

DMP - 11
DMR - 11

M8207 STATIC DIAG. #1
CZDMPB0

AH - E226B - MC
FICHE 1 OF 1

NOV 1980
COPYRIGHT © 79 80
MADE IN USA



A large grid of 10 columns and 15 rows of small, illegible technical diagrams or data tables. Each cell contains a small-scale version of the same type of diagram or data, likely representing a detailed static diagram for a specific component or system. The diagrams are too small to read but appear to be organized in a structured, repeating pattern.

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-E225B-MC
PRODUCT NAME: CZDMPB0 M8207 STATIC DIAG #1
PRODUCT DATE: AUG. 1980
MAINTAINER: DIAGNOSTICS MERRIMACK
AUTHOR: ED BADGER

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) 1979,1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86

TABLE OF CONTENTS

- 1.0 INTRODUCTION
 - 1.1 PROGRAM ABSTRACT
 - 1.2 HARDWARE INTRODUCTION
- 2.0 HARDWARE REQUIREMENTS
- 3.0 PRELIMINARY PROGRAM REQUIREMENTS
- 4.0 GENERAL PROGRAM CONSIDERATIONS
 - 4.1 DIAGNOSTIC SUPERVISOR
 - 4.2 EXECUTION TIME
- 5.0 PROGRAM LOAD MEDIA
- 6.0 OPERATING INSTRUCTIONS
 - 6.1 LOADING AND STARTING PROCEDURES
 - 6.1.1 LOADING PROCEDURES
 - 6.1.2 STARTING PROCEDURES
 - 6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION
 - 6.2 INITIAL DIALOGUE
 - 6.3 PROGRAM OPTIONS
 - 6.3.1 START COMMAND
 - 6.3.2 RESTART COMMAND
 - 6.3.3 CONTINUE COMMAND
 - 6.3.4 PROCEED COMMAND
 - 6.3.5 ADD COMMAND
 - 6.3.6 DROP COMMAND
 - 6.3.7 PRINT COMMAND
 - 6.3.8 DISPLAY COMMAND
 - 6.3.9 FLAGS COMMAND
 - 6.3.10 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 TEST DESCRIPTIONS
- 8.0 ERROR INFORMATION
 - 8.1 ERROR REPORTING

87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142

1.0 INTRODUCTION

1.1 PROGRAM ABSTRACT

THIS DIAGNOSTIC WAS DESIGNED TO TEST OUT THE M8200, M8204, C9 M8207 MICROPROCESSOR. IT IS THE FIRST OF TWO DIAGNOSTICS FOR THESE OPTIONS.

THE PROGRAM WAS IMPLEMENTED USING THE DIAGNOSTIC SUPERVISOR.

THROUGH DIALOGUE WITH THE OPERATOR, THE PROGRAM WILL ALLOW MODIFICATION OF DEVICE PARAMETERS, SUCH AS UNIBUS ADDRESS, VECTOR ADDRESS, AND PROCESSOR TYPE.

1.2 HARDWARE INTRODUCTION

THE M820X MICROPROCESSOR USES AN EIGHT BIT DATA PATH WITH A SIXTEEN BIT INSTRUCTION MEMORY. THE INSTRUCTION MEMORY AND DATA MEMORY ARE TWO SEPARATE MEMORIES. THE MICROPROCESSOR IS DESIGNED FOR MOVING DATA AT HIGH RATES TO WORK AS A HIGH SPEED LINK BETWEEN PROCESSORS WHEN USED WITH A LINE UNIT. THE M8200 AND M8207 HAVE PROM INSTRUCTION MEMORIES. THE M8204 HAS WRITEABLE CONTROL STORE. THE MEMORY SIZES BETWEEN ALL THREE PROCESSORS VARY ALSO.

2.0 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE M8207 STATIC LOGIC TESTS:

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

3.0 PRELIMINARY PROGRAM REQUIREMENTS

THE PROCESSOR AND MEMORY SHOULD BE THOROUGHLY TESTED PREVIOUS TO RUNNING THIS DIAGNOSTIC.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 DIAGNOSTIC SUPERVISOR

THIS PROGRAM IS COMPATIBLE WITH THE STANDALONE DIAGNOSTIC

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{E 1} PAGE 5
PROGRAM DOCUMENT

SEQ 0004

143

SUPERVISOR, AND MUST BE LOADED TO BE CO-RESIDENT WITH THE

144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198

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 TOTAL TIME REQUIRED TO RUN THE M8207 STATIC TESTS IS ABOUT 30 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.

4.7 MEMORY PARITY OPTION

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

4.8 ERROR LOGGING

THE NUMBER OF ERRORS WHICH HAVE OCCURRED ON EACH DEVICE UNDER TEST SINCE THE LAST START OR RESTART COMMAND IS KEPT IN AN ERROR LOG. THIS LOG MAY BE PRINTED BY USING THE "PRINT" COMMAND (SEE SECTION 6.3.8).

5.0 PROGRAM LOAD MEDIA

199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254

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 (DR>)
- C) ENTER STA<CR>
- D) ANSWER HARDWARE AND SOFTWARE QUESTIONS
- E) GET END OF PASS MESSAGES OR ERROR MESSAGES
- F) TO END EXECUTION, ENTER CONTROL/C

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
CZDMP-B-0

255

M8207 DIAG.#1 OF 2

256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311

UNIT IS M8200,M8204,OR M8207
DR>

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

6.3 PROGRAM OPTIONS

6.3.1 START COMMAND

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

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

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

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

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

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:

312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367

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) CONTAIN-
ING THE ERROR
IER INHIBIT ERROR REPORTING
IBE INHIBIT BASIC ERROR REPORTS
IXE INHIBIT EXTENDED ERROR REPORTS
PRI DIRECT ALL MESSAGES TO A LINE PRINTER
PNT PRINT NUMBER OF TEST BEING EXECUTED
BOE BELL ON ERROR
UAM RUN IN UNATTENDED MODE, BYPASSING MANUAL
INTERVENTION TESTS
ISR INHIBIT STATISTICAL REPORTS
IDU INHIBIT DROPPING OF UNITS BY DIAGNOSTIC
LOT LOOP ON TEST

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

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

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

6.3.1.5 EFFECT OF START COMMAND

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

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

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{K 1} PAGE 11
PROGRAM DOCUMENT

SEQ 0010

368

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

369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424

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(START)/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 DIALOGUE. 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

CZDMPB MB207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 M 1
PROGRAM DOCUMENT

SEQ 0012

425

COMMAND.

426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479

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 SUBSEI 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 WHFN 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>

480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534

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

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

6.3.4.2 EFFECT OF PROCEED COMMAND

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

535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589

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

6.3.7 PRINT COMMAND

PRI(NT)

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.

590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643

6.3.10 ZFLAGS COMMAND

ZFL(AGS)

6.3.10.1 EFFECT OF ZFLAGS COMMAND

ALL FLAGS ARE CLEARED.

6.3.11 CONTROL CHARACTERS

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

A CONTROL Z (Z) ENTERED DURING ONE OF THE THREE OPERATOR DIALOGUES- INITAIL DIALOGUE (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 SURPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR UNTIL ANOTHER O IS TYPED, WHICH RESTORES NORMAL TELETYPE OUTPUT.

6.3.12 HARDWARE PARAMETERS

THE FOLLOWING 4 QUESTION 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 CARRIGE RETURN RESPONSE.

1. WHICH MICRO-PROCESSOR: (O) 7?

THE ALLOWABLE RESPONSES ARE 0 (M8200), 4 (M8204), AND THE DEFAULT 7 (M8207).

2. MICRO-PROCESSOR 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.

644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699

3. MICRO-PROCESSOR VECTOR ADDRESS: (0) 300?

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

4. MICRO-PROCESSOR PRIORITY LEVEL: (0) 5?

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

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.

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.

700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754

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 1
<QUESTION 1> ? 75
<QUESTION 2> ? 0-6
<QUESTION 3> ? 76

UNIT 21
<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.

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 A 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).

755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810

7.0 TEST DESCRIPTION

***** TEST 1 *****
*VERIFY THAT REFERENCING UNIBUS DEVICE REGISTERS
*DOES NOT CAUSE A TIMEOUT TRAP

***** TEST 2 *****
*VERIFY THAT RUN CAN BE CLEARED

***** TEST 3 *****
*UNIBUS REGISTER WORD DUAL ADDRESSING TEST
*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
*THE SEQUENCE:
* 1. CLEAR REGISTER
* 2. WRITE PATTERN
* 3. VERIFY PATTERN
* 4. DO ALL 4 REGISTERS
* 5. READ ALL BACK IF ERRORS,
* DUAL ADDRESS PROBLEM.
*
* 1 IN REG 0
* 2 IN REG 2
* 3 IN REG 4
* 4 IN REG 6

***** TEST 4 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*FLOAT A ONE THROUGH BSEL 0
*CLEAR BIT0, VERIFY BIT0 WAS CLEARED

***** TEST 5 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT9, VERIFY BIT9 WAS SET
*CLEAR BIT9, VERIFY BIT9 WAS CLEARED

***** TEST 6 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT11, VERIFY BIT11 WAS SET
*CLEAR BIT11, VERIFY BIT11 WAS CLEARED

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 H 2 PAGE 21
PROGRAM DOCUMENT

SEQ 0020

811

812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867

***** TEST 7 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT12, VERIFY BIT12 WAS SET
*CLEAR BIT 12, VERIFY BIT 12 WAS CLEARED

***** TEST 8 *****
*CONTROL OUT REGISTER WRITE/READ TEST
*FLOAT A ONE THROUGH SEL2

***** TEST 9 *****
*PORT4 REGISTER WRITE/READ TEST
*FLOAT A ONE THROUGH PORT4 REGISTER
*FLOAT A ZERO THROUGH PORT4 REGISTER

***** TEST 10 *****
*PORT6 REGISTER WRITE/READ TEST
*FLOAT A ONE THROUGH PORT6 REGISTER
*FLOAT A ZERO THROUGH PORT6 REGISTER

***** TEST 11 *****
*UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING

***** TEST 12 *****
*MAINTENANCE INSTRUCTION REGISTER TEST
*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
*AND ALL ONES'. VERIFY THAT IS IS CLEARED ON A BUS RESET.

***** TEST 13 *****
*MICRO PROCESSOR TEST
*LOAD KMP06 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
*VERIFY INSTRUCTION EXECUTED PROPERLY
*INSTRUCTION SHOULD MOVE IBUS*4 TO IBUS*5, IBUS*4 IS ALL 1'S
*AND IBUS*5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4

868

869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924

***** TEST 14 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 0
FLOAT A 0 THROUGH IBUS REGISTER 0

***** TEST 15 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 2
FLOAT A 0 THROUGH IBUS REGISTER 2

***** TEST 16 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 4
FLOAT A 0 THROUGH IBUS REGISTER 4

***** TEST 17 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 5
FLOAT A 0 THROUGH IBUS REGISTER 5

***** TEST 18 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 10
FLOAT A 0 THROUGH IBUS REGISTER 10

***** TEST 19 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 11
FLOAT A 0 THROUGH IBUS REGISTER 11

***** TEST 20 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 0
FLOAT A 0 THROUGH IBUS REGISTER 0

***** TEST 21 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST

CZD'PB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 L 2
PAGE 25
PROGRAM DOCUMENT

SEQ 0024

925

*FLOAT A 1 THROUGH IBUS REGISTER 1

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

*FLOAT A 0 THROUGH IBUS REGISTER 1

***** TEST 22 *****
*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
*FLOAT A 1 THROUGH IBUS REGISTER 2
*FLOAT A 0 THROUGH IBUS REGISTER 2

***** TEST 23 *****
*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
*FLOAT A 1 THROUGH IBUS REGISTER 3
*FLOAT A 0 THROUGH IBUS REGISTER 3

***** TEST 24 *****
*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
*FLOAT A 1 THROUGH IBUS REGISTER 4
*FLOAT A 0 THROUGH IBUS REGISTER 4

***** TEST 25 *****
*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
*FLOAT A 1 THROUGH IBUS REGISTER 5
*FLOAT A 0 THROUGH IBUS REGISTER 5

***** TEST 26 *****
*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
*FLOAT A 1 THROUGH IBUS REGISTER 6
*FLOAT A 0 THROUGH IBUS REGISTER 6

***** TEST 27 *****
MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH IBUS REGISTER 7
FLOAT A 0 THROUGH IBUS REGISTER 7

***** TEST 28 *****
*MICRO PROCESSOR IBUS DUAL ADDRESS TEST
*WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
*READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING

982

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

***** TEST 29 *****
*MICRO PROCESSOR BR REGISTER TEST
*FLOAT A 1 THROUGH THE BR
*FLOAT A 0 THROUGH THE BR

***** TEST 30 *****
*SCRATCH PAD TEST
*FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
*FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION

***** TEST 31 *****
*SCRATCH PAD DUAL ADDRESSING TEST
*WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
*READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING

***** TEST 32 *****
*INTERRUPT TEST
*TEST THAT DEVICE CAN INTERRUPT TO VECTOR A

***** TEST 33 *****
*INTERRUPT TEST
*TEST THAT DEVICE CAN INTERRUPT TO VECTOR B

***** TEST 34 *****
*PRIORITY INTERRUPT TEST
*SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
*THE M8200,4,7 LEVEL, VERIFY THAT M8200,4,7 DOES NOT INTERRUPT

***** TEST 35 *****
*PRIORITY INTERRUPT TESTS
*SET PS TO ALL BR LEVELS LESS THAN THE M8200,4,7 LEVEL
*VERIFY THAT ALL M8200,4,7 WILL INTERRUPT

***** TEST 36 *****
*NPR TEST
*TEST OF DATO, 1 WORD FROM UPROC TO !1 MEMORY

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{C 3} PAGE 29
PROGRAM DOCUMENT

SEQ 0028

1039

1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
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

***** TEST 37 *****
*NPR TEST
*TEST OF DATI, 1 WORD FROM 11 MEMORY TO UPROC

***** TEST 38 *****
*NPR TEST
*TEST OF DATOB, 1 BYTE FROM UPROC TO 11 MEMORY

***** TEST 39 *****
*TEST OF EA BITS 16 AND 17
*DO A DATO TO AN ADDRESS USING OUT BA BITS 16 AND 17
*VERIFY CORRECT RESULTS

***** TEST 40 *****
*TEST OF EA BITS 16 AND 17
*DO A DATI USING IN BA BITS 16 AND 17
*VERIFY CORRECT RESULTS
*IN ORDER TO DO THIS TEST, WE WILL READ THE DATA FROM THE
*CONSOL TTY CSR IF ONE EXISTS
*IF NO COSOL TTY CSR AT ADDRESS 177560, THIS TEST
*WILL BE SKIPPED

***** TEST 41 *****
*NPR NON-EXISTENT MEMORY TEST
*DO A DATO TO A NON -EXISTENT ADDRESS
*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11

***** TEST 42 *****
*NPR NON-EXISTENT MEMORY TEST
*DO A DATI FROM A NON-EXISTENT ADDRESS
*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11

***** TEST 43 *****
*NPR TEST
*USING DATO, NPR A BINARY COUNT (0-377)
*FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY

1096

1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
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

***** TEST 44 *****
*ALU C BIT TEST
*TEST THAT AN ADD OF 377 AND 377 WILL SET THE C BIT

***** TEST 45 *****
*ALU TEST
*TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
*ALU FUNCTION (B) CODE=11
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 46 *****
*ALU TEST
*TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
*ALU FUNCTION (A) CODE=10
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 47 *****
*ALU TEST
*TEST OF ALU FUNCTION A OR NOTB WITH C BIT CLEARED
*ALU FUNCTION (A OR NOTB) CODE=12
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 48 *****
*ALU TEST
*TEST OF ALU FUNCTION A AND B WITH C BIT CLEARED
*ALU FUNCTION (A AND B) CODE=13
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 49 *****
*ALU TEST
*TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
*ALU FUNCTION (A OR B) CODE=14
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

1153

1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
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

***** TEST 50 *****

*ALU TEST
*TEST OF ALU FUNCTION A XOR B WITH C BIT
*ALU FUNCTION (A XOR B) CODE=15
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 51 *****

*ALU TEST
*TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
*ALU FUNCTION (A PLUS B) CODE=00
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 52 *****

*ALU TEST
*TEST OF ALU FUNCTION 2A W.C WITH C BIT CLEARED
*ALU FUNCTION (A PLUS A PLUS C) CODE=6
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 53 *****

*ALU TEST
*TEST OF ALU FUNCTION SUB WITH C BIT CLEARED
*ALU FUNCTION (A-B) CODE=16
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 54 *****

*ALU TEST
*TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
*ALU FUNCTION (A PLUS B PLUS C) CODE=01
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 55 *****

*ALU TEST
*TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED
*ALU FUNCTION (A-B-C) CODE=2
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 PAGE 35
PROGR M DOCUMENT

1 3

SEQ 0034

1210

1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266

***** TEST 56 *****
*ALU TEST
*TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
*ALU FUNCTION (A PLUS 1) CODE=3
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 57 *****
*ALU TEST
*TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
*ALU FUNCTION (A PLUS A) CODE=5
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 58 *****
*ALU TEST
*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
*ALU FUNCTION (A PLUS C) CODE=4
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 59 *****
*ALU TEST
*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED
*ALU FUNCTION (A-B-1) CODE=17
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 60 *****
*ALU TEST
*TEST OF ALU FUNCTION DEC A WITH C BIT CLEARED
*ALU FUNCTION (A-1) CODE=7
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 61 *****
*ALU TEST
*TEST OF ALU FUNCTION SEL B WITH C BIT SET
*ALU FUNCTION (B) CODE=11

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{K 3} PAGE 37
PROGRAM DOCUMENT

SEQ 0036

1267

*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA

1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
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

*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 62 *****

*ALU TEST
*TEST OF ALU FUNCTION SEL A WITH C BIT SET
*ALU FUNCTION (A) CODE=10
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 63 *****

*ALU TEST
*TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
*ALU FUNCTION (A OR NOTB) CODE=12
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 64 *****

*ALU TEST
*TEST OF ALU FUNCTION A AND B WITH C BIT SET
*ALU FUNCTION (A AND B) CODE=13
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 65 *****

*ALU TEST
*TEST OF ALU FUNCTION A OR B WITH C BIT SET
*ALU FUNCTION (A OR B) CODE=14
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 66 *****

*ALU TEST
*TEST OF ALU FUNCTION A XOR B WITH C BIT SET
*ALU FUNCTION (A XOR B) CODE=15
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 67 *****

*ALU TEST

1324

*TEST OF ALU FUNCTION ADD WITH C BIT SET

1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
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

*ALU FUNCTION (A PLUS B) CODE=00
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 68 *****
*ALU TEST
*TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
*ALU FUNCTION (A PLUS A PLUS C) CODE=6
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 69 *****
*ALU TEST
*TEST OF ALU FUNCTION SUB WITH C BIT SET
*ALU FUNCTION (A-B) CODE=16
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 70 *****
*ALU TEST
*TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
*ALU FUNCTION (A PLUS B PLUS C) CODE=01
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 71 *****
*ALU TEST
*TEST OF ALU FUNCTION SUB W/C WITH C BIT SET
*ALU FUNCTION (A-B-C) CODE=2
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 72 *****
*ALU TEST
*TEST OF ALU FUNCTION INC A WITH C BIT SET
*ALU FUNCTION (A PLUS 1) CODE=3
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

CZDMPB M8207 STATIC DIAG. #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{B 4} PAGE 41
PROGRAM DOCUMENT

SEQ 0040

1381

***** TEST 73 *****

1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
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

*ALU TEST
*TEST OF ALU FUNCTION 2A WITH C BIT SET
*ALU FUNCTION (A PLUS A) CODE=5
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 74 *****
*ALU TEST
*TEST OF ALU FUNCTION A PLUS C WITH C BIT SET
*ALU FUNCTION (A PLUS C) CODE=4
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 75 *****
*ALU TEST
*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
*ALU FUNCTION (A-B-1) CODE=17
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

***** TEST 76 *****
*ALU TEST
*TEST OF ALU FUNCTION DEC A WITH C BIT SET
*ALU FUNCTION (A-1) CODE=7
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULT

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
1454
1455
1456
1457
1458
1459
1460

8.0 ERROR INFORMATION

8.1 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.

CZDMP DVC FTL ERR 00003 TST 029 SUB 000 PC:022626

BR REGISTER DATA TEST
UNIT=00; FAILING UNIT ADDRESS=160170

GOOD	BAD
177776	000011

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

a

1461
1462
1463
1464
1465

1466 .TITLE CZDMPB0 M8207 STATIC DIAG #1
1467 002000 .=2000
1468
1469
1470
1471
1472

1473
1474 .MCALL SVC ; INITIALIZE SUPERVISOR MACROS
1475 002000 SVC
1476
1477

1478
1479
1480
1481 002J00 BGNMOD CZDMP
1482

1483
1484 000000 \$LSTIN= 0
1485 000000 \$LSTTAG= 0
1486 000000 SVCINS= 0 ; LIST INSTRUCTIONS, SHIFTED RIGHT
1487 000000 SVCTST= 0 ; LIST TEST TAGS, SHIFTED RIGHT
1488 000000 SVCSUB= 0 ; LIST SUBTEST TAGS, SHIFTED RIGHT
1489 000000 SVCGBL= 0 ; LIST GLOBAL TAGS, SHIFTED RIGHT
1490 000000 SVCTAG= 0 ; LIST OTHER TAGS, SHIFTED RIGHT

1491
1492 ; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
1493 ; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
1494 ; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
1495 ; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.

1496
1497
1498 .ENABL AMA

1499
 1500
 1501
 1502
 1503
 1504
 1505 002000
 1506
 1507
 1508 002000
 1509 002000
 1510 002000 103
 1511 002001 132
 1512 002002 104
 1513 002003 115
 1514 002004 120
 1515 002005 000
 1516 002006 000
 1517 002007 000
 1518 002010
 1519 002010 102
 1520 002011
 1521 002011 060
 1522 002012
 1523 002012 000000
 1524 002014
 1525 002014 000170
 1526 002016
 1527 002016 034630
 1528 002020
 1529 002020 000000
 1530 002022
 1531 002022 002364
 1532 002024
 1533 002024 000000
 1534 002026
 1535 002026 040004
 1536 002030
 1537 002030 000000
 1538 002032
 1539 002032 000000
 1540 002034
 1541 002034 000000
 1542 002036
 1543 002036 000000
 1544 002040
 1545 002040 002132
 1546 002042
 1547 002042 000000
 1548 002044
 1549 002044 000000
 1550 002046
 1551 002046 000000
 1552 002050
 1553 002050 003
 1554 002051 003

.SBTTL PROGRAM HEADER

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

POINTER BGNAU,BGNDU

HEADER CZDMP,B,0,120.,0

LSNAME:: :DIAGNOSTIC NAME
 .ASCII /C/
 .ASCII /Z/
 .ASCII /D/
 .ASCII /M/
 .ASCII /P/
 .BYTE 0
 .BYTE 0
 .BYTE 0
 LSREV:: :REVISION LEVEL
 .ASCII /B/
 :0
 .ASCII /O/
 LSDEPO:: :NUMBER OF UNITS
 .WORD 0
 LSTIML:: :LONGEST TEST TIME
 .WORD 120.
 LSHPCP:: :POINTER TO H.W. QUES.
 .WORD LSHARD
 LSSPCP:: :POINTER TO S.W. QUES.
 .WORD 0
 LSHPTP:: :PTR. TO DEF. H.W. PTABLE
 .WORD LSHW
 LSSPTP:: :PTR. TO S.W. PTABLE
 .WORD 0
 LSLADP:: :DIAG. END ADDRESS
 .WORD LSLAST
 LSSTA:: :RESERVED FOR APT STATS
 .WORD 0
 LSCO:: :DIAGNOSTIC TYPE
 .WORD 0
 LSDTYP:: :APT EXPANSION
 .WORD 0
 LSAPT:: :PTR. TO DISPATCH TABLE
 .WORD L\$DISPATCH
 LSPRIO:: :DIAGNOSTIC RUN PRIORITY
 .WORD 0
 L\$ENVI:: :FLAGS DESCRIBE HOW IT WAS SETUP
 .WORD 0
 L\$EXP1:: :EXPANSION WORD
 .WORD 0
 L\$MREV:: :SVC REV AND EDIT #
 .BYTE C\$REVISION
 .BYTE C\$EDIT

1555	002052		L\$EF::		;DIAG. EVENT FLAGS
1556	002052	000000	.WORD	0	
1557	002054	000000	.WORD	0	
1558	002056		L\$SPC::		
1559	002056	000000	.WORD	0	
1560	002060		L\$DEVP::		; POINTER TO DEVICE TYPE LIST
1561	002060	003130	.WORD	L\$DVTYP	
1562	002062		L\$REPP::		;PTR. TO REPORT CODE
1563	002062	000000	.WORD	0	
1564	002064		L\$EXP4::		
1565	002064	000000	.WORD	0	
1566	002066		L\$EXP5::		
1567	002066	000000	.WORD	0	
1568	002070		L\$AUT::		;PTR. TO ADD UNIT CODE
1569	002070	011364	.WORD	L\$AU	
1570	002072		L\$DUT::		;PTR. TO DROP UNIT CODE
1571	002072	011360	.WORD	L\$DU	
1572	002074		L\$LUN::		;LUN FOR EXERCISERS TO FILL
1573	002074	000000	.WORD	0	
1574	002076		L\$DESP::		;POINTER TO DIAG. DESCRIPTION
1575	002076	002414	.WORD	L\$DESC	
1576	002100		L\$LOAD::		;GENERATE SPECIAL AUTOLOAD EMT
1577	002100	104035	EMT	ESLOAD	
1578	002102		L\$ETP::		;POINTER TO ERRIBL
1579	002102	000000	.WORD	0	
1580	002104		L\$ICP::		;PTR. TO INIT CODE
1581	002104	010570	.WORD	L\$INIT	
1582	002106		L\$CCP::		;PTR. TO CLEAN-UP CODE
1583	002106	011354	.WORD	L\$CLEAN	
1584	002110		L\$ACP::		;PTR. TO AUTO CODE
1585	002110	011256	.WORD	L\$AUTO	
1586	002112		L\$PRT::		;PTR. TO PROTECT TABLE
1587	002112	002122	.WORD	L\$PROT	
1588	002114		L\$TEST::		;TEST NUMBER
1589	002114	000000	.WORD	0	
1590	002116		L\$DLY::		;DELAY COUNT
1591	002116	000000	.WORD	0	
1592	002120		L\$HIME::		;PTR. TO HIGH MEM
1593	002120	000000	.WORD	0	
1594					
1595					
1596					
1597	002122			BGNPROT	
1598	002122		L\$PROT::		
1599	002122	177777	.WORD	-1	
1600	002124	177777	.WORD	-1	
1601	002126	177777	.WORD	-1	
1602	002130			ENDPROT	
1603					

1604
1605
1606
1607
1608
1609
1610
1611 002130
1612 002130 000114
1613 002132
1614 002132 011366
1615 002134 011514
1616 002136 011560
1617 002140 011746
1618 002142 012112
1619 002144 012246
1620 002146 012376
1621 002150 012526
1622 002152 012670
1623 002154 013054
1624 002156 013240
1625 002160 013426
1626 002162 013576
1627 002164 013704
1628 002166 014134
1629 002170 014364
1630 002172 014614
1631 002174 015044
1632 002176 015330
1633 002200 015624
1634 002202 016054
1635 002204 016304
1636 002206 016534
1637 002210 016764
1638 002212 017214
1639 002214 017444
1640 002216 017674
1641 002220 020124
1642 002222 020422
1643 002224 020652
1644 002226 021216
1645 002230 021530
1646 002232 021672
1647 002234 022034
1648 002236 022210
1649 002240 022414
1650 002242 022560
1651 002244 022730
1652 002246 023074
1653 002250 023262
1654 002252 023476
1655 002254 023714
1656 002256 024046
1657 002260 024264
1658 002262 024424
1659 002264 024630

.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 76.

.WORD 76
L\$DISPATCH: :
.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
.WORD T43
.WORD T44
.WORD T45
.WORD T46

1660	002266	025034	.WORD	T47
1661	002270	025240	.WORD	T48
1662	002272	025444	.WORD	T49
1663	002274	025650	.WORD	T50
1664	002276	026054	.WORD	T51
1665	002300	026260	.WORD	T52
1666	002302	026464	.WORD	T53
1667	002304	026672	.WORD	T54
1668	002306	027076	.WORD	T55
1669	002310	027302	.WORD	T56
1670	002312	027506	.WORD	T57
1671	002314	027712	.WORD	T58
1672	002316	030116	.WORD	T59
1673	002320	030322	.WORD	T60
1674	002322	030526	.WORD	T61
1675	002324	030732	.WORD	T62
1676	002326	031136	.WORD	T63
1677	002330	031342	.WORD	T64
1678	002332	031546	.WORD	T65
1679	002334	031752	.WORD	T66
1680	002336	032156	.WORD	T67
1681	002340	032362	.WORD	T68
1682	002342	032566	.WORD	T69
1683	002344	032772	.WORD	T70
1684	002346	033176	.WORD	T71
1685	002350	033402	.WORD	T72
1686	002352	033606	.WORD	T73
1687	002354	034012	.WORD	T74
1688	002356	034216	.WORD	T75
1689	002360	034422	.WORD	T76
1690				
1691				
1692				
1693				
1694				
1695				
1696				

:LNT.ED DIFINED AT END OF PROGRAM TO BE LAST TEST NUMBER.

1697
1698
1699
1700
1701
1702
1703
1704
1705 002362
1706 002362 000013
1707 002364
1708 002364
1709
1710 002364 000007
1711 002366 160170
1712 002370 000300
1713 002372 005000
1714 002374 000003
1715 002376 000056
1716 002400 000000
1717 002402 000000
1718 002404 000000
1719 002406 000004
1720
1721
1722 002410 000000
1723 002412
1724 002412
1725
1726
1727
1728
1729

.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.
////////////////////////////////////

BGNHW DFPTBL
.WORD L10001-L\$HW/2
L\$HW::
DFPTBL::

.WORD 7 ; MICRO CPU TYPE
.WORD 160170 ; M8200,4,7 CSR ADDRESS
.WORD 300 ; M8200,4,7 VECTOR ADDRESS
.WORD 5000 ; INTERRUPT PRIORITY LEVEL
.WORD 3 ; LINE UNIT TYPE
.WORD 56 ; SWITCH PACK #1 (DDCMP LINE #)
.WORD 0 ; SWITCH PACK #2 (BM873 BOOT ADDRESS)
.WORD 0 ; SWITCH PACK #3
.WORD 0 ; TEST CONNECTOR INSTALLED FLAG
.WORD 4 ; CONTAINS BAUD RATE 4=56K BAUD DEFAULT
; 0=2.4K , 1=4.8K , 2=9.6K , 3=19.2K , 4=56K
; 5=250K , 6=500K , 7=1 MEG BAUD
; 0=RUN SW OFF , 1=SW ON
.WORD 0
ENDHW
L10001:

1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750

002412
002412 000000
002414
002414

002414
002414

```
.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  L10002-LSSW/2  
LSSW::  
SFPTBL::  
  
          ENDSW  
L10002:
```

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
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1798
 1799
 1800
 1801
 1802
 1803
 1804
 1805
 1806

002414

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
 000040
 000037

.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

1807 000036 EF.CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
1808 000035 EF.NEW== 29. ; A NEW PASS HAS BEEN STARTED
1809 000034 EF.PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED

1810 ;
1811 ;
1812 ; PRIORITY LEVEL DEFINITIONS
1813 ;

1814 000340 PRI07== 340
1815 000300 PRI06== 300
1816 000240 PRI05== 240
1817 000200 PRI04== 200
1818 000140 PRI03== 140
1819 000100 PRI02== 100
1820 000040 PRI01== 40
1821 000000 PRI00== 0

1822 ;
1823 ; OPERATOR FLAG BITS
1824 ;

1825 000004 EVL== 4
1826 000010 LOT== 10
1827 000020 ADR== 20
1828 000040 IDU== 40
1829 000100 ISR== 100
1830 000200 UAM== 200
1831 000400 BOE== 400
1832 001000 PNT== 1000
1833 002000 PRI== 2000
1834 004000 IXE== 4000
1835 010000 IBE== 10000
1836 020000 IER== 20000
1837 040000 LOE== 40000
1838 100000 HOE== 100000

1839 ;
1840 ;
1841 ;
1842 ;
1843 ;
1844 ;*****

1845 ;* INSTRUCTION DEFINITIONS
1846 ;*****
1847 022626 POP2SP=22626 ;INCREMENT STACK TWICE

1848 ;
1849 ;
1850 ;
1851 ;*****
1852 ;* PROGRAM EVENT FLAG DEFINITIONS
1853 ;*****

1854 ;
1855 ;
1856 ;

1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867 002414
1868 002414
1869 002414 034115 030062 020067
1870 002422 044504 043501 020056
1871 002430 030443 047440 020106
1872 002436 000062
1873
1874
1875
1876
1877
1878 002440 000000
1879 002442 000000
1880
1881
1882
1883
1884 002444 000000
1885 002506 002506
1886 002506 000000
1887 002550 002550
1888
1889
1890
1891
1892 002550 000000
1893 002552 000000
1894 002554 000000
1895 002556 000000
1896 002560 000000
1897 002562 000000
1898 002564 000000
1899 002566 000000
1900 002570 000000
1901 002572 000000
1902 002574 000000
1903 002576 000000
1904 002600 000000
1905 002602 000001
1906 002604 037776
1907 002606
1908 002606 000000
1909 002610 000001
1910 002612 000001
1911 002614 000001
1912 002616 000001

```
.SBTTL GLOBAL DATA SECTION
://////
:/ THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
:/ IN MORE THAN ONE TEST.
://////
:*****
:* STORAGE FOR DEVICE REGISTERS
:*****
:   DESCRIPT      <M8207 DIAG. #1 OF 2>
LSDESC:
:   .ASCIZ /M8207 DIAG. #1 OF 2/

.EVEN

:*****
:* PROGRAM CONTROL PARAMETERS
:*****
NEXT:  .WORD  0           ;ADDRESS OF NEXT TEST TO BE EXECUTED
LOCK:  .WORD  0           ;ADDRESS FOR LOCK CURRENT DATA

:*****
:* BUFFERS FOR INPUT-OUTPUT
:*****
TEMP:  0
.=.+40
MDATA: 0
.=.+40

:*****
:* MISCELLANEOUS STORAGE
:*****
$TMP0: .WORD  0           ;SCRATCH STORAGE
LOGDEV: .WORD  0           ;LOGICAL DEVICE NUMBER
PSTACK: .WORD  0           ;BASE LEVEL PROGRAM STACK POINTER
SUBRPC: .WORD  0           ;PC OF SUBR CALL FOR ERROR REPORTS
ERRFLG: .WORD  0           ;SUBROUTINE ERROR FLAG
RETADR: .WORD  0           ;SUBR ERROR RETURN ADDRESS
STRTSW: .WORD  0           ;SWITCHES AT START OF PROGRAM
STAT:   .WORD  0           ;M8200,4,7 STATUS WORD STORAGE
CLKX:   .WORD  0           ;
MASKX:  .WORD  0           ;
SAVSP:  .WORD  0           ;STACK POINTER STORAGE
SAVPC:  .WORD  0           ;PROGRAM COUNTER STORAGE
ZERO:   .WORD  0           ;
ONE:    .WORD  1           ;
MEMLIM: .WORD  MEMEND      ;HIGHEST LOCATION FOR NPR'S
MEMSZ:  .WORD  0           ;
KMACTV: .BLKW  1           ;M8200,4,7 SELECTED ACTIVE
KNUM:   .BLKW  1           ;OCTAL NUMBER OF M8200,4,7
SAVACT: .BLKW  1           ;ORIGINAL ACTIVE DEVICES
SAVNUM: .BLKW  1           ;WORKABLE NUMBER
```

1913	002620	000000	FLAG:	.WORD	0	:SCRATCH STORAGE
1914	002622	000000	RUN:	.WORD	0	:POINTER TO RUNNING DEVICES
1915	002624	000000	MRO:	.WORD	0	
1916	002626	000000	WTYPE:	.WORD	0	
1917	002630	000000	TYPE:	.WORD	0	
1918	002632	000000	\$GDADR:	.WORD	0	:CONTAINS ADDRESS OF 'GOOD' DATA
1919	002634	000000	\$BDADR:	.WORD	0	:CONTAINS ADDRESS OF 'BAD' DATA
1920	002636	000000	\$GDDAT:	.WORD	0	:CONTAINS 'GOOD' DATA
1921	002640	000000	\$BDDAT:	.WORD	0	:CONTAINS 'BAD' DATA
1922	002642	000000		.WORD	0	:RESERVED--NOT TO BE USED
1923	002644	000000		.WORD	0	
1924	002646	000000	FTIME:	.WORD	0	
1925	002650	000000	SAVE4:	.WORD	0	
1926	002652	000000	SAVE6:	.WORD	0	

 : * DATA PATTERNS

1931	002654	000	377	000	MEMDAT:	.BYTE	0,-1,0,-1,125,252,125,252
1932	002657	377	125	252			
1933	002662	125	252				
1934	002664	000	000	377	SPDAT:	.BYTE	0,0,-1,-1,125,125,252,252
1935	002667	377	125	125			
1936	002672	252	252				

.EVEN

 : * PROGRAM CONTROL FLAGS

1942	002674	000			INIFLG:	.BYTE	0	:PROGRAM INITIALIZING FLAG
1943		002676				.EVEN		
1944	002676	000			LOKFLG:	.BYTE	0	:LOCK ON CURRENT TEST FLAG
1945	002677	000			QV.FLG:	.BYTE	0	:QUICK VERIFY FLAG
1946						.EVEN		

 : * DEFINITION OF M8200,4,7 STATUS WORDS - STAT1,STAT2,STAT3

STAT1 - BITS 00-08 IS M8200,4,7 VECTOR ADDRESS
 BIT15=1 LINE UNIT IS AN M8203
 BIT14=0 NO TEST CONNECTOR(S) USED
 BIT14=1 H-XXX TEST CONNECTOR WILL BE USED
 BIT13=0 LINE UNIT IS AN M8201
 BIT13=1 LINE UNIT IS AN M8202
 BIT12=1 NO LINE UNIT
 BITS 09-11 IS M8200,4,7 PRIORITY LEVEL

STAT2 - LOW BYTE IS SWITCH PACK #1 (DDCMP LINE NUMBER)
 HIGH BYTE IS SWITCH PACK #2 (BM873 BOOT ADDRESS)

STAT3 - BIT0=1 DO FREE RUNNING TESTS ON M8200,4,7

 STAT1: .WORD 0
 STAT2: .WORD 0

1967	002700	000000
1968	002702	000000

1969 002704 000000
1970
1971
1972
1973
1974 002706 000000
1975 002710 000000
1976 002712 000000
1977 002714 000000
1978 002716 000000
1979 002720 000000
1980 002722 000000
1981 002724 000000
1982 002726 000000
1983
1984
1985
1986 002730
1987
1988
1989 002730 000100
1990 003130
1991
1992
1993
1994
1995
1996
1997

STAT3: .WORD 0

; * POINTERS TO MB200,4,7 VECTORS AND REGISTERS

KMRVEC: 0 ; POINTER TO MB200,4,7 RCV INTRPT VECTOR
KMRLVL: 0 ; POINTER TO MB200,4,7 RCV INTRPT SERVICE PS
KMTVEC: 0 ; POINTER TO MB200,4,7 TX INTRPT VECTOR
KMTLVL: 0 ; POINTER TO MB200,4,7 TX INTRPT SERVICE PS
KMCSR: 0 ; POINTER TO MB200,4,7 CONTROL STATUS REGISTER
KMCSRH: 0 ; POINTER TO MB200,4,7 CONTROL STATUS REGISTER HIGH BYTE
KMCTL: 0 ; POINTER TO MB200,4,7 CONTROL OUT REGISTER
KMPO4: 0
KMPO6: 0 ; POINTER TO MB200,4,7 PORT REGISTER - SEL6

***** PRIMARY REG ADRS STORAGE FOR THIS UNIT *****
; THESE LOCATIONS WILL BE LOADED FOR THE CURRENT UNIT, IN INIT CODE
REGADR:

***** STACK USED FOR SUBROUTINE LINKAGE *****
.BLKW 100
SSTACK:

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
2079
2080
2081
2082

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

:-----
: MACRO'S NEEDED TO CALL SUBROUTINES
:-----

.MACRO ERROR,XYX,ZZ
MOV R4,$BDDAT
.IF B ZZ
MOV R2,$GDDAT
.ENDC
MOV MRO,$BDADR
ERRDF XXX',EM'XXX',ERR'XXX'
.ENDM
.MACRO RERROR XXX
MOV R4,$BDDAT
CLRB $BDDAT+1
CLRB $GDDAT+1
MOV R2,$GDADR
ERRDF XXX',EM'XXX',ERR'XXX'
.ENDM
.MACRO BERROR XXX
MOV R4,$BDDAT
MOV R5,$GDDAT
CLRB $BDDAT+1
CLRB $GDDAT+1
ERRDF XXX',EM'XXX',ERR'XXX'
.ENDM
.MACRO ED$CALL XY
.LIST
:***** TEST 'XY' *****
.NLIST
.ENDM
.MACRO BADHEAD
.RADIX 10
ED$CALL \T$TESTNUM+1
.RADIX 8
.ENDM
.MACRO K4ONLY ?N2
.LIST
: DON'T DO TEST IF M8200 OR M8204
.NLIST
CMP MEMSZ,#2000
BNE N2
EXIT TST
N2:
.ENDM
.MACRO MYINT
.LIST
MOV KMCSR,R1 ;GET DEVICE ADDRESS.
```



```

2083 .NLIST
2084 .ENDM
2085
2086 .MACRO ROMCLK
2087 .LIST
2088 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2089 .NLIST
2090 .ENDM
2091
2092 .MACRO MSTCLR
2093 .LIST
2094 JSR R5,.MSTCLR ;CLEAR M8200,4,7
2095 .NLIST
2096 .ENDM
2097
2098 003156 .MSTCLR:
2099 003156 112777 000100 177534 MOVB #BIT6,@KMCSRH ;SET INST.
2100 003164 142777 000300 177526 BICB #BIT6 BIT7,@KMCSRH
2101 003172 000205 RTS R5
2102
2103
2104 003174 000024 ;.BLKW 20. ;PATCH AREA.
2105
2106
2107
2108 003244 ENDBUG:
2109 ; UNSAFE TO PATCH ANY OTHER AREA.
2110
2111
2112
2113 003244 .ROMCLK:
2114 003244 152777 000002 177446 BISB #BIT1,@KMCSRH
2115 003252 012577 177450 MOV (R5)+,@KMP06
2116 003256 152777 000003 177434 BISB #BIT1 BIT0,@KMCSRH
2117 003264 142777 000007 177426 BICB #BIT2:BIT1:BIT0,@KMCSRH
2118 003272 000205 RTS R5
2119
2120 003274 CLRALL:
2121 ;CLEARS C & Z BITS AND BR
2122 003274 ROMCLK
2123 003274 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2124 003300 000400 400 ;0 TO BR
2125 003302 ROMCLK
2126 003302 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2127 003306 063220 63220 ;SP(0) TO BR
2128 003310 ROMCLK
2129 003310 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2130 003314 060400 60400 ;BR,SP(0) + BR
2131 003316 000207 RTS PC
2132
2133 003320 SETBRO:
2134 ;SETS BRO BIT
2135 003320 ROMCLK
2136 003320 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2137 003324 000401 401 ;1 TO BR
2138 003326 000207 RTS PC
    
```

```
2139
2140 003330 SETBR1: ;THIS SUBROUTINE SETS BR1 BIT
2141
2142
2143 003330 ROMCLK ;NEXT WORD IS INSTRUCTION
2144 003330 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2145 003334 000402 ;BR_002
2146 003336 000207 RTS PC
2147
2148 003340 SETBR4: ;THIS SUBROUTINE SETS BR4 BIT
2149
2150
2151 003340 ROMCLK ;NEXT WORD IS INSTRUCTION
2152 003340 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2153 003344 000402
2154 003346 000207 RTS PC
2155
2156 003350 SETBR7: ;THIS SUBROUTINE SETS BR7 BIT
2157
2158
2159 003350 ROMCLK ;NEXT WORD IS INSTRUCTION
2160 003350 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2161 003354 000600
2162 003356 000207 RTS PC
2163
2164
2165 003360 SETZ: ;THIS SUBROUTINE SETS THE Z BIT
2166
2167
2168 003360 ROMCLK ;NEXT WORD IS INSTRUCTION
2169 003360 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2170 003364 000777 ;BR_377
2171 003366 000207 RTS PC
2172
2173 003370 RAMDAT: ;THIS SUBROUTINE LOADS R4 WITH THE LOWEST
2174 ;8 BITS OF THE CRAM PC.
2175
2176
2177 003370 017635 000000 MOV @ (SP),R5 ;GOOD DATA
2178 003374 062715 000002 ADD #2,(SP) ;ADJUST STACK
2179 003400 00501 CLR (R1) ;CLEAR BIT10
2180 003402 052711 000400 BIS #BIT8,(R1) ;CLOCK INSTRUCTION IN CRAM THAT
;JUMPED TO, IT LOADS BR WITH IT
2181
2182 003406 005011 CLR (R1) ;CLR BIT8
2183 003410 ROMCLK ;NEXT WORD IS INSTRUCTION
2184 003410 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2185 003414 061225 ;MOV BR TO PORT 5
2186 003416 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' IN R4
2187 003422 000207 RTS PC ;RETURN
2188
2189
2190 003424 MEMSET: ;THIS SUBROUTINE LOADS CRAM WITH SPECIAL INSTRUCTIONS
2191 ;FOR THE CRAM JUMP TEST. ALL CRAM LOCATIONS ARE LOADED
2192 ;WITH INSTRUCTIONS THAT MOVE A 37 TO THE BR, EXCEPT THE
2193 ;FOLLOWING CRAM ADDRESSES: 0,1,4,7,525,1777. THESE LOCATIONS
2194
```

```
2195 ;CONTAIN INSTRUCTIONS WHICH LOAD THE BR WITH THE LOWEST
2196 ;8 BITS OF THAT CRAM ADDRESS.
2197
2198 003424 005000 CLR R0 ;R0 = CRAM ADDRESS
2199 003426 012711 002000 1$: MOV #BIT10,(R1) ;SET POMO
2200 003432 010061 000004 MOV R0,4(R1) ;LOAD CRAM ADDRESS
2201 003436 012761 000437 000006 MOV #4$7,6(R1) ;LOAD INSTRUCTION
2202 003444 052711 020000 BIS #BIT13,(R1) ;WRITE INSTRUCTION IN CRAM
2203 003450 005200 INC R0 ;NEXT ADDRESS
2204 003452 022700 002000 CMP #2000,R0 ;DONE YET?
2205 003456 001363 BNE 1$ ;BR IF NO
2206 003460 005000 CLR R0 ;INDEX REGISTER
2207 003462 012711 002000 2$: MOV #BIT10,(R1) ;SET ROMO
2208 003466 016061 003522 000004 MOV CRAMA(R0),4(R1) ;LOAD CRAM ADDRESS IN SEL4
2209 003474 016061 003536 000006 MOV INSTU(R0),6(R1) ;LOAD INSTRUCTION TO BE WRITTEN
2210 003502 052711 020000 BIS #BIT13,(R1) ;WRITE CRAM.
2211 003506 005720 TST (R0)+ ;NEXT
2212 003510 022700 000014 CMP #14,R0 ;DONE YET?
2213 003514 001362 BNE 2$ ;BR IF NO
2214 003516 005011 CLR (R1) ;CLEAR ALL BITS
2215 003520 000207 RTS PC ;RETURN
2216
2217 003522 000000 000001 000004 CRAMA: .WORD 0,1,4,7,1777,525
2218 003530 000007 001777 000525
2219
2220 003536 000400 INSTU: 000400 ;BR_0
2221 003540 000401 000401 ;BR_1
2222 003542 000404 000404 ;BR_4
2223 003544 000407 000407 ;BR_7
2224 003546 000777 000777 ;BR_377
2225 003550 000525 000525 ;BR_125
2226
2227 003552 SETVEC:
2228 ;THIS SUBROUTINE LOADS THE VECTORS AND VECTOR LEVELS
2229
2230 003552 012577 177130 MOV (R5)+,@KMRVEC ;LOAD BASE VECTOR
2231 003556 012577 177130 MOV (R5)+,@KMTVEC ;LOAD VECTOR + 2
2232 003562 012577 177122 MOV (R5)+,@KMRLVL ;LOAD VECTOR + 4
2233 003566 012577 177122 MOV (R5)+,@KMTLVL ;LOAD VECTOR + 6
2234 003572 000205 RTS R5 ;RETURN
2235
2236
2237 003574 NPRSET:
2238 ;THIS SUBROUTINE LOADS IBUS REGISTERS 0-7
2239 ;WITH NPR INFORMATION (INBA, OUTBA, OUT DATA)
2240
2241 003574 010246 MOV R2,-(SP) ;SAVE R2
2242 003576 005002 CLR R2 ;START AT IBUS REG 0
2243 003600 112561 000004 1$: MOVB (R5)+,4(R1) ;LOAD PORT4
2244 003604 042737 000017 003622 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
2245 003612 050237 003622 BIS R2,2$ ;ADD ADDRESS TO INSTRUCTION
2246 003616 ROMCLK
2247 003616 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
2248 003622 122100 2$: 122100 ;MOVE PORT4 TO IBUS REG
2249 003624 005202 INC R2 ;NEXT ADDRESS
2250 003626 022702 000010 CMP #10,R2 ;ALL DONE?
```

