

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

.REM @

IDENTIFICATION

PRODUCT CODE: AC-B150C-MC
PRODUCT TITLE: CVDVACO DLV11-E OFFLINE TEST
PRODUCT DATE: AUGUST, 1978
AUTHOR: ODES CHOATE
MAINTAINER: DIAGNOSTIC ENGINEERING GROUP

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) 1977,1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

34
35
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

TABLE OF CONTENTS

1.0	GENERAL PROGRAM INFORMATION.
1.1	PROGRAM PURPOSE (ABSTRACT).
1.2	SYSTEM REQUIREMENTS.
1.3	RELATED DOCUMENTS AND STANDARDS.
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES.
1.5	ASSUMPTIONS.
2.0	OPERATING INSTRUCTIONS.
2.1	LOADING AND STARTING PROCEDURES.
2.2	SPECIAL ENVIRONMENTS.
2.3	OPERATIONAL SWITCH SETTINGS
2.4	PROGRAM OPTIONS.
2.5	EXECUTION TIMES.
3.0	ERROR INFORMATION.
3.1	ERROR REPORTING PROCEDURE.
3.2	ERROR HALTS.
4.0	PERFORMANCE AND PROGRESS REPORTS.
4.1	PERFORMANCE REPORTS.
5.0	DEVICE INFORMATION TABLES.
6.0	SUMMARY OF TESTS AND SPECIAL SUBROUTINES

71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE DLV11-E SERIAL LINE INTERFACE. THE PROGRAM AS SET INITIALLY DEFAULTS TO ALL OPTIONS, EXCEPT PROGRAMMABLE BAUD RATE, ENABLED AND A WRAP CABLE CONNECTED. THE USER CAN SELECTIVELY ENABLE AND DISABLE TESTING OF THE OPTIONS BY ALTERING THE CONTENTS OF '\$USER'. THE DIAGNOSTIC IS DESIGNED TO TEST AND DETECT FAULTS TO THE LOGIC LEVEL (NOT TO THE CHIP LEVEL). THIS TEST OPERATES ON UP TO SIXTEEN(16) IDENTICALLY CONFIGURED DLV11-E SERIAL LINE INTERFACES. THE DEFAULT ADDRESSES ARE:

175610 -FIRST SERIAL LINE ADDRESS OF 16 CONSECUTIVE SERIAL LINE DEVICES.

300 - VECTOR FOR FIRST OF 16 DEVICES.

THIS PROGRAM IS DESIGNED TO RUN ON ANY PDP-11 WITH 4K OF MEMORY AND A DLV11-E (LSI-BUS) MODULE. IT CAN RUN UNDER XXDP, APT, AND ACT MONITORS, AND ON PROCESSORS WITH NO HARDWARE SWITCH REGISTER. A POWER FAILURE WILL CAUSE THE DIAGNOSTIC TO RESTART.

1.2 SYSTEM REQUIREMENTS.

1. HARDWARE REQUIREMENTS:

ANY PDP-11 FAMILY PROCESSOR
4K MEMORY - MINIMUM
H315 - CABLE TURN AROUND PLUG (OR EQUIVALENT)
MODEM CABLE - BC01V- OR BC05C-X

SOFTWARE REQUIREMENTS:

THIS DIAGNOSTIC IS DESIGNED TO RUN IN ANY OF THE FOLLOWING WAYS:

STAND ALONE
WITH APT MONITOR
WITH ACT MONITOR
WITH XXDP MONITOR (CHAINABLE)

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

1.3 RELATED DOCUMENTS AND STANDARDS.

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS
APT
ACT
SYSMAC

175-003-009-02
MD-11-DZZMA
AUTOCAT-11-QZALB
MD-11-DZQAC

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES.
NO SPECIAL DIAGNOSTICS ARE REQUIRED TO RUN BEFORE THIS, BUT
THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY
OPERATIONAL.

1.5 ASSUMPTIONS.

THIS DIAGNOSTIC ASSUMES THAT THE OPERATOR HAS INITIALIZED
LOCATION '\$USWR' AND '\$DEVN' TO THE PROPER VALUES.
THE (H) JUMPER MUST BE REMOVED FROM ALL DLV11-E'S UNDER TEST.

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED
MEDIA.

THIS DIAGNOSTIC HAS ONLY ONE (1) STARTING ADDRESS. 200 FOR
START AND RESTART.

THE USER CAN SELECT A SPECIFIC TEST TO BE EXECUTED BY SETTING
SWITCH 8 IN THE SWITCH REGISTER AND THE TEST NUMBER (IN OCTAL)
IN THE LOWER BYTE. (NOTE: ALL TESTS PREVIOUS TO THE SELECTED
ONE ARE EXECUTED WITHOUT ITERATIONS.)

2.2 SPECIAL ENVIRONMENTS.

THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING
UNDER APT,ACT,XXDP MONITORS, AS DESCRIBED IN THEIR RESPECTIVE
PROCEDURES MANJAL AND SYSMAC PACKAGE.

164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217

2.3 OPERATIONAL SWITCH SETTINGS

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

- 1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: 'SWR=XXXXXX NEW=' (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE 'NEW-' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.
 - C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT.)

DYNAMIC SWITCH REGISTER

- BIT 15 - HALT ON ERROR
14 - LOOP ON TEST
13 - INHIBIT ERROR TYPEOUTS
12 - (UNUSED)
11 - INHIBIT ITERATIONS
10 - BELL ON ERROR
9 - LOOP ON ERROR
8 - LOOP ON TEST IN SWR<7:0>
7:0 - TEST NUMBER TO LOOP ON (USED WITH BIT 8)

218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266

2.4 PROGRAM OPTIONS.

THIS PROGRAM WILL SUPPORT TESTING OF MULTIPLE DLV11-E'S. IT REQUIRES THE ADDRESS OF THE FIRST RCSR (STORED AT '\$BASE') AND ITS INTERRUPT VECTOR (STORED AT '\$VECT1'); AND WILL BE ABLE TO ADDRESS ANY DLV11-E STARTING AT THE SPECIFIED BASE ADDRESS UP TO 16 CONSECUTIVE DEVICES.

EXAMPLES: \$BASE: 175610
\$VECT1: 300

THE PROGRAM WILL BE ABLE TO TEST ANY DLV11-E WITHIN THE ADDRESS RANGE 175610 --> 176000

\$BASE AND \$VECT1 DEFAULT TO 175610 AND 300 RESPECTIVELY. THE PROGRAM ASSOCIATES UNIT NUMBERS AS FOLLOWS: (NUMBERS IN PARENTHESIS ARE OCTAL)

UNIT#0 -- BASE ADDRESS STORED AT '\$BASE'
ASSOCIATED BASE VECTOR STORED AT '\$VECT1'
UNIT#1 -- BASE ADDRESS + (10)
BASE VECTOR + (10)

UP TO

UNIT#15 -- BASE ADDRESS + (170)
BASE VECTOR + (170)

LOCATION '\$DEVN' IS USED AS A BIT MAP TO INDICATE WHICH UNIT NUMBERS ARE PRESENT AND WILL BE TESTED.

BIT 15	BIT 1	BIT 0
UNIT 15	UNIT #1	UNIT #0

A BIT MAP CAN BE ENTERED AT '\$DEVN' PRIOR TO STARTING THE PROGRAM.

