107	5109	5110	5111	5112	5113	5114
5116	5117	5118	5119	5120	5121	5122	5123	5124	5126	5127	5128	5129
5130	5131	5132	5134	5135	5136	5137	5138	5139	5140	5141	5142	5144
5145	5146	5147	5148	5149	5151	5152	5153	5154	5155	5156	5157	5158
5159	5162	5171	5172	5173	5174	5175	5176	5177	5179	5180	5181	5182
5183	5184	5185	5187	5188	5189	5190	5191	5193	5194	5195	5196	5197
5198	5199	5201	5202	5203	5204	5205	5206	5207	5209	5210	5211	5212
5213	5214	5215	5217	5218	5219	5220	5221	5222	5223	5224	5225	5227
5228	5229	5230	5231	5232	5234	5235	5236	5237	5238	5239	5240	5241
5242	5244	5245	5246	5247	5248	5249	5250	5252	5253	5254	5255	5256
5257	5258	5259	5260	5262	5263	5264	5265	5266	5267	5269	5270	5271
5272	5273	5274	5275	5276	5277	5280	5289	5290	5291	5292	5293	5294
5296	5297	5298	5299	5300	5301	5302	5304	5305	5306	5307	5308	5310
5311	5312	5313	5314	5315	5316	5318	5319	5320	5321	5322	5323	5324
5326	5327	5328	5329	5330	5331	5332	5333	5334	5336	5337	5338	5339
5340	5341	5343	5344	5345	5346	5347	5348	5349	5350	5351	5353	5354
5355	5356	5357	5358	5359	5361	5362	5363	5364	5365	5366	5367	5368
5369	5371	5372	5373	5374	5375	5376	5378	5379	5380	5381	5382	5383
5384	5385	5386	5389	5398	5399	5400	5401	5402	5403	5404	5406	5407
5408	5409	5410	5412	5413	5414	5415	5416	5417	5418	5420	5421	5422
5423	5424	5425	5426	5428	5429	5430	5431	5432	5433	5434	5435	5436
5438	5439	5440	5441	5442	5443	5445	5446	5447	5448	5449	5450	5451
5452	5453	5455	5456	5457	5458	5459	5460	5461	5463	5464	5465	5466
5467	5468	5469	5470	5471	5473	5474	5475	5476	5477	5478	5480	5481
5482	5483	5484	5485	5486	5487	5488	5491	5500	5501	5502	5503	5504
5505	5507	5508	5509	5510	5511	5512	5513	5515	5516	5517	5518	5519
5521	5522	5523	5524	5525	5526	5527	5529	5530	5531	5532	5533	5534
5535	5537	5538	5539	5540	5541	5542	5543	5545	5546	5547	5548	5549
5550	5551	5552	5553	5555	5556	5557	5558	5559	5560	5562	5563	5564
5565	5566	5567	5568	5569	5570	5572	5573	5574	5575	5576	5577	5578
5580	5581	5582	5583	5584	5585	5586	5587	5588	5590	5591	5592	5593
5594	5595	5597	5598	5599	5600	5601	5602	5603	5604	5605	5608	5617
5618	5619	5620	5621	5622	5623	5625	5626	5627	5628	5629	5631	5632
5633	5634	5635	5636	5637	5639	5640	5641	5642	5643	5644	5645	5647

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

SSLSYM= 010000	2098#	2312#	2332#	4870#	4943#	5053#	5162#	5280#	5389#	5491#	5608#	5675#	5694#
	5804#	5836#	5855#	5883#	5908#	5950#	5982#	6004#	6030#	6064#	6105#	6148#	6203#
	6266#	6285#	6310#	6329#	6354#	6373#	6398#	6417#	6442#	6579#	6677#	6753#	6780#
	6824#	6847#	6890#	6910#	6953#	6986#	7034#	7066#	7094#	7120#	7123#	7200#	7203#
	7350#	7402#	7451#	7518#	7630#	7689#	7772#	7810#	7901#	7991#	8061#	8151#	8190#
	8343#	8447#	8502#	8573#	8608#	8664#	8717#	8790#	8835#	8877#			
TEOM = 000002	2625#	8248	8271										
TERR = 000200	2622#												
TEST = 000001	2663#	6992											
TESTMD= 000004	2579#												
TIMFLG 002336	2776#												
TMP0 002506	2841#	3520*	3523*	4966	5193	5310							
TMP1 002510	2842#	3538*	3539*	4967	5194	5311							
TMP2 002512	2843#												
TMP3 002514	2844#												
TMP4 002516	2845#												
TMP5 002520	2846#												
TMP6 002522	2847#												
TMP7 002524	2848#												
TSOM = 000001	2626#												
TSTCON 002452	2817#												
TSTNUM 002420	2801#	6472*	6478										
TXAB = 000004	2624#												
TXABT = 002000	2684#												
TXCHAR 006100	3825#	7376	7379	7382	7385	7388	7391	7394	7428	7431	7434	7437	7440
	7443	7482	7485	7488	7491	7494	7497	7547	7600	7655	7658	7663	7668
	7673	7676	7679	7682	7711	7714	7717	7724	7729	7734	7739	7744	7749
	7754	7759	7762	7765									
TXDATA= 000040	2576#	4467	4477										
TXEN = 000100	2491#	2563#	6089	7480	8220	8222							
TXEOM = 001000	2685#	3220	3221	3222	3223	3229	3230	7383	7386	7389	7392	7395	7435
	7438	7441	7444	7489	7492	7495	7498	7674	7677	7680	7683	7760	7763
	7766	8440	8638	8756									
TXGA = 000010	2623#												
TXGOA = 004000	2683#	8638											
TXLENO= 000040	2671#	4208	7653	7666	7722	7732	7742	7752					
TXLEN1= 000100	2670#	4208	7661	7666	7727	7732	7747	7752					
TXLEN2= 000200	2669#	4208	7653	7661	7666	7737	7742	7747	7752				
TXSOM = 000400	2686#	3213	3214	3763	7237	7577	8748						
TXWORD 002400	2793#	3651*	3653	3656	3763*	3764*	3829*	4404*	7237*	7262*	7316*	7577*	7794*
	7840*	7931*	8021*	8091*	8171*	8384*	8406*	8436*	8440*	8472*	8527*	8594*	8638*
	8690*	8748*	8756*										
TX0 = 000001	2459#	2617#											
TX1 = 000002	2458#	2616#											
TX2 = 000004	2457#	2615#											
TX3 = 000010	2456#	2614#											
TX4 = 000020	2455#	2613#											
TX5 = 000040	2454#	2612#											
TX6 = 000100	2453#	2611#											
TX7 = 000200	2452#	2610#											
TSARGC= 000001	2136#	2137#	2138#	2139#	2140#	2141#	4861#	4867	4877#	4883	4885#	4889	4891#
	4897	4899#	4905	4907#	4913	4915#	4923	4925#	4930	4932#	4940	4952#	4958
	4960#	4964	4966#	4972	4974#	4980	4982#	4988	4990#	4998	5000#	5005	5007#
	5015	5017#	5023	5025#	5033	5035#	5040	5042#	5050	5062#	5067	5069#	5075
	5077#	5081	5083#	5089	5091#	5097	5099#	5107	5109#	5114	5116#	5124	5126#
	5132	5134#	5142	5144#	5149	5151#	5159	5171#	5177	5179#	5185	5187#	5191