```

2251 003632 001362          BNE      1$          ;BR IF NO
2252 003634 012602          MOV      (SP)+,R2    ;RESTORE R2
2253 003636 000205          RTS      R5          ;RETURN
2254
2255
2256 003640          MEMLD:
2257          ;THIS SUBROUTINE LOADS THE FIRST 8 LOCATIONS OF MAIN
2258          ;MEMORY WITH THIS DATA: 0,-1,,0,-1,125,252,125,252
2259
2260 003640 013637 002550      MOV      @(SP)+,$TMP0 ;PUT POINTER TO DATA IN R0
2261 003644 062746 000002      ADD      #2,-(SP)    ;ADJUST STACK
2262
2263 003650 013700 002550      MEMLD2: MOV      $TMP0,R0 ;GET ADDR.
2264 003654 012704 000010      MOV      #10,R4     ;DO 8 LOADS
2265 003660
2266 003660 004537 003244      ROMCLK   JSR      R5,.$ROMCLK ;CLOCK INSTRUCTION
2267 003664 010000          010000 ;MAR < 0
2268 003666          ROMCLK   ;CLR      MAR HI
2269 003666 004537 003244      JSR      R5,.$ROMCLK ;CLOCK INSTRUCTION
2270 003672 004000
2271 003674 112077 177024      1$:     MOV      (R0)+,@KMP04 ;LOAD PORT4
2272 003700          ROMCLK
2273 003700 004537 003244      JSR      R5,.$ROMCLK ;CLOCK INSTRUCTION
2274 003704 136500          136500 ;MOV DATA TO MEM, AUTO INC MAR
2275 003706 005304          DEC      R4         ;DECREMENT COUNT
2276 003710 001371          BNE      1$         ;BR IF NOT DONE
2277
2278 003712          ROMCLK   ;LOAD MEM ADDR. 0
2279 003712 004537 003244      JSR      R5,.$ROMCLK ;CLOCK INSTRUCTION
2280 003716 010000          10000
2281 003720 012703 000010      MOV      #10,R3     ;CHECK 8. MEM LOCS.
2282 003724 013700 002550      MOV      $TMP0,R0
2283 003730          2$:     ROMCLK   ;READ FROM MEM,PUT INTO PORT 4
2284 003730 004537 003244      JSR      R5,.$ROMCLK ;CLOCK INSTRUCTION
2285 003734 055224          55224
2286
2287 003736 112037 002636      MOV      (R0)+,$GDDAT ;EXPECTED.
2288 003742 117704 176756      MOV      @KMP04,R4  ;PECIEVED.
2289 003746 123704 002636      CMPB    $GDDAT,R4   ;OK?
2290 003752 001414          BEQ      3$
2291 003754          ERROR   36
2292 003772 104455          TRAP    C$ERDF
2293 003774 000044          .WORD  36
2294 003776 005640          .WORD  EM36
2295 004000 010432          .WORD  ERR36
2296 004002 000402          BR      4$
2297 004004 005303          3$:     DEC      R3         ;CHECKED ALL?
2298 004006 001350          BNE      2$         ;NO-DO NEXT ONE.
2299 004010          4$:
2300 004010 000207          RTS      PC         ;RETURN
2301
2302
2303 004012          SPLD:
2304          ;THIS SUBROUTINE LOADS THE FIRST 8 SCRATCH PAD
2305          ;LOCATIONS WITH: 0,0,-1,-1,125,125,252,252
2306

```

```
2307 004012 013600          MOV    @(SP)+,R0      ;PUT POINTER TO DATA IN R5
2308 004014 062746 000002    ADD    #2,-(SP)      ;ADJUST STACK
2309 004020 005004          CLR    R4            ;START AT SP ADDRESS 0
2310 004022 112077 176676    MOVB  (R0)+,@KMP04   ;LOAD PORT4 WITH DATA
2311 004026 042737 000017    BIC   #17,2$        ;CLEAR ADDRESS FIELD OF INSTRUCTION
2312 004034 050437 004044    BIS   R4,2$         ;ADD ADDRESS TO INSTRUCTION
2313 004040          ROMCLK
2314 004040 004537 003244    JSR   R5,ROMCLK     ;CLOCK INSTRUCTION
2315 004044 123100          2$: 123100          ;MOVE DATA TO SP
2316 004046 005204          INC   R4            ;INCREMENT COUNT
2317 004050 022704 000010    CMP   #10,R4       ;DONE YET?
2318 004054 001362          BNE  1$            ;BR IF NO
2319 004056 000207          RTS   PC           ;RETURN
2320
2321
2322 004060          CLRC:
2323          ;THIS SUBROUTINE CLEARS THE MICRO PROCESSOR C BIT
2324
2325 004060          ROMCLK
2326 004060 004537 003244    JSR   R5,ROMCLK     ;CLOCK INSTRUCTION
2327 004064 010000          010000          ;MAR_0
2328 004066          ROMCLK
2329 004066 004537 003244    JSR   R5,ROMCLK     ;CLOCK INSTRUCTION
2330 004072 040400          040400!<0*20>   ;CLEAR C BIT
2331 004074 000207          RTS   PC           ;RETURN
2332
2333
2334 004076          SETC:
2335          ;THIS SUBROUTINE SETS THE MICRO PROCESSOR C BIT
2336
2337 004076          ROMCLK
2338 004076 004537 003244    JSR   R5,ROMCLK     ;CLOCK INSTRUCTION
2339 004102 010003          010003          ;MAR_3
2340 004104          ROMCLK
2341 004104 004537 003244    JSR   R5,ROMCLK     ;CLOCK INSTRUCTION
2342 004110 040403          040403!<0*20>   ;SET C BIT
2343 004112 000207          RTS   PC           ;RETURN
2344
2345
2346
2347
```

2348
2349
2350
2351
2352
2353

.SBTTL GLOBAL ERROR REPORT SECTION

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

2380						
2381						
2382	004261	122	043505	051511	EM1:	.ASCIZ ®ISTER ADDRESS TEST&
2383	004266	042524	020122	042101		
2384	004274	051104	051505	020123		
2385	004302	042524	052123	000		
2386	004307	111	052502	025123	EM2:	.ASCIZ &IBUS* REGISTER DUAL ADDRESSING TEST&
2387	004314	051040	043505	051511		
2388	004322	042524	020122	052504		
2389	004330	046101	040440	042104		
2390	004336	042522	051523	047111		
2391	004344	020107	042524	052123		
2392	004352	000				
2393	004353	111	052502	020123	EM30:	.ASCIZ ''IBUS REGISTER DUAL ADDRESSING TEST''
2394	004360	042522	044507	052123		
2395	004366	051105	042040	040525		
2396	004374	020114	042101	051104		
2397	004402	051505	044523	043516		
2398	004410	052040	051505	000124		
2399	004416	051102	051040	043505	EM3:	.ASCIZ /BR REGISTER DATA TEST/
2400	004424	051511	042524	020122		
2401	004432	040504	040524	052040		
2402	004440	051505	000124			

2403	004444	041523	040522	041524	EM4: .AS/IZ /SCRATCH PAD DATA TEST/
2404	004452	020110	040520	020104	
2405	004460	040504	040524	052040	
2406	004466	051505	000124		

2407 004472 041523 040522 041524 EMS: .ASCIZ /SCRATCH PAD DUAL ADDRESSING TEST/
2408 004500 020110 040520 020104
2409 004506 052504 046101 040440
2410 004514 042104 042522 051523
2411 004522 047111 020107 042524
2412 004530 052123 000

CZDMPBO M8207 STATIC DIAG #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50 ^{D 6} PAGE 69
GLOBAL ERROR REPORT SECTION

SEQ 0068

2413	004533	115	044501	020116	EM6:	.ASCIZ /MAIN MEMORY DATA TEST/
2414	004540	042515	047515	054522		
2415	004546	042040	052101	020101		
2416	004554	042524	052123	000		
2417	004561	115	044501	020116	EM7:	.ASCIZ /MAIN MEMORY DUAL ADDRESSING TEST/
2418	004566	042515	047515	054522		
2419	004574	042040	040525	020114		
2420	004602	042101	051104	051505		
2421	004610	044523	043516	052040		
2422	004616	051505	000124			

2423	004622	052501	047524	046440	EM10:	.ASCIZ	/AUTO MARINC FUNCTION TEST/
2424	004630	051101	047111	020103			
2425	004636	052506	041516	044524			
2426	004644	047117	052040	051505			
2427	004652	000124					
2428	004654	050116	020122	042524	EM11:	.ASCIZ	/NPR TEST/
2429	004662	052123	000				
2430	004665	115	046125	044524	EM12:	.ASCIZ	/MULTIPLE NPR TEST/
2431	004672	046120	020105	050116			
2432	004700	020122	042524	052123			
2433	004706	000					
2434	004707	116	047117	042440	EM13:	.ASCIZ	/NON EX MEM FAILED/
2435	004714	020130	042515	020115			
2436	004722	040506	046111	042105			
2437	004730	000					
2438	004731	120	047522	051107	EM14:	.ASCIZ	/PROGRAM CLOCK TEST/
2439	004736	046501	041440	047514			
2440	004744	045503	052040	051505			
2441	004752	000124					
2442	004754	046101	020125	052506	EM15:	.ASCIZ	/ALU FUNCTION WITH C BIT CLEAR TEST/
2443	004762	041516	044524	047117			
2444	004770	053440	052111	020110			
2445	004776	020103	044502	020124			
2446	005004	046103	040505	020122			
2447	005012	042524	052123	000			

2448	005017	120	053517	051105	EM16:	.ASCIZ /POWER FAIL: BUS INIT WAS NOT BLOCKED/
2449	005024	043040	044501	035114		
2450	005032	041040	051525	044440		
2451	005040	044516	020124	040527		
2452	005046	020123	047516	020124		
2453	005054	046102	041517	042513		
2454	005062	000104				
2455	005064				EM35:	
2456	005064	047506	041522	020105	EM17:	.ASCIZ /FORCE POWER FAIL ERROR/
2457	005072	047520	042527	020122		
2458	005100	040506	046111	042440		
2459	005106	051122	051117	000		
2460	005113	116	044517	042523	EM20:	.ASCIZ /NOISE TEST ON IBUS*,IBUS,SPAD,MEMORY/
2461	005120	052040	051505	020124		
2462	005126	047117	044440	052502		
2463	005134	025123	044454	052502		
2464	005142	026123	050123	042101		
2465	005150	046454	046505	051117		
2466	005156	000131				
2467	005160	046101	020125	020103	EM21:	.ASCIZ /ALU C BIT TEST FAILURE/
2468	005166	044502	020124	042524		
2469	005174	052123	043040	044501		
2470	005202	052514	042522	000		
2471	005207	124	046511	020105	EM22:	.ASCIZ /TIME OUT ERROR/
2472	005214	052517	020124	051105		
2473	005222	047522	000122			
2474	005226	046101	020125	052506	EM23:	.ASCIZ /ALU FUNCTION TEST WITH C BIT SET/
2475	005234	041516	044524	047117		
2476	005242	052040	051505	020124		
2477	005250	044527	044124	041440		
2478	005256	041040	052111	051440		
2479	005264	052105	000			
2480	005267	125	041520	051440	EM24:	.ASCIZ /UPC SEQUENCE ERROR/
2481	005274	050505	042525	041516		
2482	005302	020105	051105	047522		
2483	005310	000122				
2484	005312	050125	043040	044501	EM31:	.ASCIZ 'UP FAILED TO INTERRUPT'
2485	005320	042514	020104	047524		
2486	005326	044440	052116	051105		
2487	005334	052522	052120	000		
2488	005341	125	020120	047111	EM32:	.ASCIZ 'UP INTERRUPTED TO WRONG VECTOR'
2489	005346	042524	051122	050125		
2490	005354	042524	020104	047524		
2491	005362	053440	047522	043516		
2492	005370	053040	041505	047524		
2493	005376	000122				
2494	005400	047125	054105	042520	EM33:	.ASCIZ 'UNEXPECTED INTERRUPT FROM UP'
2495	005406	052103	042105	044440		
2496	005414	052116	051105	052522		
2497	005422	052120	043040	047522		
2498	005430	020115	050125	000		
2499	005435	101	052514	043040	EM34:	.ASCIZ 'ALU FLAG TEST'
2500	005442	040514	020107	042524		
2501	005450	052123	000			
2502	005453	110	046105	020114	EM25:	.ASCIZ /HELL RAISER TEST/
2503	005460	040522	051511	051105		

2504	005466	052040	051505	000124	
2505	005474	040515	047111	040524	EM26: .ASCIZ /MAINTANCE REGISTER ERROR/
2506	005502	041516	020105	042522	
2507	005510	044507	052123	051105	
2508	005516	042440	051122	051117	
2509	005524	000			
2510	005525	111	052502	025123	EM27: .ASCIZ 'IBUS* WRITE/READ ERROR'
2511	005532	053440	044522	042524	
2512	005540	051057	040505	020104	
2513	005546	051105	047522	000122	
2514	005554	047111	052123	052522	EM28: .ASCIZ /INSTRUCTION TEST FAILURE/
2515	005562	052103	047511	020116	
2516	005570	042524	052123	043040	
2517	005576	044501	052514	042522	
2518	005604	000			
2519	005605	111	052502	027523	EM29: .ASCIZ 'IBUS/OBUS WRITE/READ ERROR'
2520	005612	041117	051525	053440	
2521	005620	044522	042524	051057	
2522	005626	040505	020104	051105	
2523	005634	047522	000122		
2524					
2525	005640	047511	020120	040515	EM36: .ASCIZ 'IOP MAIN MEM. LOAD ERROR-RUN MCPU MEM. DIAG.'
2526	005646	047111	046440	046505	
2527	005654	020056	047514	042101	
2528	005662	042440	051122	051117	
2529	005670	051055	047125	046440	
2530	005676	050103	020125	042515	
2531	005704	027115	042040	040511	
2532	005712	027107	000		
2533	005715	000			EM37: .ASCIZ //
2534					
2535					

2536

CZDMPBO M8207 STATIC DIAG #1 MACY11 30A(1052) 20-AUG-80 07:50 ^{1 6} PAGE 74
CZDMPB.P11 20-AUG-80 07:48 GLOBAL ERROR REPORT SECTION

SEQ 0073

2537 005716 000

DHO: .ASCIZ //

2560
2561
2562

2563
2564

: MACRO'S NEEDED TO REPORT ERRORS

2565
2566
2567
2568
2569
2570
2571
2572
2573

```
-----  
.MACRO MDT0  
.ENDM  
  
.MACRO MDT1  
PRINTB #T,M1,$GDDAT,$BDDAT,$GDADR  
.ENDM  
  
.MACRO MDT2
```

```
2574 PRINTB #TFM2,$GDDAT,$BDDAT
2575 .ENDM
2576
2577 .MACRO MDT5
2578 PRINTB #TFM5,$GDDAT,$BDDAT
2579 .ENDM
```

2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593

```
.MACRO MDT27  
  PRINTB #TFM27,PRO,SGDDAT,SBDDAT  
.ENDM  
  
.MACRO SMD,ERNB,ERHM,ERFM  
.MLIST  
: ERNB = ERROR NUMBER  
: ERFM = FORMAT NUMBER  
: ERHM = HEADER NUMBER  
.LIST  
  BGNMSG ERR'ERNB'  
  PRINTB #FM1,#DH'ERHM'  
  MDT'ERFM'  
  ENDMMSG
```

```

2594
2595
2596
2597
2598
2599 006054
2600 006054
2601 006054 012746 005750
2602 006060 012746 004114
2603 006064 012746 000002
2604 006070 010600
2605 006072 104414
2606 006074 062706 000006
2607 006100 013746 002640
2608 006104 013746 002636
2609 006110 012746 004146
2610 006114 012746 000003
2611 006120 010600
2612 006122 104414
2613 006124 062706 000010
2614 006130
2615 006130 104423
2616 006132
2617 006132
2618 006132 012746 005750
2619 006136 012746 004114
2620 006142 012746 000002
2621 006146 010600
2622 006150 104414
2623 006152 062706 000006
2624 006156 013746 002640
2625 006162 013746 002636
2626 006166 012746 004146
2627 006172 012746 000003
2628 006176 010600
2629 006200 104414
2630 006202 062706 000010
2631 006206
2632 006206 104423
2633 006210
2634 006210
2635 006210 012746 005750
2636 006214 012746 004114
2637 006220 012746 000002
2638 006224 010600
2639 006226 104414
2640 006230 062706 000006
2641 006234 013746 002640
2642 006240 013746 002636
2643 006244 012746 004146
2644 006250 012746 000003
2645 006254 010600
2646 006256 104414
2647 006260 062706 000010
2648 006264
2649 006264 104423
    
```

```

.ENDM
ERR1:: SMD 1,2,2
MOV #DH2,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #6,SP
MOV $BDDAT,-(SP)
MOV $GDDAT,-(SP)
MOV #TFM2,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #10,SP
L10003: TRAP C$MSG
SMD 2,2,2
ERR2:: MOV #DH2,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #6,SP
MOV $BDDAT,-(SP)
MOV $GDDAT,-(SP)
MOV #TFM2,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #10,SP
L10004: TRAP C$MSG
SMD 3,2,2
ERR3:: MOV #DH2,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #6,SP
MOV $BDDAT,-(SP)
MOV $GDDAT,-(SP)
MOV #TFM2,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C$PNTB
ADD #10,SP
L10005: TRAP C$MSG
    
```

2650	006266				SMD	4,1,1
2651	006266				ERR4::	
2652	006266	012746	005717		MOV	#DH1,-(SP)
2653	006272	012746	004114		MOV	#FM1,-(SP)
2654	006276	012746	000002		MOV	#2,-(SP)
2655	006302	010600			MOV	SP,R0
2656	006304	104414			TRAP	C\$PNTB
2657	006306	062706	000006		ADD	#6,SP
2658	006312	013746	002632		MOV	\$GDADR,-(SP)
2659	006316	013746	002640		MOV	\$BDDAT,-(SP)
2660	006322	013746	002636		MOV	\$GDDAT,-(SP)
2661	006326	012746	004123		MOV	#TFM1,-(SP)
2662	006332	012746	000004		MOV	#4,-(SP)
2663	006336	010600			MOV	SP,R0
2664	006340	104414			TRAP	C\$PNTB
2665	006342	062706	000012		ADD	#12,SP
2666	006346				L10006:	
2667	006346	104423			TRAP	C\$MESSG
2668	006350				SMD	5,1,1
2669	006350				ERR5::	
2670	006350	012746	005717		MOV	#DH1,-(SP)
2671	006354	012746	004114		MOV	#FM1,-(SP)
2672	006360	012746	000002		MOV	#2,-(SP)
2673	006364	010600			MOV	SP,R0
2674	006366	104414			TRAP	C\$PNTB
2675	006370	062706	000006		ADD	#6,SP
2676	006374	013746	002632		MOV	\$GDADR,-(SP)
2677	006400	013746	002640		MOV	\$BDDAT,-(SP)
2678	006404	013746	002636		MOV	\$GDDAT,-(SP)
2679	006410	012746	004123		MOV	#TFM1,-(SP)
2680	006414	012746	000004		MOV	#4,-(SP)
2681	006420	010600			MOV	SP,R0
2682	006422	104414			TRAP	C\$PNTB
2683	006424	062706	000012		ADD	#12,SP
2684	006430				L10007:	
2685	006430	104423			TRAP	C\$MESSG
2686	006432				SMD	6,3,1
2687	006432				ERR6::	
2688	006432	012746	005764		MOV	#DH3,-(SP)
2689	006436	012746	004114		MOV	#FM1,-(SP)
2690	006442	012746	000002		MOV	#2,-(SP)
2691	006446	010600			MOV	SP,R0
2692	006450	104414			TRAP	C\$PNTB
2693	006452	062706	000006		ADD	#6,SP
2694	006456	013746	002632		MOV	\$GDADR,-(SP)
2695	006462	013746	002640		MOV	\$BDDAT,-(SP)
2696	006466	013746	002636		MOV	\$GDDAT,-(SP)
2697	006472	012746	004123		MOV	#TFM1,-(SP)
2698	006476	012746	000004		MOV	#4,-(SP)
2699	006502	010600			MOV	SP,R0
2700	006504	104414			TRAP	C\$PNTB
2701	006506	062706	000012		ADD	#12,SP
2702	006512				L10010:	
2703	006512	104423			TRAP	C\$MESSG
2704	006514				SMD	7,3,1
2705	006514				ERR7::	

2706	006514	012746	005764	MOV	#DH3,-(SP)
2707	006520	012746	004114	MOV	#FM1,-(SP)
2708	006524	012746	000002	MOV	#2,-(SP)
2709	006530	010600		MOV	SP,RO
2710	006532	104414		TRAP	C\$PNTB
2711	006534	062706	000006	ADD	#6,SP
2712	006540	013746	002632	MOV	\$GDADR,-(SP)
2713	006544	013746	002640	MOV	\$BDDAT,-(SP)
2714	006550	013746	002636	MOV	\$GDDAT,-(SP)
2715	006554	012746	004123	MOV	#TFM1,-(SP)
2716	006560	012746	000004	MOV	#4,-(SP)
2717	006564	010600		MOV	SP,RO
2718	006566	104414		TRAP	C\$PNTB
2719	006570	062706	000012	ADD	#12,SP
2720	006574			L10011:	
2721	006574	104423		TRAP	C\$MSG
2722	006576			\$MD	10,3,1
2723	006576			ERR10::	
2724	006576	012746	005764	MOV	#DH3,-(SP)
2725	006602	012746	004114	MOV	#FM1,-(SP)
2726	006606	012746	000002	MOV	#2,-(SP)
2727	006612	010600		MOV	SP,RO
2728	006614	104414		TRAP	C\$PNTB
2729	006616	062706	000006	ADD	#6,SP
2730	006622	013746	002632	MOV	\$GDADR,-(SP)
2731	006626	013746	002640	MOV	\$BDDAT,-(SP)
2732	006632	013746	002636	MOV	\$GDDAT,-(SP)
2733	006636	012746	004123	MOV	#TFM1,-(SP)
2734	006642	012746	000004	MOV	#4,-(SP)
2735	006646	010600		MOV	SP,RO
2736	006650	104414		TRAP	C\$PNTB
2737	006652	062706	000012	ADD	#12,SP
2738	006656			L10012:	
2739	006656	104423		TRAP	C\$MSG
2740	006660			\$MD	11,2,2
2741	006660			ERR11::	
2742	006660	012746	005750	MOV	#DH2,-(SP)
2743	006664	012746	004114	MOV	#FM1,-(SP)
2744	006670	012746	000002	MOV	#2,-(SP)
2745	006674	010600		MOV	SP,RO
2746	006676	104414		TRAP	C\$PNTB
2747	006700	062706	000006	ADD	#6,SP
2748	006704	013746	002640	MOV	\$BDDAT,-(SP)
2749	006710	013746	002636	MOV	\$GDDAT,-(SP)
2750	006714	012746	004146	MOV	#TFM2,-(SP)
2751	006720	012746	000003	MOV	#3,-(SP)
2752	006724	010600		MOV	SP,RO
2753	006726	104414		TRAP	C\$PNTB
2754	006730	062706	000010	ADD	#10,SP
2755	006734			L10013:	
2756	006734	104423		TRAP	C\$MSG
2757	006736			\$MD	12,2,2
2758	006736			ERR12::	
2759	006736	012746	005750	MOV	#DH2,-(SP)
2760	006742	012746	004114	MOV	#FM1,-(SP)
2761	006746	012746	000002	MOV	#2,-(SP)

2762	006752	010600		MOV	SP,RO
2763	006754	104414		TRAP	C\$PNTB
2764	006756	062706	000006	ADD	#6,SP
2765	006762	013746	002640	MOV	\$BDDAT,-(SP)
2766	006766	013746	002636	MOV	\$GDDAT,-(SP)
2767	006772	012746	004146	MOV	#TFM2,-(SP)
2768	006776	012746	000003	MOV	#3,-(SP)
2769	007002	010600		MOV	SP,RO
2770	007004	104414		TRAP	C\$PNTB
2771	007006	062706	000010	ADD	#10,SP
2772	007012				
2773	007012	104423		L10014:	TRAP C\$MSG
2774	007014				\$MD 13,0.0
2775	007014			ERR13::	
2776	007014	012746	005716	MOV	#DH0,-(SP)
2777	007020	012746	004114	MOV	#FM1,-(SP)
2778	007024	012746	000002	MOV	#2,-(SP)
2779	007030	010600		MOV	SP,RO
2780	007032	104414		TRAP	C\$PNTB
2781	007034	062706	000006	ADD	#6,SP
2782	007040			L10015:	
2783	007040	104423		TRAP	C\$MSG

2784	007042			SMD	14,2,2
2785	007042			ERR14::	
2786	007042	012746	005750	MOV	#DH2,-(SP)
2787	007046	012746	004114	MOV	#FM1,-(SP)
2788	007052	012746	000002	MOV	#2,-(SP)
2789	007056	010600		MOV	SP,R0
2790	007060	104414		TRAP	C\$PNTB
2791	007062	062706	000006	ADD	#6,SP
2792	007066	013746	002640	MOV	\$BDDAT,-(SP)
2793	007072	013746	002636	MOV	\$GDDAT,-(SP)
2794	007076	012746	004146	MOV	#TFM2,-(SP)
2795	007102	012746	000003	MOV	#3,-(SP)
2796	007106	010600		MOV	SP,R0
2797	007110	104414		TRAP	C\$PNTB
2798	007112	062706	000010	ADD	#10,SP
2799	007116			L10016:	
2800	007116	104423		TRAP	C\$MSG

2801	007120		
2802	007120		
2803	007120	012746	006014
2804	007124	012746	004114
2805	007130	012746	000002
2806	007134	010600	
2807	007136	104414	
2808	007140	062706	000006
2809	007144	013746	002640
2810	007150	013746	002636
2811	007154	012746	004163
2812	007160	012746	000003
2813	007164	010600	
2814	007166	104414	
2815	007170	062706	000010
2816	007174		
2817	007174	104423	
2818	007176		
2819	007176		
2820	007176	012746	005716
2821	007202	012746	004114
2822	007206	012746	000002
2823	007212	010600	
2824	007214	104414	
2825	007216	062706	000006
2826	007222		
2827	007222	104423	
2828	007224		
2829	007224		
2830	007224	012746	005716
2831	007230	012746	004114
2832	007234	012746	000002
2833	007240	010600	
2834	007242	104414	
2835	007244	062706	000006
2836	007250		
2837	007250	104423	
2838	007252		
2839	007252		
2840	007252	012746	005750
2841	007256	012746	004114
2842	007262	012746	000002
2843	007266	010600	
2844	007270	104414	
2845	007272	062706	000006
2846	007276	013746	002640
2847	007302	013746	002636
2848	007306	012746	004146
2849	007312	012746	000003
2850	007316	010600	
2851	007320	104414	
2852	007322	062706	000010
2853	007326		
2854	007326	104423	
2855	007330		
2856	007330		

ERR15::	SMD	15,4,5
	MOV	#DH4,-(SP)
	MOV	#FM1,-(SP)
	MOV	#2,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#6,SP
	MOV	\$BDDAT,-(SP)
	MOV	\$GDDAT,-(SP)
	MOV	#TFM5,-(SP)
	MOV	#3,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#10,SP
L10017:	TRAP	C\$MSG
	SMD	16,0,0
ERR16::	MOV	#DH0,-(SP)
	MOV	#FM1,-(SP)
	MOV	#2,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#6,SP
L10020:	TRAP	C\$MSG
	SMD	17,0,0
ERR17::	MOV	#DH0,-(SP)
	MOV	#FM1,-(SP)
	MOV	#2,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#6,SP
L10021:	TRAP	C\$MSG
	SMD	20,2,2
ERR20::	MOV	#DH2,-(SP)
	MOV	#FM1,-(SP)
	MOV	#2,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#6,SP
	MOV	\$BDDAT,-(SP)
	MOV	\$GDDAT,-(SP)
	MOV	#TFM2,-(SP)
	MOV	#3,-(SP)
	MOV	SP,RO
	TRAP	C\$PNTB
	ADD	#10,SP
L10022:	TRAP	C\$MSG
	SMD	21,0,0
ERR21::		

2857	007330	012746	005716	MOV	#DH0,-(SP)
2858	007334	012746	004114	MOV	#FM1,-(SP)
2859	007340	012746	000002	MOV	#2,-(SP)
2860	007344	010600		MOV	SP,R0
2861	007346	104414		TRAP	C\$PNTB
2862	007350	062706	000006	ADD	#6,SP
2863	007354			L10023:	
2864	007354	104423		TRAP	C\$MSG
2865	007356			\$MD	22,0,0
2866	007356			EPR22::	
2867	007356	012746	005716	MOV	#DH0,-(SP)
2868	007352	012746	004114	MOV	#FM1,-(SP)
2869	007366	012746	000002	MOV	#2,-(SP)
2870	007372	010600		MOV	SP,R0
2871	007374	104414		TRAP	C\$PNTB
2872	007376	062706	000006	ADD	#6,SP
2873	007402			L10024:	
2874	007402	104423		TRAP	C\$MSG
2875	007404			\$MD	23,4,5
2876	007404			ERR23::	
2877	007404	012746	006014	MOV	#DH4,-(SP)
2878	007410	012746	004114	MOV	#FM1,-(SP)
2879	007414	012746	000002	MOV	#2,-(SP)
2880	007420	010600		MOV	SP,R0
2881	007422	104414		TRAP	C\$PNTB
2882	007424	062706	000006	ADD	#6,SP
2883	007430	013746	002640	MOV	\$BDDAT,-(SP)
2884	007434	013746	002636	MOV	\$GDDAT,-(SP)
2885	007440	012746	004163	MOV	#TFM5,-(SP)
2886	007444	012746	000003	MOV	#3,-(SP)
2887	007450	010600		MOV	SP,R0
2888	007452	104414		TRAP	C\$PNTB
2889	007454	062706	000010	ADD	#10,SP
2890	007460			L10025:	
2891	007460	104423		TRAP	C\$MSG
2892	007462			\$MD	24,0,0
2893	007462			ERR24::	
2894	007462	012746	005716	MOV	#DH0,-(SP)
2895	007466	012746	004114	MOV	#FM1,-(SP)
2896	007472	012746	000002	MOV	#2,-(SP)
2897	007476	010600		MOV	SP,R0
2898	007500	104414		TRAP	C\$PNTB
2899	007502	062706	000006	ADD	#6,SP
2900	007506			L10026:	
2901	007506	104423		TRAP	C\$MSG
2902	007510			\$MD	25,2,2
2903	007510			ERR25::	
2904	007510	012746	005750	MOV	#DH2,-(SP)
2905	007514	012746	004114	MOV	#FM1,-(SP)
2906	007520	012746	000002	MOV	#2,-(SP)
2907	007524	010600		MOV	SP,R0
2908	007526	104414		TRAP	C\$PNTB
2909	007530	062706	000006	ADD	#6,SP
2910	007534	013746	002640	MOV	\$BDDAT,-(SP)
2911	007540	013746	002636	MOV	\$GDDAT,-(SP)
2912	007544	012746	004146	MOV	#TFM2,-(SP)

Address	Offset	PC	PSW	Instruction	Comment
2913	007550	012746	000003	MOV	#3,-(SP)
2914	007554	010600		MOV	SP,R0
2915	007556	104414		TRAP	C\$PNTB
2916	007560	062706	000010	ADD	#10,SP
2917	007564				L10027:
2918	007564	104423		TRAP	C\$MSG
2919	007566			\$MD	26,2,2
2920	007566				ERR26::
2921	007566	012746	005750	MOV	#DH2,-(SP)
2922	007572	012746	004114	MOV	#FM1,-(SP)
2923	007576	012746	000002	MOV	#2,-(SP)
2924	007602	010600		MOV	SP,R0
2925	007604	104414		TRAP	C\$PNTB
2926	007606	062706	000006	ADD	#6,SP
2927	007612	013746	002640	MOV	\$BDDAT,-(SP)
2928	007616	013746	002636	MOV	\$GDDAT,-(SP)
2929	007622	012746	004146	MOV	#TFM2,-(SP)
2930	007626	012746	000003	MOV	#3,-(SP)
2931	007632	010600		MOV	SP,R0
2932	007634	104414		TRAP	C\$PNTB
2933	007636	062706	000010	ADD	#10,SP
2934	007642				L10030:
2935	007642	104423		TRAP	C\$MSG
2936	007644			\$MD	27,27,27
2937	007644				ERR27::
2938	007644	012746	006030	MOV	#DH27,-(SP)
2939	007650	012746	004114	MOV	#FM1,-(SP)
2940	007654	012746	000002	MOV	#2,-(SP)
2941	007660	010600		MOV	SP,R0
2942	007662	104414		TRAP	C\$PNTB
2943	007664	062706	000006	ADD	#6,SP
2944	007670	013746	002640	MOV	\$BDDAT,-(SP)
2945	007674	013746	002636	MOV	\$GDDAT,-(SP)
2946	007700	013746	002624	MOV	MRO,-(SP)
2947	007704	012746	004200	MOV	#TFM27,-(SP)
2948	007710	012746	000004	MOV	#4,-(SP)
2949	007714	010600		MOV	SP,R0
2950	007716	104414		TRAP	C\$PNTB
2951	007720	062706	000012	ADD	#12,SP
2952	007724				L10031:
2953	007724	104423		TRAP	C\$MSG
2954	007726			\$MD	28,2,2
2955	007726				ERR28::
2956	007726	012746	005750	MOV	#DH2,-(SP)
2957	007732	012746	004114	MOV	#FM1,-(SP)
2958	007736	012746	000002	MOV	#2,-(SP)
2959	007742	010600		MOV	SP,R0
2960	007744	104414		TRAP	C\$PNTB
2961	007746	062706	000006	ADD	#6,SP
2962	007752	013746	002640	MOV	\$BDDAT,-(SP)
2963	007756	013746	002636	MOV	\$GDDAT,-(SP)
2964	007762	012746	004146	MOV	#TFM2,-(SP)
2965	007766	012746	000003	MOV	#3,-(SP)
2966	007772	010600		MOV	SP,R0
2967	007774	104414		TRAP	C\$PNTB
2968	007776	062706	000010	ADD	#10,SP

2969 010002
2970 010002 104423
2971 010004
2972 010004
2973 010004 012746 006030
2974 010010 012746 004114
2975 010014 012746 000002
2976 010020 010600
2977 010022 104414
2978 010024 062706 000006
2979 010030 013746 002640
2980 010034 013746 002636
2981 010040 013746 002624
2982 010044 012746 004200
2983 010050 012746 000004
2984 010054 010600
2985 010056 104414
2986 010060 062706 000012
2987 010064
2988 010064 104423
2989 010066
2990 010066
2991 010066 012746 005750
2992 010072 012746 004114
2993 010076 012746 000002
2994 010102 010600
2995 010104 104414
2996 010106 062706 000006
2997 010112 013746 002640
2998 010116 013746 002636
2999 010122 012746 004146
3000 010126 012746 000003
3001 010132 010600
3002 010134 104414
3003 010136 062706 000010
3004 010142
3005 010142 104423
3006 010144
3007 010144
3008 010144 012746 005716
3009 010150 012746 004114
3010 010154 012746 000002
3011 010160 010600
3012 010162 104414
3013 010164 062706 000006
3014 010170
3015 010170 104423
3016 010172
3017 010172
3018 010172 012746 005716
3019 010176 012746 004114
3020 010202 012746 000002
3021 010206 010600
3022 010210 104414
3023 010212 062706 000006
3024 010216

L10032:
TRAP C\$MSG
\$MD 29,27,27
ERR29::
MOV #DH27,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #6,SP
MOV \$BDDAT,-(SP)
MOV \$GDDAT,-(SP)
MOV MRO,-(SP)
MOV #TFM27,-(SP)
MOV #4,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #12,SP
L10033:
TRAP C\$MSG
\$MD 30,2,2
ERR30::
MOV #DH2,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #6,SP
MOV \$BDDAT,-(SP)
MOV \$GDDAT,-(SP)
MOV #TFM2,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #10,SP
L10034:
TRAP C\$MSG
\$MD 31,0,0
ERR31::
MOV #DH0,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #6,SP
L10035:
TRAP C\$MSG
\$MD 32,0,0
ERR32::
MOV #DH0,-(SP)
MOV #FM1,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #6,SP
L10036:

3025	010216	104423		TRAP	C\$MSG
3026	010220			\$MD	33,2,2
3027	010220			ERR33::	
3028	010220	012746	005750	MOV	#DH2,-(SP)
3029	010224	012746	004114	MOV	#FM1,-(SP)
3030	010230	012746	000002	MOV	#2,-(SP)
3031	010234	010600		MOV	SP,R0
3032	010236	104414		TRAP	C\$PNTB
3033	010240	062706	000006	ADD	#6,SP
3034	010244	013746	002640	MOV	\$BDDAT,-(SP)
3035	010250	013746	002636	MOV	\$GDDAT,-(SP)
3036	010254	012746	004146	MOV	#TFM2,-(SP)
3037	010260	012746	000003	MOV	#3,-(SP)
3038	010264	010600		MOV	SP,R0
3039	010266	104414		TRAP	C\$PNTB
3040	010270	062706	000010	ADD	#10,SP
3041	010274			L10037:	
3042	010274	104423		TRAP	C\$MSG
3043	010276			\$MD	34,2,2
3044	010276			ERR34::	
3045	010276	012746	005750	MOV	#DH2,-(SP)
3046	010302	012746	004114	MOV	#FM1,-(SP)
3047	010306	012746	000002	MOV	#2,-(SP)
3048	010312	010600		MOV	SP,R0
3049	010314	104414		TRAP	C\$PNTB
3050	010316	062706	000006	ADD	#6,SP
3051	010322	013746	002640	MOV	\$BDDAT,-(SP)
3052	010326	013746	002636	MOV	\$GDDAT,-(SP)
3053	010332	012746	004146	MOV	#TFM2,-(SP)
3054	010336	012746	000003	MOV	#3,-(SP)
3055	010342	010600		MOV	SP,R0
3056	010344	104414		TRAP	C\$PNTB
3057	010346	062706	000010	ADD	#10,SP
3058	010352			L10040:	
3059	010352	104423		TRAP	C\$MSG
3060	010354			\$MD	35,2,2
3061	010354			ERR35::	
3062	010354	012746	005750	MOV	#DH2,-(SP)
3063	010360	012746	004114	MOV	#FM1,-(SP)
3064	010364	012746	000002	MOV	#2,-(SP)
3065	010370	010600		MOV	SP,R0
3066	010372	104414		TRAP	C\$PNTB
3067	010374	062706	000006	ADD	#6,SP
3068	010400	013746	002640	MOV	\$BDDAT,-(SP)
3069	010404	013746	002636	MOV	\$GDDAT,-(SP)
3070	010410	012746	004146	MOV	#TFM2,-(SP)
3071	010414	012746	000003	MOV	#3,-(SP)
3072	010420	010600		MOV	SP,R0
3073	010422	104414		TRAP	C\$PNTB
3074	010424	062706	000010	ADD	#10,SP
3075	010430			L10041:	
3076	010430	104423		TRAP	C\$MSG
3077	010432			\$MD	36,2,2
3078	010432			ERR36::	
3079	010432	012746	005750	MOV	#DH2,-(SP)
3080	010436	012746	004114	MOV	#FM1,-(SP)

3081	010442	012746	000002	MOV	#2,-(SP)
3082	010446	010600		MOV	SP,R0
3083	010450	104414		TRAP	C\$PNTB
3084	010452	062706	000006	ADD	#6,SP
3085	010456	013746	002640	MOV	\$BDDAT,-(SP)
3086	010462	013746	002636	MOV	\$GDDAT,-(SP)
3087	010466	012746	004146	MOV	#TFM2,-(SP)
3088	010472	012746	000003	MOV	#3,-(SP)
3089	010476	010600		MOV	SP,R0
3090	010500	104414		TRAP	C\$PNTB
3091	010502	062706	000010	ADD	#10,SP
3092	010506			L10042:	
3093	010506	104423		TRAP	C\$MSG
3094					
3095	010510			BGNMSG	ERR37
3096	010510			ERR37::	
3097	010510			PRINTF	#FM1,#EM1
3098	010510	012746	004261	MOV	#EM1,-(SP)
3099	010514	012746	004114	MOV	#FM1,-(SP)
3100	010520	012746	000002	MOV	#2,-(SP)
3101	010524	010600		MOV	SP,R0
3102	010526	104417		TRAP	C\$PNTF
3103	010530	062706	000006	ADD	#6,SP
3104	010534			PRINTF	#TFM37,\$GDADR
3105	010534	013746	002632	MOV	\$GDADR,-(SP)
3106	010540	012746	004225	MOV	#TFM37,-(SP)
3107	010544	012746	000002	MOV	#2,-(SP)
3108	010550	010600		MOV	SP,R0
3109	010552	104417		TRAP	C\$PNTF
3110	010554	062706	000006	ADD	#6,SP
3111	010560			ENDMSG	
3112	010560			L10043:	
3113	010560	104423		TRAP	C\$MSG
3114					
3115					
3116					

3117
3118
3119
3120
3121
3122
3123
3124
3125 010562
3126 010562
3127
3128
3129 010562
3130 010562 000167
3131 010564 000000
3132
3133
3134 010566
3135 010566
3136 010566 104425
3137
3138
3139
3140
3141

.SBTTL REPORT CODING SECTION

:++
: THE REPORT CODING SECTION CONTAINS THE
: 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
:--

LSRPT:: BGNRPT

EXIT RPT
.WORD JSJMP
.WORD L10044-2-

L10044: ENDRPT
TRAP CSRPT

3142
3143
3144
3145
3146
3147
3148
3149 010570
3150 010570
3151
3152
3153 010570 012705 003'30
3154
3155 010574 010637 002554
3156 010600 005737 002646
3157 010604 001011
3158 010606 013737 000004 002650
3159 010614 013737 000006 002652
3160 010622 012737 000001 002646
3161 010630 013737 002650 000004
3162 010636 013737 002652 000006
3163
3164 010644
3165 010644 012700 000040
3166 010650 104447
3167 010652
3168 010652 103414
3169
3170 010654
3171 010654 012700 000035
3172 010660 104447
3173 010662
3174 010662 103410
3175
3176 010664
3177 010664 012700 000036
3178 010670 104447
3179 010672
3180 010672 103570
3181
3182
3183 010674
3184 010674 012700 000037
3185 010700 104447
3186 010702
3187 010702 103003
3188 010704
3189
3190 010704 012737 177777 002552
3191
3192
3193
3194
3195 010712
3196 010712 005237 002552
3197 010716 023737 002552 002012

```
.SBTTL INITIALIZE SECTION  
://////  
:/ THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED  
:/ AT THE BEGINNING OF EACH PASS.  
://////  
      BGNINIT  
LSINIT::  
:INITIALIZE SUBROUTINE STACK  
      MOV     #SSTACK,R5  
:STORE BASE LEVEL PROGRAM STACK POINTER  
      MOV     SP,PSTACK  
      TST     FTIME  
      BNE     1$  
      MOV     @#4,SAVE4  
      MOV     @#6,SAVE6  
      MOV     #1,FTIME  
1$:    MOV     SAVE4,@#4  
      MOV     SAVE6,@#6  
:SEE IF PROGRAM JUST STARTED, BR IF YES  
      READEF  #EF.START  
      MOV     #EF.START,R0  
      TRAP    CSREFG  
      BCOMPLETE NEWST  
      BCS     NEWST  
:SEE IF THIS IS A NEW PASS, BR IF YES  
      READEF  #EF.NEW  
      MOV     #EF.NEW,R0  
      TRAP    CSREFG  
      BCOMPLETE NEWST  
      BCS     NEWST  
:SEE IF PROGRAM WAS JUST CONTINUED  
      READEF  #EF.CONTINUE  
      MOV     #EF.CONTINUE,R0  
      TRAP    CSREFG  
      BCOMPLETE ENDIT  
      BCS     ENDIT  
:SEE IF PROGRAM JUST RESTARTED, BR IF NOT  
      READEF  #EF.RESTART  
      MOV     #EF.RESTART,R0  
      TRAP    CSREFG  
      BCOMPLETE GETPRM  
      BCC     GETPRM  
NEWST:  
:RESET LOGICAL DEVICE TO -1  
      MOV     #-1,LOGDEV  
:GET UNIBUS ADRS, VECTOR, PRIORITY LEVEL, LINE UNIT, SWITCH  
:   PACKS, TEST CONNECTOR INFO. FOR THIS M8200,4,7 (CURRENT LOGICAL  
:   DEVICE).  
GETPRM:  
      INC     LOGDEV  
      CMP     LOGDEV,LSUNIT
```

3198	010724	002367			BGE	NEWST
3199	010726				GPHARD	LOGDEV,R1
3200	010726	013700	002552		MOV	LOGDEV,R0
3201	010732	104442			TRAP	C\$GPHRD
3202	010734	010001			MOV	R0,R1
3203	010736				BNCOMPLETE	GETPRM
3204	010736	103365			BCC	GETPRM
3205					;GET ADDRESS OF M8200,4,7	
3206	010740	012137	002626		MOV	(R1)+,WTYPE
3207	010744	011137	002716		MOV	(R1),KMCSR
3208					;GET POINTER TO M8200,4,7 CSR HI BYTE	
3209	010750	011137	002720		MOV	(R1),KMCSRH
3210	010754	005237	002720		INC	KMCSRH
3211					;GET POINTER TO M8200,4,7 CTL OUT REG	
3212	010760	011137	002722		MOV	(R1),KMCTL
3213	010764	062737	000002	002722	ADD	#2,KMCTL
3214					;GET POINTER TO M8200,4,7 PORT REG - SEL 4	
3215	010772	011137	002724		MOV	(R1),KMPO4
3216	010776	062737	000004	002724	ADD	#4,KMPO4
3217					;GET POINTER TO M8200,4,7 PORT REG - SEL 6	
3218	011004	012137	002726		MOV	(R1)+,KMPO6
3219	011010	062737	000006	002726	ADD	#6,KMPO6
3220					;GET POINTER TO RCV VECTOR	
3221	011016	011137	002706		MOV	(R1),KMRVEC
3222					;GET POINTER TO RCV PRIORITY LEVEL	
3223	011022	011137	002710		MOV	(R1),KMRLVL
3224	011026	062737	000002	002710	ADD	#2,KMRLVL
3225					;GET POINTER TO TX VECTOR	
3226	011034	011137	002712		MOV	(R1),KMTVEC
3227	011040	062737	000004	002712	ADD	#4,KMTVEC
3228					;GET POINTER TO TX PRIORITY LEVEL	
3229	011046	011137	002714		MOV	(R1),KMTLVL
3230	011052	062737	000006	002714	ADD	#6,KMTLVL
3231					;PUT VECTOR INTO STAT1	
3232	011060	012137	002700		MOV	(R1)+,STAT1
3233					;PUT PRIORITY INTO STAT1	
3234	011064	052137	002700		BIS	(R1)+,STAT1
3235					;SEE IF NO LINE UNIT, SET BIT IF YES	
3236	011070	005711			TST	(R1)
3237	011072	001004			BNE	50000\$
3238	011074	052737	010000	002700	BIS	#BIT12,STAT1
3239	011102	000416			BR	4\$
3240	011104				50000\$:	
3241					;SEE IF M8201 LINE UNIT, SET BIT IF YES	
3242	011104	021127	000001		CMP	(R1),#1
3243	011110	001001			BNE	50001\$
3244	011112	000412			BR	4\$
3245	011114				50001\$:	
3246					;SEE IF M8202 LINE UNIT, SET BIT IF YES	
3247	011114	021127	000002		CMP	(R1),#2
3248	011120	001004			BNE	50002\$
3249	011122	052737	020000	002700	BIS	#BIT13,STAT1
3250	011130	000403			BR	4\$
3251	011132				50002\$:	
3252					;SET BIT FOR M8203 LINE UNIT	
3253	011132	052737	100000	002700	BIS	#BIT15,STAT1