EXAMPLE:
\$BASE: 175610
\$VECTOR: 300
\$DEVN: 13

THE PROGRAM WILL TEST-

UNIT#0 175610 300
UNIT#1 175620 310
UNIT#3 175640 330

267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322

OPTIONS

LOCATION \$USWR CONTAINS ALL THE USER SELECTABLE OPTIONS. THE VALUES IN THIS WORD MUST CONFORM TO THE ACTUAL BOARD CONFIGURATION. THE DEFAULT VALUE OF \$USWR IS AS FOLLOWS:

<u>BIT POSITION</u>	<u>DEFINITION</u>	<u>DEFAULT VALUE</u>
0-3	#OF DATA BITS	10(8) = 8
4	PARITY ENABLED	0 = NO
5	EVEN ODD PARITY	0 = ODD
6	COMMON SPEED	1 = YES
7	PROGRAMMABLE BAUD RATE	0 = NO
8-11	BAUD RATE OFFSET (SEE FOLLOWING NOTE)	05(8) = 110 BAUD
12	BREAK GENERATION ENABLED	1 = YES
13	CABLE TERMINATED (H315)	1 = YES
14	(-FR) AND (-FD) JUMPERS IN	1 = YES
15	(NOT DEFINED)	

NOTE

THIS DIAGNOSTIC DOES NOT TEST THE PARITY LOGIC.

WHEN THE PROGRAMMABLE BAUD RATE OPTION IS ENABLED THE PROGRAMMABLE BAUD RATE TEST WILL EXIT WITH THE BAUD RATE SET TO THE SELECTED VALUE. TO CHANGE THE DEFAULT VALUE OF 110 BAUD REPLACE BITS <11:8> WITH THE OFFSET INDICATED IN THE TABLE AT THE END OF THE PBR TEST.

DLV11-E INDIVIDUAL TEST REQUIREMENTS

	<u>TESTS NOT EXECUTED</u>	<u>IF BIT</u>
APT TEST (BIT 1 OF \$ENV)	T25, T37	1
PROGRAMMABLE BAUD RATE (BIT 7 OF \$USWR)	T32	1
BREAK GENERATION CIRCUIT (BIT 12 OF \$USWR)	T2, T40	0
CABLE TERMINATED (BIT 13 OF \$USWR)	T15, T16, T17, T20, T21 T22, T23, PARTS OF T34, T36	1
(-FR) & (-FD) JUMPERS IN (BIT 14 OF \$USWR)	T5, T6, T16, T17, T20, T22, T23	1

ALL OTHER TESTS WILL RUN REGARDLESS OF ANY BIT SETTINGS.

323
324
325
326
327
328
329
330
331
332
333
334
335
336
337

2.5 EXECUTION TIMES.

EXECUTION TIMES ARE FOR AN LSI-11 PROCESSOR WITH ALL OPTIONS
ENABLED ON THE DLV11-E (EXCEPT FOR PROGRAMMABLE BAUD RATE), AT
110 BAUD.

FIRST PASS- 90 SECONDS
ADDITIONAL PASSES 95 SECONDS
ADDITIONAL DEVICES 95 SECONDS

THE TEST TIME IS BAUD RATE DEPENDANT; HIGHER BAUD GIVES
SHORTER PASS TIMES.

338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

SINCE THIS DIAGNOSTIC WAS DESIGNED TO FIT IN 4-K OF MEMORY THE ERROR TYPEOUT IS VERY BRIEF. THE FORMAT OF THE ERROR TYPEOUT IS AS FOLLOWS:

TEST#____,ERROR#____,PC=____,ADDRESS=____,VECTOR=____

WHERE ALL VALUES TYPED ARE OCTAL.
THE ADDRESS AND VECTOR REFER TO THE FAILING DLV11-E.
FOR FURTHER INFORMATION THE LISTING MUST BE CONSULTED.
BITS 15,13,10 AND 9 OF THE SWITCH REGISTER CONTROL THE SEQUENCE OF EVENTS AFTER AN ERROR IS CAUGHT.

BIT 15 - CAUSES THE PROGRAM TO HALT IN THE ERROR ROUTINE. CONTINUING THE PROGRAM CAUSES IT TO PROCEED.

BIT 13 - DISABLES THE PRINTING OF THE ERROR MESSAGE.

BIT 10 - CAUSES THE BELL TO RING ON ERROR.

BIT 9 - CAUSES THE DIAGNOSTIC TO LOOP FROM BEGINNING OF TEST TO ERROR.

THE ERROR ROUTINE SUPPORTS THE CONTROL G FUNCTION.

3.2 ERROR HALTS.

THE ONLY HALT IN THIS DIAGNOSTIC IS IN THE ERROR ROUTINE, AND IS EXECUTED ONLY IF BIT 15 OF THE SWITCH REGISTER IS A ONE WHEN AN ERROR OCCURS.

4.0 PERFORMANCE AND PROGRESS REPORTS.

4.1 PERFORMANCE REPORTS.

AS EACH DEVICE COMPLETES ONE PASS OF THE DIAGNOSTIC THE FOLLOWING WILL BE TYPED:

CSR:____,VECTOR:____,ERRORS:____

WHERE 'CSR:____' IS THE DEVICE CSR UNDER TEST
'VECTOR:____' IS THE ASSOCIATED VECTOR
AND 'ERRORS:____' IS THE TOTAL NUMBER OF ERRORS ON THIS DEVICE ON THIS PASS.

NOTE
THIS IS TYPED AFTER THE DEVICE HAS COMPLETED ITS PASS.

394
395
396
397

AFTER ALL DEVICES HAVE BEEN EXERCISED AN END PASS STATEMENT IS
TYPED: 'ENDPASS#-----.'

398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425

5.0 DEVICE INFORMATION TABLES.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCSR	DATA	RING	CLR	CAR	RCVR	REC			RCVR	RCVR	DATA		SEC	REQ	DTR	
	INT		SEND	DET	ACT	REC			DONE	IE	IE		XMIT	SEND		
RBUF	ERRO	OR	FR	P												
	R	ERR	ERR	ERR												
TCSR	PROGRAMMABLE BAUD				PBR				XMIT	XMIT				MAIN	BREA	
	RATE		SELECT	ENAB					RDY	IE				T	K	
TBUF																

NOTE

BLANK BOXES INDICATE UNUSED AND RESERVED BIT POSITIONS. SEE THE LISTING FOR AN EXPLANATION OF THE BITS.

426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

TEST 1 ADDRESSABILITY

THIS TEST VERIFIES THAT THE ADDRESS AS PLACED IN THE
HARDWARE P-TABLE TO BE CORRECT AND THE DLV11-E
RESPONDS TO THAT ADDRESS SPACE.

THE FOLLOWING 8 TESTS TEST ALL 'READ WRITE' BITS

TEST 2 BREAK - TCSR0 SET, CLEAR, RESET

TEST 3 MAINT - TCSR2 SET, CLEAR, RESET

TEST 4 XMITE - TCSR6 SET, CLEAR, RESET

TEST 5 DTR - RCSR1 SET, CLEAR

NOTE

RESET DOES NOT CLEAR THIS BIT. WE CANNOT TEST
FOR AN INITIAL CONDITION AS THIS BIT IS
UNDEFINED UPON POWER UP AND INIT DOESN'T
AFFECT IT.

TEST 6 REQSEND - RCSR2 SET, CLEAR, RESET

THIS TEST ASSUMES THAT JUMPER FR IS IN.

TEST 7 SECXMIT - RCSR3 SET, CLEAR, RESET

TEST 10 DATAIE - RCSR5 SET, CLEAR, RESET

TEST 11 RCYRIE - RCSR6 SET, CLEAR, RESET

482

483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538

THE FOLLOWING 4 TESTS VERIFY THAT RESET (INIT) INITIALIZES
READ ONLY BITS.