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- USER SYMBOLS

T\$SAVL= 177777	2098#												
T\$SEGL= 177777	2098#	6004#	6023#	6025	6285#	6303#	6305	6329#	6347#	6349	6373#	6391#	6393
	6417#	6435#	6437	6780#	6801	6817#	6819	6847#	6868	6883#	6885	6910#	6931
	6946#	6948	6986#	7010	7026#	7028	7066#	7086#	7088				
T\$SEKO= 010000	6004#	6023	6285#	6303	6329#	6347	6373#	6391	6417#	6435	6780#	6801	6817
	6847#	6868	6883	6910#	6931	6946	6986#	7010	7026	7066#	7086		
T\$SUBN= 000000	2098#	5929#	5963#	5998#	6042#	6076#	6122#	6166#	6219#	6279#	6323#	6367#	6411#
	6470#	6596#	6694#	6775#	6842#	6905#	6979#	7052#	7062#	7099#	7142#	7146#	7226#
	7371#	7423#	7474#	7539#	7646#	7706#	7788#	7827#	7918#	8008#	8078#	8165#	8208#
	8368#	8466#	8521#	8589#	8630#	8684#	8741#						
T\$TAGL= 177777	2098#												
T\$TAGN= 010101	2098#	2303#	2325#	4859#	4875#	4950#	5060#	5169#	5287#	5396#	5498#	5615#	5690#
	5708#	5726#	5818#	5850#	5869#	5905#	5930#	5964#	5999#	6043#	6077#	6123#	6167#
	6220#	6280#	6324#	6368#	6412#	6471#	6597#	6695#	6776#	6843#	6906#	6980#	7053#
	7063#	7100#	7143#	7147#	7227#	7372#	7424#	7475#	7540#	7647#	7707#	7789#	7828#
	7919#	8009#	8079#	8166#	8209#	8369#	8467#	8522#	8590#	8631#	8685#	8742#	8817#
	8870#												
T\$TEMP= 000000	2242#	2243#	2244#	2245#	2246#	2247#	2248#	2249#	2250#	2251#	2252#	2253#	2254#
	2255#	2256#	2257#	2258#	2259#	2260#	2261#	2262#	2263#	2264#	2265#	2266#	2267#
	2268#	2269#	2270#	2271#	2272#	2273#	2274#	2275#	2276#	2277#	2278#	2279#	2280#
	2281#	2282#	2283#	2284#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#
	5607#	5674#	5693#	5713#	5803#	5835#	5854#	5882#	5907#	5949#	5981#	6023#	6029#
	6063#	6104#	6147#	6195#	6196	6202#	6256#	6257	6265#	6303#	6309#	6347#	6353#
	6391#	6397#	6435#	6441#	6578#	6614#	6615	6634#	6635	6652#	6653	6669#	6670
	6676#	6727#	6728	6743#	6744	6752#	6800#	6801#	6817#	6823#	6867#	6868#	6883#
	6889#	6930#	6931#	6946#	6952#	7009#	7010#	7026#	7033#	7086#	7093#	7119#	7122#
	7176#	7177	7192#	7193	7199#	7202#	7308#	7309	7337#	7338	7349#	7401#	7450#
	7517#	7629#	7688#	7771#	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#
	8572#	8607#	8663#	8716#	8780#	8821#	8826#	8833#	8875#	8898#			
T\$TEST= 000052	2098#	5929#	5963#	5998#	6042#	6076#	6122#	6166#	6219#	6279#	6323#	6367#	6411#
	6470#	6596#	6694#	6775#	6842#	6905#	6979#	7052#	7062	7099	7142#	7146	7226#
	7371#	7423#	7474#	7539#	7646#	7706#	7788#	7827#	7918#	8008#	8078#	8165#	8208#
	8368#	8466#	8521#	8589#	8630#	8684#	8741#	8904					
T\$TSTM= 177777	2098#	3569	3579	3593	3603	3714	3724	3886	3922	3932	3946	3956	3995
	4005	4043	4053	4065	4075	4225	4234	4257	4267	4280	4290	4303	4313
	4326	4336	4472	4482	4866	4870	4882	4888	4896	4904	4912	4922	4929
	4939	4943	4957	4963	4971	4979	4987	4997	5004	5014	5022	5032	5039
	5049	5053	5066	5074	5080	5088	5096	5106	5113	5123	5131	5141	5148
	5158	5162	5176	5184	5190	5198	5206	5214	5224	5231	5241	5249	5259
	5266	5276	5280	5293	5301	5307	5315	5323	5333	5340	5350	5358	5368
	5375	5385	5389	5403	5409	5417	5425	5435	5442	5452	5460	5470	5477
	5487	5491	5504	5512	5518	5526	5534	5542	5552	5559	5569	5577	5587
	5594	5604	5608	5622	5628	5636	5644	5654	5661	5671	5675	5694	5745
	5751	5757	5763	5783	5804	5822	5831	5836	5855	5872	5879	5883	5908
	5933	5942	5950	5975	5982	6004	6017	6024	6030	6057	6064	6082	6098
	6105	6141	6148	6190	6195	6203	6251	6256	6266	6285	6297	6304	6310
	6329	6341	6348	6354	6373	6385	6392	6398	6417	6429	6436	6442	6482
	6508	6517	6536	6545	6564	6573	6579	6609	6614	6629	6634	6647	6652
	6664	6669	6677	6722	6727	6738	6743	6753	6780	6795	6800	6811	6818
	6824	6847	6862	6867	6877	6884	6890	6910	6925	6930	6940	6947	6953
	6986	7004	7009	7020	7027	7034	7063	7066	7080	7087	7094	7100	7113
	7120	7123	7147	7171	7176	7187	7192	7200	7203	7303	7308	7332	7337
	7350	7402	7451	7509	7518	7559	7572	7588	7611	7623	7630	7689	7772
	7810	7862	7881	7901	7971	7991	8043	8061	8131	8151	8182	8190	8293
	8343	8447	8502	8573	8608	8664	8717	8790					
T\$TSTS= 000001	2098#	5930#	5964#	5999#	6043#	6077#	6123#	6167#	6220#	6280#	6324#	6368#	6412#

CZDMRF .P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

ENDSRV	1#	2098#													
ENDSUB	1#	2098#	7092	7118	7198										
ENDSW	1#	2098#	2330												
ENDTST	1#	2098#	5948	5980	6028	6062	6103	6146	6201	6264	6308	6352	6396	6440	6577
	6675	6751	6822	6888	6951	7032	7121	7201	7348	7400	7449	7516	7628	7687	7770
	7808	7899	7989	8059	8149	8188	8341	8445	8500	8571	8606	8662	8715	8788	
EQUALS	1#	2098#	2357												
ERRDF	1#	2098#	3568	3578	3592	3602	3713	3723	3885	3921	3931	3945	3955	3994	4004
	4042	4052	4064	4074	4224	4233	4256	4266	4279	4289	4302	4312	4325	4335	4471
	4481	5941	5974	6016	6056	6097	6140	6189	6250	6296	6340	6384	6428	6507	6535
	6563	6608	6628	6646	6663	6721	6737	6794	6810	6861	6876	6924	6939	7003	7019
	7079	7112	7170	7186	7302	7331	7508	7558	7571	7587	7610	7622	7861	7880	7970
	8042	8130	8181	8292											
ERRHRD	1#	2098#													
ERROR	1#	2098#													
ERRSF	1#	2098#													
ERRSOF	1#	2098#													
ERRTBL	1#	2098#													
ESCAPE	1#	2098#	6194	6255	6613	6633	6651	6668	6726	6742	6799	6866	6929	7008	7175
	7191	7307	7336												
EXIT	1#	2098#													
FEQUAL	1#	2098#													
GETBYT	1#	2098#													
GETPRI	1#	2098#													
GETWOR	1#	2098#													
GMANIA	1#	2098#													
GMANID	1#	2098#													
GMANIL	1#	2098#													
GPHARD	1#	2098#	5781												
GPRMA	1#	2098#	8820												
GPRMD	1#	2098#	8825												
GPRML	1#	2098#													
HEADER	1#	2098#	2134												
INLOOP	1#	2098#													
IOSETU	1#	2098#													
IOSTAR	1#	2098#													
KT11	1#	2098#													
LASTAD	1#	2098#	8899												
MANUAL	1#	2098#													
MEMORY	1#	2098#													
MSBYTE	1#	2098#	2135#	2141	2142	2143									
MSCHEC	1#	2098#													
MSCNTO	1#	2098#	8821#	8826#											
MSCOUN	1#	2098#	4861#	4877#	4885#	4891#	4899#	4907#	4915#	4925#	4932#	4952#	4960#	4966#	4974#
	4982#	4990#	5000#	5007#	5017#	5025#	5035#	5042#	5062#	5069#	5077#	5083#	5091#	5099#	5109#
	5116#	5126#	5134#	5144#	5151#	5171#	5179#	5187#	5193#	5201#	5209#	5217#	5227#	5234#	5244#
	5252#	5262#	5269#	5289#	5296#	5304#	5310#	5318#	5326#	5336#	5343#	5353#	5361#	5371#	5378#
	5398#	5406#	5412#	5420#	5428#	5438#	5445#	5455#	5463#	5473#	5480#	5500#	5507#	5515#	5521#
	5529#	5537#	5545#	5555#	5562#	5572#	5580#	5590#	5597#	5617#	5625#	5631#	5639#	5647#	5657#
	5664#	5875#	6478#	6514#	6542#	6570#									
MSDATA	1#	2098#	2135#	2144	2146	2148	2150	2152	2154	2156	2158	2160	2162	2164	2166
	2168	2170	2172	2174#	2176	2178	2181	2184	2186	2188	2190	2192	2194	2196	2198
	2200	2202	2204	2206	2208	2210	2212	2214	2216	2218	3260#	3268#			
MSDECR	1#	2098#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#	5607#	5674#	5693#	5713#
	5803#	5835#	5854#	5882#	5907#	5949#	5981#	6023#	6029#	6063#	6104#	6147#	6202#	6265#	6303#
	6309#	6347#	6353#	6391#	6397#	6435#	6441#	6578#	6676#	6752#	6817#	6823#	6883#	6889#	6946#