```

3254 011140          4$:
3255          .SET BIT IN STAT1 FOR TEST CONNECTOR
3256 011140 056137 000006 002700      BIS      6(R1),STAT1
3257 011146 062701 000002          ADD      #2,R1
3258          ;SET SWITCH PACK #1 IN STAT2 LOW BYTE
3259 011152 012137 002702          MOV      (R1)+,STAT2
3260          ;SET SWITCH PACK #2 IN STAT2 HIGH BYTE
3261 011156 111137 002703          MOVB    (R1),STAT2+1
3262
3263          ;INCREMENT LOGICAL UNIT (DEVICE) NUMBER
3264          :      INC      LOGDEV
3265 011162 000240          :      NOP
3266 011164 000240          :      NOP
3267
3268 011166 012737 002000 002606      MOV      #2000,MEMSZ
3269 011174 005037 002630          CLR      TYPE
3270 011200 123727 002626 000000      CMPB    WTYPE,#0
3271 011206 001422          BEQ     ENDIT
3272 011210 123727 002626 000004      CMPB    WTYPE,#4      ;KMC?
3273 011216 001004          BNE     5$
3274 011220 012737 000001 002630      MOV      #1,TYPE
3275 011226 000412          BR      ENDIT
3276 011230 012737 003777 002606 5$:  MOV      #3777,MEMSZ
3277 011236 123727 002626 000006      CMPB    WTYPE,#6
3278 011244 001003          BNE     ENDIT
3279 011246 012737 000001 002630      MOV      #1,TYPE
3280 011254          ENDIT:
3281 011254          L10045: ENDINIT
3282 011254          L10045: TRAP    C$INIT
3283 011254 104411
3284
3285          .EVEN
3286 011256          L$AUTO: BGNAUTO
3287 011256
3288          ;DEVICE DOES NOT HAVE A 'READY'
3289 011256 013701 002716          MOV      KMCSR,R1      ;R1 CONTAINS BASE M8200,4,7 ADDRESS
3290 011262 012705 000004          MOV      #4,R5      ;4 REGISTERS TO BE TESTED
3291 011266 012737 011320 000004          MOV      #2$ ,4      ;SET UP TIMEOUT TRAP
3292 011274 012737 000340 000006          MOV      #340,6      ;LEVEL 7
3293 011302 005711          1$:  TST      (R1)      ;REFERENCE DEVICE REGISTER
3294 011304 000240          NOP
3295 011306 062701 000002          ADD      #2,R1      ;NEXT REGISTER
3296 011312 005305          DEC      R5      ;DEC REGISTER COUNT
3297 011314 001372          BNE     1$      ;BR IF NOT LAST REGISTER
3298 011316 000407          BR      3$
3299 011320 062706 000004          2$:  ADD      #4,SP
3300 011324 010137 002632          MOV      R1,$GDADR
3301 011330          DODU    LOGDEV
3302 011330 013700 002552          MOV      LOGDEV,R0
3303 011334 104451          TRAP    C$DODU
3304
3305 011336 013737 002650 000004 3$:  MOV      SAVE4,4
3306 011344 013737 002652 000006      MOV      SAVE6,6
3307 011352          ENDAUTO
3308 011352          L10046:
3309 011352 104461          TRAP    C$AUTO
  
```

CZDMPBO M8207 STATIC DIAG #1 MACY11 30A(1052) 20-AUG-80 07:50 E 8
CZDMPB.P11 20-AUG-80 07:48 INITIALIZE SECTION PAGE 96

SEQ 0095

3310

3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330

011354
011354
011354
011354 104433

011356
011356
011356 104412

```
.SBTTL CLEANUP CODING SECTION  
:////////////////////  
:/ THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED  
:/ AT THE END OF EACH PASS.  
:////////////////////  
  
L$CLEAN:: BGNCLN  
          BRESET  
          TRAP   C$RESET  
  
L10047:  ENDCLN  
          TRAP   C$CLEAN
```

3331
3332
3333
3334
3335
3336
3337
3338 011360
3339 011360
3340
3341 011360
3342 011360 104433
3343 011362
3344 011362
3345 011362 104453
3346
3347
3348
3349
3350

```
.SBTTL DROP UNIT SECTION  
:////////////////////  
:/ THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE  
:/ TO NO LONGER BE TESTED.  
:////////////////////  
                BGNDU  
LSDU::  
:ISSUE UNIBUS RESET TO CLEAN UP  
                BRESET  
                TRAP   C$RESET  
                ENDDU  
L10050:  
                TRAP   C$DU
```


3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369

011364
011364
011364
011364
011364 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.
:/

LSAU:: BGAU
 ENDAU
L10051: TRAP CS AU

```

3370 .SBTTL HARDWARE TESTS
3371
3372
3373
3374
3375 011366 BADHEAD
3376 :***** TEST 1 *****
3377 :*VERIFY THAT REFERENCING UNIBUS DEVICE REGISTERS
3378 :*DOES NOT CAUSE A TIME OUT TRAP
3379 011366 BADHEAD
3380 :***** TEST 1 *****
3381
3382 011366 BGNTST
3383 011366 T1::
3384 011366 013701 002716 MOV KMCSR,R1 ;R1 CONTAINS BASE M8200,4,7 ADDRESS
3385 011372 012705 000004 MOV #4,R5 ;4 REGISTERS TO BE TESTED
3386 011376 012737 011434 000004 MOV #2$,4 ;SET UP TIMEOUT TRAP
3387 011404 012737 000340 000006 MOV #340,6 ;LEVEL 7
3388 011412 005711 1$: TST (R1) ;REFERENCE DEVICE REGISTER
3389 011414 000240 NOP
3390 011416 ESCAPE TST
3391 011416 104410 TRAP C$ESCAPE
3392 011420 000072 .WORD L10052-.
3393 011422 062701 000002 ADD #2,R1 ;NEXT REGISTER
3394 011426 005305 DEC R5 ;DEC REGISTER COUNT
3395 011430 001370 BNE 1$ ;BR IF NOT LAST REGISTER
3396 011432 000417 BR 3$
3397 011434 062706 000004 2$: ADD #4,SP
3398 011440 010137 002632 MOV R1,$GDADR
3399 011444 ERROR 37 ;TIME-OUT ERROR
3400 011462 104455 TRAP C$ERDF
3401 011464 000045 .WORD 37
3402 011466 005715 .WORD EM37
3403 011470 010510 .WORD ERR37
3404
3405 011472 013737 002650 000004 3$: MOV SAVE4,4
3406 011500 013737 002652 000006 MOV SAVE6,6
3407 011506 ESCAPE TST
3408 011506 104410 TRAP C$ESCAPE
3409 011510 000002 .WORD L10052-.
3410 011512 ENDTST
3411 011512 L10052:
3412 011512 104401 TRAP C$ETST
3413
3414 011514 BADHEAD
3415 :***** TEST 2 *****
3416 :*VERIFY THAT RUN CAN BE CLEARED
3417 011514 BADHEAD
3418 :***** TEST 2 *****
3419
3420 011514 BGNTST
3421 011514 T2::
3422 011514 MYINT
3423 011514 013701 002716 MOV KMCSR,R1 ;GET DEVICE ADDRESS.
3424 011520 005011 CLR (R1) ;CLEAR KMCSR
3425 011522 005002 CLR R2 ;CLEAR 'EXPECTED'
  
```

```
3426 011524 011104      MOV      (R1),R4          ;PUT KMCSR IN 'FOUND'
3427 011526 001413      BEQ      1$              ;BR IF CLEARED
3428 011530              ERROR    26              ;ERROR KMCSR NOT CLEARED
3429 011546 104455      TRAP    C$ERDF
3430 011550 000032      .WORD   26
3431 011552 005474      .WORD   EM26
3432 011554 007566      .WORD   ERR26
3433 011556              1$:
3434 011556              ENDTST
3435 011556              L'0053:
3436 011556 104401      TRAP    C$ETST
3437
3438 011560      BADHEAD
3439      :***** TEST 3 *****
3440      :*UNIBUS REGISTER WORD DUAL ADDRESSING TEST
3441      :*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
3442      :*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
3443      :*THE SEQUENCE:
3444      :* 1. CLEAR REGISTER
3445      :* 2. WRITE PATTERN
3446      :* 3. VERIFY PATTERN
3447      :* 4. DO ALL 4 REGISTERS
3448      :* 5. READ ALL BACK IF ERRORS,
3449      :*    DUAL ADDRESS PROBLEM.
3450      :
3451      : 1 IN REG 0
3452      : 2 IN REG 2
3453      : 3 IN REG 4
3454      : 4 IN REG 6
3455 011560      BADHEAD
3456      :***** TEST 3 *****
3457
3458 011560      BGNTST
3459 011560      T3::
3460 011560
3461 011560 013701 002716      MYINT
3462      MOV      KMCSR,R1          ;GET DEVICE ADDRESS.
3463      MSTCLR          ;R1 CONTAINS BASE M8200,4,7 ADDRESS
3464 011564 004537 003156      JSR      R5, .MSTCLR          ;MASTER CLEAR M8200,4,7
3465 011570 012702 000001      MOV      #1,R2              ;CLEAR M8200,4,7
3466 011574              ;START PATTERN AT 1
3467 011574 104404      BGNSEG
3468 011576 005011      TRAP    C$BSEG
3469 011600 010211      1$: CLR      (R1)              ;CLEAR REGISTER
3470 011602 011104      MOV      R2,(R1)          ;WRITE M8200,4,7 REGISTER WITH PATTERN
3471 011604 020204      MOV      (R1),R4          ;READ M8200,4,7 REGISTER INTO 'FOUND'
3472 011606 001413      CMP      R2,R4            ;IS DATA CORRECT
3473 011610              BEQ      2$              ;BR IS YES
3474 011626 104455      ERROR    26              ;DATA ERROR
3475 011630 000032      TRAP    C$ERDF
3476 011632 005474      .WORD   26
3477 011634 007566      .WORD   EM26
3478 011636              .WORD   ERR26
3479 011636 104410      2$: ESCAPE  SEG
3480 011640 000014      TRAP    C$ESCAPE
3481 011642 005721      .WORD   10000$-
3481      TST      (R1)+          ;NEXT REGISTER
```

```
3482 011644 005202          INC      R2          ;INCREMENT DATA PATTERN
3483 011646 022702 000005    CMP      #5,R2       ;LAST REGISTER?
3484 011652 001351          BNE      1$          ;BR IF NO
3485 011654          ENDSEG
3486 011654          10000$:
3487 011654 104405          TRAP     C$ESEG
3488 011656 013701 002716    MOV      KMCSR,R1    ;BASE MB200,4,7 ADDRESS TO R1
3489 011662 012702 000001    MOV      #1,R2       ;RESTART PATTERN AT 1
3490 011666          BGNSEG
3491 011666 104404          TRAP     C$BSEG
3492 011670          3$:
3493 011670 011104          MOV      (R1),R4     ;READ COMM. MICR-PROCESSOR FAMILY REGISTER INTO 'FOUND'
3494 011672 020204          CMP      R2,R4       ;IS DATA CORRECT
3495 011674 001413          BEQ      4$          ;BR IF YES
3496 011676          ERROR     2        ;DUAL ADDRESSING ERROR
3497 011714 104455          TRAP     C$ERDF
3498 011716 000002          .WORD   2
3499 011720 004307          .WORD   EM2
3500 011722 006132          .WORD   ERR2
3501 011724          4$:
3502 011724 104410          ESCAPE   SEG
3503 011726 000014          TRAP     C$ESCAPE
3504 011730 005721          .WORD   10001$-
3505 011732 005202          TST      (R1)+       ;NEXT REGISTER
3506 011734 022702 000005    INC      R2          ;INCREMENT PATTERN
3507 011740 001353          CMP      #5,R2       ;LAST REGISTER?
3508 011742          BNE      3$          ;BR IF NO
3509 011742          ENDSEG
3510 011742 104405          10001$:
3511 011744          TRAP     C$ESEG
3512 011744          ENDTST
3513 011744 104401          L10054:
3514          TRAP     C$ETST
3515 011746          BADHEAD
3516          ;***** TEST 4 *****
3517          ;*CONTROL STATUS REGISTER WRITE/READ TEST
3518          ;*FLOAT A ONE THROUGH BSEL 0
3519          ;*CLEAR BIT0, VERIFY BIT0 WAS CLEARED
3520 011746          BADHEAD
3521          ;***** TEST 4 *****
3522          BGNTST
3523          T4::
3524          MSTCLR
3525          JSR      R5,.MSTCLR ;MASTER CLEAR MB200,4,7
3526 011746 004537 003156          CLR      MRO         ;CLEAR MB200,4,7
3527 011752 005037 002624          MOV      #BIT0,R2    ;INDICATE BSEL0
3528 011756 012702 000001          BGNSEG
3529 011762          1$:
3530 011762 104404          TRAP     C$BSEG
3531 011764 013701 002716    MOV      KMCSR,R1    ;PUT REGISTER ADDRESS IN R1
3532 011770 010237 002636    MOV      R2,$GDDAT
3533 011774 013711 002636    MOV      $GDDAT,(R1) ;WRITE BIT 0
3534 012000 011104          MOV      (R1),R4     ;READ CONTROL STATUS REGISTER
3535 012002 023704 002636    CMP      $GDDAT,R4   ;IS DATA CORRECT
3536 012006 001411          BEQ      2$          ;BR IF YES
3537 012010          ERROR     27,YES   ;DATA ERROR
```

3538	012022	104455		TRAP	C\$ERDF	
3539	012024	000033		.WORD	27	
3540	012026	005525		.WORD	EM27	
3541	012030	007644		.WORD	ERR27	
3542	012032		2\$:	ESCAPE	SEG	
3543	012032	104410		TRAP	C\$ESCAPE	
3544	012034	000052		.WORD	10000\$-	
3545	012036	040211	3\$:	BIC	R2,(R1)	;CLEAR BSELO
3546	012040	005037		CLR	\$GDDAT	;CLEAR "EXPECTED"
3547	012044	011104	002636	MOV	(R1),R4	;READ CONTROL STATUS REGISTER
3548	012046	001413		BEQ	4\$;BR IF ZERO
3549	012050			ERROR	2	;DATA ERROR BSEL NOT CLEARED
3550	012066	104455		TRAP	C\$ERDF	
3551	012070	000002		.WORD	2	
3552	012072	004307		.WORD	EM2	
3553	012074	006132		.WORD	ERR2	
3554	012076		4\$:	F\$CAPE	SEG	
3555	012076	104410		TRAP	C\$ESCAPE	
3556	012100	000006		.WORD	10000\$-	
3557	012102	106302		ASLB	R2	
3558	012104	001327		BNE	1\$	
3559	012106			ENDSEG		
3560	012106		10000\$:			
3561	012106	104405		TRAP	C\$ESEG	
3562	012110		ENDTST			
3563	012110		L10055:			
3564	012110	104401		TRAP	C\$ETST	
3565						
3566						

```
3567
3568
3569
3570
3571
3572 012112          BADHEAD
3573                  :***** TEST 5 *****
3574                  :*CONTROL STATUS REGISTER WRITE/READ TEST
3575                  :*SET BIT9, VERIFY BIT9 WAS SET
3576                  :*CLEAR BIT9, VERIFY BIT9 WAS CLEARED
3577 012112          BADHEAD
3578                  :***** TEST 5 *****
3579
3580 012112          BGNTST
3581 012112          T5::
3582 012112
3583 012112 004537 003156  MSTCLR          ;MASTER CLEAR M8200,4,7
3584 012116          JSR          R5,.MSTCLR      ;CLEAR M8200,4,7
3585 012116 104404          BGNSEG
3586 012120 013701 002716  TRAP          C$BSEG
3587 012124 012702 001000  1$:          MOV          KMCSR,R1      ;PUT REGISTER ADDRESS IN R1
3588 012130 010211          MOV          #BIT9,R2      ;PUT DATA IN 'EXPECTED'
3589 012132 011104          MOV          R2,(R1)      ;WRITE BIT 9
3590 012134 020204          MOV          (R1),R4      ;READ CONTROL STATUS REGISTER
3591 012136 001413          MOV          (R1),R4      ;IS DATA CORRECT
3592 012140          CMP          R2,R4
3593 012156 104455          BEQ          2$
3594 012160 000032          ERROR          26      ;BR IF YES
3595 012162 005474          TRAP          C$ERDF      ;DATA ERROR
3596 012164 007566          .WORD          26
3597 012166          .WORD          EM26
3598 012166 104410          .WORD          ERR26
3599 012170 000002          2$:          ESCAPE          SEG
3600 012172          TRAP          C$ESCAPE
3601 012172          .WORD          10000$-.
3602 012172 104405          10000$:          ENDSEG
3603 012174          TRAP          C$ESEG
3604 012174 104404          BGNSEG
3605 012176 042711 001000  3$:          TRAP          C$BSEG
3606 012202 005002          BIC          #BIT9,(R1)      ;CLEAR BIT 9
3607 012204 011104          CLR          R2          ;CLEAR 'EXPECTED'
3608 012206 001416          MOV          (R1),R4      ;READ CONTROL STATUS REGISTER
3609 012210          BEQ          4$          ;BR IF ZERO
3610 012226 104455          ERROR          26      ;DATA ERROR BIT9 NOT CLEARED
3611 012230 000032          TRAP          C$ERDF
3612 012232 005474          .WORD          26
3613 012234 007566          .WORD          EM26
3614 012236          .WORD          ERR26
3615 012236 104410          ESCAPE          SEG
3616 012240 000002          TRAP          C$ESCAPE
3617 012242          .WORD          10001$-.
3618 012242          10001$:          ENDSEG
3619 012242 104405          TRAP          C$ESEG
3620 012244          4$:
3621 012244          ENDTST
3622 012244          L10056:
```

```

3623 012244 104401 TRAP C$ETST
3624
3625 012246 BADHEAD
3626 :***** TEST 6 *****
3627 :*CONTROL STATUS REGISTER WRITE/READ TEST
3628 :*SET BIT11, VERIFY BIT11 WAS SET
3629 :*CLEAR BIT11, VERIFY BIT11 WAS CLEARED
3630 012246 BADHEAD
3631 :***** TEST 6 *****
3632
3633 012246 BGNTST
3634 012246 T$:
3635 012246 MSTCLR ;MASTER CLEAR M8200,4,7
3636 012246 004537 003156 JSR R5,.MSTCLR ;CLEAR M8200,4,7
3637 012252 BGNSEG
3638 012252 104404 TRAP C$BSEG
3639 012254 013701 002716 1$: MOV KMCSR,R1 ;PUT REGISTER ADDRESS IN R1
3640 012260 012702 004000 MOV #BIT11,R2 ;PUT DATA IN 'EXPECTED'
3641 012264 010211 MOV R2,(R1) ;WRITE BIT 11
3642 012266 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
3643 012270 020204 CMP R2,R4 ;IS DATA CORRECT
3644 012272 001413 BEQ 2$ ;BR IF YES
3645 012274 ERROR 26 ;DATA ERROR
3646 012312 104455 TRAP C$ERDF
3647 012314 000032 .WORD 26
3648 012316 005474 .WORD EM26
3649 012320 007566 .WORD ERR26
3650 012322 2$: ESCAPE SEG
3651 012322 104410 TRAP C$ESCAPE
3652 012324 000002 .WORD 10000$-.
3653 012326 ENDSEG
3654 012326 10000$:
3655 012326 104405 TRAP C$ESEG
3656 012330 BGNSEG
3657 012330 104404 TRAP C$BSEG
3658 012332 042711 004000 3$: BIC #BIT11,(R1) ;CLEAR BIT 11
3659 012336 005002 CLR R2 ;CLEAR 'EXPECTED'
3660 012340 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
3661 012342 001414 BEQ 4$ ;BR IF ZERO
3662 012344 ERROR 26 ;DATA ERROR BIT11 NOT CLEARED
3663 012362 104455 TRAP C$ERDF
3664 012364 000032 .WORD 26
3665 012366 005474 .WORD EM26
3666 012370 007566 .WORD ERR26
3667 012372 ENDSEG
3668 012372 10001$:
3669 012372 104405 TRAP C$ESEG
3670 012374 4$:
3671 012374 ENDTST
3672 012374 L10057:
3673 012374 104401 TRAP C$ETST
3674
3675 012376 BADHEAD
3676 :***** TEST 7 *****
3677 :*CONTROL STATUS REGISTER WRITE/READ TEST
3678 :*SET BIT12, VERIFY BIT12 WAS SET
  
```

```
3679                                     ;*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
3680 012376                               BADHEAD
3681                                     ;***** TEST 7 *****
3682
3683 012376                               BGNTST
3684 012376                               T7:
3685 012376                               MSTCLR                               ;MASTER CLEAR M8200,4,7
3686 012376 004537 003156                 JSR R5, .MSTCLR                               ;CLEAR M8200,4,7
3687 012402                               BGNSEG
3688 012402 104404                         TRAP C$BSEG
3689 012404 013701 002716                 1$: MOV KMCSR, R1                               ;PUT REGISTER ADDRESS IN R1
3690 012410 012702 010000                 MOV #BIT12, R2                               ;PUT DATA IN 'EXPECTED'
3691 012414 010211                         MOV R2, (R1)                                ;WRITE BIT 12
3692 012416 011104                         MOV (R1), R4                                ;READ CONTROL STATUS REGISTER
3693 012420 020204                         CMP R2, R4                                  ;IS DATA CORRECT
3694 012422 001413                         BEQ 2$                                       ;BR IF YES
3695 012424                               ERROR 26                                     ;DATA ERROR
3696 012442 104455                         TRAP C$ERDF
3697 012444 000032                         .WORD 26
3698 012446 005474                         .WORD EM26
3699 012450 007566                         .WORD ERR26
3700 012452                               2$: ESCAPE SEG
3701 012452 104410                         TRAP C$ESCAPE
3702 012454 000002                         .WORD 10000$-
3703 012456                               FNDSEG
3704 012456                               10000$:
3705 012456 104405                         TRAP C$ESEG
3706 012460                               BGNSEG
3707 012460 104404                         TRAP C$BSEG
3708 012462 042711 010000                 3$: BIC #BIT12, (R1)                          ;CLEAR BIT 12
3709 012466 005002                         CLR R2                                       ;CLEAR 'EXPECTED'
3710 012470 011104                         MOV (R1), R4                                ;READ CONTROL STATUS REGISTER
3711 012472 001414                         BEQ 4$                                       ;BR IF ZERO
3712 012474                               ERROR 26                                     ;DATA ERPOR BIT12 NOT CLEARED
3713 012512 104455                         TRAP C$ERDF
3714 012514 000032                         .WORD 26
3715 012516 005474                         .WORD EM26
3716 012520 007566                         .WORD ERR26
3717 012522                               ENDSEG
3718 012522                               10001$:
3719 012522 104405                         TRAP C$ESEG
3720 012524                               4$:
3721 012524                               ENDTST
3722 012524                               L10060:
3723 012524 104401                         TRAP C$ETST
3724
3725 012526                               BADHEAD
3726                                     ;***** TEST 8 *****
3727                                     ;*CONTROL OUT REGISTER WRITE/READ TEST
3728                                     ;*FLOAT A ONE THROUGH SEL2
3729 012526                               BADHEAD
3730                                     ;***** TEST 8 *****
3731
3732 012526                               BGNTST
3733 012526                               T8:
3734 012526                               MSTCLR                               ;MASTER CLEAR M8200,4,7
```



```

3735 012526 004537 003156          JSR    R5, .MSTCLR          ;CLEAR MB200,4,7
3736 012532 012737 000002 002624  MOV    #2, MRO
3737 012540 012702 000001          MOV    #1, R2
3738 012544          BGNSEG
3739 012544 104404          TRAP   C$BSEG
3740
3741 012546 013701 002722          1$:   MOV    KMCTL, R1          ;PUT REGISTER ADDRESS IN R1
3742 012552 010237 002636          MOV    R2, $GDDAT          ;PUT DATA IN 'EXPECTED'
3743 012556 013711 002636          MOV    $GDDAT, (R1)        ;WRITE BIT 0
3744 012562 011104          MOV    (R1), R4            ;READ CONTROL OUT REGISTER
3745 012564 023704 002636          CMP    $GDDAT, R4          ;IS DATA CORRECT
3746 012570 001411          BEQ    2$                  ;BR IF YES
3747 012572          ERROR  27, YES            ;DATA ERROR
3748 012604 104455          TRAP   C$ERDF
3749 012606 000033          .WORD  27
3750 012610 005525          .WORD  EM27
3751 012612 007644          .WORD  ERR27
3752 012614          2$:   ESCAPE SEG
3753 012614 104410          TRAP   C$ESCAPE
3754 012616 000046          .WORD  10000$-
3755 012620 040211          3$:   BIC    R2, (R1) ;CLEAR BIT
3756 012622 005037 002636          CLR    $GDDAT ;CLEAR 'EXPECTED'
3757 012626 011104          MOV    (R1), R4            ;READ CONTROL OUT REGISTER
3758 012630 001411          BEQ    4$                  ;BR IF ZERO
3759 012632          ERROR  27, YES            ;DATA ERROR BIT0 NOT CLEARED
3760 012644 104455          TRAP   C$ERDF
3761 012646 000033          .WORD  27
3762 012650 005525          .WORD  EM27
3763 012652 007644          .WORD  ERR27
3764 012654          4$:   ESCAPEE SEG
3765 012654 104410          TRAP   C$ESCAPE
3766 012656 000006          .WORD  10000$-
3767 012660 006302          ASL    R2
3768 012662 001331          BNE    1$
3769 012664          ENDSEG
3770 012664          10000$:
3771 012664 104405          TRAP   C$ESEG
3772 012666          ENDTST
3773 012666          L10061:
3774 012666 104401          TRAP   C$ETST
3775
3776
3777
3778
3779
3780
3781 012670          BADHEAD
3782          ;***** TEST 9 *****
3783          ;*PORT4 REGISTER WRITE/READ TEST
3784          ;*FLOAT A ONE THROUGH PORT4 REGISTER
3785          ;*FLOAT A ZERO THROUGH PORT4 REGISTER
3786 012670          BADHEAD
3787          ;***** TEST 9 *****
3788
3789
3790 012670          BGNTST

```

```

3791 012670
3792 012670 012737 000004 002624 19:: MOV #4,MRO
3793 012676 MSTCLR ;MASTER CLEAR M8200,4,7
3794 012676 004537 003156 JSR R5,,MSTCLR ;CLEAR M8200,4,7
3795 012702 013701 002724 MOV KMP04,R1 ;PUT REGISTER ADDRESS IN R1
3796 012706 012702 000001 MOV #1,R2 ;START WITH BIT0
3797 012712
3798 012712 104404
3799 012714 64$: TRAP C$BSEG
3800 012714 010211 MOV R2,(R1) ;WRITE PORT4 REGISTER
3801 012716 011104 MOV (R1),R4 ;READ PORT4 REGISTER
3802 012720 020204 CMP R2,R4 ;COMPARE EXPECTED AND FOUND
3803 012722 001413 BEQ 65$ ;BR IF OK
3804 012724 ERROR 27 ;WRITE/READ ERROR
3805 012742 104455 TRAP C$ERDF
3806 012744 000033 .WORD 27
3807 012746 005525 .WORD EM27
3808 012750 007644 .WORD ERR27
3809 012752 65$: ESCAPE SEG
3810 012752 104410 TRAP C$ESCAPE
3811 012754 000010 .WORD 10000$-.
3812 012756 000241 CLC ;CLEAR CARRY
3813 012760 006102 ROL R2 ;SHIFT TO NEXT BIT
3814 012762 001354 BNE 64$ ;BR IF NOT DONE YET?
3815 012764
3816 012764 10000$: ENDSEG
3817 012764 104405 TRAP C$ESEG
3818 012766 012702 000001 MOV #1,R2 ;START WITH BIT0
3819 012772
3820 012772 104404 TRAP C$BSEG
3821 012774 66$:
3822 012774 005102 COM R2 ;CHANGE TO A FLOATING ZERO
3823 012776 010211 MOV R2,(R1) ;WRITE PORT4 REGISTER
3824 013000 011104 MOV (R1),R4
3825 013002 020204 CMP R2,R4 ;COMPARE EXPECTED AND FOUND
3826 013004 001413 BEQ 67$ ;BR IF OK
3827 013006 ERROR 27 ;WRITE/READ ERROR
3828 013024 104455 TRAP C$ERDF
3829 013026 000033 .WORD 27
3830 013030 005525 .WORD EM27
3831 013032 007644 .WORD ERR27
3832 013034 67$: ESCAPE SEG
3833 013034 104410 TRAP C$ESCAPE
3834 013036 000012 .WORD 10001$-.
3835 013040 005102 COM R2 ;CHANGE BACK TO A FLOATING ONE
3836 013042 000241 CLC ;CLEAR CARRY
3837 013044 006102 ROL R2 ;SHIFT TO NEXT BIT
3838 013046 001352 BNE 66$ ;BR IF NOT DONE YET?
3839 013050
3840 013050 10001$: ENDSEG
3841 013050 104405 TRAP C$ESEG
3842 013052
3843 013052
3844 013052 104401 L10062: TRAP C$ETST
3845
3846 013054 BADHEAD
  
```

```
3847 :***** TEST 10 *****
3848 :*PORT6 REGISTER WRITE/READ TEST
3849 :*FLOAT A ONE THROUGH PORT6 REGISTER
3850 :*FLOAT A ZERO THROUGH PORT6 REGISTER
3851 013054 BADHEAD
3852 :***** TEST 10 *****
3853
3854 013054 BGNTST
3855 013054 T10::
3856 013054 012737 000006 002624 MOV #6,MPO
3857 013062 MSTCLR ;MASTER CLEAR M8200,4,7
3858 013062 004537 003156 JSR R5,.MSTCLR ;CLEAR M8200,4,7
3859 013066 013701 002726 MOV KMP06,R1 ;PUT REGISTER ADDRESS IN R1
3860 013072 012702 000001 MOV #1,R2 ;START WITH BIT0
3861 013076 BGNSEG
3862 013076 104404 TRAP C$BSEG
3863 013100
3864 013100 010211 64$: MOV R2,(R1) ;WRITE PORT6 REGISTER
3865 013102 011104 MOV (R1),R4 ;READ PORT6 REGISTER
3866 013104 020204 CMP R2,R4 ;COMPARE EXPECTED AND FOUND
3867 013106 001413 BEQ 65$ ;BR IF OK
3868 013110 ERROR 27 ;WRITE/READ ERROR
3869 013126 104455 TRAP C$ERDF
3870 013130 000033 .WORD 27
3871 013132 005525 .WORD EM27
3872 013134 007644 .WORD ERR27
3873 013136 65$: ESCAPE SEG
3874 013136 104410 TRAP C$ESCAPE
3875 013140 000010 .WORD 10000$-.
3876 013142 000241 CLC ;CLEAR CARRY
3877 013144 006105 ROL R5 ;SHIFT TO NEXT BIT
3878 013146 001354 BNE 64$ ;BR IF NOT DONE YET?
3879 013150 ENDSEG
3880 013150 10000$:
3881 013150 104405 TRAP C$ESEG
3882 013152 012702 000001 MOV #1,R2 ;START WITH BIT0
3883 013156 BGNSEG
3884 013156 104404 TRAP C$BSEG
3885 013160 66$:
3886 013160 005102 COM R2 ;CHANGE TO A FLOATING ZERO
3887
3888 013162 010211 MOV R2,(R1) ;WRITE PORT6 REGISTER
3889 013164 011104 MOV (R1),R4 ;READ PORT6 REGISTER
3890 013166 020204 CMP R2,R4 ;COMPARE EXPECTED AND FOUND
3891 013170 001413 BEQ 67$ ;BR IF OK
3892 013172 ERROR 27 ;WRITE/READ ERROR
3893 013210 104455 TRAP C$ERDF
3894 013212 000033 .WORD 27
3895 013214 005525 .WORD EM27
3896 013216 007644 .WORD ERR27
3897 013220 67$: ESCAPE SEG
3898 013220 104410 TRAP C$ESCAPE
3899 013222 000012 .WORD 10001$-.
3900 013224 005102 COM R2 ;CHANGE BACK TO A FLOATING ONE
3901 013226 000241 CLC ;CLEAR CARRY
3902 013230 006102 ROL R2 ;SHIFT TO NEXT BIT
```

```

3903 013232 001352          BNE      668          ;BR IF NOT DONE YET?
3904 013234          ENDSEG
3905 013234          10001$:
3906 013234 104405          TRAP    C$ESEG
3907 013236          ENDTST
3908 013236          L10063:
3909 013236 104401          TRAP    C$ETST
3910
3911 013240          BADHEAD
3912          ;***** TEST 11 *****
3913          ;*UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
3914          ;*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
3915          ;*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
3916 013240          BADHEAD
3917          ;***** TEST 11 *****
3918
3919 013240          BGNTST
3920 013240          T11::
3921 013240
3922 013240 013701 002716          MYINT
3923 013244          MOV     KMCSR,R1          ;GET DEVICE ADDRESS.
3924 013244 004537 003156          MSTCLR          ;MASTER CLEAR M8200,4,7
3925 013250 012702 000001          JSR     R5, .MSTCLR          ;CLEAR M8200,4,7
3926 013254          MOV     #1,R2          ;START PATTERN AT 1
3927 013254 104404          BGNSEG
3928 013256 105011          1$: TRAP    C$BSEG
3929 013260 110211          CLR    (R1)          ;CLEAR REGISTER
3930 013262 111104          MOV    R2,(R1)          ;WRITE M8200,4,7 REGISTER WITH PATTERN
3931 013264 120204          MOV    (R1),R4          ;READ M8200,4,7 REGISTER INTO 'FOUND'
3932 013266 001413          CMP    R2,R4          ;IS DATA CORRECT
3933 013270          BEQ    2$          ;BR IF YES
3934 013306 104455          ERROR  2          ;DATA ERROR
3935 013310 000002          TRAP    C$ERDF
3936 013312 004307          .WORD  2
3937 013314 006132          .WORD  EM2
3938 013316          .WORD  ERR2
3939 013316 104410          2$: ESCAPE  SEG
3940 013320 000024          TRAP    C$ESCAPE
3941 013322 105721          .WORD  10000$-
3942 013324 005202          TSTB   (R1)+          ;NEXT REGISTER
3943 013326 022702 000011          INC    R2          ;INCREMENT DATA PATTERN
3944 013332 001351          CMP    #11,R2          ;LAST REGISTER?
3945 013334 013701 002716          BNE    1$          ;BR IF NO
3946 013340 012702 000001          MOV    KMCSR,R1          ;BASE M8200,4,7 ADDRESS TO R1
3947 013344          MOV    #1,R2          ;RESTART PATTERN AT 1
3948 013344          ENDSEG
3949 013344 104405          10000$: TRAP    C$ESEG
3950 013346          BGNSEG
3951 013346 104404          TRAP    C$BSEG
3952 013350          3$:
3953 013350 111104          MOV    (R1),R4          ;READ COMM.MICRO-PROCESSOR FAMILY REGISTER INTO 'FOUND'
3954 013352 120204          CMP    R2,R4          ;IS DATA CORRECT
3955 013354 001413          BEQ    4$          ;BR IF YES
3956 013356          ERROR  2          ;DUAL ADDRESSING ERROR
3957 013374 104455          TRAP    C$ERDF
3958 013376 000002          .WORD  2

```

3959	013400	004307						
3960	013402	006132						
3961	013404			4\$:	.WORD	EM2		
3962	013404	104410			.WORD	ERR2		
3963	013406	000014			ESCAPE	SEG		
3964	013410	105721			TRAP	C\$ESCAPE		
3965	013412	005202			.WORD	10001\$-		
3966	013414	022702	000011		TSTB	(R1)+		:NEXT REGISTER
3967	013420	001353			INC	R2		:INCREMENT PATTERN
3968	013422				CMP	#11,R2		:LAST REGISTER?
3969	013422				BNE	3\$:BR IF NO
3970	013422	104405			ENDSEG			
3971	013424				10001\$:	TRAP	C\$ESEG	
3972	013424				FNDTST			
3973	013424	104401			L10064:	TRAP	C\$ETST	
3974								
3975	013426				BADHEAD			
3976					:***** TEST 12 *****			
3977					:*MAINTENANCE INSTRUCTION REGISTER TEST			
3978					:*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'			
3979					:*AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A BUS RESET.			
3980	013426				BADHEAD			
3981					:***** TEST 12 *****			
3982								
3983	013426				BGNTST			
3984	013426				T12::			
3985								:R1 CONTAINS BASE M8200,4,7 ADDRESS
3986	013426				MSTCLR			:MASTER CLEAR M8200,4,7
3987	013426	004537	003156		JSR	R5,.MSTCLR		:CLEAR M8200,4,7
3988	013432				MYINT			
3989	013432	013701	002716		MOV	KMCSR,R1		:GET DEVICE ADDRESS.
3990	013436				BGNSEG			
3991	013436	104404			TRAP	C\$BSEG		
3992	013440	012711	003000		MOV	#BIT9:BIT10,(R1)		:SEL6 IS NOW THE IR
3993	013444	005002			CLR	R2		:PUT 'EXPECTED' IN \$GDDAT
3994	013446	010261	000006	1\$:	MOV	R2,6(R1)		:CLEAR THE IR
3995	013452	016104	000006		MOV	6(R1),R4		:READ THE IR
3996	013456	020204			CMP	R2,R4		:IS IT CLEARED?
3997	013460	001413			BEG	2\$:BR IF YES
3998	013462				ERROR	26		:ERROR IR IS NOT CLEAR
3999	013500	104455			TRAP	C\$ERDF		
4000	013502	000032			.WORD	26		
4001	013504	005474			.WORD	EM26		
4002	013506	007566			.WORD	ERR26		
4003	013510				2\$:	ESCAPE	SEG	
4004	013510	104410			TRAP	C\$ESCAPE		
4005	013512	000002			.WORD	10000\$-		
4006	013514				ENDSEG			
4007	013514				10000\$:	TRAP	C\$ESEG	
4008	013514	104405			MOV	#-1,R2		:PUT 'EXPECTED' IN \$GDDAT
4009	013516	012702	177777		BGNSEG			
4010	013522				TRAP	C\$BSEG		
4011	013522	104404			3\$:	MOV	R2,6(R1)	:WRITE ALL ONES TO THE IR
4012	013524	010261	000006		MOV	6(R1),R4		:READ THE IR
4013	013530	016104	000006		CMP	R2,R4		:IS IT ALL ONES?
4014	013534	020204						

4015 013536 001413
4016 013540
4017 013556 104455
4018 013560 000032
4019 013562 005474
4020 013564 007566
4021 013566
4022 013566 104410
4023 013570 000002
4024 013572
4025 013572
4026 013572 104405
4027 013574
4028 013574
4029 013574 104401
4030
4031

BEO 48
ERROR 26
TRAP C\$ERDF
.WORD 26
.WORD EM26
.WORD ERR26
4\$: ESCAPE SEG
TRAP C\$ESCAPE
.WORD 10001\$-
ENDSEG
10001\$:
TRAP C\$ESEG
ENDTST
L10065:
TRAP C\$ETST

;BR IF YES
;ERROR IR IS NOT = ALL ONES

4032
4033 013576
4034
4035
4036
4037
4038
4039
4040 013576
4041
4042
4043 013576
4044 013576
4045 013576
4046 013576 013701 002716
4047 013602
4048 013602 004537 003156
4049 013606 012761 000377 000004
4050 013614 012711 001000
4051 013620 012761 121105 000006
4052 013626 052711 001400
4053 013632 000240
4054 013634 012702 177777
4055 013640 116104 000004
4056 013644 020204
4057 013646 001413
4058 013650
4059 013666 104455
4060 013670 000034
4061 013672 005554
4062 013674 007726
4063
4064 013676
4065 013676 104410
4066 013700 000002
4067
4068 013702
4069 013702
4070 013702 104401
4071
4072 013704
4073
4074
4075
4076
4077 013704
4078
4079
4080 013704
4081 013704
4082 013704
4083 013704 004537 003156
4084 013710 012737 000000 002624
4085 013716 012705 000001
4086
4087 013722

BADHEAD
:***** TEST 13 *****
:*MICRO PROCESSOR TEST
:*LOAD KMPO6 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
:*VERIFY INSTRUCTION EXECUTED PROPERLY
:*INSTRUCTION SHOULD MOVE IBUS*4 TO IBUS*5, IBUS*4 IS ALL 1'S
:*AND IBUS*5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4
BADHEAD
:***** TEST 13 *****

EGNTST
T13::

MYINT
MOV KMCSR,R1 ;GET DEVICE ADDRESS.
MSTCLR
JSR R5, MSTCLR ;CLEAR M8200,4,7
MOV #377,4(R1) ;PORT4 HI BYTE=1'S
MOV #BIT9,(R1) ;SET ROMI
MOV #121105,6(R1) ;INSTR TO PORT 6.
BIS #BIT8!BIT9,(R1) ;CLK INSTR.
NOP
MOV #-1,R2 ;EXPECT ALL ONES.
MOVB 4(R1),R4 ;READ FOUND.
CMP R2,R4 ;DATA CORRECT?
BEQ 1\$
ERROR 28
TRAP C\$ERDF
.WORD 28
.WORD EM28
.WORD ERR28

1\$:
ESCAPE TST
TRAP C\$ESCAPE
.WORD L10066-

ENDTST
L10066:

TRAP C\$ETST

BADHEAD
:***** TEST 14 *****
:*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
:*FLOAT A 1 THROUGH IBUS* REGISTER 0
:*FLOAT A 0 THROUGH IBUS* REGISTER 0
BADHEAD
:***** TEST 14 *****

BGNTST
T14::

MSTCLR ;MASTER CLEAR M8200,4,7
JSR R5, MSTCLR ;CLEAR M8200,4,7
MOV #0,MRO ;SAVE REGISTER ADDRESS FOR TYPEOUT
MOV #1,R5 ;START WITH BIT 0

MYINT

4088	013722	013701	002716		MOV	KMCSR,R1		:GET DEVICE ADDRESS.
4089	013726				BGNSEG			
4090	013726	104404			TRAP	C\$BSEG		
4091	013730			64\$:				
4092	013730	010561	000004		MOV	R5,4(R1)		:PUT PATTERN INTO PORT4
4093	013734				ROMCLK			:NEXT WORD IS INSTRUCTION, BBN
4094	013734	004537	003244		JSR	R5,..ROMCLK		:CLOCK INSTRUCTION
4095	013740	121100			121100			:MOV DATA TO IBUS* REGISTER 0
4096	013742				ROMCLK			:NEXT WORD IS INSTRUCTION, BBN
4097	013742	004537	003244		JSR	R5,..ROMCLK		:CLOCK INSTRUCTION
4098	013746	121005			121005			:READ FROM IBU* REGITER 0
4099	013750	116104	000005		MOVB	5(R1),R4		:PUT 'FOUND' INTO R4
4100	013754	120504			CMPB	R5,R4		:DATA CORRECT?
4101	013756	001414			BEQ	65\$:BR IF YES
4102	013760				BERROR	27		:ERROR
4103	014000	104455			TRAP	C\$ERDF		
4104	014002	000033			.WORD	27		
4105	014004	005525			.WORD	EM27		
4106	014006	007644			.WORD	ERR27		
4107	014010			65\$:	ESCAPE	SEG		
4108	014010	104410			TRAP	C\$ESCAPE		
4109	014012	000010			.WORD	10000\$-		
4110	014014	000241			CLC			:CLEAR CARRY
4111	014016	106105			ROLB	R5		:SHIFT BIT IN R5
4112	014020	001343			BNE	64\$:IF R2=0 THEN DONE
4113	014022				ENDSEG			
4114	014022			10000\$:				
4115	014022	104405			TRAP	C\$ESEG		
4116	014024	012705	000001		MOV	#,R5		:START WITH BIT 0
4117				:69\$:	COM	R5		:CHANGE TO FLOATING ZERO
4118	014030				BGNSEG			
4119	014030	104404			TRAP	C\$BSEG		
4120	014032			67\$:				
4121	014032	005105			COM	R5		
4122	014034	010561	000004		MOV	R5,4(R1)		:PUT PATTERN INTO PORT4
4123	014040				ROMCLK			:NEXT WORD IS INSTRUCTION, BBN
4124	014040	004537	003244		JSR	R5,..ROMCLK		:CLOCK INSTRUCTION
4125	014044	121100			121100			:MOV DATA TO IBUS* REGISTER 0
4126	014046				ROMCLK			:NEXT WORD IS INSTRUCTION, BBN
4127	014046	004537	003244		JSR	R5,..ROMCLK		:CLOCK INSTRUCTION
4128	014052	121005			121005			:READ FROM IBUS* REGISTER 0
4129	014054	116104	000005		MOVB	5(R1),R4		:PUT 'FOUND' INTO R4
4130	014060	120504			CMPB	R5,R4		:DATA CORRECT?
4131	014062	001414			BEQ	68\$:BR IF YES
4132	014064				BERROR	27		:ERROR
4133	014104	104455			TRAP	C\$ERDF		
4134	014106	000033			.WORD	27		
4135	014110	005525			.WORD	EM27		
4136	014112	007644			.WORD	ERR27		
4137	014114			68\$:	ESCAPE	SEG		
4138	014114	104410			TRAP	C\$ESCAPE		
4139	014116	000012			.WORD	10001\$-		
4140	014120	005105			COM	R5		:CHANGE TO FLOATING 1
4141	014122	000241			CLC			:CLEAR CARRY
4142	014124	106105			ROLB	R5		:SHIFT BIT IN R5
4143	014126	001341			BNE	67\$:IF R2=0 THEN DONE


```
4144 014130          ENDSEG
4145 014130          10001$:
4146 014130 104405   TRAP    C$ESEG
4147 014132          ENDTST
4148 014132          L10067:
4149 014132 104401   TRAP    C$ETST
4150
4151 014134          BADHEAD
4152                :***** TEST 15 *****
4153                :*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
4154                :*FLOAT A 1 THROUGH IBUS* REGISTER 2
4155                :*FLOAT A 0 THROUGH IBUS* REGISTER 2
4156 014134          BADHEAD
4157                :***** TEST 15 *****
4158
4159 014134          BGNTST
4160 014134          T15::
4161 014134          MSTCLR          ;MASTER CLEAR M8200,4,7
4162 014134 004537 003156          JSR    R5, .MSTCLR          ;CLEAR M8200,4,7
4163 014140 012737 000002 002624  MOV    #2, MRO          ;SAVE REGISTER ADDRESS FOR TYPEOUT
4164 014146 012705 000001          MOV    #1, R5          ;START WITH BIT 0
4165 014152          MYINT
4166 014152 013701 002716          MOV    KMCSR, R1          ;GET DEVICE ADDRESS.
4167 014156          BGNSEG
4168 014156 104404          TRAP    C$3SEG
4169 014160          64$:
4170 014160 010561 000004          MOV    R5, 4(R1)          ;PUT PATTERN INTO PORT4
4171 014164          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4172 014164 004537 003244          JSR    R5, .ROMCLK          ;CLOCK INSTRUCTION
4173 014170 121102          121100!2          ;MOV DATA TO IBUS* REGISTER 0
4174 014172          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4175 014172 004537 003244          JSR    R5, .ROMCLK          ;CLOCK INSTRUCTION
4176 014176 121045          121005!<2*20>          ;READ FROM IBUS* REGISTER 2
4177 014200 116104 000005          MOVB   5(R1), R4          ;PUT 'FOUND' INTO R4
4178 014204 120504          CMPB   R5, R4          ;DATA CORRECT?
4179 014206 001414          BEQ    65$          ;BR IF YES
4180 014210          BERROR 27          ;ERROR
4181 014230 104455          TRAP    C$ERDF
4182 014232 000033          .WORD 27
4183 014234 005525          .WORD EM27
4184 014236 007644          .WORD ERR27
4185 014240          65$:
4186 014240 104410          ESCAPE SEG
4187 014242 000010          TRAP    C$ESCAPE
4188 014244 000241          .WORD 10000$-.
4189 014246 106105          CLC
4190 014250 001343          ROLB   R5          ;CLEAR CARRY
4191 014252          BNE    64$          ;SHIFT BIT IN R2
4192 014252          ENDSEG          ;IF R2=0 THEN DONE
4193 014252 104405          10000$:
4194 014254 012705 000001          TRAP    C$ESEG
4195          :69$:
4196 014260          COM    R5          ;START WITH BIT 0
4197 014260 104404          BGNSEG          ;CHANGE TO FLOATING ZERO
4198 014262          TRAP    C$BSEG
4199 014262 005105          COM    R5
```