TEST 12 RCVRDONE - RCSR 7 - IS CLEARED BY INIT
---- --

TEST 13 XMITRDY - TCSR 7 - IS SET BY INIT
---- --

TEST 14 DATAINT - RCSR 15 - IS CLEARED BY INIT.
---- --

TEST 15 RCVRACT - RCSR 11 - 15 CLEARED BY INIT
---- --

THE FOLLOWING 4 TESTS VERIFY THAT THE EIA SIGNALS CAN BE
TRANSMITTED AND RECEIVED THROUGH THE CABLE.

TEST 16 CARDET SETS AND CLEARS AS DTR SETS AND CLEARS
---- --

TEST 17 CLRSEND SETS AND CLEARS AS DTR SETS AND CLEARS
---- --

TEST 20 RING SETS AND CLEARS AS REQSEND SETS AND CLEARS
---- --

TEST 21 SECREC SETS AND CLEARS AS SECXMIT SETS AND CLEARS
---- --

TEST 22 DATAINT (RCSR-15) SETS WHEN DTR CHANGES STATE AND THAT
DATAINT IS CLEARED AFTER READING RCSR

NOTE

DTR IS TIED TO BOTH CARDET AND CLRSEND BY THE
H315.

TEST 23 DATAINT SETS WHEN RING SETS AND THAT DATAINT
DOES NOT SET WHEN RING CLEARS

TEST 24 DATAINT SETS WHEN SECREC CHANGES STATE
---- --

539
540

541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592

TEST 25 XMIT RDY - TCSR 7 - CLEARS WHEN TBUF IS LOADED
---- --
WITH A CHARACTER AND THAT IT SETS WITHIN A
REASONABLE AMOUNT OF TIME.

TEST 26 OUTPUTTING A CHAR FROM TBUF (WITH MAINT SET)
---- --
RESULTS IN RCVRDONE SETTING WITHIN A
REASONABLE AMOUNT OF TIME AND THAT RESET
CLEARS THE BIT.

TEST 27 RCVRDONE IS CLEARED BY READING RBUF
---- --

TEST 30 RCVRACT - RCSR 11 - SETS WHEN A START BIT IS
---- --
RECEIVED AND CLEARS WHEN RCVRDONE - RCSR 7 -
SETS

TEST 31 OVERRUN BIT - RBUF 14
---- --

TEST 32 PROGRAMMABLE BAUD RATE TEST TEST AT ALL SPEEDS
---- --
AVAILABLE A COMPARISON WILL BE MADE TO SEE IF
NEW TIME IS LESS THAN PREVIOUS.

TEST 33 TRANSMITTER INTERRUPT LOGIC TEST
---- --
LOGICALLY THIS IS 4 SEPARATE TESTS
A) DOES TRANSMITTER INTERRUPT LOGIC WORK
B) AT PRIORITY OF 0
C) AND ONLY ONCE
D) BUT NOT WITH INTERRUPT ENABLE CLEAR

TEST 34 RECEIVER INTERRUPT LOGIC TEST THIS TEST COVERS ALL
---- --
OF THE RECEIVER SIDE OF THE INTERRUPT LOGIC, BOTH
DATASET AND CHARACTER MODES.

TEST 35 TEST ACTUAL DATA TRANSFERED NON-INTERRUPT
---- --
MAINTENANCE BIT SET

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

TEST 36 TEST DATA THROUGH CABLE
---- --

TEST 37 FULL DATA TRANSFER WITH INTERRUPTS AND MAINTENANCE
---- --
MODE.

TEST 40 TEST BREAK GENERATION LOGIC TRANSMIT KNOWN CHAR
---- --
WITH BREAK SET AND COMPARE RECEIVED WITH 0.

TEST 41 NOT A TEST - SEND BACK TO LOOP
---- --

NOTE

FOR ALL OF THE FOLLOWING ROUTINES THE USE
OF (R5) IS PART OF THE LINKAGE MECHANISM
BETWEEN THE CALLER AND THE CALLED.

ROUTINE:TIMER

THIS ROUTINE IS USED TO TEST THE STATUS OF
ANY BIT IN ANY REGISTER.

INPUTS:

HOWLONG THE MAXIMUM AMOUNT OF TIME TO
SPEND IN THIS ROUTINE.
WHICHBIT A MASK WITH THE BIT(S) SET THAT
ARE TO BE CHECKED
REG A POINTER TO THE REGISTER TO BE
CHECKED
SETCLR THE DESIRED RESULTS -- EITHER SET
OR CLEAR

OUTPUT:

THE 'C' BIT IS SET TO INDICATE AN ERROR BUT IT
IS TESTED BY THE IF.ERROR STATEMENT.

636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677

ROUTINE:DATLNG

THIS ROUTINE SETS UP A MASK FOR DATA, WITH -

INPUT: NOTHING IS PASSED TO THIS ROUTINE BUT GLOBAL
INFORMATION IS ASSUMED TO EXIST:
\$USWR-- THE WORD FOR SOFTWARE PARAMETERS
DATA-- A MASK FOR THE LOCATION OF THE OCTAL
NUMBER OF DATA BITS

OUTPUT-----

MASK-- A MASK OF BINARY ZEROS RIGHT-JUSTIFIED
THE NUMBER OF WHICH IS DEFINED IN \$USWR WORD.

ROUTINE:WAIT

THIS ROUTINE IS USED TO DELAY EXECUTION OF THE
MAIN PROGRAM FOR A SPECIFIED AMOUNT OF TIME.
THIS IS ACCOMPLISHED BY INCREMENTING A
REGISTER UP TO A LIMIT. THE INNER LOOP IS SET
TO APPROXIMATE 1 MILLI SEC.

SERVICE ROUTINE: INTSRV

THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT

'INTFLAG' EACH TIME IT IS CALLED. IT ASSUMES
THAT THE MAIN CALLING ROUTINE WILL KNOW WHAT
TO LOOK FOR.

ROUTINE:CYCLE

THIS ROUTINE CAUSES ADRS TO POINT TO THE
ADDRESS OF DLV11-E UNDER TEST, ADRS +2 TO
POINT TO THE VECTOR OF THE DLV11-E UNDER TEST.
IT KEEPS TRACK OF THE CURRENT DEVICE AND BIT
MASKS.

```

678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733

```

```

@
.TITLE MAINDEC-ZZ-CVDVA-B
.*COPYRIGHT (C) 1977
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY ODES CHOATE
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-(3)), JAN 19, 1977.
.*
.SBTTL OPERATIONAL SWITCH SETTINGS
.*
.*      SWITCH          USE
.*      -----          -
.*      15             HALT ON ERROR
.*      14             LOOP ON TEST
.*      13             INHIBIT ERROR TYPEOUTS
.*      11             INHIBIT ITERATIONS
.*      10             BELL ON ERROR
.*      9              LOOP ON ERROR
.*      8              LOOP ON TEST IN SWR<7:0>

.SBTTL BASIC DEFINITIONS

.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL

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

.*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0                ;;GENERAL REGISTER
R1= %1                ;;GENERAL REGISTER
R2= %2                ;;GENERAL REGISTER
R3= %3                ;;GENERAL REGISTER
R4= %4                ;;GENERAL REGISTER
R5= %5                ;;GENERAL REGISTER
R6= %6                ;;GENERAL REGISTER
R7= %7                ;;GENERAL REGISTER
SP= %6                ;;STACK POINTER
PC= %7                ;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
PRO= 0                ;;PRIORITY LEVEL 0