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

	6952#	7026#	7033#	7086#	7093#	7119#	7122#	7199#	7202#	7349#	7401#	7450#	7517#	7629#	7688#
	7771#	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#	8572#	8607#	8663#	8716#	8789#
	8833#	8875#	8898#												
M\$DEFA	1#	2098#	8821#	8826#											
M\$SENDE	1#	2098#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#	5607#	5674#	5693#	5803#
	5835#	5854#	5882#	5907#	5949#	5981#	6023#	6029#	6063#	6104#	6147#	6202#	6265#	6303#	6309#
	6347#	6353#	6391#	6397#	6435#	6441#	6578#	6676#	6752#	6817#	6823#	6883#	6889#	6946#	6952#
	7026#	7033#	7086#	7093#	7119#	7122#	7199#	7202#	7349#	7401#	7450#	7517#	7629#	7688#	7771#
	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#	8572#	8607#	8663#	8716#	8789#	8833#
	8875#	8898#													
M\$ERRI	1#	2098#	3569#	3579#	3593#	3603#	3714#	3724#	3886#	3922#	3932#	3946#	3956#	3995#	4005#
	4043#	4053#	4065#	4075#	4225#	4234#	4257#	4267#	4280#	4290#	4303#	4313#	4326#	4336#	4472#
	4482#	5942#	5975#	6017#	6057#	6098#	6141#	6190#	6251#	6297#	6341#	6385#	6429#	6508#	6536#
	6564#	6609#	6629#	6647#	6664#	6722#	6738#	6795#	6811#	6862#	6877#	6925#	6940#	7004#	7020#
	7080#	7113#	7171#	7187#	7303#	7332#	7509#	7559#	7572#	7588#	7611#	7623#	7862#	7881#	7971#
	8043#	8131#	8182#	8293#											
M\$ESCA	1#	2098#	6195#	6196	6256#	6257	6614#	6615	6634#	6635	6652#	6653	6669#	6670	6727#
	6728	6743#	6744	6800#	6867#	6930#	7009#	7176#	7177	7192#	7193	7308#	7309	7337#	7338
M\$ESCS	1#	2098#	6195#	6256#	6614#	6634#	6652#	6669#	6727#	6743#	6800#	6801	6867#	6868	6930#
	6931	7009#	7010	7176#	7192#	7308#	7337#								
M\$EXCP	1#	2098#	8821#	8826#											
M\$EXIT	1#	2098#													
M\$EXSE	1#	2098#													
M\$EXTJ	1#	2098#													
M\$GEN	1#	2098#	2104#	2135#	2144#	2146#	2148#	2150#	2152#	2154#	2156#	2158#	2160#	2162#	2164#
	2166#	2168#	2170#	2172#	2174#	2176#	2178#	2181#	2184#	2186#	2188#	2190#	2192#	2194#	2196#
	2198#	2200#	2202#	2204#	2206#	2208#	2210#	2212#	2214#	2216#	2218#	2241#	2304#	2305#	2311#
	2326#	2327#	2331#	3260#	3268#	4859#	4869#	4875#	4942#	4950#	5052#	5060#	5161#	5169#	5279#
	5287#	5388#	5396#	5490#	5498#	5607#	5615#	5674#	5690#	5693#	5708#	5726#	5803#	5818#	5835#
	5850#	5854#	5869#	5882#	5905#	5907#	5929#	5949#	5963#	5981#	5998#	6023#	6029#	6042#	6063#
	6076#	6104#	6122#	6147#	6166#	6202#	6219#	6265#	6279#	6303#	6309#	6323#	6347#	6353#	6367#
	6391#	6397#	6411#	6435#	6441#	6470#	6578#	6596#	6676#	6694#	6752#	6775#	6817#	6823#	6842#
	6883#	6889#	6905#	6946#	6952#	6979#	7026#	7033#	7052#	7062#	7086#	7093#	7099#	7119#	7122#
	7142#	7146#	7199#	7202#	7226#	7349#	7371#	7401#	7423#	7450#	7474#	7517#	7539#	7629#	7646#
	7688#	7706#	7771#	7788#	7809#	7827#	7900#	7918#	7990#	8008#	8060#	8078#	8150#	8165#	8189#
	8208#	8342#	8368#	8446#	8466#	8501#	8521#	8572#	8589#	8607#	8630#	8663#	8684#	8716#	8741#
	8789#	8818#	8834#	8871#	8876#	8903#									
M\$GENB	1#	2098#													
M\$GETS	1#	2098#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#	5607#	5674#	5693#	5713#
	5803#	5835#	5854#	5882#	5907#	5949#	5981#	6023#	6029#	6063#	6104#	6147#	6202#	6265#	6303#
	6309#	6347#	6353#	6391#	6397#	6435#	6441#	6578#	6676#	6752#	6801#	6817#	6823#	6868#	6883#
	6889#	6931#	6946#	6952#	7010#	7026#	7033#	7086#	7093#	7119#	7122#	7199#	7202#	7349#	7401#
	7450#	7517#	7629#	7688#	7771#	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#	8572#
	8607#	8663#	8716#	8789#	8833#	8875#	8898#								
M\$GETT	1#	2098#	6195#	6256#	6614#	6634#	6652#	6669#	6727#	6743#	6800#	6801	6867#	6868	6930#
	6931	7009#	7010	7176#	7192#	7308#	7337#								
M\$GNGB	1#	2098#	2104#	2135#	2144#	2146#	2148#	2150#	2152#	2154#	2156#	2158#	2160#	2162#	2164#
	2166#	2168#	2170#	2172#	2174#	2176#	2178#	2181#	2184#	2186#	2188#	2190#	2192#	2194#	2196#
	2198#	2200#	2202#	2204#	2206#	2208#	2210#	2212#	2214#	2216#	2218#	2240#	2241	2303#	2304
	2305	2325#	2326	2327	3260#	3268#	4859#	4875#	4950#	5060#	5169#	5287#	5396#	5498#	5615#
	5690#	5708#	5726#	5818#	5850#	5869#	5905#	8817#	8818	8870#	8871	8900#	8903		
M\$GNIN	1#	2098#	2135#	2136	2137	2138	2139	2140	2141#	2142#	2143#	2144#	2145	2146#	2147
	2148#	2149	2150#	2151	2152#	2153	2154#	2155	2156#	2157	2158#	2159	2160#	2161	2162#
	2163	2164#	2165	2166#	2167	2168#	2169	2170#	2171	2172#	2173	2174#	2175	2176#	2177
	2178#	2179	2180	2181#	2182	2183#	2184#	2185	2186#	2187	2188#	2189	2190#	2191	2192#
	2193	2194#	2195	2196#	2197	2198#	2199	2200#	2201	2202#	2203	2204#	2205	2206#	2207