```
4200 014264 010561 000004      MOV      R5,4(R1)          ;PUT PATTERN INTO PORT4
4201 014270                    ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
4202 014270 004537 003244      JSR      R5,.ROMCLK       ;CLOCK INSTRUCTION
4203 014274 121102                    121100!2                 ;MOV DATA TO IBUS* REGISTER 2
4204 014276                    ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
4205 014276 004537 003244      JSR      R5,.ROMCLK       ;CLOCK INSTRUCTION
4206 014302 121045                    121005!<2*20>           ;READ FROM IBUS* REGISTER 2
4207 014304 116104 000005      MOV      5(R1),R4         ;PUT 'FOUND' INTO R4
4208 014310 120504                    CMPB     R5,R4            ;DATA CORRECT?
4209 014312 001414                    BEQ      68$              ;BR IF YES
4210 014314                    BERROR  27                ;ERROR
4211 014334 104455                    TRAP    C$ERDF
4212 014336 000033                    .WORD   27
4213 014340 005525                    .WORD   EM27
4214 014342 007644                    .WORD   ERR27
4215 014344                    68$: ESCAPE  SEG
4216 014344 104410                    TRAP    C$ESCAPE
4217 014346 000012                    .WORD   10001$-.
4218 014350 005105                    COM     R5                ;CHANGE TO FLOATING 1
4219 014352 000241                    CLC
4220 014354 106105                    ROLB   R5                ;CLEAR CARRY
4221 014356 001341                    BNE    67$              ;SHIFT BIT IN R2
4222 014360                    ENDSEG                    ;IF R2=0 THEN DUNE
4223 014360                    10001$:
4224 014360 104405                    TRAP    C$ESEG
4225 014362                    ENDTST
4226 014362                    L10070:
4227 014362 104401                    TRAP    C$ETST
4228
4229 014364                    BADHEAD
4230                    ;***** TEST 16 *****
4231                    ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
4232                    ;*FLOAT A 1 THROUGH IBUS* REGISTER 4
4233                    ;*FLOAT A 0 THROUGH IBUS* REGISTER 4
4234 014364                    BADHEAD
4235                    ;***** TEST 16 *****
4236
4237 014364                    BGNTST
4238 014364                    T16::
4239 014364                    MSTCLR                    ;MASTER CLEAR M8200,4,7
4240 014364 004537 003156      JSR      R5,.MSTCLR       ;CLEAR M8200,4,7
4241 014370 012737 000004 002624  MOV      #4,MRO           ;SAVE REGISTER ADDRESS FOR TYPEOUT
4242 014376 012705 000001      MOV      #1,R5            ;START WITH BIT 0
4243 014402                    MYINT
4244 014402 013701 002716      MOV      KMCSR,R1         ;GET DEVICE ADDRESS.
4245 014406                    BGNSEG
4246 014406 104404                    TRAP    C$BSEG
4247 014410                    64$:
4248 014410 010561 000004      MOV      R5,4(R1)          ;PUT PATTERN INTO PORT4
4249 014414                    ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
4250 014414 004537 003244      JSR      R5,.ROMCLK       ;CLOCK INSTRUCTION
4251 014420 121104                    121100!4                 ;MOV DATA TO IBUS* REGISTER 4
4252 014422                    ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
4253 014422 004537 003244      JSR      R5,.ROMCLK       ;CLOCK INSTRUCTION
4254 014426 121105                    121005!<4*20>           ;READ FROM IBUS* REGISTER 4
4255 014430 116104 000005      MOV      5(R1),R4         ;PUT 'FOUND' INTO R4
```

```
4256 014434 120504          CMPB    R5,R4          ;DATA CORRECT?
4257 014436 001414          BEQ     65$           ;BR IF YES
4258 014440                BERROR  27           ;ERROR
4259 014460 104455          TRAP    C$ERDF
4260 014462 000033          .WORD  27
4261 014464 005525          .WORD  EM27
4262 014466 007644          .WORD  ERR27
4263 014470                65$:  ESCAPE  SEG
4264 014470 104410          TRAP    C$ESCAPE
4265 014472 000010          .WORD  10000$-.
4266 014474 000241          CLC
4267 014476 106105          ROLB   R5            ;CLEAR CARRY
4268 014500 001343          BNE    64$           ;SHIFT BIT IN R2
4269 014502                ENDSEG              ;IF R2=0 THEN DONE
4270 014502                10000$:
4271 014502 104405          TRAP    C$ESEG
4272 014504 012705 000001      MOV     #1,R5        ;START WITH BIT 0
4273                :69$:  COM     R5            ;CHANGE TO FLOATING ZERO
4274 014510                BGNSEG
4275 014510 104404          TRAP    C$BSEG
4276 014512                67$:
4277 014512 005105          COM     R5
4278 014514 010561 000004      MOV     R5,4(R1)    ;PUT PATTERN INTO PORT4
4279 014520                ROMCLK              ;NEXT WORD IS INSTRUCTION, BBN
4280 014520 004537 003244      JSR     R5,.ROMCLK  ;CLOCK INSTRUCTION
4281 014524 121104                121100!4          ;MOV DATA TO IBUS* REGISTER 4
4282 014526                ROMCLK              ;NEXT WORD IS INSTRUCTION, BBN
4283 014526 004537 003244      JSR     R5,.ROMCLK  ;CLOCK INSTRUCTION
4284 014532 121105                121005!<4*20>    ;READ FROM IBUS* REGISTER 4
4285 014534 116104 000005      MOVB   5(R1),R4     ;PUT "FOUND" INTO R4
4286 014540 120504          CMPB   R5,R4        ;DATA CORRECT?
4287 014542 001414          BEQ    68$           ;BR IF YES
4288 014544                BERROR  27           ;ERROR
4289 014564 104455          TRAP    C$ERDF
4290 014566 000033          .WORD  27
4291 014570 005525          .WORD  EM27
4292 014572 007644          .WORD  ERR27
4293 014574                68$:  ESCAPE  SEG
4294 014574 104410          TRAP    C$ESCAPE
4295 014576 000012          .WORD  10001$-.
4296 014600 005105          COM     R5            ;CHANGE TO FLOATING 1
4297 014602 000241          CLC
4298 014604 106105          ROLB   R5            ;CLEAR CARRY
4299 014606 001341          BNE    67$           ;SHIFT BIT IN R2
4300 014610                ENDSEG              ;IF R2=0 THEN DONE
4301 014610                10001$:
4302 014610 104405          TRAP    C$ESEG
4303 014612                ENDTST
4304 014612                L10071:
4305 014612 104401          TRAP    C$ETST
4306
4307 014614          BADHEAD
4308          ;***** TEST 17 *****
4309          ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
4310          ;*FLOAT A 1 THROUGH IBUS* REGISTER 5
4311          ;*FLOAT A 0 THROUGH IBUS* REGISTER 5
```

```
4312 014614          BADHEAD
4313                  ;***** TEST 17 *****
4314
4315 014614          BGNTST
4316 014614          T17::
4317 014614          MSTCLR          ;MASTER CLEAR M8200,4,7
4318 014614 004537 003156          JSR      R5, .MSTCLR          ;CLEAR M8200,4,7
4319 014620 012737 000005 002624  MOV      #5, MRO          ;SAVE REGISTER ADDRESS FOR TYPEOUT
4320 014626 012705 000001          MOV      #1, R5          ;START WITH BIT 0
4321 014632          MYINT
4322 014632 013701 002716          MOV      KMCSR, R1          ;GET DEVICE ADDRESS.
4323 014636          BGNSEG
4324 014636 104404          TRAP     C$BSEG
4325 014640          64$:
4326 014640 010561 000004          MOV      R5, 4(R1)          ;PUT PATTERN INTO PORT4
4327 014644          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4328 014644 004537 003244          JSR      R5, .ROMCLK          ;CLOCK INSTRUCTION
4329 014650 121105          121100!5          ;MOV DATA TO IBUS* REGISTER 5
4330 014652          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4331 014652 004537 003244          JSR      R5, .ROMCLK          ;CLOCK INSTRUCTION
4332 014656 121125          121005!<5*20>          ;READ FROM IBUS* REGISTER 5
4333 014660 116104 000005          MOV      5(R1), R4          ;PUT 'FOUND' INTO R4
4334 014664 120504          CMPB    R5, R4          ;DATA CORRECT?
4335 014666 001414          BEQ     65$          ;BR IF YES
4336 014670          BERROR 27          ;ERROR
4337 014710 104455          TRAP     C$ERDF
4338 014712 000033          .WORD   27
4339 014714 005525          .WORD   EM27
4340 014716 007644          .WORD   ERR27
4341 014720          65$:
4342 014720 104410          ESCAPE  SEG
4343 014722 000010          TRAP     C$ESCAPE
4344 014724 000241          .WORD   10000$-.
4345 014726 106105          CLC          ;CLEAR CARRY
4_46 014730 001343          ROLB    R5          ;SHIFT BIT IN R5
4347 014732          BNE     64$          ;IF R5=0 THEN DONE
4348 014732          ENDSEG
4349 014732 104405          10000$:
4350 014734 012705 000001          TRAP     C$ESEG
4351 014740          69$:
4352 014740          COM     R5          ;START WITH BIT 0
4353 014740 104404          BGNSEG          ;CHANGE TO FLOATING ZERO
4354 014742          TRAP     C$BSEG
4355 014742 005105          67$:
4356 014744 010561 000004          COM     R5
4357 014750          MOV      R5, 4(R1)          ;PUT PATTERN INTO PORT4
4358 014750 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4359 014754 121105          JSR      R5, .ROMCLK          ;CLOCK INSTRUCTION
4360 014756          121100!5          ;MOV DATA TO IBUS* REGISTER 5
4361 014756 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4362 014762 121125          JSR      R5, .ROMCLK          ;CLOCK INSTRUCTION
4363 014764 116104 000005          121005!<5*20>          ;READ FROM IBUS* REGISTER 5
4364 014770 120504          MOV      5(R1), R4          ;PUT 'FOUND' INTO R4
4365 014772 001414          CMPB    R5, R4          ;DATA CORRECT?
4366 014774          BEQ     68$          ;BR IF YES
4367 015014 104455          BERROR 27          ;ERROR
4367 015014 104455          TRAP     C$ERDF
```

```

4368 015016 000033          .WORD 27
4369 015020 005525          .WORD EM27
4370 015022 007644          .WORD ERR27
4371 015024                68$: ESCAPE SEG
4372 015024 104410          TRAP C$ESCAPE
4373 015026 000012          .WORD 10001$-.
4374 015030 005105          COM R5          ;CHANGE TO FLOATING 1
4375 015032 000241          CLC          ;CLEAR CARRY
4376 015034 106105          ROLB R5        ;SHIFT BIT IN R5
4377 015036 001341          BNE 67$       ;IF R5=0 THEN DONE
4378 015040                ENDSEG
4379 015040                10001$:
4380 015040 104405          TRAP C$ESEG
4381 015042                ENDTST
4382 015042                L10072:
4383 015042 104401          TRAP C$ETST
4384
4385 015044                BADHEAD
4386                ;***** TEST 18 *****
4387                ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
4388                ;*FLOAT A 1 THROUGH IBUS* REGISTER 10
4389                ;*FLOAT A 0 THROUGH IBUS* REGISTER 10
4390 015044                BADHEAD
4391                ;***** TEST 18 *****
4392
4393 015044                BGNTST
4394 015044                T18:
4395 015044                MSTCLR          ;MASTER CLEAR M8200,4,7
4396 015044 004537 003156      JSR R5, MSTCLR          ;CLEAR M8200,4,7
4397 015050 012737 000010 002624  MOV #10, MRO          ;SAVE REGISTER ADDRESS FOR TYPEOUT
4398 015056 012705 000001          MOV #1, R5          ;START WITH BIT 0
4399 015062                MYINT
4400 015062 013701 002716      MOV KMCSR, R1        ;GET DEVICE ADDRESS.
4401 015066                BGNSEG
4402 015066 104404                TRAP C$BSEG
4403 015070                64$:
4404 015070 010561 000004      MOV R5, 4(R1)        ;PUT PATTERN INTO PORT4
4405 015074 042761 000141 000004  BIC #141, 4(R1)      ;CLEAR UNWANTED BITS
4406 015102                ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4407 015102 004537 003244      JSR R5, ROMCLK        ;CLOCK INSTRUCTION
4408 015106 121110          121100!10          ;MOV DATA TO IBUS* REGISTER 10
4409 015110                ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4410 015110 004537 003244      JSR R5, ROMCLK        ;CLOCK INSTRUCTION
4411 015114 121205          121005!<10*20>      ;READ FROM IBUS* REGISTER 10
4412 015116 010502          MOV R5, R2
4413 015120 042705 000141      BIC #141, R5        ;CLEAR UNWANTED BITS
4414 015124 116104 000005      MOVB 5(R1), R4      ;PUT "FOUND" INTO R4
4415 015130 120504          CMPB R5, R4        ;DATA CORRECT?
4416 015132 001414          BEQ 65$          ;BR IF YES
4417 015134                BERROR          ;ERROR
4418 015154 104455          TRAP C$ERDF
4419 015156 000033          .WORD 27
4420 015160 005525          .WORD EM27
4421 015162 007644          .WORD ERR27
4422 015164                65$: ESCAPE SEG
4423 015164 104410          TRAP C$ESCAPE

```

```

4424 015166 000012          .WORD 10000$-.
4425 015170 010205          MOV R2,R5
4426 015172 000241          CLC
4427 015174 106105          ROLB R5 ;CLEAR CARRY
4428 015176 001334          BNE 64$ ;SHIFT BIT IN R5
4429 015200                ENDSEG ;IF R5=0 THEN DONE
4430 015200                10000$:
4431 015200 104405          TRAP C$ESEG
4432 015202 012705 000001    MOV #1,R5 ;START WITH BIT 0
4433                :69$: COM R5 ;CHANGE TO FLOATING ZERO
4434 015206                BGNSEG
4435 015206 104404          TRAP C$BSEG
4436 015210                67$:
4437 015210 005105          COM R5
4438 015212 010561 000004    MOV R5,4(R1) ;PUT PATTERN INTO PORT4
4439 015216 042761 000141 000004 BIC #141,4(R1) ;CLEAR UNWANTED BITS
4440 015224                ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4441 015224 004537 003244    JSR R5,ROMCLK ;CLOCK INSTRUCTION
4442 015230 121110          121100!10 ;MOV DATA TO IBUS* REGISTER 10
4443 015232                ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4444 015232 004537 003244    JSR R5,ROMCLK ;CLOCK INSTRUCTION
4445 015236 121205          121005!<10*20> ;READ FROM IBUS* REGISTER 10
4446 015240 010502          MOV R5,R2
4447 015242 042705 000141    BIC #141,R5 ;CLEAR UNWANTED BITS
4448 015246 116104 000005    MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
4449 015252 120504          CMPB R5,R4 ;DATA CORRECT?
4450 015254 001414          BEQ 68$ ;BR IF YES
4451 015256                BERROR ;L:ROR
4452 015276 104455          TRAP C$ERDF
4453 015300 000033          .WORD 27
4454 015302 005525          .WORD EM27
4455 015304 007644          .WORD ERR27
4456 015306                68$: ESCAPE SEG
4457 015306 104410          TRAP C$ESCAPE
4458 015310 000014          .WORD 10001$-.
4459 015312 010205          MOV R2,R5
4460 015314 005105          COM R5 ;CHANGE TO FLOATING 1
4461 015316 000241          CLC ;CLEAR CARRY
4462 015320 106105          ROLB R5 ;SHIFT BIT IN R5
4463 015322 001332          BNE 67$ ;IF R5=0 THEN DONE
4464 015324                ENDSEG
4465 015324                10001$:
4466 015324 104405          TRAP C$ESEG
4467 015326                ENDTST
4468 015326                L10073:
4469 015326 104401          TRAP C$ETST
4470
4471 015330                BADHEAD
4472                ;***** TEST 19 *****
4473                ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
4474                ;*FLOAT A 1 THROUGH IBUS* REGISTER 11
4475                ;*FLOAT A 0 THROUGH IBUS* REGISTER 11
4476 015330                BADHEAD
4477                ;***** TEST 19 *****
4478
4479 015330                BGNTST
  
```

4480	015330				T19::				
4481	015330					MSTCLR			;MASTER CLEAR M8200,4,7
4482	015330	004537	003156			JSR	R5, .MSTCLR		;CLEAR M8200,4,7
4483	015334	012737	000011	002624		MOV	#11, MRO		;SAVE REGISTER ADDRESS FOR TYPEOUT
4484	015342	012705	000001			MOV	#1, R5		;START WITH BIT 0
4485	015346					MYINT			
4486	015346	013701	002716			MOV	KMCSR, R1		;GET DEVICE ADDRESS.
4487	015352					BGNSEG			
4488	015352	104404				TRAP	C\$BSEG		
4489	015354				61\$:				
4490	015354	010561	000004			MOV	R5, 4(R1)		;PUT PATTERN INTO PORT4
4491	015360	042761	000262	000004		BIC	#262, 4(R1)		;CLEAR UNWANTED BITS
4492	015366					ROMCLK			;NEXT WORD IS INSTRUCTION, BBN
4493	015366	004537	003244			JSR	R5, .ROMCLK		;CLOCK INSTRUCTION
4494	015372	121111				121100!11			;MOV DATA TO IBUS* REGISTER 11
4495	015374					ROMCLK			;NEXT WORD IS INSTRUCTION, BBN
4496	015374	004537	003244			JSR	R5, .ROMCLK		;CLOCK INSTRUCTION
4497	015400	121225				121005!<11*20>			;READ FROM !BUS* REGISTER 11
4498	015402	010502				MOV	R5, R2		
4499	015404	042705	000262			BIC	#262, R5		;CLEAR UNWANTED BITS
4500	015410	116104	000005			MOVB	5(R1), R4		;PUT 'FOUND' INTO R4
4501	015414	042704	000020			BIC	#20, R4		
4502	015420	120504				CMPB	R5, R4		;DATA CORRECT?
4503	015422	001414				BEQ	65\$;BR IF YES
4504	015424					BERROR	27		;ERROR
4505	015444	104455				TRAP	C\$ERDF		
4506	015446	000033				.WORD	27		
4507	015450	005525				.WORD	EM27		
4508	015452	007644				.WORD	ERR27		

4509	015454			65\$:	ESCAPE	SEG			
4510	015454	104410			TRAP	C\$ESCAPE			
4511	015456	000012			.WORD	10000\$-			
4512	015460	010205			MOV	R2,R5			
4513	015462	000241			CLC				;CLEAR CARRY
4514	015464	106105			ROLB	R5			;SHIFT BIT IN R5
4515	015466	001332			BNE	64\$;IF R5=0 THEN DONE
4516	015470				ENDSEG				
4517	015470			10000\$:					
4518	015470	104405			TRAP	C\$ESEG			
4519	015472	012705	000001		MOV	#1,R5			;START WITH BIT 0
4520				:59\$:	COM	R5			;CHANGE TO FLOATING ZERO
4521	015476				BGNSEG				
4522	015476	104404			TRAP	C\$BSEG			
4523	015500			67\$:					
4524	015500	005105			COM	R5			
4525	015502	010561	000004		MOV	R5,4(R1)			;PUT PATTERN INTO PORT4
4526	015506	042761	000262	000004	BIC	#262,4(R1)			;CLEAR UNWANTED BITS
4527	015514				ROMCLK				;NEXT WORD IS INSTRUCTION, BBN
4528	015514	004537	003244		JSR	R5,.ROMCLK			;CLOCK INSTRUCTION
4529	015520	121111			121100!11				;MOV DATA TO IBUS* REGISTER 11
4530	015522				ROMCLK				;NEXT WORD IS INSTRUCTION, BBN
4531	015522	004537	003244		JSR	R5,.ROMCLK			;CLOCK INSTRUCTION
4532	015526	121225			121005!<11*20>				;READ FROM IBUS* REGISTER 11


```
4533 015530 010502          MOV      R5,R2
4534 015532 042705 000262    BIC      #262,R5          ;CLEAR UNWANTED BITS
4535 015536 052705 000020    BIS      #20,R5          ;ADD THESE BITS
4536 015542 116104 000005    MOVB    5(R1),R4        ;PUT 'FOUND' INTO R4
4537 015546 120504          CMPB    R5,R4          ;DATA CORRECT?
4538 015550 001414          BEQ     68$            ;BR IF YES
4539 015552          BERROR  27            ;ERROR
4540 015572 104455          TRAP   C$ERDF
4541 015574 000033          .WORD  27
4542 015576 005525          .WORD  EM27
4543 015600 007644          .WORD  ERR27
4544 015602          €9$:  ESCAPE  SEG
4545 015602 104410          TRAP   C$ESCAPE
4546 015604 000014          .WORD  10001$-
4547 015606 010205          MOV     R2,R5
4548 015610 005105          COM
4549 015612 000241          CLC
4550 015614 106105          ROLB   R5              ;CHANGE TO FLOATING 1
4551 015616 001330          BNE    67$            ;CLEAR CARRY
4552 015620          ENDSEG              ;SHIFT BIT IN R5
4553 015620          10001$:            ;IF R5=0 THEN DONE
4554 015620 104405          TRAP   C$ESEG
4555 015622          ENDTST
4556 015622          L10074:
4557 015622 104401          TRAP   C$ETST
4558
4559 015624          BADHEAD
4560          ;***** TEST 20 *****
4561          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4562          ;*FLOAT A 1 THROUGH IBUS REGISTER 0
4563          ;*FLOAT A 0 THROUGH IBUS REGISTER 0
4564 015624          BADHEAD
4565          ;***** TEST 20 *****
4566
4567 015624          BGNTST
4568 015624          T20::
4569 015624          MSTCLR          ;MASTER CLEAR M8200.4,7
4570 015624 004537 003156    JSR     R5, .MSTCLR      ;CLEAR M8200.4,7
4571 015630 012737 000000 002624    MOV     #0,MRO          ;SAVE REGISTER ADDRESS FOR TYPEOUT
4572 015636 012705 000001          MOV     #1,R5          ;START WITH BIT 0
4573 015642          MYINT
4574 015642 013701 002716    MOV     KMCSR,R1        ;GET DEVICE ADDRESS.
4575 015646          BGNSEG
4576 015646 104404          TRAP   C$BSEG
4577 015650          64$:
4578 015650 010561 000004          MOV     R5,4(R1)        ;PUT PATTERN INTO PORT4
4579 015654          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4580 015654 004537 003244    JSR     R5, .ROMCLK      ;CLOCK INSTRUCTION
4581 015660 122100          122100          ;MOV DATA TO IBUS* REGISTER 0
4582 015662          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4583 015662 004537 003244    JSR     R5, .ROMCLK      ;CLOCK INSTRUCTION
4584 015666 021005          21005          ;READ FROM IBUS* REGISTER 0
4585 015670 116104 000005    MOVB    5(R1),R4        ;PUT 'FOUND' INTO R4
4586 015674 120504          CMPB    R5,R4          ;DATA CORRECT?
4587 015676 001414          BEQ     65$            ;BR IF YES
4588 015700          BERROR  29            ;ERROR
```

```
4589 015720 104455 TRAP C$ERDF
4590 015722 000035 .WORD 29
4591 015724 005605 .WORD EM29
4592 015726 010004 .WORD ERR29
4593 015730 65$: ESCAPE SEG
4594 015730 104410 TRAP C$ESCAPE
4595 015732 000010 .WORD 10000$-.
4596 015734 000241 CLC ;CLEAR CARRY
4597 015736 106105 ROLB R5 ;SHIFT BIT IN R5
4598 015740 001343 BNE 64$ ;IF R5=0 THEN DONE
4599 015742 ENDSEG
4600 015742 10000$:
4601 015742 104405 TRAP C$ESEG
4602 015744 012705 000001 MOV #1,R5 ;START WITH BIT 0
4603 ;69$: COM R5 ;CHANGE TO FLOATING ZERO
4604 015750 BGNSEG
4605 015750 104404 TRAP C$BSEG
4606 015752 67$:
4607 015752 005105 COM R5
4608 015754 010561 000004 MOV R5,4(R1) ;PUT PATTERN INTO PORT4
4609 015760 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4610 015760 004537 003244 JSR R5,ROMCLK ;CLOCK INSTRUCTION
4611 015764 122100 122100 ;MOV DATA TO IBUS* REGISTER 0
4612 015766 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4613 015766 004537 003244 JSR R5,ROMCLK ;CLOCK INSTRUCTION
4614 015772 021005 21005 ;READ FROM IBUS* REGISTER 0
4615 015774 116104 000005 MOV#B 5(R1),R4 ;PUT 'FOUND' INTO R4
4616 016000 120504 CMPB R5,R4 ;DATA CORRECT?
4617 016002 001414 BEQ 68$ ;BR IF YES
4618 016004 BERROR 29 ;ERROR
4619 016024 104455 TRAP C$ERDF
4620 016026 000035 .WORD 29
4621 016030 005605 .WORD EM29
4622 016032 010004 .WORD ERR29
4623 016034 68$: ESCAPE SEG
4624 016034 104410 TRAP C$ESCAPE
4625 016036 000012 .WORD 10001$-.
4626 016040 005105 COM R5 ;CHANGE TO FLOATING 1
4627 016042 000241 CLC ;CLEAR CARRY
4628 016044 106105 ROLB R5 ;SHIFT BIT IN R5
4629 016046 001341 BNE 67$ ;IF R5=0 THEN DONE
4630 016050 ENDSEG
4631 016050 10001$:
4632 016050 104405 TRAP C$ESEG
4633 016052 ENDTST
4634 016052 L10075:
4635 016052 104401 TRAP C$ETST
4636
4637 016054 BADHEAD
4638 ;***** TEST 21 *****
4639 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4640 ;*FLOAT A 1 THROUGH IBUS REGISTER 1
4641 ;*FLOAT A 0 THROUGH IBUS REGISTER 1
4642 016054 BADHEAD
4643 ;***** TEST 21 *****
4644
```

```

4645 016054          BGNST
4646 016054          T21::
4647 016054          MSTCLR          ;MASTER CLEAR M8200,4,7
4648 016054 004537 003156          JSR      R5,.,MSTCLR          ;CLEAR M8200,4,7
4649 016060 012737 000001 002624 MOV      #1,MRO          ;SAVE REGISTER ADDRESS FOR TYPEOUT
4650 016066 012705 000001          MOV      #1,R5          ;START WITH BIT 0
4651 016072          MYINT
4652 016072 013701 002716          MOV      KMCSR,R1          ;GET DEVICE ADDRESS.
4653 016076          BGNSEG
4654 016076 104404          TRAP     C$BSEG
4655 016100          64$:
4656 016100 010561 000004          MOV      R5,4(R1)          ;PUT PATTERN INTO PORT4
4657 016104          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4658 016104 004537 003244          JSR      R5,.,ROMCLK          ;CLOCK INSTRUCTION
4659 016110 122101          122100!1          ;MOV DATA TO IBUS* REGISTER 1
4660 016112          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4661 016112 004537 003244          JSR      R5,.,ROMCLK          ;CLOCK INSTRUCTION
4662 016116 021025          21005!<1*20>          ;READ FROM IBUS* REGISTER 1
4663 016120 116104 000005          MOVB    5(R1),R4          ;PUT 'FOUND' INTO R4
4664 016124 120504          CMPB    R5,R4          ;DATA CORRECT?
4665 016126 001414          BEQ     65$          ;BR IF YES
4666 016130          BERROR  29          ;ERROR
4667 016150 104455          TRAP     C$ERDF
4668 016152 000035          .WORD  29
4669 016154 005605          .WORD  EM29
4670 016156 010004          .WORD  ERR29
4671 016160          65$:
4672 016160 104410          ESCAPE  SEG
4673 016162 000010          TRAP     C$ESCAPE
4674 016164 000241          .WORD  10000$-.
4675 016166 106105          CLC          ;CLEAR CARRY
4676 016170 001343          ROLB    R5          ;SHIFT BIT IN R5
4677 016172          BNE     64$          ;IF R5=0 THEN DONE
4678 016172          10000$:
4679 016172 104405          TRAP     C$ESEG
4680 016174 012705 000001          MOV      #1,R5          ;START WITH BIT 0
4681          ;69$:
4682 016200          COM     R5          ;CHANGE TO FLOATING ZERO
4683 016200 104404          BGNSEG
4684 016202          TRAP     C$BSEG
4685 016202 005105          67$:
4686 016204 010561 000004          COM     R5
4687 016210          MOV      R5,4(R1)          ;PUT PATTERN INTO PORT4
4688 016210 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4689 016214 122101          JSR      R5,.,ROMCLK          ;CLOCK INSTRUCTION
4690 016216          122100!1          ;MOV DATA TO IBUS* REGISTER 1
4691 016216 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4692 016222 021025          JSR      R5,.,ROMCLK          ;CLOCK INSTRUCTION
4693 016224 116104 000005          21005!<1*20>          ;READ FROM IBUS* REGISTER 1
4694 016230 120504          MOVB    5(R1),R4          ;PUT 'FOUND' INTO R4
4695 016232 001414          CMPB    R5,R4          ;DATA CORRECT?
4696 016234          BEQ     68$          ;BR IF YES
4697 016254 104455          BERROR  29          ;ERROR
4698 016256 000035          TRAP     C$ERDF
4699 016260 005605          .WORD  29
4700 016262 010004          .WORD  EM29
          .WORD  ERR29
  
```

```
4701 016264          68$: ESCAPE SEG
4702 016264 104410   TRAP C$ESCAPE
4703 016266 000012   .WORD 10001$-.
4704 016270 005105   COM R5          :CHANGE TO FLOATING 1
4705 016272 000241   CLC           :CLEAR CARRY
4706 016274 106105   ROLB R5       :SHIFT BIT IN R5
4707 016276 001341   BNE 67$       :IF R5=0 THEN DONE
4708 016300          ENDSEG
4709 016300          10001$: TRAP C$ESEG
4710 016300 104405
4711 016302          ENDTST
4712 016302          L'0076: TRAP C$ETST
4713 016302 104401
4714
4715 016304          BADHEAD
4716          :***** TEST 22 *****
4717          :*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4718          :*FLOAT A 1 THROUGH IBUS REGISTER 2
4719          :*FLOAT A 0 THROUGH IBUS REGISTER 2
4720 016304          BADHEAD
4721          :***** TEST 22 *****
4722
4723 016304          BGNST
4724 016304          T22::
4725 016304          MSTCLR          :MASTER CLEAR M8200,4,7
4726 016304 004537 003156 JSR R5, MSTCLR          :CLEAR M8200,4,7
4727 016310 012737 000002 002624 MOV #2, MRO          :SAVE REGISTER ADDRESS FOR TYPEOUT
4728 016316 012705 000001 MOV #1, R5          :START WITH BIT 0
4729 016322          MYINT
4730 016322 013701 002716 MOV KMCSR, R1       :GET DEVICE ADDRESS.
4731 016326          BGNSEG
4732 016326 104404   TRAP C$BSEG
4733 016330          64$:
4734 016330 010561 000004 MOV R5, 4(R1)       :PUT PATTERN INTO PORT4
4735 016334          ROMCLK          :NEXT WORD IS INSTRUCTION, BBN
4736 016334 004537 003244 JSR R5, ROMCLK       :CLOCK INSTRUCTION
4737 016340 122102          122100!2 :MOV DATA TO IBUS* REGISTER 2
4738 016342          ROMCLK          :NEXT WORD IS INSTRUCTION, BBN
4739 016342 004537 003244 JSR R5, ROMCLK       :CLOCK INSTRUCTION
4740 016346 021045          21005!<2*20> :READ FROM IBUS* REGISTER 2
4741 016350 116104 000005 MOVB 5(R1), R4       :PUT 'FOUND' INTO R4
4742 016354 120504          CMPB R5, R4          :DATA CORRECT?
4743 016356 001414          BEQ 65$          :BR IF YES
4744 016360          BERROR 29          :ERROR
4745 016400 104455   TRAP C$ERDF
4746 016402 000035   .WORD 29
4747 016404 005605   .WORD EM29
4748 016406 010004   .WORD ERR29
4749 016410          65$: ESCAPE SEG
4750 016410 104410   TRAP C$ESCAPE
4751 016412 000010   .WORD 10000$-.
4752 016414 000241   CLC           :CLEAR CARRY
4753 016416 106105   ROLB R5       :SHIFT BIT IN R5
4754 016420 001343   BNE 64$       :IF R5=0 THEN DONE
4755 016422          ENDSEG
4756 016422          10000$:
```

```

4757 016422 104405          TRAP      C$ESEG
4758 016424 012705 000001  MOV      #1,R5          ;START WITH BIT 0
4759          ;69$:      COM      R5          ;CHANGE TO FLOATING ZERO
4760 016430          BGNSEG
4761 016430 104404          TRAP      C$BSEG
4762 016432          67$:
4763 016432 005105          COM      R5
4764 016434 010561 000004  MOV      R5,4(R1)      ;PUT PATTERN INTO PORT4
4765 016440          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4766 016440 004537 003244  JSR      R5,.ROMCLK    ;CLOCK INSTRUCTION
4767 016444 122102          122100!2  ;MOV DATA TO IBUS* REGISTER 2
4768 016446          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
4769 016446 004537 003244  JSR      R5,.ROMCLK    ;CLOCK INSTRUCTION
4770 016452 021045          21005!<2*20> ;READ FROM IBUS* REGISTER 2
4771 016454 116104 000005  MOVB     5(R1),R4      ;PUT 'FOUND' INTO R4
4772 016460 120504          CMPB     R5,R4        ;DATA CORRECT?
4773 016462 001414          BEQ      68$          ;BR IF YES
4774 016464          BERROR    29          ;ERROR
4775 016504 104455          TRAP      C$ERDF
4776 016506 000035          .WORD    29
4777 016510 005605          .WORD    EM29
4778 016512 010004          .WORD    ERR29
4779 016514          68$:      ESCAPE    SEG
4780 016514 104410          TRAP      C$ESCAPE
4781 016516 000012          .WORD    10001$-
4782 016520 005105          COM      R5          ;CHANGE TO FLOATING 1
4783 016522 000241          CLC
4784 016524 106105          ROLB     R5          ;CLEAR CARRY
4785 016526 001341          BNE     67$          ;SHIFT BIT IN R5
4786 016530          ENDSEG          ;IF R5=0 THEN DONE
4787 016530          10001$:
4788 016530 104405          TRAP      C$ESEG
4789 016532          ENDTST
4790 016532          L10077:
4791 016532 104401          TRAP      C$ETST
4792
4793 016534          BADHEAD
4794          ;***** TEST 23 *****
4795          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4796          ;*FLOAT A 1 THROUGH IBUS REGISTER 3
4797          ;*FLOAT A 0 THROUGH IBUS REGISTER 3
4798 016534          BADHEAD
4799          ;***** TEST 23 *****
4800
4801 016534          BGNTST
4802 016534          T23::
4803 016534          MSTCLR          ;MASTER CLEAR M8200,4,7
4804 016534 004537 003156  JSR      R5,.MSTCLR    ;CLEAR M8200,4,7
4805 016540 012737 000003 002624  MOV      #3,MRO        ;SAVE REGISTER ADDRESS FOR TYPEOUT
4806 016546 012705 000001  MOV      #1,R5        ;START WITH BIT 0
4807 016552          MYINT
4808 016552 013701 002716  MOV      KMCSR,R1     ;GET DEVICE ADDRESS.
4809 016556          BGNSEG
4810 016556 104404          TRAP      C$BSEG
4811 016560          64$:
4812 016560 010561 000004  MOV      R5,4(R1)      ;PUT PATTERN INTO PORT4
  
```

4813	016564			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
4814	016564	004537	003244	JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
4815	016570	122103		122100!3		;MOV DATA TO IBUS* REGISTER 3
4816	016572			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
4817	016572	004537	003244	JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
4818	016576	021065		21005!<3*20>		;READ FROM IBUS* REGISTER 3
4819	016600	116104	000005	MOVB	5(R1),R4	;PUT 'FOUND' INTO R4
4820	016604	120504		CMPB	R5,R4	;DATA CORRECT?
4821	016606	001414		BEQ	65\$;BR IF YES
4822	016610			BERROR	29	;ERROR
4823	016630	104455		TRAP	C\$ERDF	
4824	016632	000035		.WORD	29	
4825	016634	005605		.WORD	EM29	
4826	016636	010004		.WORD	ERR29	
4827	016640			65\$: ESCAPE	SEG	
4828	016640	104410		TRAP	C\$ESCAPE	
4829	016642	000010		.WORD	10000\$-	
4830	016644	000241		CLC		;CLEAR CARRY
4831	016646	106105		ROLB	R5	;SHIFT BIT IN R5
4832	016650	001343		BNE	64\$;IF R5=0 THEN DONE
4833	016652			ENDSEG		
4834	016652			10000\$:		
4835	016652	104405		TRAP	C\$ESEG	
4836	016654	012705	000001	MOV	#1,R5	;START WITH BIT 0
4837				:69\$: COM	R5	;CHANGE TO FLOATING ZERO
4838	016660			BGNSEG		
4839	016660	104404		TRAP	C\$BSEG	
4840	016662			67\$:		
4841	016662	005105		COM	R5	
4842	016664	010561	000004	MOV	R5,4(R1)	;PUT PATTERN INTO PORT4
4843	016670			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
4844	016670	004537	003244	JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
4845	016674	122103		122100!3		;MOV DATA TO IBUS* REGISTER 3
4846	016676			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
4847	016676	004537	003244	JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
4848	016702	021065		21005!<3*20>		;READ FROM IBUS* REGISTER 3
4849	016704	116104	000005	MOVB	5(R1),R4	;PUT 'FOUND' INTO R4
4850	016710	120504		CMPB	R5,R4	;DATA CORRECT?
4851	016712	001414		BEQ	68\$;BR IF YES
4852	016714			BERROR	29	;ERROR
4853	016734	104455		TRAP	C\$ERDF	
4854	016736	000035		.WORD	29	
4855	016740	005605		.WORD	EM29	
4856	016742	010004		.WORD	ERR29	
4857	016744			68\$: ESCAPE	SEG	
4858	016744	104410		TRAP	C\$ESCAPE	
4859	016746	000012		.WORD	10001\$-	
4860	016750	005105		COM	R5	;CHANGE TO FLOATING 1
4861	016752	000241		CLC		;CLEAR CARRY
4862	016754	106105		ROLB	R5	;SHIFT BIT IN R5
4863	016756	001341		BNE	67\$;IF R5=0 THEN DONE
4864	016760			ENDSEG		
4865	016760			10001\$:		
4866	016760	104405		TRAP	C\$ESEG	
4867	016762			ENDTST		
4868	016762			L10100:		

```
4869 016762 104401 TRAP C$ETST
4870
4871 016764 BADHEAD
4872 :***** TEST 24 *****
4873 :*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4874 :*FLOAT A 1 THROUGH IBUS REGISTER 4
4875 :*FLOAT A 0 THROUGH IBUS REGISTER 4
4876 016764 BADHEAD
4877 :***** TEST 24 *****
4878
4879 016764 BGNTST
4880 016764 T?4::
4881 016764 MSTCLR ;MASTER CLEAR M8200,4,7
4882 016764 004537 003156 JSR R5, .MSTCLR ;CLEAR M8200,4,7
4883 016770 012737 000004 002624 MOV #4, MRO ;SAVE REGISTER ADDRESS FOR TYPEOUT
4884 016776 012705 000001 MOV #1, R5 ;START WITH BIT 0
4885 017002 MYINT
4886 017002 013701 002716 MOV KMCSR, R1 ;GET DEVICE ADDRESS.
4887 017006 BGNSEG
4888 017006 104404 TRAP C$BSEG
4889 017010
4890 017010 010561 000004 64$: MOV R5, 4(R1) ;PUT PATTERN INTO PORT4
4891 017014 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4892 017014 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
4893 017020 122104 122100!4 ;MOV DATA TO IBUS* REGISTER 4
4894 017022 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4895 017022 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
4896 017026 021105 21005!<4*20> ;READ FROM IBUS* REGISTER 4
4897 017030 116104 000005 MOVB 5(R1), R4 ;PUT "FOUND" INTO R4
4898 017034 120504 CMPB R5, R4 ;DATA CORRECT?
4899 017036 001414 BEQ 65$ ;BR IF YES
4900 017040 BERROR 29 ;ERROR
4901 017060 104455 TRAP C$ERDF
4902 017062 000035 .WORD 29
4903 017064 005605 .WORD EM29
4904 017066 010004 .WORD ERR29
4905 017070 65$: ESCAPE SEG
4906 017070 104410 TRAP C$ESCAPE
4907 017072 000010 .WORD 10000$-.
4908 017074 000241 CLC ;CLEAR CARRY
4909 017076 106105 ROLB R5 ;SHIFT BIT IN R5
4910 017100 001343 BNE 64$ ;IF R5=0 THEN DONE
4911 017102 ENDSEG
4912 017102 10000$:
4913 017102 104405 TRAP C$ESEG
4914 017104 012705 000001 MOV #1, R5 ;START WITH BIT 0
4915 017110 ;69$: COM R5 ;CHANGE TO FLOATING ZERO
4916 017110 BGNSEG
4917 017110 104404 TRAP C$BSEG
4918 017112
4919 017112 005105 67$: COM R5
4920 017114 010561 000004 MOV R5, 4(R1) ;PUT PATTERN INTO PORT4
4921 017120 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4922 017120 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
4923 017124 122104 122100!4 ;MOV DATA TO IBUS* REGISTER 4
4924 017126 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
```

```

4925 017126 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
4926 017132 021105 21005!<4*20> ;READ FROM IBUS* REGISTER 4
4927 017134 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
4928 017140 120504 CMPB R5,R4 ;DATA CORRECT?
4929 017142 001414 BEQ 68$ ;BR IF YES
4930 017144 BERROR 29 ;ERROR
4931 017164 104455 TRAP C$ERDF
4932 017166 000035 .WORD 29
4933 017170 005605 .WORD EM29
4934 017172 010004 .WORD ERR29
4935 017174 68$: ESCAPE SEG
4936 017174 104410 TRAP C$ESCAPE
4937 017176 000012 .WORD 10001$-.
4938 017200 005105 COM R5 ;CHANGE TO FLOATING 1
4939 017202 000241 CLC ;CLEAR CARRY
4940 017204 106105 ROLB R5 ;SHIFT BIT IN R5
4941 017206 001341 BNE 67$ ;IF R5=0 THEN DONE
4942 017210 ENDSEG
4943 017210 10001$: TRAP C$ESEG
4944 017210 104405 ENDTST
4945 017212 L10101: TRAP C$ETST
4946 017212 104401
4947 017212 104401
4948 017214
4949 017214 BADHEAD
4950 017214 ;***** TEST 25 *****
4951 017214 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
4952 017214 ;*FLOAT A 1 THROUGH IBUS REGISTER 5
4953 017214 ;*FLOAT A 0 THROUGH IBUS REGISTER 5
4954 017214 BADHEAD
4955 017214 ;***** TEST 25 *****
4956 017214
4957 017214 BGNTST
4958 017214 T25::
4959 017214 MSTCLR ;MASTER CLEAR M8200,4,7
4960 017214 004537 003156 JSR R5,.MSTCLR ;CLEAR M8200,4,7
4961 017220 012737 000005 002624 MOV #5,MRO ;SAVE REGISTER ADDRESS FOR TYPEOUT
4962 017226 012705 000001 MOV #1,R5 ;START WITH BIT 0
4963 017232 MYINT
4964 017232 013701 002716 MOV KMCSR,R1 ;GET DEVICE ADDRESS.
4965 017236 BGNSEG
4966 017236 104404 TRAP C$BSEG
4967 017240 64$:
4968 017240 010561 000004 MOV R5,4(R1) ;PUT PATTERN INTO PORT4
4969 017244 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4970 017244 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
4971 017250 122105 122100!5 ;MOV DATA TO IBUS* REGISTER 5
4972 017252 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
4973 017252 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
4974 017256 021125 21005!<5*20> ;READ FROM IBUS* REGISTER 5
4975 017260 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
4976 017264 120504 CMPB R5,R4 ;DATA CORRECT?
4977 017266 001414 BEQ 65$ ;BR IF YES
4978 017270 BERROR 29 ;ERROR
4979 017310 104455 TRAP C$ERDF
4980 017312 000035 .WORD 29
  
```



```
4981 017314 005605 .WORD EM29
4982 017316 010004 .WORD ERR29
4983 017320 65$: ESCAPE SEG
4984 017320 104410 TRAP C$ESCAPE
4985 017322 000010 .WORD 10000$-.
4986 017324 000241 CLC ;CLEAR CARRY
4987 017326 106105 ROLB R5 ;SHIFT BIT IN R5
4988 017330 001343 BNE 64$ ;IF R5=0 THEN DONE
4989 017332 ENDSEG
4990 017332 10000$:
4991 017332 104405 TRAP C$ESEG
4992 017334 012705 000001 MOV #1,R5 ;START WITH BIT 0
4993 017334 69$: COM R5 ;CHANGE TO FLOATING ZERO
4994 017340 BGNSEG
4995 017340 104404 TRAP C$BSEG
4996 017342 67$:
4997 017342 005105 COM R5
4998 017344 010561 000004 MOV R5,4(R1) ;PUT PATTERN INTO PORT4
4999 017350 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5000 017350 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
5001 017354 122105 122100!5 ;MOV DATA TO IBUS* REGISTER 5
5002 017356 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5003 017356 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
5004 017362 021125 21005!<5*20> ;READ FROM IBUS* REGISTER 5
5005 017364 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
5006 017370 120504 CMPB R5,R4 ;DATA CORRECT?
5007 017372 001414 BEQ 68$ ;BR IF YES
5008 017374 BERROR 29 ;ERROR
5009 017414 104455 TRAP C$ERDF
5010 017416 000035 .WORD 29
5011 017420 005605 .WORD EM29
5012 017422 010004 .WORD ERR29
5013 017424 68$: ESCAPE SEG
5014 017424 104410 TRAP C$ESCAPE
5015 017426 000012 .WORD 10001$-.
5016 017430 005105 COM R5 ;CHANGE TO FLOATING 1
5017 017432 000241 CLC ;CLEAR CARRY
5018 017434 106105 ROLB R5 ;SHIFT BIT IN R5
5019 017436 001341 BNE 67$ ;IF R5=0 THEN DONE
5020 017440 ENDSEG
5021 017440 10001$:
5022 017440 104405 TRAP C$ESEG
5023 017442 ENDTST
5024 017442 L10102:
5025 017442 104401 TRAP C$ETST
5026
5027 017444 BADHEAD
5028 ;***** TEST 26 *****
5029 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
5030 ;*FLOAT A 1 THROUGH IBUS REGISTER 6
5031 ;*FLOAT A 0 THROUGH IBUS REGISTER 6
5032 017444 BADHEAD
5033 ;***** TEST 26 *****
5034
5035 017444 BGNST
5036 017444 T26::
```

5037	017444			MSTCLR		;MASTER CLEAR MB200,4,7
5038	017444	004537	003156	JSR	R5,,MSTCLR	;CLEAR MB200,4,7
5039	017450	012737	000006	MOV	#6,MRO	;SAVE REGISTER ADDRESS FOR TYPEOUT
5040	017456	012705	000001	MOV	#1,R5	;START WITH BIT 0
5041	017462			MYINT		
5042	017462	013701	002716	MOV	KMCSR,R1	;GET DEVICE ADDRESS.
5043	017466			BGNSEG		
5044	017466	104404		TRAP	C\$BSEG	
5045	017470					
5046	017470	010561	000004	MOV	R5,4(R1)	;PUT PATTERN INTO PORT4
5047	017474			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
5048	017474	004537	003244	JSR	R5,,ROMCLK	;CLOCK INSTRUCTION
5049	017500	122106		122100!6		;MOV DATA TO IBUS* REGISTER 6
5050	017502			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
5051	017502	004537	003244	JSR	R5,,ROMCLK	;CLOCK INSTRUCTION
5052	017506	021145		21005!<6*20>		;READ FROM IBUS* REGISTER 6
5053	017510	116104	000005	MOVB	5(R1),R4	;PUT 'FOUND' INTO R4
5054	017514	120504		CMPB	R5,R4	;DATA CORRECT?
5055	017516	001414		BEQ	65\$;BR IF YES
5056	017520			BERROR	29	;ERROR
5057	017540	104455		TRAP	C\$ERDF	
5058	017542	000035		.WORD	29	
5059	017544	005605		.WORD	EM29	
5060	017546	010004		.WORD	ERR29	
5061	017550			ESCAPE	SEG	
5062	017550	104410		TRAP	C\$ESCAPE	
5063	017552	000010		.WORD	10000\$-	
5064	017554	000241		CLC		;CLEAR CARRY
5065	017556	106105		ROLB	R5	;SHIFT BIT IN R5
5066	017560	001343		BNE	64\$;IF R5=0 THEN DONE
5067	017562			ENDSEG		
5068	017562					
5069	017562	104405		TRAP	C\$ESEG	
5070	017564	012705	000001	MOV	#1,R5	;START WITH BIT 0
5071				COM	R5	;CHANGE TO FLOATING ZERO
5072	017570			BGNSEG		
5073	017570	104404		TRAP	C\$BSEG	
5074	017572					
5075	017572	005105		COM	R5	
5076	017574	010561	000004	MOV	R5,4(R1)	;PUT PATTERN INTO PORT4
5077	017600			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
5078	017600	004537	003244	JSR	R5,,ROMCLK	;CLOCK INSTRUCTION
5079	017604	122106		122100!6		;MOV DATA TO IBUS* REGISTER 6
5080	017606			ROMCLK		;NEXT WORD IS INSTRUCTION, BBN
5081	017606	004537	003244	JSR	R5,,ROMCLK	;CLOCK INSTRUCTION
5082	017612	021145		21005!<6*20>		;READ FROM IBUS* REGISTER 6
5083	017614	116104	000005	MOVB	5(R1),R4	;PUT 'FOUND' INTO R4
5084	017620	120504		CMPB	R5,R4	;DATA CORRECT?
5085	017622	001414		BEQ	68\$;BR IF YES
5086	017624			BERROR	29	;ERROR
5087	017644	104455		TRAP	C\$ERDF	
5088	017646	000035		.WORD	29	
5089	017650	005605		.WORD	EM29	
5090	017652	010004		.WORD	ERR29	
5091	017654			ESCAPE	SEG	
5092	017654	104410		TRAP	C\$ESCAPE	