```

734	000040	PR1=	40	::PRIORITY LEVEL 1
735	000100	PR2=	100	::PRIORITY LEVEL 2
736	000140	PR3=	140	::PRIORITY LEVEL 3
737	000200	PR4=	200	::PRIORITY LEVEL 4
738	000240	PR5=	240	::PRIORITY LEVEL 5
739	000300	PR6=	300	::PRIORITY LEVEL 6
740	000340	PR7=	340	::PRIORITY LEVEL 7

741
742 ;*'SWITCH REGISTER' SWITCH DEFINITIONS

743	100000	SW15=	100000
744	040000	SW14=	40000
745	020000	SW13=	20000
746	010000	SW12=	10000
747	004000	SW11=	4000
748	002000	SW10=	2000
749	001000	SW09=	1000
750	000400	SW08=	400
751	000200	SW07=	200
752	000100	SW06=	100
753	000040	SW05=	40
754	000020	SW04=	20
755	000010	SW03=	10
756	000004	SW02=	4
757	000002	SW01=	2
758	000001	SW00=	1
759		.EQUIV	SW09,SW9
760		.EQUIV	SW08,SW8
761		.EQUIV	SW07,SW7
762		.EQUIV	SW06,SW6
763		.EQUIV	SW05,SW5
764		.EQUIV	SW04,SW4
765		.EQUIV	SW03,SW3
766		.EQUIV	SW02,SW2
767		.EQUIV	SW01,SW1
768		.EQUIV	SW00,SW0

769
770 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

771	100000	BIT15=	100000
772	040000	BIT14=	40000
773	020000	BIT13=	20000
774	010000	BIT12=	10000
775	004000	BIT11=	4000
776	002000	BIT10=	2000
777	001000	BIT09=	1000
778	000400	BIT08=	400
779	000200	BIT07=	200
780	000100	BIT06=	100
781	000040	BIT05=	40
782	000020	BIT04=	20
783	000010	BIT03=	10
784	000004	BIT02=	4
785	000002	BIT01=	2
786	000001	BIT00=	1
787		.EQUIV	BIT09,BIT9
788		.EQUIV	BIT08,BIT8
789		.EQUIV	BIT07,BIT7

```

790 .EQUIV BIT06,BIT6
791 .EQUIV BIT05,BIT5
792 .EQUIV BIT04,BIT4
793 .EQUIV BIT03,BIT3
794 .EQUIV BIT02,BIT2
795 .EQUIV BIT01,BIT1
796 .EQUIV BIT00,BIT0
797
798 ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
799 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
800 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
801 000014 TBITVEC=14 ;:'T' BIT
802 000014 TRTVEC= 14 ;:TRACE TRAP
803 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
804 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
805 000024 PWRVEC= 24 ;:POWER FAIL
806 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
807 000034 TRAPVEC=34 ;:'TRAP' TRAP
808 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
809 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
810 000240 PIRQVEC 240 ;:PROGRAM INTERRUPT REQUEST VECTOR

```

```

811 000004 ILLMEM= 4
812 000001 ADRS= R1
813 000002 GOOD= R2
814 000003 BAD= R3
815 000001 REGISTER=R1
816 000002 BIT= R2
817 000003 FUNCT= R3
818 000002 LEAD= R2
819 000004 FOLLOW= R4
820 175610 DLADDR= 175610

```

```

821
822 ; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
823 177777 SET= -1
824 000000 CLR= 0
825
826

```

```

827 ;*****
828 ; RCSR REGISTER BIT NAMES
829 ;*****
830 100000 DATAINT= BIT15 ; DATASET INTERRUPT
831 040000 RING= BIT14 ; RINGING SIGNAL INDICATOR
832 020000 CLRSEND= BIT13 ; CLEAR TO SEND FROM DATASET
833 010000 CARDET= BIT12 ; CARRIER DETECT
834 004000 RCVRACT= BIT11 ; RECEIVER ACTIVE INDICATOR
835 002000 SECREC= BIT10 ; SECONDARY RECEIVE
836 ; UNUSED BIT09
837 ; UNUSED BIT08
838 000200 RCVRDONE= BIT07 ; RECEIVER DONE
839 000100 RCVRIE= BIT06 ; RECEIVER INTERRUPT ENABLE
840 000040 DATAIE= BIT05 ; DATASET INTERRUPT ENABLE
841 ; UNUSED BIT04
842 000010 SECXMIT= BIT03 ; SECONDARY TRANSMIT DATA
843 000004 REQSEND= BIT02 ; REQUEST TO SEND
844 000002 DTR= BIT01 ; DATA TERMINAL READY
845 ; UNUSED BIT00

```

846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892

100000
040000
020000
010000

000200
000100
000040
000020
000010
000004
000002
000001

100000
040000
020000
010000
004000

000200
000100

000004
000001

```

:*****
:          ; RBUF REGISTER BIT NAMES
:*****
          ERROR=          BIT15          ; ERROR INDICATOR
          ORERR=          BIT14          ; OVERRUN ERROR
          FRERR=          BIT13          ; FRAMING ERROR
          PERR=           BIT12          ; PARITY ERROR
          ; UNUSED          BIT11
          ; UNUSED          BIT10
          ; UNUSED          BIT09
          ; UNUSED          BIT08
          RDATA7=         BIT07          ; \
          RDATA6=         BIT06          ; .
          RDATA5=         BIT05          ; /
          RDATA4=         BIT04          ; \
          RDATA3=         BIT03          ; /
          RDATA2=         BIT02          ;
          RDATA1=         BIT01          ;
          RDATA0=         BIT00          ; /
          RECEIVED DATA BITS

:*****
:          ; TCSR REGISTER BIT NAMES
:*****
          PBAUD3=         BIT15          ; \
          PBAUD2=         BIT14          ; \ PROGRAMMABLE BAUD
          PBAUD1=         BIT13          ; / RATE BITS
          PBAUD0=         BIT12          ; /
          PBAUDSET=       BIT11          ; ENABLE SETTING OF
          ; PROGRAMMABLE BAUDE RATE
          ; UNUSED          BIT10
          ; UNUSED          BIT09
          ; UNUSED          BIT08
          XMITRDY         BIT07          ; TRANSMITTER READY
          XMITIE          BIT06          ; TRANSMITTER INTERRUPT ENABLE
          ; UNUSED          BIT05
          ; UNUSED          BIT04
          ; UNUSED          BIT03
          MAINT=          BIT02          ; MAINTENANCE SET BIT
          ; UNUSED          BIT01
          BREAK           BIT00          ; SEND BREAK (CONTINUOUS SPACE)

:*****
:          ; TBUF REGISTER BIT NAMES
:*****
          ; UNUSED          BIT15
    
```

```

893          : UNUSED      BIT14
894          : UNUSED      BIT13
895          : UNUSED      BIT12
896          : UNUSED      BIT11
897          : UNUSFD     BIT10
898          : UNUSED      BIT09
899          : UNUSED      BIT08
900          000200      TDATA7=     BIT07
901          000100      TDATA6=     BIT06
902          000040      TDATA5=     BIT05
903          000020      TDATA4=     BIT04
904          000010      TDATA3=     BIT03
905          000004      TDATA2=     BIT02
906          000002      TDATA1=     BIT01
907          000001      TDATA0=     BIT00
          : \
          : |
          : | \ TRANSMITTER DATA BUFFER
          : |
          : /
  
```

```

910          :*****
911          : FLAG BITS TO BE USE OR CLEARED IN $USWR.
  