CZDMRF.P11

03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

2208#	2209	2210#	2211	2212#	2213	2214#	2215	2216#	2217	2218#	2219	2240#	2242#	2243#
2244#	2245#	2246#	2247#	2248#	2249#	2250#	2251#	2252#	2253#	2254#	2255#	2256#	2257#	2258#
2259#	2260#	2261#	2262#	2263#	2264#	2265#	2266#	2267#	2268#	2269#	2270#	2271#	2272#	2273#
2274#	2275#	2276#	2277#	2278#	2279#	2280#	2281#	2282#	2283#	2303#	2325#	3260#	3261	3262
3268#	3269	3276	3569#	3570#	3571#	3572#	3579#	3580#	3581#	3582#	3593#	3594#	3595#	3596#
3603#	3604#	3605#	3606#	3714#	3715#	3716#	3717#	3724#	3725#	3726#	3727#	3886#	3887#	3888#
3889#	3922#	3923#	3924#	3925#	3932#	3933#	3934#	3935#	3946#	3947#	3948#	3949#	3956#	3957#
3958#	3959#	3995#	3996#	3997#	3998#	4005#	4006#	4007#	4008#	4043#	4044#	4045#	4046#	4053#
4054#	4055#	4056#	4065#	4066#	4067#	4068#	4075#	4076#	4077#	4078#	4225#	4226#	4227#	4228#
4234#	4235#	4236#	4237#	4257#	4258#	4259#	4260#	4267#	4268#	4269#	4270#	4280#	4281#	4282#
4283#	4290#	4291#	4292#	4293#	4303#	4304#	4305#	4306#	4313#	4314#	4315#	4316#	4326#	4327#
4328#	4329#	4336#	4337#	4338#	4339#	4472#	4473#	4474#	4475#	4482#	4483#	4484#	4485#	4861#
4862#	4863#	4864#	4865	4866#	4867	4870#	4877#	4878#	4879#	4880#	4881	4882#	4883	4885#
4886#	4887	4888#	4889	4891#	4892#	4893#	4894#	4895	4896#	4897	4899#	4900#	4901#	4902#
4903	4904#	4905	4907#	4908#	4909#	4910#	4911	4912#	4913	4915#	4916#	4917#	4918#	4919#
4920#	4921	4922#	4923	4925#	4926#	4927#	4928	4929#	4930	4932#	4933#	4934#	4935#	4936#
4937#	4938	4939#	4940	4943#	4952#	4953#	4954#	4955#	4956	4957#	4958	4960#	4961#	4962
4963#	4964	4966#	4967#	4968#	4969#	4970	4971#	4972	4974#	4975#	4976#	4977#	4978	4979#
4980	4982#	4983#	4984#	4985#	4986	4987#	4988	4990#	4991#	4992#	4993#	4994#	4995#	4996
4997#	4998	5000#	5001#	5002#	5003	5004#	5005	5007#	5008#	5009#	5010#	5011#	5012#	5013
5014#	5015	5017#	5018#	5019#	5020#	5021	5022#	5023	5025#	5026#	5027#	5028#	5029#	5030#
5031	5032#	5033	5035#	5036#	5037#	5038	5039#	5040	5042#	5043#	5044#	5045#	5046#	5047#
5048	5049#	5050	5053#	5062#	5063#	5064#	5065	5066#	5067	5069#	5070#	5071#	5072#	5073
5074#	5075	5077#	5078#	5079	5080#	5081	5083#	5084#	5085#	5086#	5087	5088#	5089	5091#
5092#	5093#	5094#	5095	5096#	5097	5099#	5100#	5101#	5102#	5103#	5104#	5105	5106#	5107
5109#	5110#	5111#	5112	5113#	5114	5116#	5117#	5118#	5119#	5120#	5121#	5122	5123#	5124
5126#	5127#	5128#	5129#	5130	5131#	5132	5134#	5135#	5136#	5137#	5138#	5139#	5140	5141#
5142	5144#	5145#	5146#	5147	5148#	5149	5151#	5152#	5153#	5154#	5155#	5156#	5157	5158#
5159	5162#	5171#	5172#	5173#	5174#	5175	5176#	5177	5179#	5180#	5181#	5182#	5183	5184#
5185	5187#	5188#	5189	5190#	5191	5193#	5194#	5195#	5196#	5197	5198#	5199	5201#	5202#
5203#	5204#	5205	5206#	5207	5209#	5210#	5211#	5212#	5213	5214#	5215	5217#	5218#	5219#
5220#	5221#	5222#	5223	5224#	5225	5227#	5228#	5229#	5230	5231#	5232	5234#	5235#	5236#
5237#	5238#	5239#	5240	5241#	5242	5244#	5245#	5246#	5247#	5248	5249#	5250	5252#	5253#
5254#	5255#	5256#	5257#	5258	5259#	5260	5262#	5263#	5264#	5265	5266#	5267	5269#	5270#
5271#	5272#	5273#	5274#	5275	5276#	5277	5280#	5289#	5290#	5291#	5292	5293#	5294	5296#
5297#	5298#	5299#	5300	5301#	5302	5304#	5305#	5306	5307#	5308	5310#	5311#	5312#	5313#
5314	5315#	5316	5318#	5319#	5320#	5321#	5322	5323#	5324	5326#	5327#	5328#	5329#	5330#
5331#	5332	5333#	5334	5336#	5337#	5338#	5339	5340#	5341	5343#	5344#	5345#	5346#	5347#
5348#	5349	5350#	5351	5353#	5354#	5355#	5356#	5357	5358#	5359	5361#	5362#	5363#	5364#
5365#	5366#	5367	5368#	5369	5371#	5372#	5373#	5374	5375#	5376	5378#	5379#	5380#	5381#
5382#	5383#	5384	5385#	5386	5389#	5398#	5399#	5400#	5401#	5402	5403#	5404	5406#	5407#
5408	5409#	5410	5412#	5413#	5414#	5415#	5416	5417#	5418	5420#	5421#	5422#	5423#	5424
5425#	5426	5428#	5429#	5430#	5431#	5432#	5433#	5434	5435#	5436	5438#	5439#	5440#	5441
5442#	5443	5445#	5446#	5447#	5448#	5449#	5450#	5451	5452#	5453	5455#	5456#	5457#	5458#
5459	5460#	5461	5463#	5464#	5465#	5466#	5467#	5468#	5469	5470#	5471	5473#	5474#	5475#
5476	5477#	5478	5480#	5481#	5482#	5483#	5484#	5485#	5486	5487#	5488	5491#	5500#	5501#
5502#	5503	5504#	5505	5507#	5508#	5509#	5510#	5511	5512#	5513	5515#	5516#	5517	5518#
5519	5521#	5522#	5523#	5524#	5525	5526#	5527	5529#	5530#	5531#	5532#	5533	5534#	5535
5537#	5538#	5539#	5540#	5541	5542#	5543	5545#	5546#	5547#	5548#	5549#	5550#	5551	5552#
5553	5555#	5556#	5557#	5558	5559#	5560	5562#	5563#	5564#	5565#	5566#	5567#	5568	5569#
5570	5572#	5573#	5574#	5575#	5576	5577#	5578	5580#	5581#	5582#	5583#	5584#	5585#	5586
5587#	5588	5590#	5591#	5592#	5593	5594#	5595	5597#	5598#	5599#	5600#	5601#	5602#	5603
5604#	5605	5608#	5617#	5618#	5619#	5620#	5621	5622#	5623	5625#	5626#	5627	5628#	5629
5631#	5632#	5633#	5634#	5635	5636#	5637	5639#	5640#	5641#	5642#	5643	5644#	5645	5647#
5648#	5649#	5650#	5651#	5652#	5653	5654#	5655	5657#	5658#	5659#	5660	5661#	5662	5664#
5665#	5666#	5667#	5668#	5669#	5670	5671#	5672	5675#	5694#	5744#	5745#	5747#	5750#	5751#