```
5093 017656 000012 .WORD 10001$-.
5094 017660 005105 COM R5 ;CHANGE TO FLOATING 1
5095 017662 000241 CLC ;CLEAR CARRY
5096 017664 106105 ROLB R5 ;SHIFT BIT IN R5
5097 017666 001341 BNE 67$ ;IF R5=0 THEN DONE
5098 017670 ENDSEG
5099 017670 10001$: TRAP C$ESEG
5100 017670 104405
5101 017672
5102 017672 ENDTST
5103 017672 104401 L'0103: TRAP C$ETST
5104
5105 017674 BADHEAD
5106 ;***** TEST 27 *****
5107 ;*MICRO PROCEOR IBUS* REGISTER WRITE/READ TEST
5108 ;*FLOAT A 1 THOUGH IBUS* REGISTER 7
5109 ;*FLOAT A 0 THROUGH IBUS* REGISTER 7
5110 017674 BADHEAD
5111 ;***** TEST 27 *****
5112
5113 017674 BGNTST
5114 017674 T27::
5115 017674 MSTCLR ;MASTER CLEAR M8200.4,7
5116 017674 004537 003156 JSR R5, .MSTCLR ;CLEAR M8200.4,7
5117 017700 012737 000007 002624 MOV #7, MRO ;SAVE REGISTER ADDRESS FOR TYPEOUT
5118 017706 012705 000001 MOV #1, R5 ;START WITH BIT 0
5119 017712 MYINT
5120 017712 013701 002716 MOV KMCSR, R1 ;GET DEVICE ADDRESS.
5121 017716 BGNSEG
5122 017716 104404 TRAP C$BSEG
5123 017720 64$:
5124 017720 010561 000004 MOV R5, 4(R1) ;PUT PATTERN INTO PORT4
5125 017724 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5126 017724 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
5127 017730 122107 122100!7 ;MOV DATA TO IBUS* REGISTER 7
5128 017732 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5129 017732 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
5130 017736 021165 21005!<7*20> ;READ FROM IBUS* REGISTER 7
5131 017740 116104 000005 MOVB 5(R1), R4 ;PUT 'FOUND' INTO R4
5132 017744 120504 CMPB R5, R4 ;DATA CORRECT?
5133 017746 001414 BEQ 65$ ;BR IF YES
5134 017750 BERROR 29 ;ERROR
5135 017770 104455 TRAP C$ERDF
5136 017772 000035 .WORD 29
5137 017774 005605 .WORD EM29
5138 017776 010004 .WORD ERR29
5139 020000 65$: ESCAPE SEG
5140 020000 104410 TRAP C$ESCAPE
5141 020002 000010 .WORD 10000$-.
5142 020004 000241 CLC ;CLEAR CARRY
5143 020006 106105 ROLB R5 ;SHIFT BIT IN R5
5144 020010 001343 BNE 64$ ;IF R5=0 THEN DONE
5145 020012 ENDSEG
5146 020012 10000$:
5147 020012 104405 TRAP C$ESEG
5148 020014 012705 000001 MOV #1, R5 ;START WITH BIT 0
```

```

5149          ;69$: COM      R5          ;CHANGE TO FLOATING ZERO
5150 020020   BGNSEG
5151 020020 104404 TRAP      C$BSEG
5152 020022   ;
5153 020022 005105   ;
5154 020024 010561 000004   ;
5155 020030   ;
5156 020030 004537 003244   ;
5157 020034 122107   ;
5158 020036   ;
5159 020036 004537 003244   ;
5160 020042 021165   ;
5161 020044 116104 000005   ;
5162 020050 120504   ;
5163 020052 001414   ;
5164 020054   ;
5165 020074 104455   ;
5166 020076 000035   ;
5167 020100 005605   ;
5168 020102 010004   ;
5169 020104   ;
5170 020104 104410   ;
5171 020106 000012   ;
5172 020110 005105   ;
5173 020112 000241   ;
5174 020114 106105   ;
5175 020116 001341   ;
5176 020120   ;
5177 020120   ;
5178 020120 104405   ;
5179 020122   ;
5180 020122   ;
5181 020122 104401   ;
5182          ;
5183 020124   ;
5184          ;
5185          ;
5186          ;
5187          ;
5188 020124   ;
5189          ;
5190          ;
5191 020124   ;
5192 020124   ;
5193 020124   ;
5194 020124 004537 003156   ;
5195 020130 012705 000001   ;
5196 020134 005002   ;
5197 020136   ;
5198 020136 013701 002716   ;
5199 020142   ;
5200 020142 104404   ;
5201 020144 010203   ;
5202 020146 010561 000004   ;
5203 020152 042737 000017 020170   ;
5204 020160 050337 020170   ;

;69$: COM      R5          ;CHANGE TO FLOATING ZERO
      BGNSEG
      TRAP      C$BSEG

;67$: COM      R5
      MOV      R5,4(R1)      ;PUT PATTERN INTO PORT4
      ROMCLK      ;NEXT WORD IS INSTRUCTION, BBN
      JSR      R5, .ROMCLK   ;CLOCK INSTRUCTION
      122100.7      ;MOV DATA TO IBUS* REGISTER 7
      ROMCLK      ;NEXT WORD IS INSTRUCTION, BBN
      JSR      R5, .ROMCLK   ;CLOCK INSTRUCTION
      21005!<7*20>      ;READ FROM IBUS* REGISTER 7
      MOV      5(R1),R4      ;PUT 'FOUND' INTO R4
      CMP      R5,R4         ;DATA CORRECT?
      BEQ      68$          ;BR IF YES
      BERROR 29             ;ERROR
      TRAP      C$ERDF
      .WORD    29
      .WORD    EM29
      .WORD    ERR29
;68$: ESCAPE  SEG
      TRAP      C$ESCAPE
      .WORD    10001$-.
      COM      R5          ;CHANGE TO FLOATING 1
      CLC      ;CLEAR CARRY
      ROLB    R5          ;SHIFT BIT IN R5
      BNE     67$        ;IF R5=0 THEN DONE
      ENDSEG

10001$: TRAP      C$ESEG

ENDTST
L10104: TRAP      C$ETST

BADHEAD
;***** TEST 28 *****
;*MICRO PROCESSOR IBUS DUAL ADDRESS TEST
;*WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
;*READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING
BADHEAD
;***** TEST 28 *****

BGNTST
T28:: MSTCLR      ;MASTER CLEAR M8200,4,7
      JSR      R5, .MSTCLR   ;CLEAR M8200,4,7
      MOV      #1,R5        ;START WITH A ONE
      CLR      R2          ;R2 CONTAINS ADDRESS OF REGISTER
      MYINT
      MOV      KMCSR,R1     ;GET DEVICE ADDRESS.
      BGNSEG
      TRAP      C$BSEG
;1$: MOV      R2,R3        ;R3=REGISTER ADDRESS
      MOV      R5,4(R1)     ;WRITE DATA TO PORT4
      BIC     #17,5$       ;CLEAR ADDRESS FIELD OF INSTRUCTION
      BIS     R3,5$        ;ADD ADDRESS TO INSTRUCTION
  
```

5205	020164				ROMCLK				:NEXT WORD IS INSTRUCTION, BBN
5206	020164	004537	003244		JSR	R5, .ROMCLK			:CLOCK INSTRUCTION
5207	020170	122100		5\$:	122100				:MOVE DATA TO IBUS REGISTER
5208	020172	006303			ASL	R3			:SHIFT ADDRESS
5209	020174	006303			ASL	R3			:4 TIMES TO GET
5210	020176	006303			ASL	R3			:IT TO BITS 4-7
5211	020200	006303			ASL	R3			:OF NEXT INSTRUCTION
5212	020202	042737	000360	020220	BIC	#360,6\$:CLEAR ADDRESS FIELD
5213	020210	050337	020220		BIS	R3,6\$:ADD ADDRESS TO INSTRUCTION
5214	020214				ROMCLK				:NEXT WORD IS INSTRUCTION, BBN
5215	020214	004537	003244		JSR	R5, .ROMCLK			:CLOCK INSTRUCTION
5216	020220	021005		6\$:	21005				:READ FROM IBUS REGISTER
5217	020222	116104	000005		MOVB	5(R1),R4			:PUT 'FOUND' IN R4
5218	020226	120504			CMPB	R5,R4			:IS DATA CORRECT?
5219	020230	001414			BEQ	2\$:BR IF YES
5220	020232				BERROR	29			:DATA ERROR
5221	020252	104455			TRAP	C\$ERDF			
5222	020254	000035			.WORD	29			
5223	020256	005605			.WORD	EM29			
5224	020260	010004			.WORD	ERR29			
5225	020262			2\$:	ESCAPE	SEG			
5226	020262	104410			TRAP	C\$ESCAPE			
5227	020264	000014			.WORD	10000\$-			
5228	020266	005205			INC	R5			:INCREMENT PATTERN
5229	020270	005202			INC	R2			:INCREMENT REGISTER ADDRESS
5230	020272	022702	000010		CMP	#7+1,R2			:LAST ADDRESS DONE?
5231	020276	001322			BNE	1\$:BR IF NO
5232	020300				ENDSEG				
5233	020300			10000\$:					
5234	020300	104405			TRAP	C\$ESEG			
5235	020302	012705	000001		MOV	#1,R5			:RESTART PATTERN TO 1
5236	020306	005002			CLR	R2			:RESTART AT ADDRESS 0
5237	020310				BGNSEG				
5238	020310	104404			TRAP	C\$BSEG			
5239	020312	005003			CLR	R3			:RESTART AT ADDRESS 0
5240	020314	042737	000360	020332	BIC	#360,7\$:CLEAR ADDRESS FIELD OF INSTRUCTION
5241	020322	050337	020332	3\$:	BIS	R3,7\$:ADD ADDRESS TO INSTRUCTION
5242	020326				ROMCLK				:NEXT WORD IS INSTRUCTION, BBN
5243	020326	004537	003244		JSR	R5, .ROMCLK			:CLOCK INSTRUCTION
5244	020332	021005		7\$:	21005				:READ FROM IBUS REGISTER
5245	020334	116104	000005		MOVB	5(R1),R4			:PUT 'FOUND' IN \$GDDAT
5246	020340	120504			CMPB	R5,R4			:DATA CORRECT?
5247	020342	001414			BEQ	4\$:BR IF YES
5248	020344				BERROR	30			:DUAL ADDRESSING ERROR
5249	020364	104455			TRAP	C\$ERDF			
5250	020366	000036			.WORD	30			
5251	020370	004353			.WORD	EM30			
5252	020372	010066			.WORD	ERR30			
5253	020374			4\$:	ESCAPE	SEG			
5254	020374	104410			TRAP	C\$ESCAPE			
5255	020376	000020			.WORD	10001\$-			
5256	020400	005205			INC	R5			:INCREMENT PATTERN
5257	020402	005202			INC	R2			:NEXT ADDRESS
5258	020404	062703	000020		ADD	#20,R3			:ADD 1 TO ADDRESS IN R3(SHIFTED 4 TIMES)
5259	020410	022702	000010		CMP	#7+1,R2			:LAST ADDRESS DONE?
5260	020414	001337			BNE	3\$:BR IF NO

5261 020416
5262 020416
5263 020416 104405
5264 020420
5265 020420
5266 020420 104401
5267
5268 020422
5269
5270
5271
5272
5273 020422
5274
5275
5276 020422
5277 020422
5278
5279 020422
5280 020422 004537 003156
5281 020426 012702 000001
5282 020432
5283 020432 013701 002716
5284 020436
5285 020436 104404
5286 020440
5287 020440 010261 000004
5288 020444
5289 020444 004537 003244
5290 020450 120500
5291 020452
5292 020452 004537 003244
5293 020456 061225
5294 020460 116104 000005
5295 020464 120204
5296 020466 001414
5297 020470
5298 020510 104455
5299 020512 000003
5300 020514 004416
5301 020516 006210
5302 020520
5303 020520 104410
5304 020522 000010
5305 020524 000241
5306 020526 106102
5307 020530 001343
5308 020532
5309 020532
5310 020532 104405
5311 020534 012702 000001
5312 020540
5313 020540
5314 020540 104404
5315 020542
5316 020542 005102

10001\$: ENDSEG
TRAP C\$ESEG
ENDTST
L10105: TRAP C\$ETST

BADHEAD
:***** TEST 29 *****
:*MICRO PROCESSOR BR REGISTER TEST
:*FLOAT A 1 THOUGH THE BR
:*FLOAT A 0 THOUGH THE BR
BADHEAD
:***** TEST 29 *****

BGNTST
T29: :

MSTCLR ;R1 CONTAINS BASE M8200,4,7 ADDRESS
JSR R5, MSTCLR ;MASTER CLEAR COMM. MICRO-PROCESSOR FAMILY
MOV #1, R2 ;CLEAR M8200,4,7
MYINT ;START PATTERN WITH BIT0
MOV KMCSR, R1 ;GET DEVICE ADDRESS.

64\$:

BGNSEG
TRAP C\$BSEG
MOV R2, 4(R1) ;WRITE PATTERN IN PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5, ROMCLK ;CLOCK INSTRUCTION
120500 ;MOVE DATA TO THE BR REGISTER
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5, ROMCLK ;CLOCK INSTRUCTION
061225 ;MOVE BR TO PORT 5
MOVB 5(R1), R4 ;PUT "FOUND" IN R4
CMPB R2, R4 ;IS DATA CORRECT?
BEQ 65\$;BR IF YES
BERROR 3 ;DATA ERROR
TRAP C\$ERDF

65\$:

.WORD 3
.WORD EM3
.WORD ERR3
ESCAPE SEG
TRAP C\$ESCAPE
. WORD 10000\$-.
CLC ;CLEAR CARRY
ROLB R2 ;SHIFT BIT IN R2
BNE 64\$;DONE IF R2=0
ENDSEG

10000\$:

TRAP C\$ESEG
MOV #1, R2 ;START PATTERN WITH BIT0

69\$:

BGNSEG
TRAP C\$BSEG

67\$:

COM R2

CZDMPBO MB207 STATIC DIAG #1
CZDMPB.P11 20-AUG-80 07:48

MACY11 30A(1052) 20-AUG-80 07:50
HARDWARE TESTS

SEQ 0136

```

5317 020544 010261 000004      MOV      R2,4(R1)      ;WRITE PATTERN IN PORT4
5318 020550                      ROMCLK                ;NEXT WORD IS INSTRUCTION, BBN
5319 020550 004537 003244      JSR      R5,.ROMCLK   ;CLOCK INSTRUCTION
5320 020554 120500                      120500                ;MOVE DATA TO THE BR REGISTER
5321 020556                      ROMCLK                ;NEXT WORD IS INSTRUCTION, BBN
5322 020556 004537 003244      JSR      R5,.ROMC K   ;CLOCK INSTRUCTION
5323 020562 061225                      061225                ;MOVE BR TO PORT 5
5324 020564 116104 000005      MOVB     5(R1),R4      ;PUT 'FOUND' IN $GDDAT
5325 020570 010205                      MOV      R2,R5
5326 020572 120204                      CMPB     R2,R4         ;DATA CORRECT?
5327 020574 001414                      BEQ      68$           ;BR IF YES
5328 020576                      BERROR   3            ;DATA ERROR
5329 020616 104455                      TRAP     C$ERDF
5330 020620 000003                      .WORD   3
5331 020622 004416                      .WORD   EM3
5332 020624 006210                      .WORD   ERR3
5333 020626                      68$: ESCAPE   SEG
5334 020626 104410                      TRAP     C$ESCAPE
5335 020630 000016                      .WORD   10001$-.
5336                      ;FAILED TO CLEAR
5337 020632 105061 000001      70$: CLRB     1(R1)    ;BRG
5338 020636 005102                      COM      R2            ;CHANGE BACK TO A ONE
5339 020640 000241                      CLC                      ;CLEAR CARRY
5340 020642 106102                      ROLB     R2            ;SHIFT BIT IN R5
5341 020644 001336                      BNE     67$           ;DONE IF R5=0
5342 020646                      ENDSEG
5343 020646                      10001$:
5344 020646 104405                      TRAP     C$ESEG
5345 020650                      ENDTST
5346 020650                      L10106:
5347 020650 104401                      TRAP     C$ETST
5348
5349 020652                      BADHEAD
5350                      ;***** TEST 30 *****
5351                      ;*SCRATCH PAD TEST
5352                      ;*FLOAT A 1 THOUGH EACH SCRATCH PAD LOCATION
5353                      ;*FLOAT A 0 THOUGH EACH SCRATCH PAD LOCATION
5354 020652                      BADHEAD
5355                      ;***** TEST 30 *****
5356
5357 020652                      BGNTST
5358 020652                      T30::
5359 020652                      MYINT
5360 020652 013701 002716      MOV      KMCSR,R1     ;GET DEVICE ADDRESS.
5361 020656                      MSTCLR                ;MASTER CLEAR MB200,4,7
5362 020656 004537 003156      JSR      R5,.MSTCLR   ;CLEAR MB200,4,7
5363 020662 005002                      CLR      R2            ;START AT ADDRESS ZERO
5364 020664 012705 000001      MOV      #1,R5        ;START WITH BIT0
5365 020670                      BGNSUB
5366 020670                      T30.1:
5367 020670 104402                      TRAP     C$BSUB
5368 020672                      1$: BGNSEG
5369 020672 104404                      TRAP     C$BSEG
5370 020674 042737 000017 020716 64$: BIC      #17,65$     ;CLEAR ADDRESS FIELD OF INSTRUCTION
5371 020702 050237 020716                      BIS      R2,65$       ;ADD ADDRESS TO INSTRUCTION
5372 020706 010561 000004                      MOV      R5,4(R1)     ;WRITE PATTERN IN PORT4

```

5373	020712				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5374	020712	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5375	020716	123100		65\$:	123100		;WRITE SCRATCH PAD(ADDRESS IN R2)	
5376	020720	042737	000017	020736	BIC	#17,66\$;CLEAR ADDRESS FIELD OF INSTRUCTION	
5377	020726	050237	020736		BIS	R2,66\$;ADD ADDRESS TO INSTRUCTION	
5378	020732				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5379	020732	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5380	020736	040600		66\$:	040600		;MOVE SP TO BR	
5381	020740				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5382	020740	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5383	020744	061225			061225		;MOVE BR TO PORT5	
5384	020746	010537	002636		MOV	R5,\$GDDAT	;PUT 'EXPECTED' IN \$GDDAT	
5385	020752	116104	000005		MOVB	5(R1),R4	;PUT 'FOUND' IN R4	
5386	020756	123704	002636		CMPB	\$GDDAT,R4	;DATA CORRECT	
5387	020762	001414			BEQ	67\$;BR IF YES	
5388	020764				RERROR	4	;DATA ERROR	
5389	021004	104455			TRAP	C\$ERDF		
5390	021006	000004			.WORD	4		
5391	021010	004444			.WORD	EM4		
5392	021012	006266			.WORD	ERR4		
5393	021014			67\$:	ESCAPE	SEG		
5394	021014	104410			TRAP	C\$ESCAPE		
5395	021016	000010			.WORD	10000\$-		
5396	021020	000241			CLC		;CLEAR CARRY	
5397	021022	106105			ROLB	R5	;SHIFT BIT IN R5	
5398	021024	001323			BNE	64\$;DONE IF R5=0	
5399	021026				ENDSEG			
5400	021026			10000\$:				
5401	021026	104405			TRAP	C\$ESEG		
5402	021030	012705	000001		MOV	#1,R5	;START WITH BIT0	
5403	021034				BGNSEG			
5404	021034	104404			TRAP	C\$BSEG		
5405								
5406	021036	005105		73\$:	COM	R5	;CHANGE TO FLOATING ZERO	
5407	021040	042737	000017	021062	69\$:	BIC	#17,70\$;CLEAR ADDRESS FIELD OF INSTRUCTION
5408	021046	050237	021062		BIS	R2,70\$;ADD ADDRESS TO INSTRUCTION	
5409	021052	010561	000004		MOV	R5,4(R1)	;WRITE PATTERN IN PORT4	
5410	021056				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5411	021056	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5412	021062	123100		70\$:	123100		;WRITE SCRATCH PAD(ADDRESS IN R2)	
5413	021064	042737	000017	021102	BIC	#17,71\$;CLEAR ADDRESS FIELD OF INSTRUCTION	
5414	021072	050237	021102		BIS	R2,71\$;ADD ADDRESS TO INSTRUCTION	
5415	021076				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5416	021076	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5417	021102	040600		71\$:	040600		;MOVE SP TO BR	
5418	021104				ROMCLK		;NEXT WORD IS INSTRUCTION, BBN	
5419	021104	004537	003244		JSR	R5, .ROMCLK	;CLOCK INSTRUCTION	
5420	021110	061225			061225		;MOVE BR TO PORT5	
5421	021112	010537	002636		MOV	R5,\$GDDAT	;PUT 'EXPECTED' IN \$GDDAT	
5422	021116	116104	000005		MOVB	5(R1),R4	;PUT 'FOUND' IN \$GDDAT	
5423	021122	123704	002636		CMPB	\$GDDAT,R4	;DATA CORRECT?	
5424	021126	001414			BEQ	72\$;BR IF YES	
5425	021130				RERROR	4	;DATA ERROR	
5426	021150	104455			TRAP	C\$ERDF		
5427	021152	000004			.WORD	4		
5428	021154	004444			.WORD	EM4		


```

5429 021156 006266
5430 021160
5431 021160 104410
5432 021162 000032
5433 021164 005105
5434 021166 000241
5435 021170 106105
5436 021172 001321
5437 021174
5438 021174
5439 021174 104405
5440 021176 012705 000001
5441 021202 005202
5442 021204 022702 000020
5443 021210 001230
5444 021212
5445 021212
5446 021212 104403
5447 021214
5448 021214
5449 021214 104401
5450
5451 021216
5452
5453
5454
5455
5456 021216
5457
5458
5459 021216
5460 021216
5461 021216
5462 021216 004537 003156
5463 021222 012705 000001
5464 021226 005003
5465 021230
5466 021230 013701 002716
5467 021234
5468 021234 104404
5469 021236 010302
5470 021240 042737 000017 021262
5471 021246 050237 021262
5472 021252 010561 000004
5473 021256
5474 021256 004537 003244
5475 021262 123100
5476 021264 042737 000017 021302

72$: .WORD ERR4
      ESCAPE TST
      TRAP C$ESCAPE
      .WORD L10107-.
      COM R5 ;CHANGE BACK TO A ONE
      CLC ;CLEAR CARRY
      ROLB R5 ;SHIFT BIT IN R5
      BNE 73$ ;DONE IF R5=0

ENDSEG
10001$: TRAP C$ESEG
        MOV #1,R5 ;RESTART AT BIT 0
        INC R2 ;NEXT SP ADDRESS
        CMP #20,R2 ;LAST ADDRESS?
        BNE 1$ ;BR IF NO
        ENDSUB

L10110: TRAP C$ESUB

ENDTST
L10107: TRAP C$ETST

BADHEAD
:***** TEST 31 *****
:*SCRATCH PAD DUAL ADDRESSING TEST
:*WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
:*READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING
BADHEAD
:***** TEST 31 *****

BGNTST
T31:: MSTCLR ;MASTER CLEAR M8200,4,7
      JSR R5,.MSTCLR ;CLEAR M8200,4,7
      MOV #1,R5 ;START WITH A 1
      CLR R3 ;ADDRESS 0
      MYINT
      MOV KMCSR,R1 ;GET DEVICE ADDRESS.
      BGNSEG
      TRAP C$BSEG
      MOV R3,R2 ;MOVE ADDRESS TO R2
      BIC #17,2$ ;CLEAR ADDRESS FIELD
      BIS R2,2$ ;ADD ADDRESS TO INSTRUCTION
      MOV R5,4(R1) ;WRITE PATTERN IN PORT4
      ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
      JSR R5,.ROMCLK ;CLOCK INSTRUCTION
      123100 ;WRITE SP(ADDRESS IN R2)
      BIC #17,3$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
  
```

5477 021272 050237 021302
5478 021276
5479 021276 004537 003244

BIS R2,3\$
ROMCLK
JSR R5,.ROMCLK

;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, BBN
;CLOCK INSTRUCTION

```

5480 021302 060600          3$: 60600          ;MOVE SP TO BR
5481 021304              ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
5482 021304 004537 003244   JSR      R5, .ROMCLK  ;CLOCK INSTRUCTION
5483 021310 061225          61225          ;MOVE BR TO PORT5
5484 021312 010537 002636   MOV      R5,$GDDAT   ;PUT 'EXPECTED' IN $GDDAT
5485 021316 116104 000005   MOV     5(R1),R4     ;PUT 'FOUND' IN R4
5486 021322 123704 002636   CMP     $GDDAT,R4   ;DATA CORRECT
5487 021326 001414          BEQ     4$           ;BR IF YES
5488 021330              RRROR      4           ;DATA ERROR
5489 021350 104455          TRAP    C$ERDF
5490 021352 000004          .WORD   4
5491 021354 004444          .WORD   EM4
5492 021356 006266          .WORD   ERR4
5493 021360              4$: ESCAPE    SEG
5494 021360 104410          TRAP    C$ESCAPE
5495 021362 000014          .WORD   10000$-.
5496 021364 005205          INC     R5           ;INCREMENT PATTERN
5497 021366 005203          INC     R3           ;NEXT ADDRESS
5498 021370 022703 000020   CMP     #20,R3      ;LAST ADDRESS DONE?
5499 021374 001320          BNE     1$           ;BR IF NO
5500 021376              10000$:
5501 021376              TRAP    C$ESEG
5502 021376 104405          MOV     #1,R5        ;RESTART PATTERN AT 1
5503 021400 012705 000001   CLR     R3           ;RESTART AT ADDRESS ZERO
5504 021404 005003          BGNSEG
5505 021406              TRAP    C$BSEG
5506 021406 104404          MOV     R3,R2        ;PUT ADDRESS IN R2
5507 021410 010302          BIC     #17,6$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
5508 021412 042737 000017 021430 69$: BIS     R2,6$      ;ADD ADDRESS TO INSTRUCTION
5509 021420 050237 021430   ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
5510 021424              JSR      R5, .ROMCLK  ;CLOCK INSTRUCTION
5511 021424 004537 003244   6$: 60600          ;MOVE SP TO BR
5512 021430 060600              ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
5513 021432              JSR      R5, .ROMCLK  ;CLOCK INSTRUCTION
5514 021432 004537 003244   61225          ;MOVE BR TO PORT5
5515 021436 061225          MOV     R5,$GDDAT   ;PUT 'EXPECTED' IN $GDDAT
5516 021440 010537 002636   MOV     5(R1),R4     ;PUT 'FOUND' IN $GDDAT
5517 021444 116104 000005   CMP     $GDDAT,R4   ;DATA CORRECT?
5518 021450 123704 002636   BEQ     7$           ;BR IF YES
5519 021454 001414          RRROR      5           ;SP ADDRESSING ERROR
5520 021456              TRAP    C$ERDF
5521 021476 104455          .WORD   5
5522 021500 000005          .WORD   EM5
5523 021502 004472          .WORD   ERR5
5524 021504 006350          7$: ESCAPE    SEG
5525 021506              TRAP    C$ESCAPE
5526 021506 104410          .WORD   10001$-.
5527 021510 000014          INC     R5           ;INCREMENT PATTERN
5528 021512 005205          INC     R3           ;NEXT ADDRESS
5529 021514 005203          CMP     #20,R3      ;LAST ADDRESS DONE?
5530 021516 022703 000020   BNE     5$           ;BR IF NO
5531 021522 001332          ENDSEG
5532 021524              10001$:
5533 021524              TRAP    C$ESEG
5534 021524 104405          5535 021526
5535 021526              ENDTST
  
```

```
5536 021526          L10111: TRAP  C$ETST
5537 021526 104401
5538
5539 021530          BADHEAD
5540          ;***** TEST 32 *****
5541          ;*INTERRUPT TEST
5542          ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR A
5543 021530          BADHEAD
5544          ;***** TEST 32 *****
5545
5546 021530          BGNTST
5547 021530          T32::
5548 021530
5549 021530 013701 002716          MYINT
5550 021534          MOV  KMCSR,R1          ;GET DEVICE ADDRESS.
5551 021534 104433          BRESET          ;BUS RESET
5552 021536 005011          TRAP  C$RESET
5553 021540 004537 003552          CLR  (R1)          ;CLEAR RUN
5554 021544 021664          JSR  R5,SETVEC          ;SET UP VECTORS
5555 021546 021636          3$          ;XX0
5556 021550 000340 000340          2$          ;XX4
5557 021554          .WORD 340,340          ;LEVEL 7
5558 021554 012700 000340          1$: SETPRI #PRI07          ;PS = LEVEL 7
5559 021560 104441          MOV  #PRI07,R0
5560 021562 012761 000200 000004          TRAP  C$SPRI
5561 021570          MOV  #200,4(R1)          ;WRITE PORT4
5562 021570 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
5563 021574 121111          JSR  R5,.ROMCLK          ;CLOCK INSTRUCTION
5564 021576          121111          ;SET BR RQ IN IBUS* REG 11
5565 021576 012700 000000          SETPRI #PRI00          ;ALLOW INTERRUPT
5566 021602 104441          MOV  #PRI00,R0
5567 021604 000240          TRAP  C$SPRI
5568 021606          NOP
5569 021624 104455          ERROR 31          ;NO INTERRUPT
5570 021626 000037          TRAP  C$ERDF
5571 021630 005312          .WORD 31
5572 021632 010144          .WORD EM31
5573 021634 000415          .WORD ERR31
5574 021636          BR  4$
5575 021654 104455          2$: ERROR 32          ;WRONG VECTOR
5576 021656 000040          TRAP  C$ERDF
5577 021660 005341          .WORD 32
5578 021662 010172          .WORD EM32
5579 021664 062706 000004          .WORD ERR32
5580 021670          3$: ADD  #4,SP          ;RESET STACK
5581 021670          4$:
5582 021670          ENDTST
5583 021670 104401          L10112: TRAP  C$ETST
5584
5585 021672          BADHEAD
5586          ;***** TEST 33 *****
5587          ;*INTERRUPT TEST
5588          ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR B
5589 021672          BADHEAD
5590          ;***** TEST 33 *****
5591
```

```

5592 021672          BGNTST
5593 021672          T33::
5594 021672
5595 021672 013701 002716          MYINT
5596 021676          MOV      KMCSR,R1          ;GET DEVICE ADDRESS.
5597 021676 004537 003156          MSTCLR          ;MASTER CLEAR M8200,4,7
5598 021702 004537 003552          JSR      R5,.MSTCLR          ;CLEAR M8200,4,7
5599 021706 022000          JSR      R5,SETVEC          ;SET UP VECTORS
5600 021710 022026          2$          ;XX0
5601 021712 000340 000340          3$          ;XX4
5602 021716          .WORD   340,340          ;LEVEL 7
5603 021716 012700 000340          1$: SETPRI  #PRI07          ;PS = LEVEL 7
5604 021722 104441          MOV      #PRI07,R0
5605 021724 012761 000300 000004          TRAP    C$SPRI
5606 021732          MOV      #300,4(R1)          ;WRITE PORT4
5607 021732 004537 003244          ROMCLK          ;NEXT WORD IS INSTRUCTION, BBN
5608 021736 121111          JSR      R5,.ROMCLK          ;CLOCK INSTRUCTION
5609 021740          121111          ;SET BR R0 IN IBUS* REG 11
5610 021740 012700 000000          SETPRI  #PRI00          ;ALLOW INTERRUPT
5611 021744 104441          MOV      #PRI00,R0
5612 021746 000240          TRAP    C$SPRI
5613 021750          NOP
5614 021766 104455          ERROR   31          ;NO INTERRUPT
5615 021770 000037          TRAP    C$ERDF
5616 021772 005312          .WORD   31
5617 021774 010144          .WORD   EM31
5618 021776 000415          .WORD   ERR31
5619 022000          BR      4$
5620 022016 104455          2$: ERROR   32          ;WRONG VECTOR
5621 022020 000040          TRAP    C$ERDF
5622 022022 005341          .WORD   32
5623 022024 010172          .WORD   EM32
5624 022026 062706 000004          3$: .WORD   ERR32
5625 022032          ADD     #4,SP          ;RESET STACK
5626 022032          4$:
5627 022032          ENDTST
5628 022032 104401          L10113: TRAP    C$ETST
5629
5630 022034          BADHEAD
5631          ;***** TEST 34 *****
5632          ;*PRIORITY INTERRUPT TEST
5633          ;*SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
5634          ;*THE M8200,4,7 LEVEL, VERIFY THAT M8200,4,7 DOES NOT INTERRUPT
5635 022034          BADHEAD
5636          ;***** TEST 34 *****
5637
5638 022034          BGNTST
5639 022034          T34::
5640 022034
5641 022034 013701 002716          MYINT
5642 022040          MOV      KMCSR,R1          ;GET DEVICE ADDRESS.
5643 022040 004537 003156          MSTCLR          ;MASTER CLEAR M8200,4,7
5644 022044 012704 000340          JSR      R5,.MSTCLR          ;CLEAR M8200,4,7
5645 022050          MOV      #340,R4          ;PUT LEVEL 7 IN R2
5646 022050 010400          SETPRI  R4          ;SET PRIORITY TO 7
5647 022052 104441          MOV      R4,R0
5647 022052 104441          TRAP    C$SPRI
  
```

```

5648 022054 013705 002700      MOV     STAT1,R5      ;GET BR LEVEL OF M8200,4,7
5649 022060 006205              ASR     R5            ;SHIFT R5 4 TIMES
5650 022062 006205              ASR     R5            ;TO GET PROPER LEVEL
5651 022064 006205              ASR     R5
5652 022066 006205              ASR     R5
5653 022070 042705 177437      BIC     #177437,R5    ;CLEAR UNWANTED BITS
5654 022074 010537 002636      MOV     R5,$GDDAT
5655 022100 004537 003552      JSR     R5,SETVEC    ;SET UP VECTORS
5656 022104 022150              2$
5657 022106 022150              2$
5658 022110 000340 000340      .WORD  340,340      ;A VECTOR
5659 022114 012761 000200 000004 4$: MOV     #200,4(R1)    ;B VECTOR
5660 022122              ROMCLK              ;PRIORITY 7
5661 022122 004537 003244      JSR     R5,ROMCLK    ;LOAD PORT4
5662 022126 121111              121111              ;NEXT WORD IS INSTRUCTION, BBN
5663 022130              5$: SETPRI  R4            ;CLOCK INSTRUCTION
5664 022130 010400              MOV     R4,R0        ;SET BR REQUEST
5665 022132 104441              TRAP   C$SPRI        ;PUT LEVEL IN R2 IN PS
5666 022134 000240              NOP
5667 022136 020504              CMP     R5,R4        ;IS PRESENT PS LEVEL = TO M8200,4,7 LEVEL
5668 022140 001420              BEQ    1$            ;BR IF YES
5669 022142 162704 000040      SUB     #40,R4        ;NO GET NEXT LOWER LEVEL IN R2
5670 022146 000770              BR     5$            ;AND CONTINUE WITH TEST
5671 022150              2$: BRESET
5672 022150 104433              TRAP   C$RESET
5673 022152              ERROR  33            ;ERROR UNEXPECTED INTERRUPT
5674 022170 104455              TRAP   C$ERDF
5675 022172 000041              .WORD  33
5676 022174 005400              .WORD  EM33
5677 022176 010220              .WORD  ERR33
5678 022200 000002              RTI
5679 022202              1$: MSTCLR
5680 022202 004537 003156      JSR     R5,MSTCLR    ;CLEAR M8200,4,7
5681 022206              ENDTST
5682 022206              L10114:
5683 022206 104401              TRAP   C$ETST
5684
5685 022210              BADHEAD
5686              ;***** TEST 35 *****
5687              ;*PRIORITY INTERRUPT TESTS
5688              ;*SET PS TO ALL BR LEVELS LESS THAN THE M8200,4,7 LEVEL
5689              ;*VERIFY THAT M8200,4,7 WILL INTERRUPT
5690 022210              BADHEAD
5691              ;***** TEST 35 *****
5692
5693 022210              BGNTST
5694 022210              T35::
5695 022210
5696 022210 013701 002716      MYINT
5697 022214              MOV     KMCSR,R1     ;GET DEVICE ADDRESS.
5698 022214 004537 003156      MSTCLR
5699 022220 012704 000340      JSR     R5,MSTCLR    ;MASTER CLEAR M8200,4,7
5700 022224              MOV     #340,R4      ;CLEAR M8200,4,7
5701 022224 010400              SETPRI R4            ;PUT LEVEL 7 IN R2
5702 022226 104441              MOV     R4,R0        ;SET PRIORITY TO 7
5703 022230 013705 002700      TRAP   C$SPRI
5703 022230 013705 002700      MOV     STAT1,R5     ;GET BR LEVEL OF M8200,4,7

```

```

5704 022234 006205      ASR      R5      ;SHIFT R5 4 TIMES
5705 022236 006205      ASR      R5      ;TO GET PROPER LEVEL
5706 022240 006205      ASR      R5
5707 022242 006205      ASR      R5
5708 022244 042705 177437 BIC      #177437,R5 ;CLEAR UNWANTED BITS
5709 022250 010502      MOV      R5,R2   ;PUT MB200,4,7 LEVEL IN R2
5710 022252 162702 000040 SUB      #40,R2   ;GET NEXT LOWER LEVEL IN R2
5711 022256 004537 003552 JSR      R5,SETVEC ;SET UP VECTORS
5712 022262 022344      2$      ;A VECTOR
5713 022264 022352      3$      ;B VECTOR
5714 022266 000340 000340 .WORD   340,340 ;PRIORITY 7
5715 022272 012761 000200 000004 4$: MOV      #200,4(R1) ;LOAD PORT4
5716 022300      ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5717 022300 004537 003244 JSR      R5,..ROMCLK ;CLOCK INSTRUCTION
5718 022304 121111      121111 ;SET BR REQUEST
5719 022306      5$: SETPRI  R2 ;PUT LEVEL IN R2 IN PS
5720 022306 010200      MOV      R2,R0
5721 022310 104441      TRAP    C$SPRI
5722 022312 000240      NOP
5723 022314      ERROR  31 ;ERROR, NO INTERRUPT
5724 022332 104455      TRAP    C$ERDF
5725 022334 000037      .WORD   31
5726 022336 005312      .WORD   EM31
5727 022340 010144      .WORD   ERR31
5728 022342 000421      6$: BR      1$
5729 022344 012716 022342 2$: MOV      #6$, (SP) ;SET UP FOR RTI
5730 022350 000002      RTI
5731 022352      3$: ERROR  32 ;ERROR, WRONG VECTOR
5732 022370 104455      TRAP    C$ERDF
5733 022372 000040      .WORD   32
5734 022374 005341      .WORD   EM32
5735 022376 010172      .WORD   ERR32
5736 022400 012716 022406 MOV      #1$, (SP) ;SET UP FOR RTI
5737 022404 000002      RTI
5738 022406      1$: MSTCLR
5739 022406 004537 003156 JSR      R5,..MSTCLR ;CLEAR MB200,4,7
5740 022412      ENDTST
5741 022412      L10115:
5742 022412 104401      TRAP    C$ETST
5743
5744 022414      BADHEAD
5745      ;***** TEST 36 *****
5746      ;*NPR TEST
5747      ;*TEST OF DATO, 1 WORD FROM UPROC TO 11 MEMORY
5748 022414      BADHEAD
5749      ;***** TEST 36 *****
5750
5751 022414      BGNTST
5752 022414      T36::
5753 022414      BRESET
5754 022414 104433      TRAP    C$RESET ;BUS RESET
5755
5756 022416      MYINT
5757 022416 013701 002716 MOV      KMCSR,R1 ;GET DEVICE ADDRESS.
5758 022422 005011      CLR      (R1) ;CLEAR RUN
5759 022424 005061 000004 CLR      4(R1) ;CLR PORT4
  
```

```
5760 022430 004537 003574 JSR R5,NPRSET ;SET UP IBUS REG 0-7
5761 022434 000000 0 ;IN DATA
5762 022436 177777 -1 ;OUT DATA
5763 022440 022554 3$ ;IN BA
5764 022442 022552 2$ ;OUT BA
5765 022444 005037 022552 CLR 2$ ;CLEAR 2$
5766 022450 005061 000004 CLR 4(R1) ;CLEAR PORT 4
5767 022454 ROMCLK ;NOW MOVE TO IBUS* <11>
5768 022454 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
5769 022460 121111 121111
5770 022462 012761 000021 000004 MOV #21,4(R1) ;WRITE PORT4
5771 022470 ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
5772 022470 004537 003244 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
5773 022474 121110 121110 ;SET NPR BITS IN IBUS* REG 10
5774 022476 000240 NOP
5775 022500 012737 177777 002636 MOV #-1,$GDDAT ;PUT 'EXPECTED' IN $GDDAT
5776 022506 013704 022552 MOV 2$,R4 ;PUT 'FOUND' IN R4
5777 022512 023704 002636 CMP $GDDAT,R4 ;DATA CORRECT?
5778 022516 001413 BEQ 4$ ;BR IF YES
5779 022520 ERROR 11,YES ;ERROR NPR FAILED
5780 022532 104455 TRAP C$ERDF
5781 022534 000013 .WORD 11
5782 022536 004654 .WORD EM11
5783 022540 006660 .WORD ERR11
5784 022542 ESCAPE TST
5785 022542 104410 TRAP C$ESCAPE
5786 022544 000012 .WORD L10116-.
5787 022546 4$: EXIT TST
5788 022546 104432 TRAP C$EXIT
5789 022550 000006 .WORD L10116-.
5790 022552 000000 2$: 0 ;OUT BA
5791 022554 000000 3$: 0 ;IN BA
5792 022556 ENDTST
5793 022556 L10116:
5794 022556 104401 TRAP C$ETST
5795
5796 022560 BADHEAD
5797 ;***** TEST 37 *****
5798 ;*NPR TEST
5799 ;*TEST OF DAT1, 1 WORD FROM 11 MEMORY TO UPROC
5800 022560 BADHEAD
5801 ;***** TEST 37 *****
5802
5803 022560 BGNTST
5804 022560 T37::
5805 022560 MYINT
5806 022560 013701 002716 MOV KMCSR,R1 ;GET DEVICE ADDRESS.
5807 022564 MSTCLR ;MASTER CLEAR M8200,4,7
5808 022564 004537 003156 JSR R5,.MSTCLR ;CLEAR M8200,4,7
5809 022570 005061 000004 CLR 4(R1) ;CLR PORT4
5810 022574 004537 003574 JSR R5,NPRSET ;SET UP IBUS REG 0-7
5811 022600 000000 0 ;IN DATA
5812 022602 177777 -1 ;OUT DATA
5813 022604 022724 3$ ;IN BA
5814 022606 022722 2$ ;OUT BA
5815 022610 012737 177777 022724 MOV #-1,3$ ;PUT DATA IN 3$
```



```

5816 022616 012761 000001 000004      MOV      #1,4(R1)      ;WRITE PORT4
5817 022624      ROMCLK      ;NEXT WORD IS INSTRUCTION, BBN
5818 022624 004537 003244      JSR      R5,ROMCLK    ;CLOCK INSTRUCTION
5819 022630 121110      121110      ;SET NPR BITS IN IBUS* REG 11
5820 022632 000240      NOP
5821 022634 012737 177777 002636      MOV      #-1,$GDDAT    ;PUT 'EXPECTED' IN $GDDAT
5822 022642      ROMCLK      ;NEXT WORD IS INSTRUCTION, BBN
5823 022642 004537 003244      JSR      R5,ROMCLK    ;CLOCK INSTRUCTION
5824 022646 021004      021004      ;MOVE IN DATA LOW BYTE TO PORT4
5825 022650      ROMCLK      ;NEXT WORD IS INSTRUCTION, BBN
5826 022650 004537 003244      JSR      R5,ROMCLK    ;CLOCK INSTRUCTION
5827 022654 021025      021025      ;MOVE IN DATA HIGH BYTE TO PORT5
5828 022656 016104 000004      MOV      4(R1),R4     ;PUT 'FOUND' IN R4
5829 022662 023704 002636      CMP      $GDDAT,R4    ;DATA CORRECT?
5830 022666 001413      BEQ      4$          ;BR IF YES
5831 022670      ERROR      11,YES    ;ERROR NPR FAILED
5832 022702 104455      TRAP     C$ERDF
5833 022704 000013      .WORD   11
5834 022706 004654      .WORD   EM11
5835 022710 006660      .WORD   ERR11
5836 022712      ESCAPE     TST
5837 022712 104410      TRAP     C$ESCAPE
5838 022714 000012      .WORD   L10117-.
5839 022716      4$:      EXIT     TST
5840 022716 104432      TRAP     C$EXIT
5841 022720 000006      .WORD   L10117-.
5842 022722 000000      2$:      0          ;OUT BA
5843 022724 000000      3$:      0          ;IN BA
5844 022726      ENDTST
5845 022726      L10117:
5846 022726 104401      TRAP     C$ETST
5847
5848 022730      BADHEAD
5849      ;***** TEST 38 *****
5850      ;*NPR TEST
5851      ;*TEST OF DATOB, 1 BYTE FROM UPROC TO 11 MEMORY
5852 022730      BADHEAD
5853      ;***** TEST 38 *****
5854
5855 022730      BGNTST
5856 022730      T38::
5857 022730      MYINT
5858 022730 013701 002716      MOV      KMCSR,R1     ;GET DEVICE ADDRESS.
5859 022734      MSTCLR      ;MASTER CLEAR M8200,4,7
5860 022734 004537 003156      JSR      R5,MSTCLR    ;CLEAR M8200,4,7
5861 022740 005061 000004      CLR      4(R1)        ;CLR PORT4
5862 022744 004537 003574      JSR      R5,NPRSET    ;SET UP IBUS REG 0-7
5863 022750 000000      0          ;IN DATA
5864 022752 177777      -1        ;OUT DATA
5865 022754 023070      3$        ;IN BA
5866 022756 023067      2$+1     ;OUT BA
5867 022760 005037 023066      CLR      2$          ;CLEAR 2$
5868 022764 005061 000004      CLR      4(R1)        ;CLEAR PORT 4
5869 022770      ROMCLK      ;NOW MOVE IT TO IBUS*<11>
5870 022770 004537 003244      JSR      R5,ROMCLK    ;CLOCK INSTRUCTION
5871 022774 121111      121111

```

```

5872 022776 012761 000221 000004      MOV      #221,4(R1)      ;WRITE PORT4
5873 023004                                ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
5874 023004 004537 003244      JSR      R5,.ROMCLK      ;CLOCK INSTRUCTION
5875 023010 121110                                121110                    ;SET NPR BITS IN IBUS* REG 11
5876 023012 000240      NOP
5877 023014 012737 177400 002636      MOV      #177400,$GDDAT ;PUT 'EXPECTED' IN $GDDAT
5878 023022 013704 023066      MOV      2$,R4           ;PUT 'FOUND' IN R4
5879 023026 023704 002636      CMP      $GDDAT,R4      ;DATA CORRECT?
5880 023032 001413      BEQ      4$             ;BR IF YES
5881 023034                                ERROR 11,YES             ;ERROR NPR FAILED
5882 023046 104455      TRAP    C$ERDF
5883 023050 000013      .WORD   11
5884 023052 004654      .WORD   EM11
5885 023054 006660      .WORD   ERR11
5886 023056                                ESCAPE TST
5887 023056 104410      TRAP    C$ESCAPE
5888 023060 000012      .WORD   L10120-.
5889 023062                                4$: EXIT TST
5890 023062 104432      TRAP    C$EXIT
5891 023064 000006      .WORD   L10120-.
5892 023066 000000      2$: 0 ;OUT BA
5893 023070 000000      3$: 0 ;IN BA
5894 023072                                ENDTST
5895 023072                                L10120:
5896 023072 104401      TRAP    C$ETST

5898 023074                                BADHEAD
5899                                ;***** TEST 39 *****
5900                                ;*TEST OF EA BITS 16 AND 17
5901                                ;*DO A DATO TO AN ADDRESS USING OUT BA BITS 16 AND 17
5902                                ;*VERIFY CORRECT RESULTS
5903 023074                                BADHEAD
5904                                ;***** TEST 39 *****
5905