```

```

912          000017      DATA -      17
913          000020      PARITY =     20
914          000040      EVENODD =    40
915          000100      COMSPD      100
916          000200      PBR =       200
  
```

```

917          : BAUDE MUST BE ON THE UPPER
918          : BYTE BOUNDRY OF $USWR.--4 BITS
919          BAUD -      7400
920          BRK =      10000
921          CABLE =    20000
922          FRFD -     40000
  
```

```

923          :*****
924          .SBTTL TRAP CATCHER
  
```

```

925          .=0
926          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
927          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
928          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
  
```

```

929          .=174
930          DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
931          SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
932          .SBTTL STARTING ADDRESS(ES)
933          JMP      @START ;;JUMP TO STARTING ADDRESS OF PROGRAM
934          .SBTTL ACT11 HOOKS
  
```

```

935          :*****
936          ;HOOKS REQUIRED BY ACT11
937          $SVPC-.      ;SAVE PC
938          .=46
939          $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
940          .-52
941          .WORD 0      ;;2)SET LOC.52 TO ZERO
942          . $SVPC      ;; RESTORE PC
  
```

```

943          . 1000
944          .SBTTL APT PARAMETER BLOCK
  
```


949
950
951
952
953 001000
954 000024
955 000024 000200
956 000044
957 000044 001000
958 001000
959
960
961
962
963 001000
964 001000 000000
965 001002 001174
966 001004 000005
967 001006 000055
968 001010 000036
969 001012 000030

```
::*****  
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
:*****  
.$X=      ;;SAVE CURRENT LOCATION  
.=24     ;;SET POWER FAIL TO POINT TO START OF PROGRAM  
200      ;;FOR APT START UP  
.=44     ;;POINT TO APT INDIRECT ADDRESS PNTR.  
$APTHDR  ;;POINT TO APT HEADER BLOCK  
.=.$X    ;;RESET LOCATION COUNTER  
:*****  
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
:INTERFACE SPEC.
```

```
$APTHD:  
$HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
$MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)  
$STMT:  .WORD 5      ;;RUN TIM OF LONGEST TEST  
$PASTM: .WORD 45.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
$UNITM: .WORD 30.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
        .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)
```

```

970          .SBTTL COMMON TAGS
971
972          ::*****
973          ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
974          ::*USED IN THE PROGRAM.
975
976          001100          .=1100
977          001100          $CMTAG:          ;;START OF COMMON TAGS
978          001100          000000          .WORD          0          ;;CONTAINS THE TEST NUMBER
979          001102          000          $STNM: .BYTE          0          ;;CONTAINS ERROR FLAG
980          001103          000          $ERFLG: .BYTE          0          ;;CONTAINS SUBTEST ITERATION COUNT
981          001104          000000          $ICNT: .WORD          0          ;;CONTAINS SCOPE LOOP ADDRESS
982          001106          000000          $LPADR: .WORD          0          ;;CONTAINS SCOPE RETURN FOR ERRORS
983          001110          000000          $LPERR: .WORD          0          ;;CONTAINS TOTAL ERRORS DETECTED
984          001112          000000          $ERTTL: .WORD          0          ;;CONTAINS ITEM CONTROL BYTE
985          001114          000          $ITEMB: .BYTE          0          ;;CONTAINS MAX. ERRORS PER TEST
986          001115          00          $ERMAX: .BYTE          1          ;;CONTAINS PC OF LAST ERROR INSTRUCTION
987          001116          000000          $ERRPC: .WORD          0          ;;CONTAINS ADDRESS OF 'GOOD' DATA
988          001120          000000          $GDADR: .WORD          0          ;;CONTAINS ADDRESS OF 'BAD' DATA
989          001122          000000          $BDADR: .WORD          0          ;;CONTAINS 'GOOD' DATA
990          001124          000000          $GDDAT: .WORD          0          ;;CONTAINS 'BAD' DATA
991          001126          000000          $BDDAT: .WORD          0          ;;RESERVED--NOT TO BE USED
992          001130          000000          .WORD          0
993          001132          000000          .WORD          0
994          001134          000          $AUTOB: .BYTE          0          ;;AUTOMATIC MODE INDICATOR
995          001135          000          $INTAG: .BYTE          0          ;;INTERRUPT MODE INDICATOR
996          001136          000000          .WORD          0
997          001140          177570          SWR: .WORD          DSWR          ;;ADDRESS OF SWITCH REGISTER
998          001142          177570          DISPLAY: .WORD          DDISP          ;;ADDRESS OF DISPLAY REGISTER
999          001144          177560          $TKS: 177560          ;;TTY KBD STATUS
1000         001146          177562          $TKB: 177562          ;;TTY KBD BUFFER
1001         001150          177564          $TPS: 177564          ;;TTY PRINTER STATUS REG. ADDRESS
1002         001152          177566          $TPB: 177566          ;;TTY PRINTER BUFFER REG. ADDRESS
1003         001154          000          $NULL: .BYTE          0          ;;CONTAINS NULL CHARACTER FOR FILLS
1004         001155          002          $FILLS: .BYTE          2          ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1005         001156          012          $FILLC: .BYTE          12          ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
1006         001157          000          $TPFLG: .BYTE          0          ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
1007         001160          000000          $TIMES: 0          ;;MAX. NUMBER OF ITERATIONS
1008         001162          000000          $ESCAPE: 0          ;;ESCAPE ON ERROR ADDRESS
1009         001164          177607          $BELL: .ASCIZ <207><377><377>          ;;CODE FOR BELL
1010         001170          077          $QUES: .ASCII /?/          ;;QUESTION MARK
1011         001171          015          $CRLF: .ASCII <15>          ;;CARRIAGE RETURN
1012         001172          000012          $LF: .ASCIZ <12>          ;;LINE FEED
1013         ::*****
1014         .SBTTL APT MAILBOX-ETABLE
1015
1016         ::*****
1017         .EVEN
1018         001174          $MAIL:          ;;APT MAILBOX
1019         001174          000000          $MSGTY: .WORD          AMSGTY          ;;MESSAGE TYPE CODE
1020         001176          000000          $FATAL: .WORD          AFATAL          ;;FATAL ERROR NUMBER
1021         001200          000000          $TESTN: .WORD          ATESTN          ;;TEST NUMBER
1022         001202          000000          $PASS: .WORD          APASS          ;;PASS COUNT
1023         001204          000000          $DEVCT: .WORD          ADEVCT          ;;DEVICE COUNT
1024         001206          000000          $UNIT: .WORD          AUNIT          ;;I/O UNIT NUMBER
1025         001210          000000          $MSGAD: .WORD          AMSGAD          ;;MESSAGE ADDRESS