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

	5753#	5756#	5757#	5759#	5762#	5763#	5765#	5782#	5783#	5784#	5786#	5804#	5821#	5822#	5830#
	5831#	5836#	5855#	5872#	5875#	5876#	5877#	5878	5879#	5880	5883#	5908#	5932#	5933#	5942#
	5943#	5944#	5945#	5950#	5975#	5976#	5977#	5978#	5982#	6004#	6017#	6018#	6019#	6020#	6024#
	6030#	6057#	6058#	6059#	6060#	6064#	6082#	6098#	6099#	6100#	6101#	6105#	6141#	6142#	6143#
	6144#	6148#	6190#	6191#	6192#	6193#	6195#	6196#	6203#	6251#	6252#	6253#	6254#	6256#	6257#
	6266#	6285#	6297#	6298#	6299#	6300#	6304#	6310#	6329#	6341#	6342#	6343#	6344#	6348#	6354#
	6373#	6385#	6386#	6387#	6388#	6392#	6398#	6417#	6429#	6430#	6431#	6432#	6436#	6442#	6478#
	6479#	6480#	6481	6482#	6483	6508#	6509#	6510#	6511#	6514#	6515#	6516	6517#	6518	6536#
	6537#	6538#	6539#	6542#	6543#	6544	6545#	6546	6564#	6565#	6566#	6567#	6570#	6571#	6572
	6573#	6574	6579#	6609#	6610#	6611#	6612#	6614#	6615#	6629#	6630#	6631#	6632#	6634#	6635#
	6647#	6648#	6649#	6650#	6652#	6653#	6664#	6665#	6666#	6667#	6669#	6670#	6677#	6722#	6723#
	6724#	6725#	6727#	6728#	6738#	6739#	6740#	6741#	6743#	6744#	6753#	6780#	6795#	6796#	6797#
	6798#	6800#	6801#	6811#	6812#	6813#	6814#	6818#	6824#	6847#	6862#	6863#	6864#	6865#	6867#
	6868#	6877#	6878#	6879#	6880#	6884#	6890#	6910#	6925#	6926#	6927#	6928#	6930#	6931#	6940#
	6941#	6942#	6943#	6947#	6953#	6986#	7004#	7005#	7006#	7007#	7009#	7010#	7020#	7021#	7022#
	7023#	7027#	7034#	7063#	7066#	7080#	7081#	7082#	7083#	7087#	7094#	7100#	7113#	7114#	7115#
	7116#	7120#	7123#	7147#	7171#	7172#	7173#	7174#	7176#	7177#	7187#	7188#	7189#	7190#	7192#
	7193#	7200#	7203#	7303#	7304#	7305#	7306#	7308#	7309#	7332#	7333#	7334#	7335#	7337#	7338#
	7350#	7402#	7451#	7509#	7510#	7511#	7512#	7518#	7559#	7560#	7561#	7562#	7572#	7573#	7574#
	7575#	7588#	7589#	7590#	7591#	7611#	7612#	7613#	7614#	7623#	7624#	7625#	7626#	7630#	7689#
	7772#	7810#	7862#	7863#	7864#	7865#	7881#	7882#	7883#	7884#	7901#	7971#	7972#	7973#	7974#
	7991#	8043#	8044#	8045#	8046#	8061#	8131#	8132#	8133#	8134#	8151#	8182#	8183#	8184#	8185#
	8190#	8293#	8294#	8295#	8296#	8343#	8447#	8502#	8573#	8608#	8664#	8717#	8790#	8817#	8821#
	8822	8823	8824	8826#	8827	8828	8829	8830	8833#	8870#	8875#	8900#	8901#	8902#	
MSGNLS	1#	2098#	6023#	6303#	6347#	6391#	6435#	6817#	6883#	6946#	7026#	7086#			
MSGNSU	1#	2098#	7062#	7099#	7146#										
MSGNTA	1#	2098#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#	5607#	5674#	5693#	5803#
	5835#	5854#	5882#	5907#	5949#	5981#	6029#	6063#	6104#	6147#	6202#	6265#	6309#	6353#	6397#
	6441#	6578#	6676#	6752#	6823#	6889#	6952#	7033#	7093#	7119#	7122#	7199#	7202#	7349#	7401#
	7450#	7517#	7629#	7688#	7771#	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#	8572#
	8607#	8663#	8716#	8789#	8833#	8834	8875#	8876							
MSGNTE	1#	2098#	5929#	5963#	5998#	6042#	6076#	6122#	6166#	6219#	6279#	6323#	6367#	6411#	6470#
	6596#	6694#	6775#	6842#	6905#	6979#	7052#	7142#	7226#	7371#	7423#	7474#	7539#	7646#	7706#
	7788#	7827#	7918#	8008#	8078#	8165#	8208#	8368#	8466#	8521#	8589#	8630#	8684#	8741#	
MSHAPT	1#	2098#	2135#												
MSHNAP	1#	2098#	2135#	2174											
MSINCR	1#	2098#	2104#	2303#	2325#	3569#	3579#	3593#	3603#	3714#	3724#	3886#	3922#	3932#	3946#
	3956#	3995#	4005#	4043#	4053#	4065#	4075#	4225#	4234#	4257#	4267#	4280#	4290#	4303#	4313#
	4326#	4336#	4472#	4482#	4859#	4866#	4870#	4875#	4882#	4888#	4896#	4904#	4912#	4922#	4929#
	4939#	4943#	4950#	4957#	4963#	4971#	4979#	4987#	4997#	5004#	5014#	5022#	5032#	5039#	5049#
	5053#	5060#	5066#	5074#	5080#	5088#	5096#	5106#	5113#	5123#	5131#	5141#	5148#	5158#	5162#
	5169#	5176#	5184#	5190#	5198#	5206#	5214#	5224#	5231#	5241#	5249#	5259#	5266#	5276#	5280#
	5287#	5293#	5301#	5307#	5315#	5323#	5333#	5340#	5350#	5358#	5368#	5375#	5385#	5389#	5396#
	5403#	5409#	5417#	5425#	5435#	5442#	5452#	5460#	5470#	5477#	5487#	5491#	5498#	5504#	5512#
	5518#	5526#	5534#	5542#	5552#	5559#	5569#	5577#	5587#	5594#	5604#	5608#	5615#	5622#	5628#
	5636#	5644#	5654#	5661#	5671#	5675#	5690#	5694#	5708#	5726#	5745#	5751#	5757#	5763#	5783#
	5804#	5818#	5822#	5831#	5836#	5850#	5855#	5869#	5872#	5879#	5883#	5905#	5908#	5929#	5930#
	5933#	5942#	5950#	5963#	5964#	5975#	5982#	5998#	5999#	6004#	6017#	6024#	6030#	6042#	6043#
	6057#	6064#	6076#	6077#	6082#	6098#	6105#	6122#	6123#	6141#	6148#	6166#	6167#	6190#	6195#
	6203#	6219#	6220#	6251#	6256#	6266#	6279#	6280#	6285#	6297#	6304#	6310#	6323#	6324#	6329#
	6341#	6348#	6354#	6367#	6368#	6373#	6385#	6392#	6398#	6411#	6412#	6417#	6429#	6436#	6442#
	6470#	6471#	6482#	6508#	6517#	6536#	6545#	6564#	6573#	6579#	6596#	6597#	6609#	6614#	6629#
	6634#	6647#	6652#	6664#	6669#	6677#	6694#	6695#	6722#	6727#	6738#	6743#	6753#	6775#	6776#
	6780#	6795#	6800#	6811#	6818#	6824#	6842#	6843#	6847#	6862#	6867#	6877#	6884#	6890#	6905#
	6906#	6910#	6925#	6930#	6940#	6947#	6953#	6979#	6980#	6986#	7004#	7009#	7020#	7027#	7034#
	7052#	7053#	7062#	7063#	7066#	7080#	7087#	7094#	7099#	7100#	7113#	7120#	7123#	7142#	7143#