5906 023074                                BGNTST
5907 023074                                T39::
5908 023074                                MSTCLR                    ;MASTER CLEAR M8200,4,7
5909 023074 004537 003156      JSR      R5,.MSTCLR      ;CLEAR M8200,4,7
5910 023100                                MYINT
5911 023100 013701 002716      MOV      KMCSR,R1        ;GET DEVICE ADDRESS.
5912 023104 013737 002726 023132      MOV      KMPO6,1$        ;USE SEL4 FOR ADDRESS
5913 023112 013737 002726 023130      MOV      KMPO6,2$        ;USE SEL4 FOR ADDRESS
5914 023120 004537 003574      JSR      R5,NPRSET       ;LOAD BA AND DATA
5915 023124 000000      0 ;IN DATA
5916 023126 125252                                125252                    ;OUT DATA
5917 023130 000000      2$: 0 ;IN BA
5918 023132 000000      1$: 0 ;OUT BA
5919 023134 012761 000014 000004      MOV      #14,4(R1)       ;LOAD SEL 4 WITH OUT BA16 AND 17
5920 023142                                ROMCLK                    ;NEXT WORD IS INSTRUCTION, BBN
5921 023142 004537 003244      JSR      R5,.ROMCLK      ;CLOCK INSTRUCTION
5922 023146 121111                                121111                    ;SET OUTBA 16 AND 17
5923 023150 012761 000021 000004      MOV      #21,4(R1)       ;LOAD SEL4
5924 023156 012711 003000      MOV      #BIT9:BIT10,(R1)
5925 023162 012761 121110 000006      MOV      #121110,6(R1)  ;PUT INSTRUCTION IN SEL6
5926 023170 052711 000400      BIS      #BIT8,(R1)      ;CLOCK IT!
5927 023174 000240      NOP                      ;WAIT FOR NPR

```

```

5928 023176 012737 121110 002636      MOV      #121110,$GDDAT ;PUT 'EXPECTED' IN $GDDAT
5929 023204 000240                      NOP
5930 023206 000240                      NOP
5931                                     ;OK,LISTEN UP.EXPLANATION TIME.
5932                                     ;
5933                                     ;ON THE NPR OUT,THE DATA ENDED UP
5934                                     ;IN THE IBUS(NOT IBUS*) SENCE SEL A
5935                                     ;WAS ONLY SELECTED IN THE NPR CYCLE.
5936                                     ;THAT IS,WE DIDN'T REALLY DO AN NPR TO
5937                                     ;PORT 6,THE NPR OUT REALLY ENDED UP IN
5938                                     ;OUT DATA LOW,AND OUT DATA HIGH
5939                                     ;(IBUS <2> AND IBUS <3>).
5940
5941                                     ;WHAT WE'RE DOING NEXT IS READING IBUS 2&3
5942                                     ;TO SEE IF THE DATA GOT XFERRD CORRECTLY.
5943 023210                      ROMCLK
5944 023210 004537 003244      JSR      R5,ROMCLK      ;CLOCK INSTRUCTION
5945 023214 021044                      021044      ;READ IBUS <2> PUT IN PORT 4
5946 023216                      ROMCLK
5947 023216 004537 003244      JSR      R5,ROMCLK      ;CLOCK INSTRUCTION
5948 023222 021065                      021065      ;READ IBUS <3> PUT IN PORT 5
5949 023224 016104 000004      MOV      4(R1),R4      ;PUT 'FOUND' IN R4
5950 023230 023704 002636      CMP      $GDDAT,R4
5951 023234 001411                      BEQ      3$
5952 023236                      ERROR    11,YES
5953 023250 104455                      TRAP    C$ERDF
5954 023252 000013                      .WORD  11
5955 023254 004654                      .WORD  EM11
5956 023256 006660                      .WORD  ERR11
5957 023260
5958 023260                      3$:
5959 023260                      ENDTST
5960 023260 104401                      L10121:
5961                                     TRAP    C$ETST
5962 023262                      BADHEAD
5963                                     ;***** TEST 40 *****
5964                                     ;*TEST OF EA BITS 16 AND 17
5965                                     ;*DO A DATI USING IN BA BITS 16 AND 17
5966                                     ;*VERIFY CORRECT RESULTS
5967                                     ;*IN ORDER TO DO THIS TEST, WE WILL READ THE DATA FROM THE
5968                                     ;*CONSOL TTY CSR IF ONE EXSITS
5969                                     ;*IF NO CONSOL TTY CSR AT ADDRESS 177560, THIS TEST
5970                                     ;*WILL BE SKIPPED
5971 023262                      BADHEAD
5972                                     ;***** TEST 40 *****
5973
5974 023262                      BGNTST
5975 023262                      T40::
5976 023262
5977 023262 013701 002716      MYINT
5978 023266                      MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
5979 023266 004537 003156      MSTCLR   ;MASTER CLEAR M8200,4,7
5980 023272 012737 023454 000004      JSR      R5,MSTCLR     ;CLEAR M8200,4,7
5981 023300 012737 000340 000006      MOV      #TOUTT,4      ;SET UP FOR TRAP IN CASE IF NO
5982 023306 005737 177560                      MOV      #340,6 ;TTY AT ADDRESS 177560
5983 023312 012737 177560 023340      TST     177560
5983 023312 012737 177560 023340      MOV      #177560,1$    ;ADDRESS THE TTY-TRAPS HERE IF NONE.
5983 023312 012737 177560 023340                      ;USE SEL4 FOR ADDRESS

```

```

5984 023320 012737 177560 023336      MOV      #177560,28      ;USE SEL4 FOR ADDRESS
5985 023326 004537 003574                JSR      R5,NPRSET      ;LOAD BA AND DATA
5986 023332 000000                0                    ;IN DATA
5987 023334 125252                125252              ;OUT DATA
5988 023336 000000                2$: 0                ;IN BA
5989 023340 000000                1$: 0                ;OUT BA
5990 023342 012761 000015 000004      MOV      #15,4(R1)
5991 023350 012711 003000                MOV      #BIT9:BIT10,(R1); SET CROMI AND CROMO.
5992 023354 012761 121110 000006      MOV      #121110,6(R1); PUT INSTR INTO SEL6 NW*
5993 023362 052711 000400                BIS      #BIT8,(R1)    ;CLOCK IT!
5994 023366 000240                NOP                    ;WAIT FOR NPR
5995 023370                ROMCLK                ;NEXT WORD IS INSTRUCTION, BBN
5996 023370 004537 003244                JSR      R5,.ROMCLK    ;CLOCK INSTRUCTION
5997 023374 021004                021004              ;MOVE OUT DATA LB TO SEL4
5998 023376                ROMCLK                ;NEXT WORD IS INSTRUCTION, BBN
5999 023376 004537 003244                JSR      R5,.ROMCLK    ;CLOCK INSTRUCTION
6000 023402 021025                021025              ;MOVE OUT DATA HB TO SEL5
6001 023404 016104 000004                MOV      4(R1),R4      ;PUT "FOUND" IN R4
6002 023410 013737 177560 002636      MOV      177560,$GDDAT
6003 023416 042737 000200 002636      BIC      #200,$GDDAT
6004 023424 023704 002636                CMP      $GDDAT,R4
6005 023430 001413                BEQ      TOUTP          ;CORRECT RESULTS?
6006 023432                ERROR 11,YES          ;BR IF YES
6007 023444 104455                TRAP    C$ERDF        ;ERROR BA 16 AND 17 FAILED
6008 023446 000013                .WORD   11
6009 023450 004654                .WORD   EM11
6010 023452 006660                .WORD   ERR:1
6011 023454                3$:
6012 023454 062706 000004                TOUTT: ADD      #4,SP      ;UPDATE STACK POITNTER
6013 023460 013737 002652 000006      TOUTP: MOV      SAVE6,6  ;RESTORE TRAP VECTOR
6014 023466 013737 002650 000004      MOV      SAVE4,4
6015 023474                ENDTST
6016 023474                L10122:
6017 023474 104401                TRAP    C$ETST
6018
6019 023476                BADHEAD
6020                ;***** TEST 41 *****

```

```

6021 ;*NPR NON-EXISTENT MEMORY TEST
6022 ;*DO A DATO TO A NON-EXISTENT ADDRESS
6023 ;*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
6024 023476 BADHEAD
6025 ;***** TEST 41 *****
6026
6027 023476 BGNTST
6028 023476 T41::
6029 023476 MYINT
6030 023476 013701 002716 MOV KMCSR,P1 ;GET DEVICE ADDRESS.
6031 023502 MSTCLR ;MASTER CLEAR M8200,4,7
6032 023502 004537 003156 JSR R5,.MSTCLR ;CLEAR M8200,4,7
6033 023506 004537 003574 JSR R5,NPRSET ;LOAD IBUS REGISTERS 0-7
6034 023512 000000 0 ;IN DATA
6035 023514 000000 0 ;OUT DATA
6036 023516 177320 ;IN BA
6037 023520 177320 ;IN BA
6038 023522 012761 000014 000004 MOV #14,4(R1) ;SET OUT BA BITS 16+17 IN PORT4
  
```


6095	023724	004537	003574		JSR	R5,NPRSET	:LOAD IBUS REGISTERS 0-7
6096	023730	000000			0		:IN DATA
6097	023732	000000			0		:OUT DATA
6098	023734	177320			177320		:IN BA
6099	023736	177320			177320		:OUT BA
6100	023740	005061	000004		CLR	4(R1)	
6101	023744				ROMCLK		:NEXT WORD IS INSTRUCTION, BBN
6102	023744	004537	003244		JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
6103	023750	121111			121111		:CLEAR NON-EXISTENT BIT
6104	023752	012761	000015	000004	MOV	#15,4(R1)	:SET NPR REQUEST BITS IN PORT4
6105	023760				RJMCLK		:NEXT WORD IS INSTRUCTION, BBN
6106	023760	004537	003244		JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
6107	023764	121110			121110		:MOV IBUS* 4 TO IBUS* 10
6108	023766	000240			NOP		
6109	023770				ROMCLK		:NEXT WORD IS INSTRUCTION, BBN
6110	023770	004537	003244		JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
6111	023774	121225			121225		:MOV IBUS*11 TO IBUS*5
6112	023776	012737	000001	002636	MOV	#1,\$GDDAT	:PUT 'EXPECTED' IN \$GDDAT
6113	024004	116104	000005		MOVB	5(R1),R4	:PUT 'FOUND' IN R4
6114	024010	042704	177776		BIC	#177776,R4	:CLEAR UNWANTED BITS
6115	024014	023704	002636		CMP	\$GDDAT,R4	:DATA CORRECT?
6116	024020	001411			BEQ	1\$:BR IF YES
6117	024022				ERROR	13,YES	:ERROR NON-EXISTENT MEM BIT FAILED TO SET
6118	024034	104455			TRAP	C\$ERDF	
6119	024036	000015			.WORD	13	
6120	024040	004707			.WORD	EM13	
6121	024042	007014			.WORD	ERR13	
6122	024044						
6123	024044						
6124	024044						
6125	024044	104401			TRAP	C\$ETST	
6126							
6127	024046				BADHEAD		
6128					:***** TEST 43 *****		
6129					:*NPR TEST		
6130					:*USING DATO, NPR A BINARY COUNT (0-377)		
6131					:*FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY		
6132	024046				BADHEAD		
6133					:***** TEST 43 *****		
6134							
6135	024046				BGNTST		
6136	024046				T43::		
6137	024046				MYINT		
6138	024046	013701	002716		MOV	KMCSR,R1	:GET DEVICE ADDRESS.
6139	024052				MSTCLR		:MASTER CLEAR M8200,4,7
6140	024052	004537	003156		JSR	R5,.MSTCLR	:CLEAR M8200,4,7
6141	024056	005037	024260		CLR	5\$:START FLAG AT 0
6142	024062	005005			CLR	R5	:DATA
6143	024064	012702	035374		MOV	#CORMAX,R2	:ADDRESS
6144	024070						
6145	024070	010537	024120		MOV	R5,2\$:LOAD DATA
6146	024074	010237	024124		MOV	R2,4\$:LOAD BA
6147	024100	032702	000001		BIT	#BIT0,R2	:IS BA ODD?
6148	024104	001402			BEQ	.+6	:BR IF NO
6149	024106	000337	024120		SWAB	2\$:IF ODD PUT DATA IN HI-BYTE
6150	024112	004537	003574		JSR	R5,NPRSET	:LOAD NPR REGISTERS

```

6151 024116 000000          0          ; IN DATA
6152 024120 000000      2$: 0          ; OUT DATA
6153 024122 000000          0          ; IN BA
6154 024124 000000      4$: 0          ; OUT BA
6155 024126 105012          (LRB (R2) ; CLEAR MEMORY LOCATION
6156 024130 012761 000221 000004 MOV #221,4(R1) ; LOAD PORT4
6157 024136 ROMCLK ; NEXT WORD IS INSTRUCTION, BBN
6158 024136 004537 003244 JSR R5,ROMCLK ; CLOCK INSTRUCTION
6159 024142 121110 ; DO THE NPR
6160 024144 000240 NOP
6161 024146 010537 002636 MOV R5,$GDDAT ; PUT 'EXPECTED' IN $GDDAT
6162 024152 111204 MOVB (R2),R4 ; PUT 'FOUND' IN R4
6163 024154 123704 002636 CMPB $GDDAT,R4 ; IS DATA CORRECT?
6164 024160 001411 BEQ 3$ ; BR IF YES
6165 024162 ERROR 11,YES ; ERROR, DATA INCORRECT
6166 024174 104455 TRAP C$ERDF
6167 024176 000013 .WORD 11
6168 024200 004654 .WORD EM11
6169 024202 006660 .WORD ERR11
6170 024204 3$: ESCAPE TST
6171 024204 104410 TRAP C$ESCAPE
6172 024206 000054 .WORD L10125-.
6173 024210 005205 INC R5 ; NEXT CHARACTER
6174 024212 042705 177400 BIC #177400,R5 ; USE ONLY LOW BYTE
6175 024216 005737 024260 TST 5$ ; HAS MAX MEMORY BEEN REACHED YET?
6176 024222 001402 BEQ 6$ ; BR IF NO
6177 024224 005705 TST R5 ; DONE PATTERN?
6178 024226 001412 BEQ 7$ ; BR IF YES
6179 024230 005202 6$: INC R2 ; INC BA
6180 024232 023702 002604 CMP MEMLIM,R2 ; REACHED MEMORY LIMIT YET?
6181 024236 001314 BNE 1$ ; BR IF NOT
6182 024240 012702 035374 MOV #CORMAX,R2 ; RESTART BA AT FIRST ADDRESS
6183 024244 012737 177777 024260 MOV #-1,5$ ; SET FLAG TO END TEST AT END OF DATA PATTERN
6184 024252 000706 BR 1$ ; CONTINUE
6185 024254 7$:
6186 024254 EXIT TST
6187 024254 104432 TRAP C$EXIT
6188 024256 000004 .WORD L10125-.
6189 024260 000000 5$: 0 ; THIS LOCATION IS A FLAG, IT STARTS AT 0,
6190 ; AND IS SET TO -1 WHEN LAST MEMORY ADDRESS
6191 ; IS USED, TEST IS THEN ENDED WHEN PATTERN IS FINISHED
6192 024262 ENDTST
6193 024262 L10125:
6194 024262 104401 TRAP C$ETST
6195 ;$MEM1
6196 ;$MEM0
6197 ;$MEM2 1K
6198 ;$MEM3 1K
6199
6200 024264 BADHEAD
6201 ;***** TEST 44 *****
6202 ;*ALU C BIT TEST
6203 ;*TEST THAT AN ADD OF 377 AND 377 WILL SET THE C BIT
6204 024264 BADHEAD
6205 ;***** TEST 44 *****
6206

```



```

6207 024264
6208 024264
6209 024264
6210 024264 013701 002716
6211 024270
6212 024270 004537 003156
6213 024274 004737 003640
6214 024300 024414
6215 024302 004737 004012
6216 024306 024414
6217 024310
6218 024310 104404
6219 024312
6220 024312
6221 024312 004537 003244
6222 024316 010000
6223 024320
6224 024320 004537 003244
6225 024324 054400
6226 024326
6227 024326 004537 003244
6228 024332 040421
6229 024334
6230 024334 004537 003244
6231 024340 061224
6232 024342 012737 000001 002636
6233 024350 016104 000004
6234 024354 123704 002636
6235 024360 001411
6236 024362
6237 024374 104455
6238 024376 000042
6239 024400 005435
6240 024402 010276
6241 024404
6242 024404 104410
6243 024406 000002
6244 024410
6245 024410
6246 024410 104405
6247 024412
6248 024412
6249 024412 104401
6250 024414 377 000 000 000
6251 024417 000 000 000
6252 024422 000 000
6253
6254
6255
6256 024424
6257
6258
6259
6260
6261
6262

```

BGNTST
T44::

```

MYINT
MOV KMCSR,R1 ;GET DEVICE ADDRESS.
MSTCLR ;MASTER CLEAR M8200,4,7
JSR R5,,MSTCLR ;CLEAR M8200,4,7
JSR PC,MEMLD ;LOAD MAINMEM DATA
TDATA ;POINTER TO DATA
JSR PC,SPLD ;LOAD SP DATA
TDATA ;POINTER TO DATA
BGNSEG
TRAP C$BSEG

1$:
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5,,ROMCLK ;CLOCK INSTRUCTION
010000 ;MAR 0
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5,,ROMCLK ;CLOCK INSTRUCTION
054400.<0*20> ;ADD 377 AND 377, TO SET C BIT
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5,,ROMCLK ;CLOCK INSTRUCTION
040401!<1*20> ;ADD 0 AND 0 AND THE C BIT
ROMCLK ;NEXT WORD IS INSTRUCTION, BBN
JSR R5,,ROMCLK ;CLOCK INSTRUCTION
61224 ;PUT RESULTS IN PORT4
MOV #1,$GDDAT ;PUT 'EXPECTED' IN $GDDAT
MOV 4(R1),R4 ;PUT 'FOUND' IN R4
CMPB $GDDAT,R4 ;DATA CORRECT?
BEQ 2$ ;BR IF YES
ERROR 34,YES ;ERROR C BIT NOT SET
TRAP C$ERDF
.WORD 34
.WORD EM34
.WORD ERR34

2$:
ESCAPE SEG
TRAP C$ESCAPE
.WORD 10000$-.
ENDSEG

10000$:
TRAP C$ESEG

ENDTST
L10126:
TRAP C$ETST
TDATA. .BYTE -1,0,0,0,0,0,0,0

```

.EVEN

BADHEAD

```

:***** TEST 45 *****
:*ALU TEST
:*TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
:*ALU FUNCTION (B) CODE=11
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS

```

```

6263 024424          BADHEAD
6264                :***** TEST 45 *****
6265
6266 024424          BGNTST
6267 024424          T45::
6268 024424
6269 024424 013701 002716  MYINT
6270 024430          MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
6271 024430 004537 003156  MSTCLR          ;MASTER CLEAR M8200,4,7
6272 024434 005005          JSR      R5, .MSTCLR      ;CLEAR M8200,4,7
6273 024436 012702 024616  CLR      R5          ;MFM + SP ADDRESS
6274 024442 004737 003640  MOV      #5$,R2      ;POINTER TO CORRECT DATA
6275 024446 002654          JSR      PC,MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
6276 024450 004737 004012  MEMDAT          ;POINTER TO DATA
6277 024454 002664          JSR      PC,SPLD      ;LOAD 8 WORDS OF SP
6278 024456          SPDAT          ;POINTER TO DATA
6279 024456 104404          BGNSEG
6280 024460 004737 004060  TRAP     C$BSEG
6281 024464 042737 000017 024502 1$: JSR      PC,CLRC      ;CLEAR C BIT!
6282 024472 050537 024502  BIC      #17,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
6283 024476          BIS      R5,2$          ;ADD ADDRESS TO INSTRUCTION
6284 024476 004537 003244  ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6285 024502 010000          JSR      R5, .ROMCLK      ;CLOCK INSTRUCTION
6286 024504 042737 000017 024522 2$: 010000          ;LOAD MAR
6287 024512 050537 024522  BIC      #17,3$      ;CLEAR ADDRESS OF INSTRUCTION
6288 024516          BIS      R5,3$          ;ADD ADDRESS TO INSTRUCTION
6289 024516 004537 003244  ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6290 024522 040620          JSR      R5, .ROMCLK      ;CLOCK INSTRUCTION
6291 024524          ROMCLK          ;BR SEL B
6292 024524 004537 003244  JSR      R5, .ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6293 024530 061224          JSR      R5, .ROMCLK      ;CLOCK INSTRUCTION
6294 024532 111237 002636  MOVB     (R2), $GDDAT ;MOVE BR TO PORT4
6295 024536 116104 000004          ;PUT 'EXPECTED' IN $GDDAT
6296 024542 123704 002636  MOVB     4(R1), R4    ;PUT 'FOUND' IN R4
6297 024546 001411          CMPB     $GDDAT, R4    ;DATA CORRECT?
6298 024550          BEQ      4$          ;BR IF YES
6299 024562 104455          ERROR    15, YES     ;ALU ERROR
6300 024564 000017          TRAP     C$ERDF
6301 024566 004754          .WORD   15
6302 024570 007120          .WORD   EM15
6303 024572          .WORD   ERR15
6304 024572 104410          ESCAPE   SEG
6305 024574 000014          TRAP     C$ESCAPE
6306 024576 005202          .WORD   10000$-.
6307 024600 005205          INC      R2          ;NEXT DATA
6308 024602 022705 000010  INC      R5          ;NEXT ADDRESS
6309 024606 001324          CMP      #10, R5      ;DONE YET?
6310 024610          BNE     1$          ;BR IF NO
6311 024610          ENDSEG
6312 024610 104405          10000$: TRAP     C$ESEG
6313 024612          EXIT     TST
6314 024612 104432          TRAP     C$EXIT
6315 024614 000012          .WORD   L10127-.
6316 024616 000 377 000 5$: .BYTE   0,-1,0,-1,125,252,125,252
6317 024621 377 125 252
6318 024624 125 252
  
```

6319									
6320									
6321	024626								
6322	024626								
6323	024626	104401							
6324									
6325	024630								
6326									
6327									
6328									
6329									
6330									
6331									
6332	024630								
6333									
6334									
6335	024630								
6336	024630								
6337	024630								
6338	024630	013701	002716						
6339	024634								
6340	024634	004537	003156						
6341	024640	005005							
6342	024642	012702	025022						
6343	024646	004737	003640						
6344	024652	002654							
6345	024654	004737	004012						
6346	024660	002664							
6347	024662								
6348	024662	104404							
6349	024664	004737	004060						
6350	024670	042737	000017	024706	1\$:				
6351	024676	050537	024706						
6352	024702								
6353	024702	004537	003244						
6354	024706	010000			2\$:				
6355	024710	042737	000017	024726					
6356	024716	050537	024726						
6357	024722								
6358	024722	004537	003244						
6359	024726	040600			3\$:				
6360	024730								
6361	024730	004537	003244						
6362	024734	061224							
6363	024736	111237	002636						
6364	024742	116104	000004						
6365	024746	123704	002636						
6366	024752	001411							
6367	024754								
6368	024766	104455							
6369	024770	000017							
6370	024772	004754							
6371	024774	007120							
6372	024776				4\$				
6373	024776	104410							
6374	025000	000014							

.EVEN
 ENDTST
 L10127:

TRAP C\$ETST

BADHEAD
 :***** TEST 46 *****
 :*ALU TEST
 :*TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
 :*ALU FUNCTION (A) CODE=10
 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
 BADHEAD
 :***** TEST 46 *****

BGNTST
 T46::

MYINT
 MOV KCSR,R1 ;GET DEVICE ADDRESS.
 MSTCLR ;MASTER CLEAR M8200,4,7
 JSR R5, .MSTCLR ;CLEAR M8200,4,7
 CLR R5 ;MEM + SP ADDRESS
 MOV #5\$,R2 ;POINTER TO CORRECT DATA
 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
 MEMDAT ;POINTER TO DATA
 JSR PC,SPLD ;LOAD 8 WORDS OF SP
 SPDAT ;POINTER TO DATA
 BGNSEG
 TRAP C\$BSEG
 JSR PC,CLRC ;CLEAR C BIT!
 BIC #17,2\$;CLEAR ADDRESS FIELD OF INSTRUCTION
 BIS R5,2\$;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 010000 ;LOAD MAR
 BIC #17,3\$;CLEAR ADDRESS OF INSTRUCTION
 BIS R5,3\$;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 040400!<10*20> ;BR SEL A
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 61224 ;MOVE BR TO PORT4
 MOVB (R2), \$GDDAT ;PUT 'EXPECTED' IN \$GDDAT
 MOVB 4(R1), R4 ;PUT 'FOUND' IN R4
 CMPB \$GDDAT, R4 ;DATA CORRECT?
 BEQ 4\$;BR IF YES
 ERROR 15, YES ;ALU ERROR
 TRAP C\$ERDF
 .WORD 15
 .WORD EM15
 .WORD ERR15
 ESCAPE SEG
 TRAP C\$ESCAPE
 .WORD 10000\$-

6375	025002	005202				INC	R2	:NEXT DATA
6376	025004	005205				INC	R5	:NEXT DATA
6377	025006	022705	000010			CMP	#10,R5	:DONE YET?
6378	025012	001324				BNE	1\$:BR IF NO
6379	025014					ENDSEG		
6380	025014			10000\$:				
6381	025014	104405				TRAP	C\$ESEG	
6382	025016					EXIT	TST	
6383	025016	104432				TRAP	C\$EXIT	
6384	025020	000012				.WORD	L10130-	
6385	025022	000	000	377	5\$:	.BYTE	0,0,-1,-1,125,125,252,252	
6386	025025	377	125	125				
6387	025030	252	252					
6388								
6389								
6390	025032					.EVEN		
6391	025032					ENDTST		
6392	025032	104401				L10130:		
6393						TRAP	C\$ETST	
6394	025034					BADHEAD		
6395						:***** TEST 47 *****		
6396						:*ALU TEST		
6397						:*TEST OF ALU FUNCTION A OR NOTB WITH C BIT CLEARED		
6398						:*ALU FUNCTION (A OR NOTB) CODE=12		
6399						:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA		
6400						:*PERFORM THE FUNCTION, VERIFY THE RESULTS		
6401	025034					BADHEAD		
6402						:***** TEST 47 *****		
6403								
6404	025034					BGNTST		
6405	025034					T47::		
6406	025034					MYINT		
6407	025034	013701	002716			MOV	KMCSR,R1	:GET DEVICE ADDRESS.
6408	025040					MSTCLR		:MASTER CLEAR M8200,4,7
6409	025040	004537	003156			JSR	R5,.MSTCLR	:CLEAR M8200,4,7
6410	025044	005005				CLR	R5	:MEM + SP ADDRESS
6411	025046	012702	025226			MOV	#5\$,R2	:POINTER TO CORRECT DATA
6412	025052	004737	003640			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
6413	025056	002654				MEMDAT		:POINTER TO DATA
6414	025060	004737	004012			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
6415	025064	002664				SPDAT		:POINTER TO DATA
6416	025066					BGNSEG		
6417	025066	104404				TRAP	C\$BSEG	
6418	025070	004737	004060			JSR	PC,CLRC	:CLEAR C BIT!
6419	025074	042737	000017	025112	1\$:	BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
6420	025102	050537	025112			BIS	R5,2\$:ADD ADDRESS TO INSTRUCTION
6421	025106					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6422	025106	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
6423	025112	010000				010000		:LOAD MAR
6424	025114	042737	000017	025132	2\$:	BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
6425	025122	050537	025132			BIS	R5,3\$:ADD ADDRESS TO INSTRUCTION
6426	025126					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6427	025126	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
6428	025132	040640				040400!	<12*20>	:BR A OR NOTB
6429	025134					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6430	025134	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION


```

6487 025274 004737 004060 1$: JSR PC,CLRC ;CLEAR C BIT!
6488 025300 042737 000017 025316 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
6489 025306 050537 025316 BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
6490 025312 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6491 025312 004537 003244 JSR R5,ROMCLK ;CLOCK INSTRUCTION
6492 025316 010000 2$: 010000 ;LOAD MAR
6493 025320 042737 000017 025336 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
6494 025326 050537 025336 BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
6495 025332 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6496 025332 004537 003244 JSR R5,ROMCLK ;CLOCK INSTRUCTION
6497 025336 040660 3$: 040400!<13*20> ;BR A AND B
6498 025340 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6499 025340 004537 003244 JSR R5,ROMCLK ;CLOCK INSTRUCTION
6500 025344 061224 61224 ;MOVE BR TO PORT4
6501 025346 111237 002636 MOVB (R2),SGDDAT ;PUT 'EXPECTED' IN SGDDAT
6502 025352 116104 000004 MOVB 4(R1),R4 ;PUT 'FOUND' IN R4
6503 025356 123704 002636 CMPB SGDDAT,R4 ;DATA CORRECT?
6504 025362 001411 BEQ 4$ ;BR IF YES
6505 025364 ERROR 15,YES ;ALU ERROR
6506 025376 104455 TRAP C$ERDF
6507 025400 000017 .WORD 15
6508 025402 004754 .WORD EM15
6509 025404 007120 .WORD ERR15
6510 025406 4$: ESCAPE SEG
6511 025406 104410 TRAP C$ESCAPE
6512 025410 000014 .WORD 10000$-
6513 025412 005202 INC R2 ;NEXT DATA
6514 025414 005205 INC R5 ;NEXT DATA
6515 025416 022705 000010 CMP #10,R5 ;DONE YET?
6516 025422 001324 BNE 1$ ;BR IF NO
6517 025424 ENDSEG
6518 025424 10000$:
6519 025424 104405 TRAP C$ESEG
6520 025426 EXIT TST
6521 025426 104432 TRAP C$EXIT
6522 025430 000012 .WORD L10132-
6523 025432 000 000 000 5$: .BYTE 0,0,0,-1,125,0,0,252
6524 025435 377 125 000
6525 025440 000 252
6526
6527 .EVEN
6528 025442 ENDTST
6529 025442 L10132:
6530 025442 104401 TRAP C$ETST
6531
6532 025444 BADHEAD
6533 ;***** TEST 49 *****
6534 ;*ALU TEST
6535 ;*TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
6536 ;*ALU FUNCTION (A OR B) CODE=14
6537 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6538 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
6539 025444 BADHEAD
6540 ;***** TEST 49 *****
6541
6542 025444 BGNTST
  
```

```

6543 025444          T49::
6544 025444          MYINT
6545 025444 013701 002716  MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
6546 025450          MSTCLR          ;MASTER CLEAR M8200,4,7
6547 025450 004537 003156  JSR      R5,.MSTCLR    ;CLEAR M8200,4,7
6548 025454 005005          CLR      R5            ;MEM + SP ADDRESS
6549 025456 012702 025636  MOV      #5$,R2        ;POINTER TO CORRECT DATA
6550 025462 004737 003640  JSR      PC,MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
6551 025466 002654          MEMDAT          ;POINTER TO DATA
6552 025470 004737 004012  JSR      PC,SPLD       ;LOAD 8 WORDS OF SP
6553 025474 002664          SPDAT          ;POINTER TO DATA
6554 025476          BGNSEG
6555 025476 104404          TRAP      C$BSEG
6556 025500 004737 004060          JSR      PC,CLRC      ;CLEAR C BIT!
6557 025504 042737 000017 025522 1$:  BIC      #17,2$       ;CLEAR ADDRESS FIELD OF INSTRUCTION
6558 025512 050537 025522          BIS      R5,2$        ;ADD ADDRESS TO INSTRUCTION
6559 025516          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6560 025516 004537 003244          JSR      R5,.ROMCLK  ;CLOCK INSTRUCTION
6561 025522 010000          2$: 010000          ;LOAD MAR
6562 025524 042737 000017 025542  BIC      #17,3$       ;CLEAR ADDRESS OF INSTRUCTION
6563 025532 050537 025542          BIS      R5,3$        ;ADD ADDRESS TO INSTRUCTION
6564 025536          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6565 025536 004537 003244          JSR      R5,.ROMCLK  ;CLOCK INSTRUCTION
6566 025542 040700          3$: 040400!<14*20> ;BR A OR B
6567 025544          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6568 025544 004537 003244          JSR      R5,.ROMCLK  ;CLOCK INSTRUCTION
6569 025550 061224          61224          ;MOVE BR TO PORT4
6570 025552 111237 002636          MOVB    (R2),%GDDAT   ;PUT 'EXPECTED' IN %GDDAT
6571 025556 116104 000004          MOVB    4(R1),R4      ;PUT 'FOUND' IN R4
6572 025562 123704 002636          CMPB    %GDDAT,R4    ;DATA CORRECT?
6573 025566 001411          BEQ     4$            ;BR IF YES
6574 025570          ERROR          ;ALU ERROR
6575 025602 104455          TRAP    C$ERDF
6576 025604 000017          .WORD  15
6577 025606 004754          .WORD  EM15
6578 025610 007120          .WORD  ERR15
6579 025612          4$:  ESCAPE  SEG
6580 025612 104410          TRAP    C$ESCAPE
6581 025614 000014          .WORD  10000$-.
6582 025616 005202          INC     R2            ;NEXT DATA
6583 025620 005205          INC     R5            ;NEXT DATA
6584 025622 022705 000010          CMP     #10,R5        ;DONE YET?
6585 025626 001324          BNE    1$            ;BR IF NO
6586 025630          ENDSEG
6587 025630          10000$:
6588 025630 104405          TRAP    C$ESEG
6589 025632          EXIT    TST
6590 025632 104432          TRAP    C$EXIT
6591 025634 000012          .WORD  L10133-.
6592 025636          000 377 377 5$: .BYTE  0,-1,-1,-1,125,-1,-1,252
6593 025641          377 125 377
6594 025644          377 252
6595
6596          .EVEN
6597 025646          ENDTST
6598 025646          L10133:

```

```

6599 025646 104401 TRAP C$ETST
6600
6601 025650 BADHEAD
6602 :***** TEST 50 *****
6603 :*ALU TEST
6604 :*TEST OF ALU FUNCTION A XOR B WITH C BIT CLEARED
6605 :*ALU FUNCTION (A XOR B) CODE=15
6606 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6607 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
6608 025650 BADHEAD
6609 :***** TEST 50 *****
6610
6611 025650 RGNTST
6612 025650 T50::
6613 025650 MYINT
6614 025650 013701 002716 MOV KMCSR,R1 ;GET DEVICE ADDRESS.
6615 025654 MSTCLR ;MASTER CLEAR M8200,4,7
6616 025654 004537 003156 JSR R5, .MSTCLR ;CLEAR M8200,4,7
6617 025660 005005 CLR R5 ;MEM + SP ADDRESS
6618 025662 012702 026042 MOV #5$,R2 ;POINTER TO CORRECT DATA
6619 025666 004737 003640 JSR PC, MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
6620 025672 002654 MEMDAT ;POINTER TO DATA
6621 025674 004737 004012 JSR PC, SPLD ;LOAD 8 WORDS OF SP
6622 025700 002664 SPDAT ;POINTER TO DATA
6623 025702 BGNSEG
6624 025702 104404 TRAP C$BSEG
6625 025704 004737 004060 1$: JSR PC, CLRC ;CLEAR C BIT!
6626 025710 042737 000017 025726 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
6627 025716 050537 025726 BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
6628 025722 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6629 025722 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
6630 025726 010000 2$: 010000 ;LOAD MAR
6631 025730 042737 000017 025746 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
6632 025736 050537 025746 BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
6633 025742 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6634 025742 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
6635 025746 040720 3$: 040400!<15*20> ;BR A XOR B
6636 025750 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6637 025750 004537 003244 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
6638 025754 061224 61224 ;MOVE BR TO PORT4
6639 025756 111237 002636 MOVB (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
6640 025762 116104 000004 MOVB 4(R1), R4 ;PUT 'FOUND' IN R4
6641 025766 123704 002636 CMPB $GDDAT, R4 ;DATA CORRECT?
6642 025772 001411 BEQ 4$ ;BR IF YES
6643 025774 ERROR 15, YES ;ALU ERROR
6644 026006 104455 TRAP C$ERDF
6645 026010 000017 .WORD 15
6646 026012 004754 .WORD EM15
6647 026014 007120 .WORD ERR15
6648 026016 4$: ESCAPE SEG
6649 026016 104410 TRAP C$ESCAPE
6650 026020 000014 .WORD 10000$-
6651 026022 005202 INC R2 ;NEXT DATA
6652 026024 005205 INC R5 ;NEXT DATA
6653 026026 022705 000010 CMP #10, R5 ;DONE YET?
6654 026032 001324 BNE 1$ ;BR IF NO
  
```



```

6655 026034
6656 026034
6657 026034 104405
6658 026036
6659 026036 104432
6660 026040 000012
6661 026042 000 377 377 5$:
6662 026045 000 000 377
6663 026050 377 000
6664
6665
6666 026052
6667 026052
6668 026052 104401
6669
6670 026054
6671
6672
6673
6674
6675
6676
6677 026054
6678
6679
6680 026054
6681 026054
6682 026054
6683 026054 013701 002716
6684 026060
6685 026060 004537 003156
6686 026064 005005
6687 026066 012702 026246
6688 026072 004737 003640
6689 026076 002654
6690 026100 004737 004012
6691 026104 002664
6692 026106
6693 026106 104404
6694 026110 004737 004060
6695 026114 042737 000017 026132
6696 026122 050537 026132
6697 026126
6698 026126 004537 003244
6699 026132 010000
6700 026134 042737 000017 026152
6701 026142 050537 026152
6702 026146
6703 026146 004537 003244
6704 026152 040400
6705 026154
6706 026154 004537 003244
6707 026160 061224
6708 026162 111237 002636
6709 026166 116104 000004
6710 026172 123704 002636

```

```

ENDSEG
10000$:
TRAP C$ESEG
EXIT TST
TRAP C$EXIT
.WORD L10134-
.BYTE 0,-1,-1,0,0,-1,-1,0

```

```

.EVEN
ENDTST
.10134:
TRAP C$ETST

```

```

BADHEAD
:***** TEST 51 *****
:*ALU TEST
:*TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
:*ALU FUNCTION (A PLUS B) CODE=00
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS
BADHEAD
:***** TEST 51 *****

```

```

BGNTST
T51::

```

```

MYINT
MOV KMCSR,R1 ;GET DEVICE ADDRESS.
MSTCLR ;MASTER CLEAR M8200,4,7
JSR R5, .MSTCLR ;CLEAR M8200,4,7
CLR R5 ;MEM + SP ADDRESS
MOV #5$,R2 ;POINTER TO CORRECT DATA
JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
MEMDAT ;POINTER TO DATA
JSR PC,SPLD ;LOAD 8 WORDS OF SP
SPDAT ;POINTER TO DATA
BGNSEG
TRAP C$BSEG
JSR PC,CLRC ;CLEAR C BIT
BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR R5, .ROMCLK ;CLOCK INSTRUCTION
010000 ;LOAD MAR
BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR R5, .ROMCLK ;CLOCK INSTRUCTION
040400!<00*20> ;BR ADD
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
JSR R5, .ROMCLK ;CLOCK INSTRUCTION
61224 ;MOVE BR TO PORT4
MOVB (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
MOVB 4(R1), R4 ;PUT 'FOUND' IN R4
CMPB $GDDAT, R4 ;DATA CORRECT?

```

```
6711 026176 001411 BEQ 4$ :BR IF YES
6712 026200 ERROR 15,YES :ALU ERROR
6713 026212 104455 TRAP C$ERDF
6714 026214 000017 .WORD 15
6715 026216 004754 .WORD EM15
6716 026220 007120 .WORD ERR15
6717 026222 4$: ESCAPE SEG
6718 026222 104410 TRAP C$ESCAPE
6719 026224 000014 .WORD 10000$-
6720 026226 005202 INC R2 :NEXT DATA
6721 026230 005205 INC R5 :NEXT DATA
6722 026232 022705 000010 CMP #10,R5 :DONE YET?
6723 026236 001324 BNE 1$ :BR IF NO
6724 026240 ENDSEG
6725 026240 10000$:
6726 026240 104405 TRAP C$ESEG
6727 026242 EXIT TST
6728 026242 104432 TRAP C$EXIT
6729 026244 000012 .WORD L10135-
6730 026246 000 377 377 5$: .BYTE 0,-1,-1,376,252,-1,-1,124
6731 026251 376 252 377
6732 026254 377 124
6733
6734 .EVEN
6735 026256 ENDTST
6736 026256 L10135:
6737 026256 104401 TRAP C$ETST
6738
6739 026260 BADHEAD
6740 :***** TEST 52 *****
6741 :*ALU TEST
6742 :*TEST OF ALU FUNCTION 2A W.C WITH C BIT CLEARED
6743 :*ALU FUNCTION (A PLUS A PLUS C) CODE=6
6744 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6745 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
6746 026260 BADHEAD
6747 :***** TEST 52 *****
6748
6749 026260 BGNTST
6750 026260 T52::
6751 026260 MYINT
6752 026260 013701 002716 MOV KMCSR,R1 :GET DEVICE ADDRESS.
6753 026264 MSTCLR :MASTER CLEAR M8200,4,7
6754 026264 004537 003156 JSR R5,.MSTCLR :CLEAR M8200,4,7
6755 026270 005005 CLR R5 :MEM + SP ADDRESS
6756 026272 012702 026452 MOV #5$,R2 :POINTER TO CORRECT DATA
6757 026276 004737 003640 JSR PC,MEMLD :LOAD 8 WORDS OF MAIN MEMORY
6758 026302 002654 MEMDAT :POINTER TO DATA
6759 026304 004737 004012 JSR PC,SPLD :LOAD 8 WORDS OF SP
6760 026310 002664 SPDAT :POINTER TO DATA
6761 026312 BGNSEG
6762 026312 104404 TRAP C$BSEG
6763 026314 004737 004060 1$: JSR PC,CLRC :CLEAR C BIT!
6764 026320 042737 000017 026336 BIC #17,2$ :CLEAR ADDRESS FIELD OF INSTRUCTION
6765 026326 050537 026336 BIS R5,2$ :ADD ADDRESS TO INSTRUCTION
6766 026332 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
```



```

6823 026470 004537 003156      JSR    R5, .MSTCLR      ;CLEAR M8200.4,7
6824 026474 005005              CLR    R5              ;MEM + SP ADDRESS
6825 026476 012702 026660      MOV    #5$,R2         ;POINTER TO CORRECT DATA
6826 026502 004737 003640      JSR    PC,MEMLD       ;LOAD 8 WORDS OF MAIN MEMORY
6827 026506 002654              MEMDAT                ;POINTER TO DATA
6828 026510 004737 004012      JSR    PC,SPLD        ;LOAD 8 WORDS OF SP
6829 026514 002664              SPJAT                ;POINTER TO DATA
6830 026516                    BJSNSEG
6831 026516 104404              TRAP   C$BSEG
6832 026520 004737 004060      JSR    PC,CLRC        ;CLEAR C BIT!
6833 026524 042737 000017 026542 1$:  BIC    #17,2$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
6834 026532 050537 026542      BIS    R5,2$         ;ADD ADDRESS TO INSTRUCTION
6835 026536                    ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6836 026536 004537 003244      JSR    R5, .ROMCLK    ;CLOCK INSTRUCTION
6837 026542 010000 010000 2$:  010000 ;LOAD MAR
6838 026544 042737 000017 026562  BIC    #17,3$         ;CLEAR ADDRESS OF INSTRUCTION
6839 026552 050537 026562      BIS    R5,3$         ;ADD ADDRESS TO INSTRUCTION
6840 026556                    ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6841 026556 004537 003244      JSR    R5, .ROMCLK    ;CLOCK INSTRUCTION
6842 026562 040740 040400 <16*20> 3$:  ;BR _ SUB
6843 026564                    ROMCLK
6844 026564 004537 003244      JSR    R5, .ROMCLK    ;CLOCK INSTRUCTION
6845 026570 061224 61224 ;MOVE BR TO PORT4
6846 026572 111237 002636      MOVB  (R2), $GDDAT    ;PUT 'EXPECTED' IN $GDDAT
6847 026576 116104 000004      MOVB  4(R1), R4      ;PUT 'FOUND' IN R4
6848 026602 123737 002636 002636  CMPB  $GDDAT, $GDDAT ;DATA CORRECT?
6849 026610 001411 001411      BEQ   4$             ;BR IF YES
6850 026612              ERROR 15, YES      ;ALU ERROR
6851 026624 104455              TRAP  C$ERDF
6852 026626 000017              .WORD 15
6853 026630 004754              .WORD EM15
6854 026632 007120              .WORD ERR15
6855 026634 007120 4$:  ESCAPE SEG
6856 026634 104410              TRAP  C$ESCAPE
6857 026636 000014              .WORD 10000$-
6858 026640 005202              INC   R2             ;NEXT DATA
6859 026642 005205              INC   R5             ;NEXT ADDRESS
6860 026644 022705 000010      CMP   #10, R5        ;DONE YET?
6861 026650 001323              BNE  1$             ;BR IF NO
6862 026652                    ENDSEG
6863 026652 10000$:
6864 026652 104405              TRAP  C$ESEG
6865 026654                    EXIT  TST
6866 026654 104432              TRAP  C$EXIT
6867 026656 000012              .WORD L10137-
6868 026660 000 001 377 5$:  .BYTE 0,1,-1,0,0,253,125,0
6869 026663 000 000 253
6870 026666 125 000
6871
6872
6873              .EVEN
6874 026670              ENDTST
6875 026670              L10137:
6876 026670 104401              TRAP  C$ETST
6877
6878

```

6879 026672
 6880
 6881
 6882
 6883
 6884
 6885
 6886 026672
 6887
 6888
 6889 026672
 6890 026672
 6891 026672
 6892 026672 013701 002716
 6893 026676
 6894 026676 004537 003156
 6895 026702 005005
 6896 026704 012702 027064
 6897 026710 004737 003640
 6898 026714 002654
 6899 026716 004737 004012
 6900 026722 002664
 6901 026724
 6902 026724 104404
 6903 026726 004737 004060
 6904 026732 042737 000017 026750
 6905 026740 050537 026750
 6906 026744
 6907 026744 004537 003244
 6908 026750 010000
 6909 026752 042737 000017 026770
 6910 026760 050537 026770
 6911 026764
 6912 026764 004537 003244
 6913 026770 040420
 6914 026772
 6915 026772 004537 003244
 6916 026776 061224
 6917 027000 111237 002636
 6918 027004 116104 000004
 6919 027010 123704 002636
 6920 027014 001411
 6921 027016
 6922 027030 104455
 6923 027032 000017
 6924 027034 004754
 6925 027036 007120
 6926 027040
 6927 027040 104410
 6928 027042 000014
 6929 027044 005202
 6930 027046 005205
 6931 027050 022705 000010
 6932 027054 001324
 6933 027056
 6934 027056

BGNTST
 T54::

```

BADHEAD
:***** TEST 54 *****
:*ALU TEST
:*TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
:*ALU FUNCTION (A PLUS B PLUS C)      CODE=01
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS
BADHEAD
:***** TEST 54 *****

MYINT
MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
MSTCLR
;MASTER CLEAR MB200,4,7
;CLEAR MB200,4,7
JSR      R5, .MSTCLR
CLR      R5           ;MEM + SP ADDRESS
MOV      #5$,R2      ;POINTER TO CORRECT DATA
JSR      PC, MEMLD   ;LOAD 8 WORDS OF MAIN MEMORY
MEMDAT
;POINTER TO DATA
JSR      PC, SPLD   ;LOAD 8 WORDS OF SP
SPDAT
;POINTER TO DATA
BGNSEG
TRAP    C$BSEG
JSR      PC, CLRC   ;CLEAR C BIT!
BIC      #17,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
BIS      R5,2$     ;ADD ADDRESS TO INSTRUCTION
ROMCLK
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;CLOCK INSTRUCTION
JSR      R5, .ROMCLK
2$: 010000
;LOAD MAR
BIC      #17,3$    ;CLEAR ADDRESS OF INSTRUCTION
BIS      R5,3$     ;ADD ADDRESS TO INSTRUCTION
ROMCLK
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;CLOCK INSTRUCTION
3$: JSR      R5, .ROMCLK
;BR ADD W/C
ROMCLK
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;CLOCK INSTRUCTION
61224
;MOVE BR TO PORT4
MOV      (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
MOV      4(R1), R4   ;PUT 'FOUND' IN R4
CMP      $GDDAT, R4 ;DATA CORRECT?
BEQ      4$         ;BR IS YES
ERROR    15, YES    ;ALU ERROR
TRAP    C$ERDF
;WORD 15
;WORD EM15
;WORD ERR15
4$: ESCAPE SEG
TRAP    C$ESCAPE
;WORD 10000$-
INC      R2         ;NEXT DATA
INC      R5         ;NEXT ADDRESS
CMP      #10, R5   ;DONE YET?
BNE     1$         ;BR IF NO
ENDSEG
10000$:
  
```

6935	027056	104405					TRAP	C\$ESEG	
6936	027060						EXIT	TST	
6937	027060	104432					TRAP	C\$EXIT	
6938	027062	000012					.WORD	L10140-	
6939	027064	000	377	377	58:		.BYTE	0,-1,-1,376,252,-1,-1,124	
6940	027067	376	252	377					
6941	027072	377	124						
6942									
6943									
6944	027074					.EVEN			
6945	027074					ENDTST			
6946	027074	104401				L10140:	TRAP	C\$ETST	
6947									
6948									
6949	027076						BADHEAD		
6950							:***** TEST 55 *****		
6951							:*ALU TEST		
6952							:*TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED		
6953							:*ALU FUNCTION (A-B-C) CODE=2		
6954							:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA		
6955							:*PERFORM THE FUNCTION, VERIFY THE RESULTS		
6956	027076						BADHEAD		
6957							:***** TEST 55 *****		
6958									
6959	027076					BGNTST			
6960	027076					T55::			
6961	027076						MYINT		
6962	027076	013701	002716				MOV	KMCSR,R1	:GET DEVICE ADDRESS.