```

000377

1026	001212	000000	\$MSGLG: .WORD	AMSGLG	::MESSAGE LENGTH
1027	001214		\$ETABLE:		::APT ENVIRONMENT TABLE
1028	001214	000	\$ENV: .BYTE	AENV	::ENVIRONMENT BYTE
1029	001215	000	\$ENVM: .BYTE	AENVM	::ENVIRONMENT MODE BITS
1030	001216	000000	\$SWREG: .WORD	ASWREG	::APT SWITCH REGISTER
1031	001220	071110	\$USWR: .WORD	AUSWR	::USER SWITCHES
1032	001222	000000	\$CPUOP: .WORD	ACPUOP	::CPU TYPE,OPTIONS
1033			.*		BITS 15-11=CPU TYPE
1034			.*		11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1035			.*		11/70=06,PDQ=07,Q=10
1036			.*		BIT 10=REAL TIME CLOCK
1037			.*		BIT 9=FLOATING POINT PROCESSOR
1038			.*		BIT 8=MEMORY MANAGEMENT
1039	001224	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
1040	001225	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE,BLK#1
1041			.*		MEM.TYPE BYTE -- (HIGH BYTE)
1042			.*		900 NSEC CORE=001
1043			.*		300 NSEC BIPOLAR=002
1044			.*		500 NSEC MOS=003
1045	001226	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS,BLK#1
1046			.*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
1047	001230	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
1048	001231	000	\$MTYP2: .BYTE	AMTYP2	::MEM. TYPE,BLK#2
1049	001232	000000	\$MADR2: .WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
1050	001234	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
1051	001235	000	\$MTYP3: .BYTE	AMTYP3	::MEM. TYPE,BLK#3
1052	001236	000000	\$MADR3: .WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
1053	001240	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
1054	001241	000	\$MTYP4: .BYTE	AMTYP4	::MEM. TYPE,BLK#4
1055	001242	000000	\$MADR4: .WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
1056	001244	000300	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
1057	001246	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
1058	001250	175610	\$BASE: .WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
1059	001252	000001	\$DEVN: .WORD	ADEVN	::DEVICE MAP
1060	001254		\$ETEND:		
1061			.MEXIT		

1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076 001254
1077
1078 001254 175610
1079 001256 000300
1080 001260 175610
1081 001262 175612
1082 001264 175614
1083 001266 175615
1084 001270 175616
1085 001272 000000
1086 001274 000020
1087 001334 000000
1088 001336
1089
1090
1091 001336 012706 001100
1092 001342 005026
1093 001344 022706 001140
1094 001350 001374
1095 001352 012706 001100
1096
1097 001356 012737 014370 000020
1098 001364 012737 000340 000022
1099 001372 012737 014170 000030
1100 001400 012737 000340 000032
1101 001406 012737 015322 000034
1102 001414 012737 000340 000036
1103 001422 012737 012502 000024
1104 001430 012737 000340 000026
1105 001436 016767 010752 010742
1106 001444 005067 177510
1107 001450 005067 177506
1108 001454 112767 000001 177433
1109 001462 012767 001462 177416
1110 001470 012767 001470 177412
1111
1112
1113 001476 013746 000004
1114 001502 012737 001536 000004
1115 001510 012767 177570 177422
1116 001516 012767 177570 177416
1117 001524 022777 177777 177406

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;; GLOBAL DATA
DLADD: DLADDR
DLVEC: 300
RCSR: DLADDR + 0
RBUF: DLADDR + 2
TCSR: DLADDR + 4
TCSRHI: DLADDR + 5
TBUF: DLADDR + 6
I: 0
.BLKW 20 ;FOR R5 STACK
R5STACK: .WORD 0

START:

.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS (\$CMTAG) AREA
MOV # \$CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;;CLEAR MEMORY LOCATION
CMP #SWR,R6 ;;DONE?
BNE --6 ;;LOOP BACK IF NO
MOV #STACK,SP ;;SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV # \$SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
MOV #340,@IOTVEC+2 ;;LEVEL 7
MOV # \$ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
MOV #340,@EMTVEC+2 ;;LEVEL 7
MOV # \$TRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV #340,@TRAPVEC+2 ;;LEVEL 7
MOV # \$PWRDN,@PWRVEC ;;POWER FAILURE VECTOR
MOV #340,@PWRVEC+2 ;;LEVEL 7
MOV \$ENDCT,\$EOPC1 ;;SETUP END-OF-PROGRAM COUNTER
CLR \$TIMES ;;INITIALIZE NUMBER OF ITERATIONS
CLR \$ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
MOV #1,\$ERMAX ;;ALLOW ONE ERROR PER TEST
MOV #,\$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,\$LPERR ;;SETUP THE ERROR LOOP ADDRESS
;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV @ERRVEC, -(SP) ;;SAVE ERROR VECTOR
MOV #64,\$@ERRVEC ;;SET UP ERROR VECTOR
MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWITCH REGISTER
MOV #DISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
CMV #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR

```

1118 001532 001012      BNE      66$      ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1119                                ;;AND THE HARDWARE SWR IS NOT -1
1120 001534 000403      BR       65$      ;;BRANCH IF NO TIMEOUT
1121 001536 012716 001544 64$:  MOV     #65$, (SP) ;;SET UP FOR TRAP RETURN
1122 001542 000002      RTI
1123 001544 012767 000176 177366 65$:  MOV     #SWREG, SWR ;;POINT TO SOFTWARE SWR
1124 001552 012767 000174 177362      MOV     #DISPREG, DISPLAY
1125 001560 012637 000004 66$:  MOV     (SP)+, @#ERRVEC ;;RESTORE ERROR VECTOR
1126
1127 001564 005067 177412      CLR     @PASS      ;;CLEAR PASS COUNT
1128 001570 132767 000200 177417      BITB   #AP SIZE, $ENVM ;;TEST USER SIZE UNDER APT
1129 001576 001403      BEQ     67$      ;;YES, USE NON-APT SWITCH
1130 001600 012767 001216 177332      MOV     #SSWREG, SWR ;;NO, USE APT SWITCH REGISTER
1131 001606
1132 67$:
1133 .SBTTL  TYPE PROGRAM NAME
1134 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1135 INC     #-1      ;;FIRST TIME?
1136 BNE     68$      ;;BRANCH IF NO
1137 CMP     #SENDAD, @#42 ;;ACT-11?
1138 BEQ     68$      ;;BRANCH IF YES
1139 TYPE    69$      ;;TYPE ASCIZ STRING
1140 .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1141 TST     @#42      ;;ARE WE RUNNING UNDER XXDP/ACT?
1142 BNE     70$      ;;BRANCH IF YES
1143 CMPB   $ENV, #1  ;;ARE WE RUNNING UNDER APT?
1144 BEQ     70$      ;;BRANCH IF YES
1145 CMP     SWR, #SWREG ;;SOFTWARE SWITCH REG SELECTED?
1146 BNE     71$      ;;BRANCH IF NO
1147 GTSWR      ;;GET SOFT-SWR SETTINGS
1148 BR      71$
1149 MOVVB  #1, $AUTOB ;;SET AUTO-MODE INDICATOR
1150 70$:
1151 71$:
1152 BR      68$      ;;GET OVER THE ASCIZ
1153 ;;69$:
1154 .ASCIZ <CRLF>*ZZ-CVDVA-C DLV11-E OFFLINE TEST*<CRLF>
1155 68$:
1156 LET INITFLAG :- #1
1157 MOV     #1, INITFLAG
1158 LOOP:
1159 CALL    CYCLE      ; NO ARGUMENTS--ADDRS -> NEXT ADDRESS
1160 JSR     PC, CYCLE ;
1161                                ; ADDR+2 -> NEXT VECTOR
1162                                ; GET UNIT ADDRESS
1163 LET     DLADD := (ADRS)+
1164                                ; GET UNIT VECTOR
1165 LET     DLVEC :- (ADRS)
1166 LET     ADRS := DLADD
1167                                ; RCSR = DLADD + 0
1168 LET     RCSR := DLADD
1169 MOV     DLADD, RCSR
1170 LET     RBUF := DLADD + #2
1171 MOV     DLADD, RBUF
1172 ADD     #2, RBUF
1173 LET     TCSR := DLADD + #4
    
```



```

1186
1187
1188
1189
1190
1191
1192 002060 000004
1193 002062 012767 000002 177070
1194 002070 012767 000001 177102
1195 002076
1196 002076 016701 177152
1197
1198 002102
1199 002102 010146
1200 002104 012701 000004
1201 002110 012721 012026
1202 002114 012711 000340
1203 002120 012601
1204 002122
1205 002122 005067 177144
1206 002126
1207 002126
1208 002126
1209 002126 012767 002134 176754
1210
1211 002134
1212 002134 005067 007674
1213
1214
1215 002140 005711
1216 002142
1217 002142 005767 007666
1218 002146 001401
1219
1220 002150
1221 002150 104001
1222 002152
1223 002152
1224 002152
1225 002152
1226 002152 062767 000002 177112
1227 002160
1228 002160 016701 177070
1229 002164 066701 177102
1230 002170
1231 002170 026727 177076 000010
1232 002176 001353
1233 002200
1234 002200 010146
1235 002202 010246
1236 002204 012701 000004
1237 002210 010102
1238 002212 062702 000002
1239 002216 010221
1240 002220 005011
1241 002222 012602