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

	7146#	7147#	7171#	7176#	7187#	7192#	7200#	7203#	7226#	7227#	7303#	7308#	7332#	7337#	7350#
	7371#	7372#	7402#	7423#	7424#	7451#	7474#	7475#	7509#	7518#	7539#	7540#	7559#	7572#	7588#
	7611#	7623#	7630#	7646#	7647#	7689#	7706#	7707#	7772#	7788#	7789#	7810#	7827#	7828#	7862#
	7881#	7901#	7918#	7919#	7971#	7991#	8008#	8009#	8043#	8061#	8078#	8079#	8131#	8151#	8165#
	8166#	8182#	8190#	8208#	8209#	8293#	8343#	8368#	8369#	8447#	8466#	8467#	8502#	8521#	8522#
	8573#	8589#	8590#	8608#	8630#	8631#	8664#	8684#	8685#	8717#	8741#	8742#	8790#	8817#	8870#
MSIOSE	1#	2098#													
MSLDRO	1#	2098#	5744#	5750#	5756#	5762#	5782#	5821#	5830#	5932#					
MSMASK	1#	2098#													
MSMCHI	1#	2098#													
MSMCLO	1#	2098#													
MSMSK1	1#	2098#													
MSPOP	1#	2098#	2311#	2331#	4869#	4942#	5052#	5161#	5279#	5388#	5490#	5607#	5674#	5693#	5713#
	5803#	5835#	5854#	5882#	5907#	5949#	5981#	6023#	6029#	6063#	6104#	6147#	6202#	6265#	6303#
	6309#	6347#	6353#	6391#	6397#	6435#	6441#	6578#	6676#	6752#	6817#	6823#	6883#	6889#	6946#
	6952#	7026#	7033#	7086#	7093#	7119#	7122#	7199#	7202#	7349#	7401#	7450#	7517#	7629#	7688#
	7771#	7809#	7900#	7990#	8060#	8150#	8189#	8342#	8446#	8501#	8572#	8607#	8663#	8716#	8789#
MSPRIN	1#	2098#	4861#	4877#	4885#	4891#	4899#	4907#	4915#	4925#	4932#	4952#	4960#	4966#	4974#
	4982#	4990#	5000#	5007#	5017#	5025#	5035#	5042#	5062#	5069#	5077#	5083#	5091#	5099#	5109#
	5116#	5126#	5134#	5144#	5151#	5171#	5179#	5187#	5193#	5201#	5209#	5217#	5227#	5234#	5244#
	5252#	5262#	5269#	5289#	5296#	5304#	5310#	5318#	5326#	5336#	5343#	5353#	5361#	5371#	5378#
	5398#	5406#	5412#	5420#	5428#	5438#	5445#	5455#	5463#	5473#	5480#	5500#	5507#	5515#	5521#
	5529#	5537#	5545#	5555#	5562#	5572#	5580#	5590#	5597#	5617#	5625#	5631#	5639#	5647#	5657#
MSPUSH	1#	2098#	2104#	2303#	2325#	4859#	4875#	4950#	5060#	5169#	5287#	5396#	5498#	5615#	5690#
	5708#	5726#	5818#	5850#	5869#	5905#	5929#	5930	5963#	5964	5998#	5999	6004#	6042#	6043
	6076#	6077	6122#	6123	6166#	6167	6219#	6220	6279#	6280	6285#	6323#	6324	6329#	6367#
	6368	6373#	6411#	6412	6417#	6470#	6471	6596#	6597	6694#	6695	6775#	6776	6780#	6842#
	6843	6847#	6905#	6906	6910#	6979#	6980	6986#	7052#	7053	7062#	7063	7066#	7099#	7100
	7142#	7143	7146#	7147	7226#	7227	7371#	7372	7423#	7424	7474#	7475	7539#	7540	7646#
	7647	7706#	7707	7788#	7789	7827#	7828	7918#	7919	8008#	8009	8078#	8079	8165#	8166
	8208#	8209	8368#	8369	8466#	8467	8521#	8522	8589#	8590	8630#	8631	8684#	8685	8741#
	8742	8817#	8870#												
MSPUT	1#	2098#	4861#	4877#	4885#	4891#	4899#	4907#	4915#	4925#	4932#	4952#	4960#	4966#	4974#
	4982#	4990#	5000#	5007#	5017#	5025#	5035#	5042#	5062#	5069#	5077#	5083#	5091#	5099#	5109#
	5116#	5126#	5134#	5144#	5151#	5171#	5179#	5187#	5193#	5201#	5209#	5217#	5227#	5234#	5244#
	5252#	5262#	5269#	5289#	5296#	5304#	5310#	5318#	5326#	5336#	5343#	5353#	5361#	5371#	5378#
	5398#	5406#	5412#	5420#	5428#	5438#	5445#	5455#	5463#	5473#	5480#	5500#	5507#	5515#	5521#
	5529#	5537#	5545#	5555#	5562#	5572#	5580#	5590#	5597#	5617#	5625#	5631#	5639#	5647#	5657#
MSPUT1	1#	2098#	4861#	4862	4863	4864	4877#	4878	4879	4880	4885#	4886	4891#	4892	4893
	4894	4899#	4900	4901	4902	4907#	4908	4909	4910	4915#	4916	4917	4918	4919	4920
	4925#	4926	4927	4932#	4933	4934	4935	4936	4937	4952#	4953	4954	4955	4960#	4961
	4966#	4967	4968	4969	4974#	4975	4976	4977	4982#	4983	4984	4985	4990#	4991	4992
	4993	4994	4995	5000#	5001	5002	5007#	5008	5009	5010	5011	5012	5017#	5018	5019
	5020	5025#	5026	5027	5028	5029	5030	5035#	5036	5037	5042#	5043	5044	5045	5046
	5047	5062#	5063	5064	5069#	5070	5071	5072	5077#	5078	5083#	5084	5085	5086	5091#
	5092	5093	5094	5099#	5100	5101	5102	5103	5104	5109#	5110	5111	5116#	5117	5118
	5119	5120	5121	5126#	5127	5128	5129	5134#	5135	5136	5137	5138	5139	5144#	5145
	5146	5151#	5152	5153	5154	5155	5156	5171#	5172	5173	5174	5179#	5180	5181	5182
	5187#	5188	5193#	5194	5195	5196	5201#	5202	5203	5204	5209#	5210	5211	5212	5217#
	5218	5219	5220	5221	5222	5227#	5228	5229	5234#	5235	5236	5237	5238	5239	5244#
	5245	5246	5247	5252#	5253	5254	5255	5256	5257	5262#	5263	5264	5269#	5270	5271
	5272	5273	5274	5289#	5290	5291	5296#	5297	5298	5299	5304#	5305	5310#	5311	5312
	5313	5318#	5319	5320	5321	5326#	5327	5328	5329	5330	5331	5336#	5337	5338	5343#