```

6963 027102          MSTCLR          ;MASTER CLEAR MB200,4,7
6964 027102 004537 003156 JSR      R5,.MSTCLR          ;CLEAR MB200,4,7
6965 027106 005005          CLR      R5          ;MEM + SP ADDRESS
6966 027110 012702 027270 MOV     #5$,R2          ;POINTER TO CORRECT DATA
6967 027114 004737 003640 JSR     PC, MEMLD       ;LOAD 8 WORDS OF MAIN MEMORY
6968 027120 002654          MEMDAT          ;POINTER TO DATA
6969 027122 004737 004012 JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
6970 027126 002664          SPDAT          ;POINTER TO DATA
6971 027130          BGNSEG
6972 027130 104404          TRAP     C$BSEG
6973 027132 004737 004060 JSR     PC,CLRC         ;CLEAR C BIT!
6974 027136 042737 000017 027154 BIC     #17,2$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
6975 027144 050537 027154 BIS     R5,2$          ;ADD ADDRESS TO INSTRUCTION
6976 027150          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6977 027150 004537 003244 JSR     R5,.ROMCLK      ;CLOCK INSTRUCTION
6978 027154 010000          010000          ;LOAD MAR
6979 027156 042737 000017 027174 BIC     #17,3$         ;CLEAR ADDRESS OF INSTRUCTION
6980 027164 050537 027174 BIS     R5,3$          ;ADD ADDRESS TO INSTRUCTION
6981 027170          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6982 027170 004537 003244 JSR     R5,.ROMCLK      ;CLOCK INSTRUCTION
6983 027174 040440 3$: 040400!<2*20> ;BR _ SUB W/C
6984 027176          ROMCLK
6985 027176 004537 003244 JSR     R5,.ROMCLK      ;CLOCK INSTRUCTION
6986 027202 061224          61224          ;MOVE BR TO PORT4
6987 027204 111237 002636 MOVB   (R2),$GDDA1      ;PUT 'EXPECTED' IN $GDDAT
6988 027210 116104 000004 MOVB   4(R1),R4        ;PUT 'FOUND' IN R4
6989 027214 123704 002636 CMPB   $GDDAT,R4      ;DATA CORRECT?
6990 027220 001411          BEQ     4$             BR IF YES
6991 027222          ERROR          ;ALU ERROR
6992 027234 104455          TRAP     C$ERDF
6993 027236 000017          .WORD   15
6994 027240 004754          .WORD   EM15
6995 027242 007120          .WORD   ERR15
6996 027244          4$: ESCAPE          SEG
6997 027244 104410          TRAP     C$ESCAPE
6998 027246 000014          .WORD   10000$-
6999 027250 005202          INC     R2          ;NEXT DATA
7000 027252 005205          INC     R5          ;NEXT ADDRESS
7001 027254 022705 000010 CMP     #10,R5        ;DONE YET?
7002 027260 001324          BNE     1$          ;BR IF NO
7003 027262          ENDSEG
7004 027262          10000$:
7005 027262 104405          TRAP     C$ESEG
7006 027264          EXIT          TST
7007 027264 104432          TRAP     C$EXIT
7008 027266 000012          .WORD   L10141-
7009 027270 377 000 376 5$: .BYTE   -1,0,376,-1,-1,252,124,-1
7010 027273 377 377 252
7011 027276 124 377
7012
7013
7014          .EVEN
7015 027300          ENDTST
7016 027300          L10141:
7017 027300 104401          TRAP     C$ETST
7018
  
```

7019
7020 027302
7021
7022
7023
7024
7025
7026
7027 027302
7028
7029
7030 027302
7031 027302
7032 027302
7033 027302 013701 002716
7034 027306
7035 027306 004537 003156
7036 027312 012702 027474
7037 027316 005005
7038 027320 004737 003640
7039 027324 002654
7040 027326 004737 004012
7041 027332 002664
7042 027334
7043 027334 104404
7044 027336 004737 004060
7045 027342 042737 000017 027360
7046 027350 050537 027360
7047 027354
7048 027354 004537 003244
7049 027360 010000
7050 027362 042737 000017 027400
7051 027370 050537 027400
7052 027374
7053 027374 004537 003244
7054 027400 040460
7055 027402
7056 027402 004537 003244
7057 027406 061224
7058 027410 111237 002636
7059 027414 116104 000004
7060 027420 123704 002636
7061 027424 001411
7062 027426
7063 027440 104455
7064 027442 000017
7065 027444 004754
7066 027446 007120
7067 027450
7068 027450 104410
7069 027452 000014
7070 027454 005202
7071 027456 005205
7072 027460 022705 000010
7073 027464 001324
7074 027466

```
BADHEAD
:***** TEST 56 *****
:*ALU TEST
:*TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
:*ALU FUNCTION (A PLUS 1) CODE=3
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS
BADHEAD
:***** TEST 56 *****

EGNTST
T56::

MYINT
MOV KCSR,R1 ;GET DEVICE ADDRESS.
MSTCLR ;MASTER CLEAR M8200,4,7
JSR R5,.MSTCLR ;CLEAR M8200,4,7
MOV #5$,R2 ;POINTER TO CORRECT DATA
CLR R5
JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMRY
MEMDAT ;POINTER TO DATA
JSR PC,SPLD ;LOAD 8 WORDS OF SP
SPDAT ;POINTER TO DATA
BGNSEG
TRAP C$BSEG
JSR PC,CLRC ;CLEAR C BIT!
BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR R5,.ROMCLK ;CLOCK INSTRUCTION
010000 ;LOAD MAR
BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR R5,.ROMCLK ;CLOCK INSTRUCTION
040400!<3*20> ;BR INC A
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR R5,.ROMCLK ;CLOCK INSTRUCTION
61224 ;MOVE BR TO PORT4
MOV# (R2),$GDDAT ;PUT 'EXPECTED' IN $GDDAT
MOV# 4(R1),R4 ;PUT 'FOUND' IN R4
CMPB $GDDAT,R4 ;DATA CORRECT?
BEQ 4$ ;BR IF YES
ERROR 15,YES ;ALU ERROR
TRAP C$ERDF
.WORD 15
.WORD EM15
.WORD ERR15
4$: ESCAPE SEG
TRAP C$ESCAPE
.WORD 10000$-
INC R2 ;NEXT DATA
INC R5
CMP #10,R5 ;DONE YET?
BNE 1$ ;BR IF NO
ENDSEG
```


7075	027466				100008:		
7076	027466	104405				TRAP	C\$ESEG
7077	027470					EXIT	TST
7078	027470	104432				TRAP	C\$EXIT
7079	027472	000012				.WORD	L10142-
7080	027474	001	001	000	58:	.BYTE	1,1,0,0,126,126,253,253
7081	027477	000	126	126			
7082	027502	253	253				
7083							
7084							
7085	027504						
7086	027504						
7087	027504	104401				TRAP	C\$ETST
7088							
7089							
7090	027506						
7091							
7092							
7093							
7094							
7095							
7096							
7097	027506						
7098							
7099							
7100	027506						
7101	027506						
7102	027506						
7103	027506	013701	002716				
7104	027512						
7105	027512	004537	003156				
7106	027516	005005					
7107	027520	012702	027700				
7108	027524	004737	003640				
7109	027530	002654					
7110	027532	004737	004012				
7111	027536	002664					
7112	027540						
7113	027540	104404					
7114	027542	004737	004060				
7115	027546	042737	000017	027564	18:		
7116	027554	050537	027564				
7117	027560						
7118	027560	004537	003244				
7119	027564	010000					
7120	027566	042737	000017	027604	28:		
7121	027574	050537	027604				
7122	027600						
7123	027600	004537	003244				
7124	027604	040520					
7125	027606						
7126	027606	004537	003244				
7127	027612	061224					
7128	027614	111237	002636				
7129	027620	116104	000004				
7130	027624	123704	002636				

.EVEN
 ENDTST
 L10142:

BADHEAD

:***** TEST 57 *****
 :*ALU TEST
 :*TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
 :*ALU FUNCTION (A PLUS A) CODE=5
 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
 BADHEAD
 :***** TEST 57 *****

BGNTST
 T57::

MYINT
 MOV KMC SR,R1 ;GET DEVICE ADDRESS.
 MSTCLR ;MASTER CLEAR DMC11
 JSR R5,.MSTCLR ;CLEAR M8200,4,7
 CLR R5 ;MEM = SP ADDRESS
 MOV #58,R2 ;POINTER TO CORRECT DATA
 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
 MEMDAT ;POINTER TO DATA
 JSR PC,SPLD ;LOAD 8 WORDS OF SP
 SPDAT ;POINTER TO DATA
 BCNSEG
 TRAP C\$BSEG
 JSR PC,CLRC ;CLEAR C BIT'
 BIC #17,28 ;CLEAR ADDRESS FIELD OF INSTRUCTION
 BIS R5,28 ;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
 010000 ;LOAD MAR
 BIC #17,38 ;CLEAR ADDRESS OF INSTRUCTION
 BIS R5,38 ;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
 040400!<5*20> ;BR 2A
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5,.ROMCLK ;CLOCK INSTRUCTION
 61224 ;MOVE BR TO PORT4
 MOVB (R2),%GDDAT ;PUT 'EXPECTED' IN %GDDAT
 MOVB 4(R1),R4 ;PUT 'FOUND' IN R4
 CMPB %GDDAT,R4 ;DATA CORRECT?

```

7131 027630 001411      BEQ      4$          ;BR IF YES
7132 027632             ERROR    15,YES     ;ALU ERROR
7133 027644 104455      TRAP    C$ERDF
7134 027646 000017      .WORD   15
7135 027650 004754      .WORD  EM15
7136 027652 007120      .WORD  ERR15
7137 027654             4$:   ESCAPE  SEG
7138 027654 104410      TRAP    C$ESCAPE
7139 027656 000014      .WORD  10000$-
7140 027660 005202      INC     R2          ;NEXT DATA
7141 027662 005205      INC     R5          ;NEXT ADDRESS
7142 027664 022705 000010  CMP     #10,R5     ;DONE YET?
7143 027670 001324      BNE     1$          ;BR IF NO
7144 027672             ENDSEG
7145 027672             10000$:
7146 027672 104405      TRAP    C$ESEG
7147 027674             EXIT    TST
7148 027674 104432      TRAP    C$EXIT
7149 027676 000012      .WORD  L10143-
7150 027700      000      000      376  5$: .BYTE  0,0,376,376,252,252,124,124
7151 027703      376      252      252
7152 027706      124      124
7153
7154             .EVEN
7155 027710             ENDTST
7156 027710             L10143:
7157 027710 104401      TRAP    C$ETST
7158
7159
7160 027712             BADHEAD
7161             ;***** TEST 58 *****
7162             ;*ALU TEST
7163             ;*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
7164             ;*ALU FUNCTION (A PLUS C)          CODE=4
7165             ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7166             ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
7167 027712             BADHEAD
7168             ;***** TEST 58 *****
7169
7170 027712             BGNTST
7171 027712             T58::
7172 027712
7173 027712 013701 002716  MYINT
7174 027716             MOV     KMCSR,R1      ;GET DEVICE ADDRESS.
7175 027716 004537 003156  MSTCLR             ;MASTER CLEAR M8200,4,7
7176 027722 005005             JSR     R5,.MSTCLR      ;CLEAR M8200,4,7
7177 027724 012702 030104  CLR     R5          ;MEM + SP ADDRESS
7178 027730 004737 00364C  MOV     #5$,R2     ;POINTER TO CORRECT DATA
7179 027734 002654             JSR     PC,MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
7180 027736 004737 004012  MEMDAT             ;POINTER TO DATA
7181 027742 002664             JSR     PC,SPLD      ;LOAD 8 WORDS OF SP
7182 027744             SPDAT             ;POINTER TO DATA
7183 027744 104404             BGNSEG
7184 027746 004737 004060  TRAP    C$BSEG
7185 027752 042737 000017 027770 1$: JSR     PC,CLRC     ;CLEAR C BIT!
7186 027760 050537 027770  BIC     #17,2$     ;CLEAR ADDRESS FIELD OF INSTRUCTION
             BIS     R5,2$      ;ADD ADDRESS TO INSTRUCTION
    
```

7187	027764					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7188	027764	004537	003244			JSR	R5, .ROMCLK	:CLOCK INSTRUCTION
7189	027770	010000			2\$:	010000		:LOAD MAR
7190	027772	042737	000017	030010		BIC	#17, 3\$:CLEAR ADDRESS OF INSTRUCTION
7191	030000	050537	030010			BIS	R5, 3\$:ADD ADDRESS TO INSTRUCTION
7192	030004					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=304
7193	030004	004537	003244			JSR	R5, .ROMCLK	:CLOCK INSTRUCTION
7194	030010	040500			3\$:	040400!	<4*20>	:BR A PLUS C
7195	030012					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCKL PC=5305
7196	030012	004537	003244			JSR	R5, .ROMCLK	:CLOCK INSTRUCTION
7197	030016	061224				61224		:MOVE BR TO PORT4
7198	030020	111237	002636			MOVB	(R2), \$GDDAT	:PUT 'EXPECTED' IN \$GDDAT
7199	030024	116104	000004			MOVB	4(R1), R4	:PUT 'FOUND' IN R4
7200	030030	123704	002636			CMPB	\$GDDAT, R4	:DATA CORRECT?
7201	030034	001411				BEQ	4\$:BR IS YES
7202	030036					ERROR	15, YES	:ALU ERROR
7203	030050	104455				TRAP	C\$ERDF	
7204	030052	000017				.WORD	15	
7205	030054	004754				.WORD	EM15	
7206	030056	007120				.WORD	ERR15	
7207	030060				4\$:	ESCAPE	SEG	
7208	030060	104410				TRAP	C\$ESCAPE	
7209	030062	000014				.WORD	10000\$-	
7210	030064	005202				INC	R2	:NEXT DATA
7211	030066	005205				INC	R5	:NEXT ADDRESS
7212	030070	022705	000010			CMP	#10, R5	:DONE YET?
7213	030074	001324				BNE	1\$:BR IF NO
7214	030076					ENDSEG		
7215	030076				10000\$:			
7216	030076	104405				TRAP	C\$ESEG	
7217	030100					EXIT	TST	
7218	030100	104432				TRAP	C\$EXIT	
7219	030102	000012				.WORD	L10144-	
7220	030104	000	000	377	5\$:	.BYTE	0,0,-1,-1,125,125,252,252	
7221	030107	377	125	125				
7222	030112	252	252					
7223								
7224						.EVEN		
7225	030114					ENDTST		
7226	030114					L10144:		
7227	030114	104401				TRAP	C\$ETST	
7228								
7229								
7230	030116					BADHEAD		
7231						:***** TEST 59 *****		
7232						:*ALU TEST		
7233						:*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED		
7234						:*ALU FUNCTION (A-B-1) CODE=17		
7235						:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA		
7236						:*PERFORM THE FUNCTION, VERIFY THE RESULTS		
7237	030116					BADHEAD		
7238						:***** TEST 59 *****		
7239								
7240	030116					BGNTST		
7241	030116					T59::		
7242	030116					MYINT		

7243	030116	013701	002716						MOV	KMCSR,R1	;GET DEVICE ADDRESS.
7244	030122								MSTCLR		;MASTER CLEAR M8200,4,7
7245	030122	004537	003156						JSR	R5,.MSTCLR	;CLEAR M8200,4,7
7246	030126	005005							CLR	R5	;MEM + SP ADDRESS
7247	030130	012702	030310						MOV	#5\$,R2	;POINTER TO CORRECT DATA
7248	030134	004737	003640						JSR	PC,MEMLD	;LOAD 8 WORDS OF MAIN MEMORY
7249	030140	002654							MEMDAT		;POINTER TO DATA
7250	030142	004737	004012						JSR	PC,SPLD	;LOAD 8 WORDS OF SP
7251	030146	002664							SPDAT		;POINTER TO DATA
7252	030150								BGNSEG		
7253	030150	104404							TRAP	C\$BSEG	
7254	030152	004737	004060						JSR	PC,CLRC	;CLEAR C BIT!
7255	030156	042737	000017	030174	1\$:				BIC	#17,2\$;CLEAR ADDRESS FIELD OF INSTRUCTION
7256	030164	050537	030174						BIS	R5,2\$;ADD ADDRESS TO INSTRUCTION
7257	030170								ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7258	030170	004537	003244						JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
7259	030174	010000								010000	;LOAD MAR
7260	030176	042737	000017	030214	2\$:				BIC	#17,3\$;CLEAR ADDRESS OF INSTRUCTION
7261	030204	050537	030214						BIS	R5,3\$;ADD ADDRESS TO INSTRUCTION
7262	030210								ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7263	030210	004537	003244						JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
7264	030214	040760			3\$:					040400!<17*20>	BR 2'S COMP SUB
7265	030216								ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7266	030216	004537	003244						JSR	R5,.ROMCLK	;CLOCK INSTRUCTION
7267	030222	061224								61224	;MOVE BR TO PORT4
7268	030224	111237	002636						MOVB	(R2), \$GDDAT	;PUT 'EXPECTED' IN \$GDDAT
7269	030230	116104	000004						MOVB	4(R1),R4	;PUT 'FOUND' IN R4
7270	030234	123704	002636						CMPB	\$GDDAT,R4	;DATA CORRECT?
7271	030240	001411							BEQ	4\$;BR IS YES
7272	030242								ERROR	15,YES	;ALU ERROR
7273	030254	104455							TRAP	C\$ERDF	
7274	030256	000017							.WORD	15	
7275	030260	004754							.WORD	EM15	
7276	030262	007120							.WORD	ERR15	
7277	030264				4\$:				ESCAPE	SEG	
7278	030264	104410							TRAP	C\$ESCAPE	
7279	030266	000014							.WORD	10000\$-	
7280	030270	005202							INC	R2	;NEXT DATA
7281	030272	005205							INC	R5	;NEXT ADDRESS
7282	030274	022705	000010						CMP	#10,R5	;DONE YET?
7283	030300	001324							BNE	1\$;BR IF NO
7284	030302								ENDSEG		
7285	030302				10000\$:						
7286	030302	104405							TRAP	C\$ESEG	
7287	030304								EXIT	TST	
7288	030304	104432							TRAP	C\$EXIT	
7289	030306	000012							.WORD	L10145-	
7290	030310	377	000	376	5\$:				.BYTE	-1,0,376,-1,-1,252,124,-1	
7291	030313	377	377	252							
7292	030316	124	377								
7293											
7294											
7295	030320								.EVEN		
7296	030320								ENDTST		
7297	030320	104401							L10145:		
7298									TRAP	C\$ETST	

```
7299
7300 030322          BADHEAD
7301                :***** TEST 60 *****
7302                :*ALU TEST
7303                :*TEST OF ALU FUNCTION DEC A WITH C BIT CLEARED
7304                :*ALU FUNCTION (A-1)   CODE=7
7305                :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7306                :*PERFORM THE FUNCTION, VERIFY THE RESULTS
7307
7308 030322          BADHEAD
7309                :***** TEST 60 *****
7310
7311 030322          BGNSTST
7312 030322          T60::
7313 030322
7314 030322 0:3701 002716 MYINT
7315 030326          MCJ      KMCSR,R1      ;GET DEVICE ADDRESS.
7316 030326 004537 003156 MSTCLR          ;MASTER CLEAR DMC11
7317 030332          JSR      R5,.MSTCLR      ;CLEAR M8200,4,7
7318 030334 012702 030514 CLR      R5      ;MEM + SP ADDRESS
7319 030340 004737 003640 MOV      #5,R2     ;POINTER TO CORRECT DATA
7320 030344 002654          JSR      PC,MEMLD   ;LOAD 8 WORDS OF MAIN MEMMOR
7321 030346 004737 004012 MEMDAT          ;POINTER TO DATA
7322 030352 002664          JSR      PC,SPLD   ;LOAD 8 WORDS OF SP
7323 030354          SPDAT          ;POINTER TO DATA
7324 030354 104404          BGNSEG
7325 030356 004737 004060 TRAP      C$BSEG
7326 030362 042737 000017 030400 1$: JSR      PC,CLRC   ;CLEAR C BIT!
7327 030370 050537 030400 BIC      #17,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
7328 030374          BIS      R5,2$          ;ADD ADDRESS TO INSTRUCTION
7329 030374 004537 003244 ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7330 030400 010000          JSR      R5,.ROMCLK   ;CLOCK INSTRUCTION
7331 030402 042737 000017 030420 2$: JSR      PC,CLRC   ;LOAD MAR
7332 030410 050537 030420 BIC      #17,3$    ;CLEAR ADDRESS OF INSTRUCTION
7333 030414          BIS      R5,3$          ;ADD ADDRESS TO INSTRUCTION
7334 030414 004537 003244 ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7335 030420 040560          JSR      R5,.ROMCLK   ;CLOCK INSTRUCTION
7336 030422          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7337 030422 004537 003244 JSR      R5,.ROMCLK   ;CLOCK INSTRUCTION
7338 030426 061224          JSR      R5,.ROMCLK   ;MOVE BR TO PORT4
7339 030430 111237 002636 MOVB     (R2),%GDDAT ;PUT "EXPECTED" IN %GDDAT
7340 030434 116104 000004 MOVB     4(R1),R4    ;PUT "FOUND" IN R4
7341 030440 123704 002636 CMPB     %GDDAT,R4   ;DATA CORRECT?
7342 030444 001411          BEQ      4$          ;BR IF YES
7343 030446          ERROR          ;ALU ERROR
7344 030460 104455          TRAP      C$ERDF
7345 030462 000017          .WORD     15
7346 030464 004754          .WORD     EM15
7347 030466 007120          .WORD     ERR15
7348 030470          4$: ESCAPE          SEG
7349 030470 104410          TRAP      C$ESCAPE
7350 030472 000014          .WORD     10000$-.
7351 030 74 005202          INC      R2          ;NEXT DATA
7352 030476 005205          INC      R5          ;NEXT ADDRESS
7353 030500 022705 000010 CMP      #10,R5     ;DONE YET?
7354 030504 001324          BNE     1$          ;BR IF NO
```

```

7355 030506          ENDSEG
7356 030506          10000$:
7357 030506 104405   TRAP   C$ESEG
7358 030510          EXIT   TST
7359 030510 104432   TRAP   C$EXIT
7360 030512 000012   .WORD  L10146-
7361 030514   377   377   376 5$: .BYTE  -1,-1,376,376,124,124,251,251
7362 030517   376   124   124
7363 030522   251   251
7364
7365          .EVEN
7366 030524          ENDTST
7367 030524          L10146:
7368 030524 104401   TRAP   C$ETST
7369
7370
7371 030526          BADHEAD
7372          :***** TEST 61 *****
7373          :*ALU TEST
7374          :*TEST OF ALU FUNCTION SEL B WITH C BIT SET
7375          :*ALU FUNCTION (B)      CODE=11
7376          :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7377          :*PERFORM THE FUNCTION, VERIFY THE RESULTS
7378 030526          BADHEAD
7379          :***** TEST 61 *****
7380
7381 030526          BGNTST
7382 030526          T61::
7383 030526          MYINT
7384 030526 013701 002716  MOV   KMCSR,R1      ;GET DEVICE ADDRESS.
7385 030532          MSTCLR          ;MASTER CLEAR M8200,4,7
7386 030532 004537 003156  JSR   R5,.,MSTCLR   ;CLEAR M8200,4,7
7387 030536 005005          CLR   R5            ;MEM + SP ADDRESS
7388 030540 012702 030720  MOV   #5$,R2        ;POINTER TO CORRECT DATA
7389 030544 004737 003640  JSR   PC,MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
7390 030550 002654          MEMDAT          ;POINTER TO DATA
7391 030552 004737 004012  JSR   PC,SPLD        ;LOAD 8 WORDS OF SP
7392 030556 002664          SPDAT          ;POINTET TO DATA
7393 030560          BGNSEG
7394 030560 104404          TRAP   C$BSEG
7395 030562 004737 004076  JSR   PC,SETC        ;SET C BIT
7396 030566 042737 000017 030604 BIC   #17,2$        ;CLEAR ADDRESS FIELD OF INSTRUCTION
7397 030574 050537 030604  BIS   R5,2$         ;ADD ADDRESS TO INSTRUCTION
7398 030600          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7399 030600 004537 003244  JSR   R5,.,ROMCLK   ;CLOCK INSTRUCTION
7400 030604 010000          2$: 010000          ;LOAD MAR
7401 030606 042737 000017 030624 BIC   #17,3$        ;CLEAR ADDRESS OF INSTRUCTION
7402 030614 050537 030624  BIS   R5,3$         ;ADD ADDRESS TO INSTRUCTION
7403 030620          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7404 030620 004537 003244  JSR   R5,.,ROMCLK   ;CLOCK INSTRUCTION
7405 030624 040620          3$: 040400!<11*20> ;BR   SEL B
7406 030626          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7407 030626 004537 003244  JSR   R5,.,ROMCLK   ;CLOCK INSTRUCTION
7408 030632 061224          61224          ;MOVE BR TO PORT4
7409 030634 111237 002636  MOVB  (R2),%GDDAT   ;PUT 'EXPECTED' IN %GDDAT
7410 030640 116104 000004  MOVB  4(R1),R4      ;PUT 'FOUND' IN R4
  
```

```

7411 030644 123704 002636      CMPB   $GDDA',R4      ;DATA CORRECT?
7412 030650 001411              BEQ    4$              ;BR IF YES
7413 030652                      ERROR  23,YES         ;ALU ERROR
7414 030664 104455              TRAP   C$ERDF
7415 030666 000027              .WORD  23
7416 030670 005226              .WORD  EM23
7417 030672 007404              .WORD  ERR23
7418 030674                      4$:  ESCAPE  SEG
7419 030674 104410              TRAP   C$ESCAPE
7420 030676 000014              .WORD  10000$-
7421 030700 005202              INC    R2              ;NEXT DATA
7422 030702 005205              INC    R5              ;NEXT ADDRESS
7423 030704 022705 000010      CMP    #10,R5         ;DONE YET?
7424 030710 001324              BNE    1$              ;BR IF NO
7425 030712                      ENDSEG
7426 030712                      10000$:
7427 030712 104405              TRAP   C$ESEG
7428 030714                      EXIT   TST
7429 030714 104432              TRAP   C$EXIT
7430 030716 000012              .WORD  L10147-
7431 030720      000      377      000  5$: .BYTE  0,-1,0,-1,125,252,125,252
7432 030723      377      125      252
7433 030726      125      252
7434
7435                      .EVEN
7436 030730                      ENDTST
7437 030730                      L10147:
7438 030730 104401              TRAP   C$ETST
7439
7440
7441 030732                      BADHEAD
7442                      ;***** TEST 62 *****
7443                      ;*ALU TEST
7444                      ;*TEST OF ALU FUNCTION SEL A WITH C BIT SET
7445                      ;*ALU FUNCTION (A)      CODE=10
7446                      ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7447                      ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
7448 030732                      BADHEAD
7449                      ;***** TEST 62 *****
  
```



```

7506
7507 031134 .EVEN
7508 031134 ENDTST
7509 031134 104401 L10150:
7510 TRAP CSETST
7511
7512 031136 BADHEAD
7513 :***** TEST 63 *****
7514 :*ALU TEST
7515 :*TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
7516 :*ALU FUNCTION (A OR NOTB) CODE=12
7517 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7518 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
7519 031136 BADHEAD
7520 :***** TEST 63 *****
7521
7522 031136 BGNTST
7523 031136 T63::
7524 031136
7525 031136 013701 002716 MYINT
7526 031142 MOV KMC? ,R1 ;GET DEVICE ADDRESS.
7527 031142 004537 003156 MSTCLR ;MASTER CLEAR M8200,4,7
7528 031146 005005 JSR R5, .MSTCLR ;CLEAR M8200,4,7
7529 031150 012702 031330 CLR R5 ;MEM + SP ADDRESS
7530 031154 004737 003640 MOV #5$,R2 ;POINTER TO CORRECT DATA
7531 031160 002654 JSR PC, MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
7532 031162 004737 004012 MEMDAT ;POINTER TO DATA
7533 031166 002664 JSR PC, SPLD ;LOAD 8 WORDS OF SP
7534 031170 SPDAT ;POINTER TO DATA
7535 031170 104404 BGNSEG
7536 031172 004737 004076 TRAP C$BSEG
7537 031176 042737 000017 031214 1$: JSR PC, SETC ;SET C BIT!
7538 031204 050537 031214 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
7539 031210 ROMCLK ;ADD ADDRESS TO INSTRUCTION
7540 031210 004537 003244 JSR R5, .ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7541 031214 010000 031234 2$: ;CLOCK INSTRUCTION
7542 031216 042737 000017 BIC #17,3$ ;LOAD MAR
7543 031224 050537 031234 BIS R5,3$ ;CLEAR ADDRESS OF INSTRUCTION
7544 031230 ROMCLK ;ADD ADDRESS TO INSTRUCTION
7545 031230 004537 003244 JSR R5, .ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7546 031234 040640 3$: ;CLOCK INSTRUCTION
7547 031236 ROMCLK ;BR A OR NOTB
7548 031236 004537 003244 JSR R5, .ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7549 031242 061224 61224 ;CLOCK INSTRUCTION
7550 031244 111237 002636 MOVB (R2), $GDDAT ;MOVE BR TO PORT4
7551 031250 116104 000004 MOVB 4(R1), R4 ;PUT 'EXPECTED' IN $GDDAT
7552 031254 123704 002636 CMPB $GDDAT, R4 ;PUT 'FOUND' IN R4
7553 031260 001411 BEQ 4$ ;DATA CORRECT?
7554 031262 ERROR 15, YES ;BR IF YES
7555 031274 104455 TRAP C$ERDF ;ALU ERROR
7556 031276 000017 .WORD 15
7557 031300 004754 .WORD EM15
7558 031302 007120 .WORD ERR15
7559 031304 4$: ESCAPE SEG
7560 031304 104410 TRAP C$ESCAPE
7561 031306 000014 .WORD 10000$-
  
```

```

7562 031310 005202          INC      R2          ;NEXT DATA
7563 031312 005205          INC      R5          ;NEXT ADDRESS
7564 031314 022705 000010  CMP      #10,R5     ;DONE YET?
7565 031320 001324          BNE     1$          ;BR IF NO
7566 031322                ENDSEG
7567 031322                10000$:
7568 031322 104405          TRAP    C$ESEG
7569 031324                EXIT    TST
7570 031324 104432          TRAP    C$EXIT
7571 031326 000012          .WORD  L10151-
7572 031330          377      000      377  5$:  .BYTE  -1,0,-1,-1,-1,125,252,-1
7573 031333          377      377      125
7574 031336          252      377
7575
7576                .EVEN
7577 031340                ENDTST
7578 031340                L10151:
7579 031340 104401          TRAP    C$ETST
7580
7581
7582 031342                BADHEAD
7583                ;***** TEST 64 *****
7584                ;*ALU TEST
7585                ;*TEST OF ALU FUNCTION A AND B WITH C BIT SET
7586                ;ALU FUNCTION (A AND B) CODE=13
7587                ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7588                ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
7589 031342                BADHEAD
7590                ;***** TEST 64 *****
7591
7592 031342                BGNTST
7593 031342                T64::
7594 031342
7595 031342 013701 002716  MYINT
7596 031346                MOV     KMCSR,R1    ;GET DEVICE ADDRESS.
7597 031346 004537 003156  MSTCLR          ;MASTER CLEAR M8200,4,7
7598 031352 005005                JSR    R5,.MSTCLR ;CLEAR M8200,4,7
7599 031354 012702 031534  CLR     R5          ;MEM + SP ADDRESS
7600 031360 004737 003640  MOV     #5$,R2     ;POINTER TO CORRECT ADDRESS
7601 031364 002654                JSR    PC,MEMLD   ;LOAD 8 WORDS OF MAIN MEMORY
7602 031366 004737 004012  MEMDAT          ;POINTER TO DATA
7603 031372 002664                JSR    PC,SPLD   ;LOAD 8 WORDS OF SP
7604 031374                SPDAT          ;POINTER TO DATA
7605 031374 104404          BGNSEG
7606 031376 004737 004076  TRAP    C$BSEG
7607 031402 042737 000017 031420 1$:  JSR    PC,SETC   ;SET C BIT!
7608 031410 050537 031420  BIC     #17,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
7609 031414                BIS     R5,2$     ;ADD ADDRESS TO INSTRUCTION
7610 031414 004537 003244  ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7611 031420 010000                JSR    R5,.ROMCLK ;CLOCK INSTRUCTION
7612 031422 042737 000017 031440 2$:  010000          ;LOAD MAR
7613 031430 050537 031440  BIC     #17,3$    ;CLEAR ADDRESS OF INSTRUCTION
7614 031434                BIS     R5,3$     ;ADD ADDRESS TO INSTRUCTION
7615 031434 004537 003244  ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7616 031440 040660                JSR    R5,.ROMCLK ;CLOCK INSTRUCTION
7617 031442                040400!<13*20> ;BR A AND B
                                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  
```

```

7618 031442 004537 003244      JSR      R5, .ROMCLK      ;CLOCK INSTRUCTION
7619 031446 061224              61224      ;MOVE BR TO PORT4
7620 031450 111237 002636      MOVB     (R2), $GDDAT     ;PUT 'EXPECTED' IN $GDDAT
7621 031454 116104 000004      MOVB     4(R1), R4       ;PUT 'FOUND' IN R4
7622 031460 123704 002636      CMPB     $GDDAT, R4      ;DATA CORRECT?
7623 031464 001411              BEQ      4$              ;BR IF YES
7624 031466              ERROR    23, YES        ;ALU ERROR
7625 031500 104455              TRAP     C$ERDF
7626 031502 000027              .WORD    23
7627 031504 005226              .WORD    EM23
7628 031506 007404              .WORD    ERR23
7629 031510              4$:      ESCAPE    SEG
7630 031510 104410              TRAP     C$ESCAPE
7631 031512 000014              .WORD    10000$-
7632 031514 005202              INC      R2              ;NEXT DATA
7633 031516 005205              INC      R5              ;NEXT ADDRESS
7634 031520 022705 000010      CMP      #10, R5        ;DONE YET?
7635 031524 001324              BNE     1$              ;BR IF NO
7636 031526              ENDSEG
7637 031526              10000$:
7638 031526 104405              TRAP     C$ESEG
7639 031530              EXIT     TST
7640 031530 104432              TRAP     C$EXIT
7641 031532 000012              .WORD    L10152-
7642 031534      000      000      000 5$:      .BYTE    0,0,0,-1,125,0,0,252
7643 031537      377      125      000
7644 031542      000      252
7645
7646              .EVEN
7647 031544              ENDTST
7648 031544              L10152:
7649 031544 104401              TRAP     C$ETST
7650
7651
7652 031546              BADHEAD
7653              ;***** TEST 65 *****
7654              ;*ALU TEST
7655              ;*TEST OF ALU FUNCTION A OR B WITH C BIT SET
7656              ;*ALU FUNCTION (A OR B) CODE=14
7657              ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7658              ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
7659 031546              BADHEAD
7660              ;***** TEST 65 *****
7661
7662 031546              BGNTST
7663 031546              T65::
7664 031546              MYINT
7665 031546 013701 002716      MOV      KMCSR, R1      ;GET DEVICE ADDRESS.
7666 031552              MSTCLR      ;MASTER CLEAR MB200,4,7
7667 031552 004537 003156      JSR      R5, .MSTCLR    ;CLEAR MB200,4,7
7668 031556 005005              CLR      R5              ;MEM + SP ADDRESS
7669 031560 012702 031740      MOV      #5$, R2        ;POINTER TO CORRECT DATA
7670 031564 004737 003640      JSR      PC, MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
7671 031570 002654              MEMDAT      ;POINTER TO DATA
7672 031572 004737 004012      JSR      PC, SPLD       ;LOAD 8 WORDS OF SP
7673 031576 002664              SPDAT      ;POINTER TO DATA

```

```

7674 031600
7675 031600 104404
7676 031602 004737 004076
7677 031606 042737 000017 031624 1$:
7678 031614 050537 031624 BIC #17,2$ ;SET C BIT!
7679 031620 ROMCLK ;CLEAR ADDRESS FIELD OF INSTRUCTION
7680 031620 004537 003244 JSR R5,.ROMCLK ;ADD ADDRESS TO INSTRUCTION
7681 031624 010000 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7682 031626 042737 000017 031644 2$: ;CLOCK INSTRUCTION
7683 031634 050537 031644 BIC #17,3$ ;LOAD MAR
7684 031640 ROMCLK ;CLEAR ADDRESS OF INSTRUCTION
7685 031640 004537 003244 JSR R5,.ROMCLK ;ADD ADDRESS TO INSTRUCTION
7686 031644 040700 3$: ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7687 031646 ROMCLK ;CLOCK INSTRUCTION
7688 031646 004537 003244 JSR R5,.ROMCLK ;BR A OR B
7689 031652 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7690 031654 111237 002636 MOVB (R2),$GDDAT ;CLOCK INSTRUCTION
7691 031660 116104 000004 MOVB 4(R1),R4 ;MOVE BR TO PORT4
7692 031664 123704 002636 CMPB $GDDAT,R4 ;PUT 'EXPECTED' IN R4
7693 031670 001411 BEQ 4$ ;PUT 'FOUND' IN R4
7694 031672 ERROR 23,YES ;DATA CORRECT?
7695 031704 104455 TRAP C$ERDF ;BR IF YES
7696 031706 000027 .WORD 23 ;ALU ERROR
7697 031710 005226 .WORD EM23
7698 031712 007404 .WORD ERR23
7699 031714 4$: ESCAPE SEG
7700 031714 104410 TRAP C$ESCAPE
7701 031716 000014 .WORD 10000$-.
7702 031720 005202 INC R2 ;NEXT DATA
7703 031722 005205 INC R5 ;NEXT ADDRESS
7704 031724 022705 000010 CMP #10,R5 ;DONE YET?
7705 031730 001324 BNE 1$ ;BR IF NO
7706 031732 ENDSEG
7707 031732 10000$:
7708 031732 104405 TRAP C$ESEG
7709 031734 EXIT TST
7710 031734 104432 TRAP C$EXIT
7711 031736 000012 .WORD L10153-.
7712 031740 000 377 377 5$: .BYTE 0,-1,-1,-1,125,-1,-1,252
7713 031743 377 125 377
7714 031746 377 252
7715
7716 .EVEN
7717 031750 ENDTST
7718 031750 L10153:
7719 031750 104401 TRAP C$ETST
7720
7721
7722 031752 BADHEAD
7723 :***** TEST 66 *****
7724 :*ALU TEST
7725 :*TEST OF ALU FUNCTION A XOR B WITH C BIT SET
7726 :*ALU FUNCTION (A XOR B) CODE=15
7727 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7728 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
7729

```

```

7730 031752                                BADHEAD
7731                                         :***** TEST 66 *****
7732 031752                                BGNTST
7733 031752                                T66::
7734 031752
7735 031752 013701 002716                   MYINT
7736 031756                                MOV      KMCSR,R1          ;GET DEVICE ADDRESS.
7737 031756 004537 003156                   MSTCLR                                ;MASTER CLEAR M8200,4,7
7738 031762 005005                                JSR      R5,.MSTCLR        ;CLEAR M8200,4,7
7739 031764 012702 032144                   CLR      R5                ;MEM + SP ADDRESS
7740 031770 004737 003640                   MOV      #5$,R2           ;POINTER TO CORRECT DATA
7741 031774 002654                                JSR      PC,MEMLD         ;LOAD 8 WORDS OF MAIN MEMORY
7742 031776 004737 004012                   MEMDAT                                ;POINTER TO DATA
7743 032002 002664                                JSR      PC,SPLD         ;LOAD 8 WORDS OF SP
7744 032004                                SPDAT                                ;POINTER TO DATA
7745 032004 104404                                BGNSEG
7746 032006 004737 004076                   TRAP     C$BSEG
7747 032012 042737 000017 032030 1$:      JSR      PC,SETC          ;SET C BIT!
7748 032020 050537 032030                   BIC      #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
7749 032024                                BIS      R5,2$           ;ADD ADDRESS TO INSTRUCTION
7750 032024 004537 003244                   ROMCLK                                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7751 032030 010000                                JSR      R5,.ROMCLK      ;CLOCK INSTRUCTION
7752 032032 042737 000017 032050 2$:      010000                    ;LOAD MAR
7753 032040 050537 032050                   BIC      #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
7754 032044                                BIS      R5,3$           ;ADD ADDRESS TO INSTRUCTION
7755 032044 004537 003244                   ROMCLK                                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7756 032050 040720                                JSR      R5,.ROMCLK      ;CLOCK INSTRUCTION
7757 032052 040400!<15*20> 3$:      ROMCLK                    ;BR A XOR B
7758 032052 004537 003244                   JSR      R5,.ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7759 032056 061224                                61224                    ;CLOCK INSTRUCTION
7760 032060 111237 002636                   MOVB     (R2), $GDDAT     ;MOVE BR TO PORT
7761 032064 116104 000004                   MOVB     4(R1), R4        ;PUT "EXPECTED" IN $GDDAT
7762 032070 123704 002636                   CMPB     $GDDAT, R4      ;PUT "FOUND" IN R4
7763 032074 001411                                BEQ      4$              ;DATA CORRECT?
7764 032076                                ERROR    23, YES         ;BR IF YES
7765 032110 104455                                TRAP     C$ERDF          ;ALU ERROR
7766 032112 000027                                .WORD   23
7767 032114 005226                                .WORD   EM23
7768 032116 007404                                .WORD   ERR23
7769 032120                                4$:      ESCAPE  SEG
7770 032120 104410                                TRAP     C$ESCAPE
7771 032122 000014                                .WORD   10000$-
7772 032124 005202                                INC      R2              ;NEXT DATA
7773 032126 005205                                INC      R5              ;NEXT ADDRESS
7774 032130 022705 000010                   CMP      #10, R5         ;DONE YET?
7775 032134 001324                                BNE     1$              ;BR IF NO
7776 032136                                10000$:
7777 032136                                TRAP     C$ESEG
7778 032136 104405                                EXIT     TST
7779 032140                                TRAP     C$EXIT
7780 032140 104432                                .WORD   L10154-
7781 032142 000012                                .BYTE  0,-1,-1,0,0,-1,-1,0
7782 032144 000 377 377 5$:
7783 032147 000 000 377
7784 032152 377 000
7785

```

```
7786  
7787 032154 .EVEN  
7788 032154 ENDTST  
7789 032154 104401 L10154:  
7790 TRAP C$ETST  
7791  
7792 032156 BADHEAD  
7793 :***** TEST 67 *****  
7794 :*ALU TEST  
7795 :*TEST OF ALU FUNCTION ADD WITH C BIT SET  
7796 :*ALU FUNCTION (A PLUS B) CODE=00  
7797 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA  
7798 :*PERFORM THE FUNCTION, VERIFY THE RESULTS  
7799 032156 BADHEAD  
7800 :***** TEST 67 *****  
7801  
7802 032156 BGNTST  
7803 032156 T67::  
7804 032156 MYINT  
7805 032156 013701 002716 MOV KMCSR,R1 ;GET DEVICE ADDRESS.  
7806 032162 MSTCLR ;MASTER CLEAR M8200,4,7  
7807 032162 004537 003156 JSR R5, MSTCLR ;CLEAR M8200,4,7  
7808 032166 C05005 CLR R5 ;MEM + SP ADDRESS  
7809 032170 012702 032350 MOV #5$,R2 ;POINTER TO CORRECT DATA  
7810 032174 004737 003640 JSR PC, MEMLD ;LOAD 8 WORDS OF MAIN MEMORY  
7811 032200 002654 MEMDAT ;POINTER TO DATA  
7812 032202 004737 004012 JSR PC, SPLD ;LOAD 8 WORDS OF SP  
7813 032206 002664 SPDAT ;POINTER TO DATA  
7814 032210 BGNSEG  
7815 032210 104404 TRAP C$BSEG  
7816 032212 004737 004076 1$: JSR PC, SETC ;SET C BIT!  
7817 032216 042737 000017 032234 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION  
7818 032224 050537 032234 BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION  
7819 032230 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
7820 032230 004537 003244 JSR R5, ROMCLK ;CLOCK INSTRUCTION  
7821 032234 010000 2$: 010000 ;LOAD MAR  
7822 032236 042737 000017 032254 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION  
7823 032244 050537 032254 BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION  
7824 032250 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
7825 032250 004537 003244 JSR R5, ROMCLK ;CLOCK INSTRUCTION  
7826 032254 040400 3$: 040400: <00*20> ;BR ADD  
7827 032256 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
7828 032256 004537 003244 JSR R5, ROMCLK ;CLOCK INSTRUCTION  
7829 032262 061224 61224 ;MOVE BR TO PORT4  
7830 032264 111237 002636 MOVB (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT  
7831 032270 116104 000004 MOVB 4(R1), R4 ;PUT 'FOUND' IN R4  
7832 032274 123704 002636 CMPB $GDDAT, R4 ;DATA CORRECT?  
7833 032300 001411 BEQ 4$ ;BR IF YES  
7834 032302 ERROR 23, YES ;ALU ERROR  
7835 032314 104455 TRAP C$ERDF  
7836 032316 000027 .WORD 23  
7837 032320 005226 .WORD EM23  
7838 032322 007404 .WORD ERR23  
7839 032324 4$: ESCAPE SEG  
7840 032324 104410 TRAP C$ESCAPE  
7841 032326 000014 .WORD 10000$-
```

```

7842 032330 005202          INC    R2          ;NEXT DATA
7843 032332 005205          INC    R5          ;NEXT ADDRESS
7844 032334 022705 000010  CMP    #10,R5     ;DONE YET?
7845 032340 001324          BNE    1$         ;BR IF NO
7846 032342                ENDSEG
7847 032342                10000$:
7848 032342 104405          TRAP   C$ESEG
7849 032344                EXIT    TST
7850 032344 104432          TRAP   C$EXIT
7851 032346 000012          .WORD  L10155-
7852 032350          000    377    377  5$: .BYTE  0,-1,-1,376,252,-1,-1,124
7853 032353          376    252    377
7854 032356          377    124
7855
7856                .EVEN
7857 032360                ENDTST
7858 032360                L10155:
7859 032360 104401          TRAP   C$ETST
7860
7861
7862 032362                BADHEAD
7863                ;***** TEST 68 *****
7864                ;*ALU TEST
7865                ;*TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
7866                ;*ALU FUNCTION (A PLUS A PLUS C)          CODE=6
7867                ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
7868                ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
7869 032362                BADHEAD
7870                ;***** TEST 68 *****
7871
7872 032362                BGNTST
7873 032362                T68::
7874 032362                MYINT
7875 032362 013701 002716  MOV    KMCSR,R1    ;GET DEVICE ADDRESS.
7876 032366                MSTCLR    ;MASTER CLEAR M8200,4,7
7877 032366 004537 003156  JSR    R5,.MSTCLR ;CLEAR M8200,4,7
7878 032372 005005                CLR    R5          ;MEM + SP ADDRESS
7879 032374 012702 032554  MOV    #5$,R2     ;POINTER TO CORRECT DATA
7880 032400 004737 003640  JSR    PC,MEMLD   ;LOAD 8 WORDS OF MAIN MEMORY
7881 032404 002654                MEMDAT            ;POINTER TO DATA
7882 032406 004737 004012  JSR    PC,SPLD    ;LOAD 8 WORDS OF SP
7883 032412 002664                SPDAT            ;POINTER TO DATA
7884 032414
7885 032414 104404          TRAP   C$BSEG
7886 032416 004737 004076  JSR    PC,SETC    ;SET C BIT!
7887 032422 042737 000017 032440 1$: BIC    #17,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
7888 032430 050537 032440  BIS    R5,2$     ;ADD ADDRESS TO INSTRUCTION
7889 032434                ROMCLK
7890 032434 004537 003244  JSR    R5,.ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7891 032440 010000                ;CLOCK INSTRUCTION
7892 032442 042737 000017 032460 2$: BIC    #17,3$    ;LOAD MAR
7893 032450 050537 032460  BIS    R5,3$     ;CLEAR ADDRESS OF INSTRUCTION
7894 032454                ROMCLK            ;ADD ADDRESS TO INSTRUCTION
7895 032454 004537 003244  JSR    R5,.ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=55304
7896 032460 040540                ;CLOCK INSTRUCTION
7897 032462                ROMCLK            ;BR 2A W/C
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```



```

7954 032620          BGNSEG
7955 032620 104404   TRAP    C$BSEG
7956 032622 004737 004076 032644 1$:   JSR    PC,SETC      ;SET C BIT.
7957 032626 042737 000017 032644   BIC    #17,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
7958 032634 050537 032644   BIS    R5,2$       ;ADD ADDRESS TO INSTRUCTION
7959 032640          ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7960 032640 004537 003244   JSR    R5,.ROMCLK ;CLOCK INSTRUCTION
7961 032644 010000          2$:   010000   ;LOAD MAR
7962 032646 042737 000017 032664   BIC    #17,3$      ;CLEAR ADDRESS OF INSTRUCTION
7963 032654 050537 032664   BIS    R5,3$       ;ADD ADDRESS TO INSTRUCTION
7964 032660          ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7965 032660 004537 003244   JSR    R5,.ROMCLK ;CLOCK INSTRUCTION
7966 032664 040740          3$:   040400!<16*20> ;BR SUB
7967 032666          ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
7968 032666 004537 003244   JSR    R5,.ROMCLK ;CLOCK INSTRUCTION
7969 032672 061224          61224   ;MOVE BR TO PORT4
7970 032674 111237 002636   MOVB   (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
7971 032700 116104 000004   MOVB   4(R1), R4    ;PUT 'FOUND' IN R4
7972 032704 123704 002636   CMPB   $GDDAT, R4   ;DATA CORRECT?
7973 032710 001411          BEQ    4$           ;BR IF YES
7974 032712          ERROR    23, YES ;ALU ERROR
7975 032724 104455          TRAP    C$ERDF
7976 032726 000027          .WORD  23
7977 032730 005226          .WORD  EM23
7978 032732 007404          .WORD  ERR23
7979 032734          4$:   ESCAPE  SEG
7980 032734 104410          TRAP    C$ESCAPE
7981 032736 000014          .WORD  10000$-.
7982 032740 005202          INC    R2           ;NEXT DATA
7983 032742 005205          INC    R5           ;NEXT ADDRESS
7984 032744 022705 000010   CMP    #10, R5      ;DONE YET?
7985 032750 001324          BNE    1$           ;BR IF NO
7986 032752          ENDSEG
7987 032752          10000$:
7988 032752 104405          TRAP    C$ESEG
7989 032754          EXIT    TST
7990 032754 104432          TRAP    C$EXIT
7991 032756 0000i2          .WORD  L10157-.
7992 032760 000 001 377 5$:   .BYTE  0,1,-1,0,0,253,125,0
7993 032763 000 000 253
7994 032766 125 000
7995
7996          .EVEN
7997 032770          ENDTST
7998 032770          L10157:
7999 032770 104401          TRAP    C$ETST
8000
8001
8002 032772          BADHEAD
8003          ;***** TEST 70 *****
8004          ;*ALU TEST
8005          ;*TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
8006          ;*ALU FUNCTION (A PLUS B PLUS C) CODE-01
8007          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
8008          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
8009 032772          BADHEAD
  