```

```

*****
:TEST 1 ADDRESSABILITY
: THIS TEST VERIFIES THAT THE ADDRESS AS PLACED IN
: THE HARDWARE P-TABLE TO BE CORRECT AND THE DLV11-E RESPONDS
: TO THAT ADDRESS SPACE
*****
TST1: SCOPE
MOV #2,$TIMES ;:DO 2 ITERATIONS
MOV #1,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
LET ADRS := DLADD
MOV DLADD,ADRS ; SET UP INTERRUPT
SETVEC #ILLMEM,#INTSRV,#PR7
MOV R1,-(SP)
MOV #ILLMEM,R1
MOV #INTSRV,(R1)+
MOV #PR7,(R1)
MOV (SP)+,R1
LET I : #0
REPEAT
BGNSUB
;CLEAR FLAG
LET INTFLAG :- #0
;READ FLAG
IF INTFLAG NE #0 THEN
TST @ADRS
TST INTFLAG
BEQ $2
; FATAL ERROR
ERRDF 1,,NODL
ENDIF
ENDSUB
LET I :- I + #2
LET ADRS :- DLADD + I
UNTIL I EQ #8.
CLRVEC ILLMEM
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV #ILLMEM,R1
MOV R1,R2
ADD #2,R2
MOV R2,(R1)+
CLR (R1)
MOV (SP)+,R2 ;:POP STACK IN'0 R2

```

```
1242 002224 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
1243                                     ;;END OF TEST
1244 002226                                     ENDTST
1245 :*****
1246 :* THE FOLLOWING 8 TESTS TEST ALL 'READ WRITE' BITS
1247 :*****
1248
1249
```



```
1250 .....  
1251 *TEST 2 BREAK - TCSRO SET, CLEAR, RESET  
1252 * NOTE: THE (H) JUMPER MUST BE REMOVED FOR THIS  
1253 * TEST TO FUNCTION PROPERLY.  
1254 .....  
1255 TST2: SCOPE  
1256 002226 000004 MOV #10,$TIMES ;;DO 10 ITERATIONS  
1257 002230 012767 000010 176722 MOV #2,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
1258 002236 012767 000002 176734  
1259 IF #BRK NOTSETIN $USWR THEN  
1260 002244 032767 010000 176746 BIT #BRK,$USWR  
1261 002252 001004 BNE $3  
1262 002254 EXIT TST  
1263 002254 012767 000001 176676 MOV #1,$TIMES  
1264 002262 000452 BR TST3 ;;EXIT THIS TEST  
1265 002264 ENDIF  
1266 002764 $3:  
1267 ; SEE IF IT IS CLEAR  
1268 002264 ; BGNSUB  
1269 002264 012767 002272 176616 MOV #64,$LPERR  
1270  
1271 IF #BREAK SETIN @TCSR THEN  
1272 002272 032777 000001 176764 BIT #BREAK,@TCSR  
1273 002300 001401 BEQ $4  
1274 ; BREAK DID NOT RESET IN TCSR  
1275 002302 ERRHRD 2,,DIDNOT  
1276 002302 104002 ERROR 2  
1277 002304 ENDIF  
1278 002304 $4:  
1279 002304 ENDSUB  
1280  
1281 ; TRY TO SET BREAK BIT  
1282 002304 ; BGNSUB  
1283 002304 012767 002312 176576 MOV #64,$LPERR  
1284 002312 LFT @TCSR := @TCSR SET.BY #BREAK  
1285 002312 052777 000001 176744 BIS #BREAK,@TCSR  
1286 ; STUCK TO 0  
1287 002320 IF #BREAK NOTSETIN @TCSR THEN  
1288 002320 032777 000001 176736 BIT #BREAK,@TCSR  
1289 002326 001001 BNE $5  
1290 ; BREAK DID NOT SET IN TCSR  
1291 002330 ERRHRD 3,,DIDNOT  
1292 002330 104003 ERROR 3  
1293 002332 ENDIF  
1294 002332 $5:  
1295 002332 ENDSUB  
1296  
1297 ; TRY TO CLEAR A SET BIT  
1298 002332 ; BGNSUB  
1299 002332 012767 002340 176550 MOV #64,$LPERR  
1300  
1301 LET @TCSR := @TCSR CLR.BY #BREAK  
1302 002340 042777 000001 176716 BIC #BREAK,@TCSR  
1303 ; SHOULD HAVE CLEARED  
1304 002346 IF #BREAK SETIN @TCSR THEN  
1305 002346 032777 000001 176710 BIT #BREAK,@TCSR
```

```
1306 002354 001401 BEQ $6 ; BREAK DID NOT CLEAR IN TCSR  
1307 ; ERRHRD 4,,DIDNOT  
1308 002356  
1309 002356 104004 ERROR 4  
1310 002360  
1311 002360 $6: ENDF  
1312 002360 ENDSUB  
1313 ; NOW SEE IF RESET CLEARS IT  
1314 ; BGNSUB  
1315 002360  
1316 002360 012767 002366 176522 MOV #64$, $LPERR  
1317  
1318 002366 LET @TCSR := @TCSR SET.BY #BREAK  
1319 002366 052777 000001 176670 BIS #BREAK, @TCSR  
1320 ; ISSUE BUS RESET  
1321 002374 BRESÉT  
1322 002374 000005 RESET  
1323 002376 IF #BREAK SETIN @TCSR THEN  
1324 002376 032777 000001 176660 BIT #BREAK, @TCSR  
1325 002404 001401 BEQ $7  
1326 ; BREAK DID NOT RESET IN TCSR  
1327 002406 ; ERRHRD 5,,DIDNOT  
1328 002406 104005 ERROR 5  
1329 002410 ENDF  
1330 002410 $7:  
1331 002410 ENDSUB  
1332 002410 ENDTST  
1333  
1334  
1335 ;*****
```

```
1336 ::*****  
1337 :*TEST 3 MAINT - TCSR2 SET, CLEAR, RESET  
1338 :*****  
1339 002410 000004 TST3: SCOPE  
1340 002412 012767 000010 176540 MOV #10,$TIMES ;;DO 10 ITERATIONS  
1341 002420 012767 000003 176552 MOV #3,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
1342  
1343 ; SEE IF IT IS CLEAR  
1344 002426 BGNSUB  
1345 002426 012767 002434 176454 MOV #64$,$LPERR  
1346  
1347 002434 IF #MAINT SETIN @TCSR THEN  
1348 002434 032777 000004 176622 BIT #MAINT,@TCSR  
1349 002442 001401 BEQ $10  
1350 ; MAINT DID NOT RESET IN TCSR  
1351 002444 ERRHRD 6,,DIDNOT  
1352 002444 104006 ERROR 5  
1353 002446 ENDIF  
1354 002446 $10:  
1355 002446 ENDSUB  
1356  
1357 ; TRY TO SET MAINT BIT  
1358 002446 BGNSUB  
1359 002446 012767 002454 176434 MOV #64$,$LPERR  
1360 002454 LET @TCSR := @TCSR SET.BY #MAINT  
1361 002454 052777 000004 176602 BIS #MAINT,@TCSR  
1362 ; STUCK TO 0  
1363 002462 IF #MAINT NOTSETIN @TCSR THEN  
1364 002462 032777 000004 176574 BIT #MAINT,@TCSR  
1365 002470 001001 BNE $11  
1366 ; MAINT DID NOT SET IN TCSR  
1367 002472 ERRHRD 7,,DIDNOT  
1368 002472 104007 ERROR 7  
1369 002474 ENDIF  
1370 002474 $11:  
1371 002474 ENDSUB  
1372  
1373 ; TRY TO CLEAR A SET BIT  
1374 002474 BGNSUB  
1375 002474 012767 002502 176406 MOV #64$,$LPERR  
1376  
1377 002502 LET @TCSR := @TCSR CLR.BY #MAINT  
1378 002502 042777 000004 176554 BIC #MAINT,@TCSR  
1379 ; SHOULD HAVE CLEARED  
1380 002510 IF #MAINT SETIN @TCSR THEN  
1381 002510 032777 000004 176546 BIT #MAINT,@TCSR  
1382 002516 001401 BEQ $12  
1383 ; MAINT DID NOT CLEAR INTCSR  
1384 002520 ERRHRD 10,,DIDNOT  
1385 002520 104010 ERROR 10  
1386 002522 ENDIF  
1387 002522 $12:  
1388 002522 ENDSUB  
1389  
1390 ; NOW SEE IF RESET CLEARS IT  
1391 002522 BGNSUB
```

```
1392 002522 012767 002530 176360      MOV      #64$, $LPERR
1393
1394 002530
1395 002530 052777 000004 176526      BIS      #MAINT, @TCSR
1396
1397 002536
1398 002536 000005
1399 002540
1400 002540 032777 000004 176516      BIT      #MAINT, @TCSR
1401 002546 001401
1402
1403 002550
1404 002550 104011
1405 002552
1406 002552
1407 002552
1408 002552
1409
1410
1411
1412
```

```
      LET      @TCSR := @TCSR SET.BY #MAINT
      ; ISSUE BUS RESET
      BRESÉT
      IF      #MAINT SETIN @TCSR THEN
      ; MAINT DID NOT RESET IN TCSR
      ERRHRD 11,,DIDNOT
      ENDIF
      $3:
      ENDSUB
      ENDTST
```

```

1413
1414
1415
1416 002552 000004
1417 002554 012767 000010 176376
1418 002562 012767 000004 176410
1419
1420 002570 012746 000340
1421 002574 012746 002602
1422 002600 000002
1423 002602
1424
1425
1426 002602
1427 002602 012767 002610 176300
1428
1429 002610
1430 002610 032777 000100 176446
1431 002616 001401
1432
1433 002620
1434 002620 104012
1435 002622
1436 002622
1437 002622
1438
1439
1440 002622
1441 002622 012767 002630 176260
1442 002630
1443 002630 052777 000100 176426
1444
1445 002636
1446 002636 032777 000100 176420
1447 002644 001001
1448
1449 002646
1450 002646 104013
1451 002650
1452 002650
1453 002650
1454
1455
1456 002650
1457 002650 012767 002656 176232
1458
1459 002656
1460 002656 042777 000100 176400
1461
1462 002664
1463 002664 032777 000100 176372
1464 002672 001401
1465
1466 002674
1467 002674 104014
1468 002676