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

	5344	5345	5346	5347	5348	5353#	5354	5355	5356	5361#	5362	5363	5364	5365	5366
	5371#	5372	5373	5378#	5379	5380	5381	5382	5383	5398#	5399	5400	5401	5406#	5407
	5412#	5413	5414	5415	5420#	5421	5422	5423	5428#	5429	5430	5431	5432	5433	5438#
	5439	5440	5445#	5446	5447	5448	5449	5450	5455#	5456	5457	5458	5463#	5464	5465
	5466	5467	5468	5473#	5474	5475	5480#	5481	5482	5483	5484	5485	5500#	5501	5502
	5507#	5508	5509	5510	5515#	5516	5521#	5522	5523	5524	5529#	5530	5531	5532	5537#
	5538	5539	5540	5545#	5546	5547	5548	5549	5550	5555#	5556	5557	5562#	5563	5564
	5565	5566	5567	5572#	5573	5574	5575	5580#	5581	5582	5583	5584	5585	5590#	5591
	5592	5597#	5598	5599	5600	5601	5602	5617#	5618	5619	5620	5625#	5626	5631#	5632
	5633	5634	5639#	5640	5641	5642	5647#	5648	5649	5650	5651	5652	5657#	5658	5659
	5664#	5665	5666	5667	5668	5669	5875#	5876	5877	6478#	6479	6480	6514#	6515	6542#
	6543	6570#	6571												
MSRADI	1#	2098#	8821#	8826#											
MSRBRO	1#	2098#													
MSRNRO	1#	2098#	5782#	5784											
M\$SETS	1#	2098#	2104#	2303#	2325#	4859#	4875#	4950#	5060#	5169#	5287#	5396#	5498#	5615#	5690#
	5708#	5726#	5818#	5850#	5869#	5905#	5930#	5964#	5999#	6004#	6043#	6077#	6123#	6167#	6220#
	6280#	6285#	6324#	6329#	6368#	6373#	6412#	6417#	6471#	6597#	6695#	6776#	6780#	6843#	6847#
	6906#	6910#	6980#	6986#	7053#	7063#	7066#	7100#	7143#	7147#	7227#	7372#	7424#	7475#	7540#
	7647#	7707#	7789#	7828#	7919#	8009#	8079#	8166#	8209#	8369#	8467#	8522#	8590#	8631#	8685#
	8742#	8817#	8870#												
M\$STAR	1#	2098#													
M\$SVC	1#	2098#	3569	3579	3593	3603	3714	3724	3886	3922	3932	3946	3956	3995	4005
	4043	4053	4065	4075	4225	4234	4257	4267	4280	4290	4303	4313	4326	4336	4472
	4482	4861#	4866	4869#	4870	4877#	4882	4885#	4888	4891#	4896	4899#	4904	4907#	4912
	4915#	4922	4925#	4929	4932#	4939	4942#	4943	4952#	4957	4960#	4963	4966#	4971	4974#
	4979	4982#	4987	4990#	4997	5000#	5004	5007#	5014	5017#	5022	5025#	5032	5035#	5039
	5042#	5049	5052#	5053	5062#	5066	5069#	5074	5077#	5080	5083#	5088	5091#	5096	5099#
	5106	5109#	5113	5116#	5123	5126#	5131	5134#	5141	5144#	5148	5151#	5158	5161#	5162
	5171#	5176	5179#	5184	5187#	5190	5193#	5198	5201#	5206	5209#	5214	5217#	5224	5227#
	5231	5234#	5241	5244#	5249	5252#	5259	5262#	5266	5269#	5276	5279#	5280	5289#	5293
	5296#	5301	5304#	5307	5310#	5315	5318#	5323	5326#	5333	5336#	5340	5343#	5350	5353#
	5358	5361#	5368	5371#	5375	5378#	5385	5388#	5389	5398#	5403	5406#	5409	5412#	5417
	5420#	5425	5428#	5435	5438#	5442	5445#	5452	5455#	5460	5463#	5470	5473#	5477	5'80#
	5487	5490#	5491	5500#	5504	5507#	5512	5515#	5518	5521#	5526	5529#	5534	5537#	5542
	5545#	5552	5555#	5559	5562#	5569	5572#	5577	5580#	5587	5590#	5594	5597#	5604	5607#
	5608	5617#	5622	5625#	5628	5631#	5636	5639#	5644	5647#	5654	5657#	5661	5664#	5671
	5674#	5675	5693#	5694	5744#	5745	5750#	5751	5756#	5757	5762#	5763	5782#	5783	5803#
	5804	5821#	5822	5830#	5831	5835#	5836	5854#	5855	5872#	5875#	5879	5882#	5883	5907#
	5908	5932#	5933	5942	5949#	5950	5975	5981#	5982	6004#	6017	6023#	6024	6029#	6030
	6057	6063#	6064	6082#	6098	6104#	6105	6141	6147#	6148	6190	6195#	6202#	6203	6251
	6256#	6265#	6266	6285#	6297	6303#	6304	6309#	6310	6329#	6341	6347#	6348	6353#	6354
	6373#	6385	6391#	6392	6397#	6398	6417#	6429	6435#	6436	6441#	6442	6478#	6482	6508
	6514#	6517	6536	6542#	6545	6564	6570#	6573	6578#	6579	6609	6614#	6629	6634#	6647
	6652#	6664	6669#	6676#	6677	6722	6727#	6738	6743#	6752#	6753	6780#	6795	6800#	6811
	6817#	6818	6823#	6824	6847#	6862	6867#	6877	6883#	6884	6889#	6890	6910#	6925	6930#
	6940	6946#	6947	6952#	6953	6986#	7004	7009#	7020	7026#	7027	7033#	7034	7062#	7063
	7066#	7080	7086#	7087	7093#	7094	7099#	7100	7113	7119#	7120	7122#	7123	7146#	7147
	7171	7176#	7187	7192#	7199#	7200	7202#	7203	7303	7308#	7332	7337#	7349#	7350	7401#
	7402	7450#	7451	7509	7517#	7518	7559	7572	7588	7611	7623	7629#	7630	7688#	7689
	7771#	7772	7809#	7810	7862	7881	7900#	7901	7971	7990#	7991	8043	8060#	8061	8131
	8150#	8151	8182	8189#	8190	8293	8342#	8343	8446#	8447	8501#	8502	8572#	8573	8607#
	8608	8663#	8664	8716#	8717	8789#	8790								
M\$TLAB	1#	2098#	3569#	3579#	3593#	3603#	3714#	3724#	3886#	3922#	3932#	3946#	3956#	3995#	4005#
	4043#	4053#	4065#	4075#	4225#	4234#	4257#	4267#	4280#	4290#	4303#	4313#	4326#	4336#	4472#
	4482#	4866#	4870#	4882#	4888#	4896#	4904#	4912#	4922#	4929#	4939#	4943#	4957#	4963#	4971#