```

```

8010                                     :***** TEST 70 *****
8011
8012 032772                               BGNTST
8013 032772                               T70::
8014 032772
8015 032772 013701 002716                 MYINT
8016 032776                               MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
8017 032776 004537 003156                 MSTCLR                               ;MASTER CLEAR M8200,4,7
8018 033002 005005                               JSR      R5,.MSTCLR   ;CLEAR M8200,4,7
8019 033004 012702 033164                 CLR      R5           ;MEM +SP ADDRESS
8020 033010 004737 003640                 MOV      #5$,R2      ;POINTER TO CORRECT DATA
8021 033014 002654                               JSR      PC,MEMLD    ;LOAD 8 WORDS OF MAIN MEMORY
8022 033016 004737 004012                 MEMDAT                               ;POINTER TO DATA
8023 033022 002664                               JSR      PC,SPLD     ;LOAD 8 WORDS OF SP
8024 033024                               SPDAT                               ;POINTER TO DATA
8025 033024 104404                               BGNSEG
8026 033026 004737 004076                 TRAP     C$BSEG
8027 033032 042737 000017 033050 1$:     JSR      PC,SETC     ;SET C BIT!
8028 033040 050537 033050                 BIC      #17,2$     ;CLEAR ADDRESS FIELD OF INSTRUCTION
8029 033044                               BIS      R5,2$      ;ADD ADDRESS TO INSTRUCTION
8030 033044 004537 003244                 ROMCLK                               ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8031 033050 010000                               JSR      R5,.ROMCLK ;CLOCK INSTRUCTION
8032 033052 042737 000017 033070 2$:     010000                               ;LOAD MAR
8033 033060 050537 033070                 BIC      #17,3$     ;CLEAR ADDRESS OF INSTRUCTION
8034 033064                               BIS      R5,3$      ;ADD ADDRESS TO INSTRUCTION
8035 033064 004537 003244                 ROMCLK                               ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8036 033070 040420                               JSR      R5,.ROMCLK ;CLOCK INSTRUCTION
8037 033072 004537 003244                 040400!<01*20> 3$:                               ;RR - ADD W/C
8038 033072 004537 003244                 ROMCLK                               ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8039 033076 061224                               JSR      R5,.ROMCLK ;CLOCK INSTRUCTION
8040 033100 111237 002636                 61224                               ;MOVE BR TO PORT4
8041 033104 116104 000004                 MOVB     (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
8042 033110 123704 002636                 MOVB     4(R1), R4   ;PUT 'FOUND' IN R4
8043 033114 001411                               CMPB     $GDDAT, R4 ;DATA CORRECT?
8044 033116                               BEQ      4$          ;BR IF YES
8045 033130 104455                               ERROR   23,YES      ;ALU ERROR
8046 033132 000027                               TRAP     C$ERDF
8047 033134 005226                               .WORD   23
8048 033136 007404                               .WORD   EM23
8049 033140                               .WORD   ERR23
8050 033140 104410                               4$:     ESCAPE   SEG
8051 033142 000014                               TRAP     C$ESCAPE
8052 033144 005202                               .WORD   10000$-
8053 033146 005205                               INC      R2          ;NEXT DATA
8054 033150 022705 000010                 INC      R5          ;NEXT ADDRESS
8055 033154 001324                               INC      R5
8056 033156                               CMP      #10,R5     ;DONE YET?
8057 033156                               BNE     1$          ;BR IF NO
8058 033156 104405                               10000$: TRAP     C$ESEG
8059 033160                               EXIT     TST
8060 033160 104432                               TRAP     C$EXIT
8061 033162 000012                               .WORD   L10160-
8062 033164 001 000 000 5$:             .BYTE  1,0,0,-1,253,0,0,125
8063 033167 377 253 000
8064 033172 000 125
8065

```

8066
 8067 033174
 8068 033174
 8069 033174 104401
 8070
 8071
 8072 033176
 8073
 8074
 8075
 8076
 8077
 8078
 8079 033176
 8080
 8081
 8082
 8083 033176
 8084 033176
 8085 033176
 8086 033176 013701 002716
 8087 033202
 8088 033202 004537 003156
 8089 033206 005005
 8090 033210 012702 033370
 8091 033214 004737 003640
 8092 033220 002654
 8093 033222 004737 004012
 8094 033226 002664
 8095 033230
 8096 033230 104404
 8097 033232 004737 004076
 8098 033236 042737 000017 033254
 8099 033244 050537 033254
 8100 033250
 8101 033250 004537 003244
 8102 033254 010000
 8103 033256 042737 000017 033274
 8104 033264 050537 033274
 8105 033270
 8106 033270 004537 003244
 8107 033274 040440
 8108 033276
 8109 033276 004537 003244
 8110 033302 061224
 8111 033304 111237 002636
 8112 033310 116104 000004
 8113 033314 123704 002636
 8114 033320 001411
 8115 033322
 8116 033334 104455
 8117 033336 000027
 8118 033340 005226
 8119 033342 007404
 8120 033344
 8121 033344 04410

.EVEN
 ENDTST
 L10160:
 TRAP C\$ETST

BADHEAD
 :***** TEST 71 *****
 :*ALU TEST
 :*TEST OF ALU FUNCTION SUB W/C WITH C BIT SET
 :*ALU FUNCTION (A-B-C) CODE=2
 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
 BADHEAD
 :***** TEST 71 *****

BGNTST
 T71::

MYINT
 MOV KMCSR,R1 ;GET DEVICE ADDRESS.
 MSTCLR ;MASTER CLEAR M8200,4,7
 JSR R5, .MSTCLR ;CLEAR M8200,4,7
 CLR R5 ;MEM + SP ADDRESS
 MOV #5\$,R2 ;POINTER TO CORRECT DATA
 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
 MEMDAT ;POINTER TO DATA
 JSR PC,SPLD ;LOAD 8 WORDS OF SP
 SPDAT ;POINTER TO DATA
 BGNSEG
 TRAP C\$BSEG
 JSR PC,SETC ;SET C BIT!
 BIC #17,2\$;CLEAR ADDRESS FIELD OF INSTRUCTION
 BIS R5,2\$;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 1\$: 010000 ;LOAD MAR
 BIC #17,3\$;CLEAR ADDRESS OF INSTRUCTION
 BIS R5,3\$;ADD ADDRESS TO INSTRUCTION
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=55304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 2\$: 040400! <2*20> ;BR SUB W/C
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5, .ROMCLK ;CLOCK INSTRUCTION
 3\$: 61224 ;MOVE BR TO PORT4
 MOVB (R2), \$GDDAT ;PUT 'EXPECTED' IN \$GDDAT
 MOVB 4(R1), R4 ;PUT 'FOUND' IN R4
 CMPB \$GDDAT, R4 ;DATA CORRECT?
 BEQ 4\$;BR IF YES
 ERROR 23, YES ;ALU ERROR
 TRAP C\$ERDF
 .WORD 23
 .WORD EM23
 .WORD ERR23
 4\$: ESCAPE SEG
 TRAP C\$ESCAPE

```

8122 033346 000014      .WORD 10000$-.
8123 033350 005202      INC R2      ;NEXT DATA
8124 033352 005205      INC R5      ;NEXT ADDRESS
8125 033354 022705 000010  CMP #10,R5 ;DONE YET?
8126 033360 001324      BNE 1$      ;BR IF NO
8127 033362      ENDSEG
8128 033362      10000$:
8129 033362 104405      TRAP C$ESEG
8130 033364      EXIT TST
8131 033364 104432      TRAP C$EXIT
8132 033366 000012      .WORD L10161-.
8133 033370 000 001 377 5$: .BYTE 0,1,-1,0,0,253,125,0
8134 033373 000 000 253
8135 033376 125 000
8136
8137      .EVEN
8138 033400      ENDTST
8139 033400      L10161:
8140 033400 104401      TRAP C$ETST
8141
8142
8143 033402      BADHEAD
8144      ;***** TEST 72 *****
8145      ;*ALU TEST
8146      ;*TEST OF ALU FUNCTION INC A WITH C BIT SET
8147      ;*ALU FUNCTION (A PLUS 1) CODE=3
8148      ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
8149      ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
8150 033402      BADHEAD
8151      ;***** TEST 72 *****
8152
8153 033402      BGNTST
8154 033402      T72::
8155 033402
8156 033402 013701 002716      MYINT
8157 033406      MOV KMCSR,R1 ;GET DEVICE ADDRESS.
8158 033406 004537 003156      MSTCLR ;MASTER CLEAR M8200,4,7
8159 033412 005005      JSR R5, .MSTCLR ;CLEAR M8200,4,7
8160 033414 012702 033574      CLR R5 ;MEM + SP ADDRESS
8161 033420 004737 003640      MOV #5$,R2 ;POINTER TO CORRECT DATA
8162 033424 002654      JSR PC, MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
8163 033426 004737 004012      MEMDAT ;POINTER TO DATA
8164 033432 002664      JSR PC, SPLD ;LOAD 8 WORDS OF SP
8165 033434      SPDAT ;POINTER TO DATA
8166 033434 104404      BGNSEG
8167 033436 004737 004076 1$: TRAP C$BSEG
8168 033442 042737 000017 033460 JSR PC, SETC ;SET C BIT!
8169 033450 050537 033460      BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
8170 033454      BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
8171 033454 004537 003244      ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8172 033460 010000      JSR R5, .ROMCLK ;CLOCK INSTRUCTION
8173 033462 042737 000017 033500 2$: 010000 ;LOAD MAR
8174 033470 050537 033500      BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
8175 033474      BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
8176 033474 004537 003244      ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
8177 033500 040460 3$: JSR R5, .ROMCLK ;CLOCK INSTRUCTION
      040400!<3*20> ;BR _ INC A

```

8178	033502					ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8179	033502	004537	003244			JSR	R5, .ROMCLK	;CLOCK INSTRUCTION
8180	033506	061224				61224		;MOVE BR TO PORT4
8181	033510	11237	002636			MOVB	(R2), \$GDDAT	;PUT 'EXPECTED' IN \$GDDAT
8182	033514	116104	000004			MOVB	4(R1), R4	;PUT 'FOUND IN R4
8183	033520	123704	002636			CMPB	\$GDDAT, R4	;DATA CORRECT?
8184	033524	001411				BEQ	4\$;BR IF YES
8185	033526					ERROR	23, YES	;ALU ERROR
8186	033540	104455				TRAP	C\$ERDF	
8187	033542	000027				.WORD	23	
8188	033544	005226				.WORD	EM23	
8189	033546	007404				.WORD	ERR23	
8190	033550				4\$:	ESCAPE	SEG	
8191	033550	104410				TRAP	C\$ESCAPE	
8192	033552	000014				.WORD	10000\$-	
8193	033554	005202				INC	R2	;NEXT DATA
8194	033556	005205				INC	R5	;NEXT ADDRESS
8195	033560	022705	000010			CMP	#10, R5	;DONE YET?
8196	033564	001324				BNE	1\$;BR IF NO
8197	033566					ENDSEG		
8198	033566				10000\$:			
8199	033566	104405				TRAP	C\$ESEG	
8200	033570					EXIT	TST	
8201	033570	104432				TRAP	C\$EXIT	
8202	033572	000012				.WORD	L10162-	
8203	033574	001	001	000	5\$:	.BYTE	1,1,0,0,126,126,253,253	
8204	033577	000	126	126				
8205	033602	253	253					
8206								
8207						.EVEN		
8208	033604					ENDTST		
8209	033604					L10162:		
8210	033604	104401				TRAP	C\$ETST	
8211								
8212								
8213	033606					BADHEAD		
8214						:***** TEST 73 *****		
8215						;*ALU TEST		
8216						;*TEST OF ALU FUNCTION 2A WITH C BIT SET		
8217						;*ALU FUNCTION (A PLUS A) CODE=5		
8218						;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA		
8219						;*PERFORM THE FUNCTION, VERIFY THE RESULTS		
8220	033606					BADHEAD		
8221						:***** TEST 73 *****		
8222								
8223	033606					BGNTST		
8224	033606					T73::		
8225	033606					MYINT		
8226	033606	013701	002716			MOV	KMCSR, R1	;GET DEVICE ADDRESS.
8227	033612					MSTCLR		;MASTER CLEAR M8200,4,7
8228	033612	004537	003156			JSR	R5, .MSTCLR	;CLEAR M8200,4,7
8229	033616	005005				CLR	R5	;MEM + SP ADDRESS
8230	033620	012702	034000			MOV	#5\$, R2	;POINTER TO CORRECT DATA
8231	033624	004737	003640			JSR	PC, MEMLD	;LOAD 8 WORDS OF MAIN MEMORY
8232	033630	002654				MEMDAT		;POINTER TO DATA
8233	033632	004737	004012			JSR	PC, SPLD	;LOAD 8 WORDS OF SP

8234	033636	002664				SPDAT		: POINTER TO DATA
8235	033640					BGNSEG		
8236	033640	104404				TRAP	C\$BSEG	
8237	033642	004737	004076		1\$:	JSR	PC,SETC	: SET C BIT!
8238	033646	042737	000017	033664		BIC	#17,2\$: CLEAR ADDRESS FIELD OF INSTRUCTION
8239	033654	050537	033664			BIS	R5,2\$: ADD ADDRESS TO INSTRUCTION
8240	033660					ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8241	033660	004537	003244			JSR	R5,.ROMCLK	: CLOCK INSTRUCTION
8242	033664	010000			2\$:	010000		: LOAD MAR
8243	033666	042737	000017	033704		BIC	#17,3\$: CLEAR ADDRESS OF INSTRUCTION
8244	033674	050537	033704			BIS	R5,3\$: ADD ADDRESS TO INSTRUCTION
8245	033700					ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8246	033700	004537	003244			JSR	R5,.ROMCLK	: CLOCK INSTRUCTION
8247	033704	040520			3\$:	040400!	<5*20>	: BR 2A
8248	033706					ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8249	033706	004537	003244			JSR	R5,.ROMCLK	: CLOCK INSTRUCTION
8250	033712	061224				61224		: MOVE BR TO PORT4
8251	033714	111237	002636			MOVB	(R2), \$GDDAT	: PUT 'EXPECTED' IN \$GDDAT
8252	033720	116104	000004			MOVB	4(R1), R4	: PUT 'FOUND IN R4
8253	033724	123704	002636			CMPB	\$GDDAT, R4	: DATA CORRECT?
8254	033730	001411				BEQ	4\$: BR IF YES
8255	033732					ERROR	23, YES	: ALU ERROR
8256	033744	104455				TRAP	C\$ERDF	
8257	033746	000027				.WORD	23	
8258	033750	005226				.WORD	EM23	
8259	033752	007404				.WORD	ERR23	
8260	033754				4\$:	ESCAPE	SEG	
8261	033754	104410				TRAP	C\$ESCAPE	
8262	033756	000014				.WORD	10000\$-	
8263	033760	005202				INC	R2	: NEXT DATA
8264	033762	005205				INC	R5	: NEXT ADDRESS
8265	033764	022705	000010			CMP	#10, R5	: DONE YET?
8266	033770	001324				BNE	1\$: BR IF NO
8267	033772					ENDSEG		
8268	033772				10000\$:			
8269	033772	104405				TRAP	C\$ESEG	
8270	033774					EXIT	TST	
8271	033774	104432				TRAP	C\$EXIT	
8272	033776	000012				.WORD	L10163-	
8273	034000	000	000	376	5\$:	.BYTE	0,0,376,376,252,252,124,124	
8274	034003	376	252	252				
8275	034006	124	124					
8276								
8277						.EVEN		
8278	034010					ENDTST		
8279	034010					L10163:		
8280	034010	104401				TRAP	C\$ETST	
8281								
8282								
8283	034012					BADHEAD		
8284						:***** TEST 74 *****		
8285						:*ALU TEST		
8286						:*TEST OF ALU FUNCTION A PLUS C WITH C BIT SET		
8287						:*ALU FUNCTION (A PLUS C) CODE=4		
8288						:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA		
8289						:*PERFORM THE FUNCTION, VERIFY THE RESULTS		

```
8290 034012          BADHEAD
8291                :***** TEST 74 *****
8292
8293 034012          BGNTST
8294 034012          T74::
8295 034012
8296 034012 013701 002716 MYINT
8297 034016          MOV KMCSR,R1 ;GET DEVICE ADDRESS.
8298 034016 004537 003156 MSTCLR ;MASTER CLEAR M8200,4,7
8299 034022 005005          JSR R5,.MSTCLR ;CLEAR M8200,4,7
8300 034024 012702 034204 CLR R5 ;MEM + SP ADDRESS
8301 034030 004737 003640 MOV #5$,R2 ;POINTER TO CORRECT DATA
8302 034034 002654          JCR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
8303 034036 004737 004012 MEMDAT ;POINTER TO DATA
8304 034042 002664          JSR PC,SPLD ;LOAD 8 WORDS OF SP
8305 034044          SPDAT ;POINTER TO DATA
8306 034044 104404          BGNSEG
8307 034046 004737 004076 TRAP C$BSEG
8308 034052 042737 000017 034070 1$: JSR PC,SETC ;SET C BIT!
8309 034060 050537 034070 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
8310 034064          BIS R5,2$ ;ADD ADDRESS TO INSTRUCTION
8311 034064 004537 003244 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8312 034070 010000          JSR R5,.ROMCLK ;CLOCK INSTRUCTION
8313 034072 042737 000017 034110 2$: 010000 ;LOAD MAR
8314 034100 050537 034110 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
8315 034104          BIS R5,3$ ;ADD ADDRESS TO INSTRUCTION
8316 034104 004537 003244 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8317 034110 040500          JSR R5,.ROMCLK ;CLOCK INSTRUCTION
8318 034112          ROMCLK <4*20> ;BR A PLUS C
8319 034112 004537 003244 JSR R5,.ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8320 034116 061224          ;CLOCK INSTRUCTION
8321 034120 111237 002636 MOVB (R2), $GDDAT ;MOVE BR TO PORT4
8322 034124 116104 000004 MOVB 4(R1), R4 ;PUT 'EXPECTED' IN $GDDAT
8323 034130 123704 002636 CMPB $GDDAT, R4 ;PUT 'FOUND IN R4
8324 034134 001411          BEQ 4$ ;DATA CORRECT?
8325 034136          ERROR 23, YES ;BR IF YES
8326 034150 104455          TRAP C$ERDF ;ALU ERROR
8327 034152 000027          .WORD 23
8328 034154 005226          .WORD EM23
8329 034156 007404          .WORD ERR23
8330 034160          4$: ESCAPE SEG
8331 034160 104410          TRAP C$ESCAPE
8332 034162 000014          .WORD 10000$-.
8333 034164 005202          INC R2 ;NEXT DATA
8334 034166 005205          INC R5 ;NEXT ADDRESS
8335 034170 022705 000010 CMP #10, R5 ;DONE YET?
8336 034174 001324          BNE 1$ ;BR IF NO
8337 034176          ENDSEG
8338 034176          10000$:
8339 034176 104405          TRAP C$ESEG
8340 034200          EXIT TST
8341 034200 104432          TRAP C$EXIT
8342 034202 000012          .WORD L10164-.
8343 034204 001 001 000 5$: .BYTE 1,1,0,0,126,126,253,253
8344 034207 000 126 126
8345 034212 253 253
```

8346
8347
8348 034214
8349 034214
8350 034214 104401
8351
8352 034216
8353
8354
8355
8356
8357
8358
8359 034216
8360
8361
8362 034216
8363 034216

.EVEN
ENDTST
L10164:
TRAP CSETST

BADHEAD
;***** TEST 75 *****
;*ALU TEST
;*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
;*ALU FUNCTION (A-B-1) CODE=17
;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;*PERFORM THE FUNCTION, VERIFY THE RESULTS
BADHEAD
;***** TEST 75 *****

BGNTST
T75::

8364	034216					MYINT		
8365	034216	013701	002716			MOV	KMCSR,R1	:GET DEVICE ADDRESS.
8366	034222					MSTCLR		:MASTER CLEAR M8200,4,7
8367	034222	004537	003156			JSR	R5,.MSTCLR	:CLEAR M8200,4,7
8368	034226	005005				CLR	R5	:MEM + SP ADDRESS
8369	034230	012702	034410			MOV	#5\$,R2	:POINTER TO CORRECT DATA
8370	034234	004737	003640			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
8371	034240	002654				MEMDAT		:POINTER TO DATA
8372	034242	004737	004012			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
8373	034246	002664				SPDAT		:POINTER TO DATA
8374	034250					BGNSEG		
8375	034250	104404				TRAP	C\$BSEG	
8376	034252	004737	004076			JSR	PC,SETC	:SET C BIT!
8377	034256	042737	000017	034274	1\$:	BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
8378	034264	050537	034274			BIS	R5,2\$:ADD ADDRESS TO INSTRUCTION
8379	034270					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8380	034270	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
8381	034274	010000				010000		:LOAD MAR
8382	034276	042737	000017	034314	2\$:	BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
8383	034304	050537	034314			BIS	R5,3\$:ADD ADDRESS TO INSTRUCTION
8384	034310					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8385	034310	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
8386	034314	040760			3\$:	040400!	<17*20>	:BR 2'S COMP SUB
8387	034316					ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
8388	034316	004537	003244			JSR	R5,.ROMCLK	:CLOCK INSTRUCTION
8389	034322	061224				61224		:MOVE BR TO PORT4
8390	034324	111237	002636			MOVB	(R2), \$GDDAT	:PUT 'EXPECTED' IN \$GDDAT
8391	034330	116104	000004			MOVB	4(R1),R4	:PUT 'FOUND IN R4
8392	034334	123704	002636			CMPB	\$GDDAT,R4	:DATA CORRECT?
8393	034340	001411				BEQ	4\$:BR IF YES
8394	034342					ERROR	23,YES	:ALU ERROR
8395	034354	104455				TRAP	C\$ERDF	
8396	034356	000027				.WORD	23	
8397	034360	005226				.WORD	EM23	
8398	034362	007404				.WORD	ERR23	
8399	034364				4\$:	ESCAPE	SEG	
8400	034364	104410				TRAP	C\$ESCAPE	
8401	034366	000014				.WORD	10000\$-	
8402	034370	005202				INC	R2	:NEXT DATA
8403	034372	005205				INC	R5	:NEXT ADDRESS
8404	034374	022705	000010			CMP	#10,R5	:DONE YET?
8405	034400	001324				BNE	1\$:BR IF NO
8406	034402					ENDSEG		
8407	034402				10000\$:			
8408	034402	104405				TRAP	C\$ESEG	
8409	034404					EXIT	TST	
8410	034404	104432				TRAP	C\$EXIT	
8411	034406	000012				.WORD	L10165-	
8412	034410	377	000	376	5\$:	.BYTE	-1,0,376,-1,-1,252,124,-1	
8413	034413	377	377	252				
8414	034416	124	377					
8415								
8416						.EVEN		
8417	034420					ENDTST		
8418	034420					L10165:		
8419	034420	104401				TRAP	C\$ETST	

8420									
8421									
8422	034422								
8423									
8424									
8425									
8426									
8427									
8428									
8429	034422								
8430									
8431									
8432	034422								
8433	034422								
8434	034422								
8435	034422	013701	002716						
8436	034426								
8437	034426	004537	003156						
8438	034432	005005							
8439	034434	012702	034614						
8440	034440	004737	003640						
8441	034444	002654							
8442	034446	004737	004012						
8443	034452	002664							
8444	034454								
8445	034454	104404							
8446	034456	004737	004076						
8447	034462	042737	000017	034500	1\$:				
8448	034470	050537	034500						
8449	034474								
8450	034474	004537	003244						
8451	034500	010000			2\$:				
8452	034502	042737	000017	034520					
8453	034510	050537	034520						
8454	034514								
8455	034514	004537	003244						
8456	034520	040560			3\$:				
8457	034522								
8458	034522	004537	003244						
8459	034526	061224							
8460	034530	111237	002636						
8461	034534	116104	000004						
8462	034540	123704	002636						
8463	034544	001411							
8464	034546								
8465	034560	104455							
8466	034562	000027							
8467	034564	005226							
8468	034566	007404							
8469	034570				4\$:				
8470	034570	104410							
8471	034572	000014							
8472	034574	005202							
8473	034576	005205							
8474	034600	022705	000010						
8475	034604	001324							

```

BADHEAD
:***** TEST 76 *****
:*ALU TEST
:*TEST OF ALU FUNCTION DEC A WITH C BIT SET
:*ALU FUNCTION (A-1) CODE=7
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS
BADHEAD
:***** TEST 76 *****
  
```

BGNTST
T76::

```

MYINT
MOV      KMCSR,R1      ;GET DEVICE ADDRESS.
MSTCLR   ;MASTER CLEAR M8200,4,7
JSR      R5, .MSTCLR  ;CLEAR M8200,4,7
CLR      R5           ;MEM + SP ADDRESS
MOV      #5$,R2      ;POINTER TO CORRECT DATA
JSR      PC,MEMLD    ;LOAD 8 WORDS OF MAIN MEMORY
MEMDAT   ;POINTER TO DATA
JSR      PC,SPLD     ;LOAD 8 WORDS OF SP
SPDAT   ;POINTER TO DATA
BGNSEG
TRAP     C$BSEG
JSR      PC,SETC     ;SET C BIT!
BIC      #17,2$     ;CLEAR ADDRESS FIELD OF INSTRUCTION
BIS      R5,2$     ;ADD ADDRESS TO INSTRUCTION
ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR      R5, .ROMCLK ;CLOCK INSTRUCTION
010000   ;LOAD MAR
BIC      #17,3$     ;CLEAR ADDRESS OF INSTRUCTION
BIS      R5,3$     ;ADD ADDRESS TO INSTRUCTION
ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR      R5, .ROMCLK ;CLOCK INSTRUCTION
040400!<7*20> ;BR DEC A
ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR      R5, .ROMCLK ;CLOCK INSTRUCTION
61224   ;MOVE BR TO PORT4
MOV      (R2), $GDDAT ;PUT 'EXPECTED' IN $GDDAT
MOV      4(R1), R4    ;PUT 'FOUND IN R4
CMP      $GDDAT, R4  ;DATA CORRECT?
BEQ      4$         ;BR IF YES
ERROR    23, YES    ;ALU ERROR
TRAP     C$ERDF
.WORD    23
.WORD    EM23
.WORD    ERR23
ESCAPE   SEG
TRAP     C$ESCAPE
.WORD    10000$-
INC      R2         ;NEXT DATA
INC      R5         ;NEXT ADDRESS
CMP      #10, R5    ;DONE YET?
BNE      *$        ;BR IF NO
  
```


8492
8493
8494
8495
8496
8497

8498
 8499
 8500
 8501
 8502
 8503
 8504
 8505
 8506
 8507
 8508
 8509
 8510
 8511 034626
 8512 034626 000022
 8513 034630
 8514
 8515 034630
 8516 034630 000032
 8517 034632 034674
 8518 034634 000007
 8519 034636 000000
 8520 034640 000007
 8521 034642
 8522 034642 001031
 8523 034644 034753
 8524 034646 160000
 8525 034650 177776
 8526 034652
 8527 034652 002031
 8528 034654 035012
 8529 034656 000000
 8530 034660 000674
 8531 034662
 8532 034662 003032
 8533 034664 035054
 8534 034666 007000
 8535 034670 000004
 8536 034672 000007
 8537
 8538
 8539
 8540
 8541
 8542 034674
 8543
 8544 034674
 8545
 8546 034674 044127 041511 020110
 8547 034702 044515 051103 026517
 8548 034710 051120 041517 051505
 8549 034716 047523 035122 000
 8550 034723 060 046475 031070
 8551 034730 030060 032054 046475
 8552 034736 031070 032060 033454
 8553 034744 046475 031070 033460

.SBTTL HARDWARE PARAMETER CODING SECTION

```

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

```

BGNHRD
.WORD L10167-L$HARD:2
L$HARD::

GPRMD WMP,0,0,7,0,7,YES
.WORD T$CODE
.WORD WMP
.WORD 7
.WORD T$LLOLIM
.WORD T$HILIM
GPRMA ADDRES,2,0,160000,177776,YES
.WORD T$CODE
.WORD ADDRES
.WORD T$LLOLIM
.WORD T$HILIM
GPRMA VECTOR,4,0,0,674,YES
.WORD T$CODE
.WORD VECTOR
.WORD T$LLOLIM
.WORD T$HILIM
GPRMD PRIRTY,6,0,7000,4,7,YES
.WORD T$CODE
.WORD PRIRTY
.WORD 7000
.WORD T$LLOLIM
.WORD T$HILIM
: GPRMD LNUNIT,10,0,3,0,3,YES
: GPRMD SWPAC1,12,0,377,0,377,YES
: GPRMD SWPAC2,14,0,377,0,377,YES
: GPRMD LOOPBK,16,0,40000,0,1,YES
  
```

```

ENDHRD
.EVEN
L10167:
WMP: .ASCIZ 'WHICH MICRO-PROCESSOR:'
.ASCIZ ''0=M8200,4=M8204,7=M8207''
  
```

8554	034752	000				
8555	034753	115	041511	047522	ADDRES: ASCIZ	/MICRO-PROCESSOR CSR ADDRESS : /
8556	034760	050055	047522	042503		
8557	034766	051523	051117	041440		
8558	034774	051123	040440	042104		
8559	035002	042522	051523	035040		
8560	035010	000040				
8561	035012	044515	051103	026517	VECTOR: .ASCIZ	/MICRO-PROCESSOR VECTOR ADDRESS : /
8562	035020	051120	041517	051505		
8563	035026	047523	020122	042526		
8564	035034	052103	051117	040440		
8565	035042	042104	042522	051523		
8566	035050	035040	000040			
8567	035054	044515	051103	026517	PRIRTY: .ASCIZ	/MICRO-PROCESSOR PRIORITY LEVEL : /
8568	035062	051120	041517	051505		
8569	035070	047523	020122	051120		
8570	035076	047511	044522	054524		
8571	035104	046040	053105	046105		
8572	035112	035040	000040			
8573	035116	044127	041511	020110	LNUNIT: .ASCIZ	/WHICH LINE UNIT (0-3)? 0=NONE,1=M8201,2=M8202,3=M8203 : /
8574	035124	044514	042516	052440		
8575	035132	044516	020124	030050		
8576	035140	031455	037451	030040		
8577	035146	047075	047117	026105		
8578	035154	036461	034115	030062		
8579	035162	026061	036462	034115		
8580	035170	030062	026062	036463		
8581	035176	034115	030062	020063		
8582	035204	020072	000			
8583	035207	123	044527	041524	SWPAC1: .ASCIZ	/SWITCH PACK #1 (DDCMP LINE #) : /
8584	035214	020110	040520	045503		
8585	035222	021440	020061	042050		
8586	035230	041504	050115	046040		
8587	035236	047111	020105	024443		
8588	035244	035040	000040			
8589	035250	053523	052111	044103	SWPAC2: .ASCIZ	/SWITCH PACK #2 (BM873 BOOT ADR) : /
8590	035256	050040	041501	020113		
8591	035254	031043	024040	046502		
8592	035272	033470	020063	047502		
8593	035300	052117	040440	051104		
8594	035306	020051	020072	000		
8595	035313	127	046111	020114	LOOPBK: .ASCIZ	/WILL TEST CONNECTOR(S) BE USED ? 0=NO,1=YES : /
8596	035320	042524	052123	041440		
8597	035326	047117	042516	052103		
8598	035334	051117	051450	020051		
8599	035342	042502	052440	042523		
8600	035350	020104	020077	036460		
8601	035356	047516	030454	054475		
8602	035364	051505	035040	000040		
8603						
8604					.EVEN	
8605						
8606						
8607						
8608						
8609						

CZDMPBO M8207 STATIC DIAG #1 MACY11 30A(1052) 20-AUG-80 07:50 F 16 PAGE 201
CZDMPB.P11 20-AUG-80 07:48 HARDWARE PARAMETER CODING SECTION

SEQ 0200

8610

ADDRS	034753	CSERHR=	000056	EM13	004707	ERR30	010066	G	G\$RADB=	000000
ADR	= 000020	CSERRO=	000060	EM14	004731	ERR31	010144	G	G\$RADD=	000040
ASSEMB=	000010	CSERSF=	000054	EM15	004754	ERR32	010172	G	G\$RADL=	000120
BIT0	= 000001	CSERSO=	000057	EM16	005017	ERR33	010220	G	G\$RADO=	000020
BIT00	= 000001	CSESCA=	000010	EM17	005064	ERR34	010276	G	G\$XFER=	000004
BIT01	= 000002	CSSESEG=	000005	EM2	004307	ERR35	010354	G	G\$YES	= 000010
BIT02	= 000004	CSSESUB=	000003	EM20	005113	ERR36	010432	G	HELP	= 000000
BIT03	= 000010	CSSETST=	0000^1	EM21	005160	ERR37	010510	G	HCE	= 100000
BIT04	= 000020	CS\$EXIT=	000032	EM22	005207	ERR4	006266	G	IBE	= 010000
BIT05	= 000040	CS\$GETB=	000026	EM23	005226	ERR5	006350	G	IDU	= 000040
BIT06	= 000100	CS\$GETW=	000027	EM24	005267	ERR6	006432	G	IER	= 020000
BIT07	= 000200	CS\$GMAN=	000043	EM25	005453	ERR7	006514	G	INIFLG	002674
BIT08	= 000400	CS\$GPHR=	000042	EM26	005474	EVL	= 000004	G	INSTU	003536
BIT09	= 001000	CS\$GPLO=	000030	EM27	005525	E\$END	= 002100		ISR	= 000100
BIT1	= 000002	CS\$GPRI=	000040	EM28	005554	E\$LOAD=	000035		IXE	= 004000
BIT10	= 002000	CS\$INIT=	000011	EM29	005605	FLAG	002620		ISAU	= 000041
BIT11	= 004000	CS\$INLP=	000020	EM3	004416	FM1	004114		ISAUTO=	000041
BIT12	= 010000	CS\$MANI=	000050	EM30	004353	FTIME	002646		ISCLN	= 000041
BIT13	= 020000	CS\$MEM	= 000031	EM31	005312	F\$AU	= 000015		ISDU	= 000041
BIT14	= 040000	CS\$MSG	= 000023	EM32	005341	F\$AUTO=	000020		ISHRD	= 000041
BIT15	= 100000	CS\$OPEN=	000034	EM33	005400	F\$BGN	= 000040		ISINIT=	000041
BIT2	= 000004	CS\$PNTB=	000014	EM34	005435	F\$CLEA=	000007		ISMOD	= 000041
BIT3	= 000010	CS\$PNTF=	000017	EM35	005064	F\$DU	= 000016		ISMSG	= 000041
BIT4	= 000020	CS\$PNTS=	000016	EM36	005640	F\$END	= 000041		ISPROT=	000040
BIT5	= 000040	CS\$PNTX=	000015	EM37	005715	F\$HARD=	000004		ISPTAB=	000041
BIT6	= 000100	CS\$QIO	= 000377	EM4	004444	F\$HW	= 000013		IS\$PWR	= 000041
BIT7	= 000200	CS\$RDBU=	000007	EM5	004472	F\$INIT=	000006		ISRPT	= 000041
BIT8	= 000400	CS\$REFG=	000047	EM6	004533	F\$JMP	= 000050		ISSEG	= 000041
BIT9	= 001000	CS\$RESE=	000033	EM7	004561	F\$MOD	= 000000		ISSETU=	000041
BOE	= 000400	CS\$REVI=	000003	ENDBUG	003244	F\$MSG	= 000011		ISSFT	= 000041
CLKX	002570	CS\$RFLA=	000021	ENDIT	011254	F\$PROT=	000021		ISSRV	= 000041
CLRALL	003274	CS\$RPT	= 000025	ERRFLG	002560	F\$PWR	= 000017		ISSUB	= 000041
CLRC	004060	CS\$SEFG=	000046	ERR1	006054	F\$RPT	= 000012		ISTST	= 000041
CORMAX	035374	CS\$SPRI=	000041	ERR10	006576	F\$SEG	= 000003		JSJMP	= 000167
CRAMA	003522	CS\$SVEC=	000037	ERR11	006660	F\$SOFT=	000005		KMACTV	002110
CZDMF	002000	CS\$TPPI=	000013	ERR12	006736	F\$SRV	= 000010		KMCSR	002116
C\$AU	= 000052	DFPTBL	002364	ERR13	007014	F\$SUB	= 000002		KMCSRH	002720
C\$AUTO=	000061	DH0	005716	ERR14	007042	F\$SW	= 000014		KMCTL	002722
C\$BRK	= 000022	DH1	005717	ERR15	007120	F\$TEST=	000001		KMNUM	002612
C\$BSEG=	000004	DH2	005750	ERR16	007176	GETPRM	010712		KMPO4	002724
C\$BSUB=	000002	DH27	006030	ERR17	007224	G\$CNT0=	000200		KMPO6	002726
C\$CEFG=	000045	DH3	005764	ERR2	006132	G\$DELM=	000372		KMRLVL	002710
C\$CLCK=	000062	DH4	006014	ERR20	007252	G\$DISP=	000003		KMRVEC	002706
C\$CLEA=	000012	DIAGMC=	000000	ERR21	007330	G\$EXCP=	000400		KMTLVL	002714
C\$CLOS=	000035	EF.CON=	000036	ERR22	007356	G\$HILI=	000002		KMTVEC	002712
C\$CLP1=	000006	EF.NEW=	000035	ERR23	007404	G\$LOLI=	000001		LNUNIT	035116
C\$CVEC=	000036	EF.PWR=	000034	ERR24	007462	G\$NO	= 000000		LOCK	002442
C\$DCLN=	000044	EF.RES=	000037	ERR25	007510	G\$OFFS=	000400		LOE	= 040000
C\$DODU=	000051	EF.STA=	000040	ERR26	007566	G\$OFISI=	000376		LOGDEV	002552
C\$DRPT=	000024	EM1	004261	ERR27	007644	G\$PRMA=	000001		LOKFLG	002676
C\$DU	= 000053	EM10	004622	ERR28	007726	G\$PRMD=	000002		LOOPBK	035313
C\$EDIT=	000003	EM11	004654	ERR29	010004	G\$PRML=	000000		LOT	= 000010
C\$ERDF=	000055	EM12	004665	ERR3	006210	G\$RADA=	000140		LTN.ED=	000114

LSACP	002110	G	L10001	002412	L10066	013702	L10153	031750	RETADR	002562
LSAPT	002036	G	L10002	002414	L10067	014132	L10154	032154	RUN	002622
LSAU	011364	G	L10003	006130	L10070	014362	L10155	032360	SAVACT	002614
LSAUT	002070	G	L10004	006206	L10071	014612	L10156	032564	SAVE4	002650
LSAUTO	011256	G	L10005	006264	L10072	015042	L10157	032770	SAVE6	002652
LSCCP	002106	G	L10006	006346	L10073	015326	L10160	033174	SAVNUM	002616
LSCLEA	011354	G	L10007	006430	L10074	015622	L10161	033400	SAVPC	002576
LSCO	002032	G	L10010	006512	L10075	016052	L10162	033604	SAVSP	002574
LSDEPO	002011	G	L10011	006574	L10076	016302	L10163	034010	SETBRO	003320
LSDESC	002414	G	L10012	006656	L10077	016532	L10164	034214	SETBR1	003330
LSDESP	002076	G	L10013	006734	L10100	016762	L10165	034420	SETBR4	003340
LSDEVP	002060	G	L10014	007012	L10101	017212	L10166	034624	SETBR7	003350
LSDISP	002132	G	L10015	007040	L10102	017442	L10167	034674	SETC	004076
LSDLY	002116	G	L10016	007116	L10103	017672	L10170	035374	SETVEC	003552
LSDTP	002040	G	L10017	007174	L10104	020122	MASKX	002572	SETZ	003360
LSDTYP	002034	G	L10020	007222	L10105	020420	MDATA	002506	SFPTBL	002414 G
LSDU	011360	G	L10021	007250	L10106	020650	MEMDAT	002654	SPDAT	002664
LSDUT	002072	G	L10022	007326	L10107	021214	MEMEND	037776	SPLD	004012
LSDVTY	003130	G	L10023	007354	L10110	021212	MEMLD	003640	SSTACK	003130
LSEF	002052	G	L10024	007402	L10111	021526	MEMLD2	003650	STAT	002566
LSENV1	002044	G	L10025	007460	L10112	021670	MEMLIM	002604	STAT1	002700
LSETP	002102	G	L10026	007506	L10113	022032	MEMSET	003424	STAT2	002702
LSEXP1	002046	G	L10027	007564	L10114	022206	MEMSZ	002606	STAT3	002704
LSEXP4	002064	G	L10030	007642	L10115	022412	MRO	002624	STOP	023272
LSEXP5	002066	G	L10031	007724	L10116	022556	NEWST	010704	STRTSW	002564
LSHARD	034630	G	L10032	010002	L10117	022726	NEXT	002440	SUBRPC	002556
LSHIME	002120	G	L10033	010064	L10120	023072	NPRSET	003574	SVCGBL=	000000
LSHPCP	002016	G	L10034	010142	L10121	023260	ONE	002602	SVCINS=	000000
LSHPTP	002022	G	L10035	010170	L10122	023474	OSAPTS=	000000	SVCSUB=	000000
LSHW	002364	G	L10036	010216	L10123	023712	OSAU =	000001	SVCTAG=	000000
LSICP	002104	G	L10037	010274	L10124	024044	OSBGNR=	000000	SVCTST=	000000
LSINIT	010570	G	L10040	010352	L10125	024262	OSBGNS=	000000	SWPAC1	035207
LSLADP	002026	G	L10041	010430	L10126	024412	OSDU =	000001	SWPAC2	035250
LSLAST	040004	G	L10042	010506	L10127	024626	OSERRT=	000000	SLSYM=	010000
LSLOAD	002100	G	L10043	010560	L10130	025032	OSGNSW=	000000	TDATA	024414
LSLUN	002074	G	L10044	010566	L10131	025236	OSPOIN=	000001	TEMP	002444
LSMREV	002050	G	L10045	011254	L10132	025442	OSSETU=	000000	TFM1	004123
LSNAME	002000	G	L10046	011352	L10133	025646	PNT =	001000 G	TFM2	004146
LSPRIO	002042	G	L10047	011356	L10134	026052	POP2SP=	022626	TFM27	004200
LSPROT	002122	G	L10050	011362	L10135	026256	PRI =	002000 G	TFM37	004225
LSPRT	002112	G	L10051	011364	L10136	026462	PRIPTY	035054	TFM5	004163
LSREPP	002062	G	L10052	011512	L10137	026670	PRI00 =	000000 G	TOUTP	023460
LSREV	002010	G	L10053	011556	L10140	027074	PRI01 =	000040 G	TOUTT	023454
LSRPT	010562	G	L10054	011744	L10141	027300	PRI02 =	000100 G	TYPE	002630
LSOFT	035374	G	L10055	012110	L10142	027504	PRI03 =	000140 G	TSARGC=	000002
LSSPC	002056	G	L10056	012244	L10143	027710	PRI04 =	000200 G	TSCODE=	003032
LSSPCP	002020	G	L10057	012374	L10144	030114	PRI05 =	000240 G	TSERRN=	000027
LSPTP	002024	G	L10060	012524	L10145	030320	PRI06 =	000300 G	TSEXCP=	000000
LSSTA	002030	G	L10061	012666	L10146	030524	PRI07 =	000340 G	TSFLAG=	000040
LSW	002414	G	L10062	013052	L10147	030730	PSTACK	002554	TSGMAN=	000000
LSTEST	002114	G	L10063	013236	L10150	031134	QV.FLG	002677	TSHILI=	000007
LSIML	002014	G	L10064	013424	L10151	031340	RAMDAT	003370	TSLAST=	000001
LSUNIT	002012	G	L10065	013574	L10152	031544	REGADR	002730	T\$LOLI=	000004

T\$LSYM= 010000	T\$\$PRO= 010000	T27 017674 G	T5 012112 G	T73 033606 G
T\$LTNO= 000114	T\$\$RPT= 010044	T28 020124 G	T50 025650 G	T74 034012 G
T\$NEST= 177777	T\$\$SEG= 010000	T29 020422 G	T51 026054 G	T75 034216 G
T\$NSO = 000000	T\$\$SOF= 010170	T3 011560 G	T52 026260 G	T76 034422 G
T\$NS1 = 000005	T\$\$SUB= 010110	T30 020652 G	T53 026464 G	T8 012526 G
T\$NS2 = 000003	T\$\$SW = 010002	T30.1 020670	T54 026672 G	T9 012670 G
T\$NS3 = 000003	T\$\$TES= 010166	T31 021216 G	T55 027076 G	UAM = 000200 G
T\$PTNU= 000000	T1 011366 G	T32 021530 G	T56 027302 G	VECTOR 035012
T\$SAVL= 177777	T10 013054 G	T33 021672 G	T57 027506 G	WMP 034674
T\$SEGL= 177777	T11 013240 G	T34 022034 G	T58 027712 G	WTYPE 002626
T\$SEKO= 010000	T12 013426 G	T35 022210 G	T59 030116 G	X\$ALWA= 000000
T\$SUBN= 000000	T13 013576 G	T36 022414 G	T6 012246 G	X\$FALS= 000040
T\$TAGL= 177777	T14 013704 G	T37 022560 G	T60 030322 G	X\$OFFS= 000400
T\$TAGN= 010171	T15 014134 G	T38 022730 G	T61 030526 G	X\$TRUE= 000020
T\$TEMP= 000000	T16 014364 G	T39 023074 G	T62 030732 G	ZERO 002600
T\$TEST= 000114	T17 014614 G	T4 011746 G	T63 031136 G	\$BDADR 002634
T\$TSTM= 177777	T18 015044 G	T40 023262 G	T64 031342 G	\$BDDAT 002640
T\$TSTS= 000001	T19 015330 G	T41 023476 G	T65 031546 G	\$GDADR 002632
T\$SAU = 010051	T2 011514 G	T42 023714 G	T66 031752 G	\$GDDAT 002636
T\$SAUT= 010046	T20 015624 G	T43 024046 G	T67 032156 G	\$LSTIN= 000000
T\$SCLE= 010047	T21 016054 G	T44 024264 G	T68 032362 G	\$LSTTA= 000000
T\$SDU = 010050	T22 016304 G	T45 024424 G	T69 032566 G	\$TMPO 002550
T\$SHAR= 010167	T23 016534 G	T46 024630 G	T7 012376 G	= 040004
T\$SHW = 010001	T24 016764 G	T47 025034 G	T70 032772 G	.MSTCL 003156
T\$SINI= 010045	T25 017214 G	T48 025240 G	T71 033176 G	.ROMCL 003244
T\$MSG= 010043	T26 017444 G	T49 025444 G	T72 033402 G	

. ABS. 040004 000

ERRORS DETECTED: 0

CZDMPB,CZDMPB/SOL/NL:TOC=SVC34R.MLB,CZDMPB.P11
 RUN-TIME: 46 55 .6 SECONDS
 RUN-TIME RATIO: 118/102=1.1
 CORE USED: 17K (33 PAGES)