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

MSTSTL

MSWORD

4979#	4987#	4997#	5004#	5014#	5022#	5032#	5039#	5049#	5053#	5066#	5074#	5080#	5088#	5096#
5106#	5113#	5123#	5131#	5141#	5148#	5158#	5162#	5176#	5184#	5190#	5198#	5206#	5214#	5224#
5231#	5241#	5249#	5259#	5266#	5276#	5280#	5293#	5301#	5307#	5315#	5323#	5333#	5340#	5350#
5358#	5368#	5375#	5385#	5389#	5403#	5409#	5417#	5425#	5435#	5442#	5452#	5460#	5470#	5477#
5487#	5491#	5504#	5512#	5518#	5526#	5534#	5542#	5552#	5559#	5569#	5577#	5587#	5594#	5604#
5608#	5622#	5628#	5636#	5644#	5654#	5661#	5671#	5675#	5694#	5745#	5751#	5757#	5763#	5783#
5804#	5822#	5831#	5836#	5855#	5872#	5879#	5883#	5908#	5933#	5942#	5950#	5975#	5982#	6004#
6017#	6024#	6030#	6057#	6064#	6082#	6098#	6105#	6141#	6148#	6190#	6195#	6203#	6251#	6256#
6266#	6285#	6297#	6304#	6310#	6329#	6341#	6348#	6354#	6373#	6385#	6392#	6398#	6417#	6429#
6436#	6442#	6482#	6508#	6517#	6536#	6545#	6564#	6573#	6579#	6609#	6614#	6629#	6634#	6647#
6652#	6664#	6669#	6677#	6722#	6727#	6738#	6743#	6753#	6780#	6795#	6800#	6811#	6818#	6824#
6847#	6862#	6867#	6877#	6884#	6890#	6910#	6925#	6930#	6940#	6947#	6953#	6986#	7004#	7009#
7020#	7027#	7034#	7063#	7066#	7080#	7087#	7094#	7100#	7113#	7120#	7123#	7147#	7171#	7176#
7187#	7192#	7200#	7203#	7303#	7308#	7332#	7337#	7350#	7402#	7451#	7509#	7518#	7559#	7572#
7588#	7611#	7623#	7630#	7689#	7772#	7810#	7862#	7881#	7901#	7971#	7991#	8043#	8061#	8131#
8151#	8182#	8190#	8293#	8343#	8447#	8502#	8573#	8608#	8664#	8717#	8790#			
1#	2098#	3569#	3579#	3593#	3603#	3714#	3724#	3886#	3922#	3932#	3946#	3956#	3995#	4005#
4043#	4053#	4065#	4075#	4225#	4234#	4257#	4267#	4280#	4290#	4303#	4313#	4326#	4336#	4472#
4482#	4866#	4870#	4882#	4888#	4896#	4904#	4912#	4922#	4929#	4939#	4943#	4957#	4963#	4971#
4979#	4987#	4997#	5004#	5014#	5022#	5032#	5039#	5049#	5053#	5066#	5074#	5080#	5088#	5096#
5106#	5113#	5123#	5131#	5141#	5148#	5158#	5162#	5176#	5184#	5190#	5198#	5206#	5214#	5224#
5231#	5241#	5249#	5259#	5266#	5276#	5280#	5293#	5301#	5307#	5315#	5323#	5333#	5340#	5350#
5358#	5368#	5375#	5385#	5389#	5403#	5409#	5417#	5425#	5435#	5442#	5452#	5460#	5470#	5477#
5487#	5491#	5504#	5512#	5518#	5526#	5534#	5542#	5552#	5559#	5569#	5577#	5587#	5594#	5604#
5608#	5622#	5628#	5636#	5644#	5654#	5661#	5671#	5675#	5694#	5745#	5751#	5757#	5763#	5783#
5804#	5822#	5831#	5836#	5855#	5872#	5879#	5883#	5908#	5933#	5942#	5950#	5975#	5982#	6004#
6017#	6024#	6030#	6057#	6064#	6082#	6098#	6105#	6141#	6148#	6190#	6195#	6203#	6251#	6256#
6266#	6285#	6297#	6304#	6310#	6329#	6341#	6348#	6354#	6373#	6385#	6392#	6398#	6417#	6429#
6436#	6442#	6482#	6508#	6517#	6536#	6545#	6564#	6573#	6579#	6609#	6614#	6629#	6634#	6647#
6652#	6664#	6669#	6677#	6722#	6727#	6738#	6743#	6753#	6780#	6795#	6800#	6811#	6818#	6824#
6847#	6862#	6867#	6877#	6884#	6890#	6910#	6925#	6930#	6940#	6947#	6953#	6986#	7004#	7009#
7020#	7027#	7034#	7063#	7066#	7080#	7087#	7094#	7100#	7113#	7120#	7123#	7147#	7171#	7176#
7187#	7192#	7200#	7203#	7303#	7308#	7332#	7337#	7350#	7402#	7451#	7509#	7518#	7559#	7572#
7588#	7611#	7623#	7630#	7689#	7772#	7810#	7862#	7881#	7901#	7971#	7991#	8043#	8061#	8131#
8151#	8182#	8190#	8293#	8343#	8447#	8502#	8573#	8608#	8664#	8717#	8790#			
1#	2098#	2174#	2183	2240#	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251
2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266
2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281
2282	2283	3569#	3570	3571	3572	3579#	3580	3581	3582	3593#	3594	3595	3596	3603#
3604	3605	3606	3714#	3715	3716	3717	3724#	3725	3726	3727	3886#	3887	3888	3889
3922#	3923	3924	3925	3932#	3933	3934	3935	3946#	3947	3948	3949	3956#	3957	3958
3959	3995#	3996	3997	3998	4005#	4006	4007	4008	4043#	4044	4045	4046	4053#	4054
4055	4056	4065#	4066	4067	4068	4075#	4076	4077	4078	4225#	4226	4227	4228	4234#
4235	4236	4237	4257#	4258	4259	4260	4267#	4268	4269	4270	4280#	4281	4282	4283
4290#	4291	4292	4293	4303#	4304	4305	4306	4313#	4314	4315	4316	4326#	4327	4328
4329	4336#	4337	4338	4339	4472#	4473	4474	4475	4482#	4483	4484	4485	5942#	5943
5944	5945	5975#	5976	5977	5978	6017#	6018	6019	6020	6057#	6058	6059	6060	6098#
6099	6100	6101	6141#	6142	6143	6144	6190#	6191	6192	6193	6251#	6252	6253	6254
6297#	6298	6299	6300	6341#	6342	6343	6344	6385#	6386	6387	6388	6429#	6430	6431
6432	6508#	6509	6510	6511	6536#	6537	6538	6539	6564#	6565	6566	6567	6609#	6610
6611	6612	6629#	6630	6631	6632	6647#	6648	6649	6650	6664#	6665	6666	6667	6722#
6723	6724	6725	6738#	6739	6740	6741	6795#	6796	6797	6798	6811#	6812	6813	6814
6862#	6863	6864	6865	6877#	6878	6879	6880	6925#	6926	6927	6928	6940#	6941	6942
6943	7004#	7005	7006	7007	7020#	7021	7022	7023	7080#	7081	7082	7083	7113#	7114
7115	7116	7171#	7172	7173	7174	7187#	7188	7189	7190	7303#	7304	7305	7306	7332#
7333	7334	7335	7509#	7510	7511	7512	7559#	7560	7561	7562	7572#	7573	7574	7575

CZDMRF.P11 03-NOV-81 10:29

CROSS REFERENCE TABLE -- MACRO NAMES

	7588#	7589	7590	7591	7611#	7612	7613	7614	7623#	7624	7625	7626	7862#	7863	7864
	7865	7881#	7882	7883	7884	7971#	7972	7973	7974	8043#	8044	8045	8046	8131#	8132
	8133	8134	8182#	8183	8184	8185	8293#	8294	8295	8296	8821#	8826#	8901	8902	
MSXFER	1#	2098#													
OPEN	1#	2098#													
POINTE	1#	2098#	2126												
PRINTB	1#	2098#	4860	4876	4884	4890	4898	4951	4959	4965	4973	5061	5068	5076	5082
	5170	5178	5186	5192	5200	5288	5295	5303	5309	5397	5405	5411	5499	5506	5514
	5520	5528	5616	5624	5630										
PRINTF	1#	2098#	5874	6477	6513	6541	6569								
PRINTS	1#	2098#													
PRINTX	1#	2098#	4906	4914	4924	4931	4981	4989	4999	5006	5016	5024	5034	5041	5090
	5098	5108	5115	5125	5133	5143	5150	5208	5216	5226	5233	5243	5251	5261	5268
	5317	5325	5335	5342	5352	5360	5370	5377	5419	5427	5437	5444	5454	5462	5472
	5479	5536	5544	5554	5561	5571	5579	5589	5596	5638	5646	5656	5663		
READBU	1#	2098#													
READEF	1#	2098#	5743	5749	5755	5761									
RFLAGS	1#	2098#													
SETPRI	1#	2098#	5820	5931											
SETVEC	1#	2098#													
SLASH	1#	2098#													
STARS	1#	2098#													
SVC	1#	2096#	2097												
XFER	1#	2098#													
XFERF	1#	2098#													
XFERT	1#	2098#													

. ABS. 036054 000

ERRORS DETECTED: 0

CZDMRF/I,CZDMRF.SEQ/CRF/SOL/NL:TOC=SVC34R.MLB,CZDMRF.P11

RUN-TIME: 35 45 4 SECONDS

RUN-TIME RATIO: 144/85=1.6

CORE USED: 19K (37 PAGES)