```

```

*****
:TEST 4 XMITIE - TCSR6 SET, CLEAR, RESET
*****
TST4: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #4,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
;; USE PRIORITY OF 7
MOV #PR7,-(P) ;;PUT NEW PS ON STACK
MOV #64$,-(SP) ;;PUT NEW PC ON STACK
RTI ;;POP NEW PC AND PS

64$:
; SEE IF IT IS CLEAR
BGNSUB
MOV #65$,$LPERR
IF #XMITIE SETIN @TCSR THEN
BIT #XMITIE,@TCSR
BEQ $14
; XMITIE DID NOT RESET IN TCSR
ERRHRD 12,,DIDNOT
ENDIF
ENDSUB

$14:
; TRY TO SET XMITIE BIT
BGNSUB
MOV #64$,$LPERR
LET @TCSR := @TCSR SET.BY #XMITIE
BIS #XMITIE,@TCSR
; STUCK TO 0
IF #XMITIE NOTSETIN @TCSR THEN
; XMIT DID NOT RESET IN TCSR
ERRHRD 13,,DIDNOT
ENDIF
ENDSUB

$15:
; TRY TO CLEAR A SET BIT
BGNSUB
MOV #64$,$_PERR
LET @TCSR :- @TCSR CLR.BY #XMITIE
BIC #XMITIE,@TCSR
; SHOULD HAVE CLEARED
IF #XMITIE SETIN @TCSR THEN
; XMIT DID NOT CLEAR IN TCSR
ERRHRD 14,,DIDNOT
ENDIF
ENDIF

```

```
1469 002676 $16:
1470 002676 ENDSUB
1471
1472 ; NOW SEE IF RESET CLEARS IT
1473 002676 ; BGNSUB
1474 002676 012767 002704 176204 MOV #64$, $LPERR
1475
1476 002704 LET @TCSR := @TCSR SET.BY #XMITIE
1477 002704 052777 000100 176352 BIS #XMITIE, @TCSR
1478 ; ISSUE BUS RESET
1479 002712 BRESÉT
1480 002712 000005 RESET
1481 002714 IF #XMITIE SETIN @TCSR THEN
1482 002714 032777 000100 176342 BIT #XMITIE, @TCSR
1483 002722 001401 BEQ $17
1484 ; XMIT DID NOT RESET IN TCSR
1485 002724 ERRHRD 15,, DIDNOT
1486 002724 104015 ERROR 15
1487 002726 ENDF
1488 002726 $17:
1489 002726 ENDSUB
1490 002726 ENDTST
1491
1492
1493
1494 ;*****
```

```

1495
1496
1497
1498
1499
1500
1501
1502
1503 002726 000004
1504 002730 012767 000010 176222
1505 002736 012767 000005 176234
1506 002744
1507 002744 032767 040000 176246
1508 002752 001004
1509 002754
1510 002754 012767 000001 176176
1511 002762 000441
1512 002764
1513 002764
1514
1515 002764
1516 002764 012767 002772 176116
1517 002772
1518 002772 042777 000002 176260
1519
1520 003000
1521 003000 032777 000002 176252
1522 003006 001401
1523
1524 003010
1525 003010 104016
1526 003012
1527 003012
1528 003012
1529
1530
1531 003012
1532 003012 012767 003020 176070
1533
1534 003020
1535 003020 052777 000002 176232
1536 003026
1537 003026 032777 000002 176224
1538 003034 001001
1539
1540 003036
1541 003036 104017
1542 003040
1543 003040
1544 003040
1545
1546
1547 003040
1548 003040 012767 003046 176042
1549 003046
1550 003046 042777 000002 176204

```

```

*****
*TEST 5      DTR - RCSR1  SET, CLEAR
*           NOTE:  RESET DOES NOT CLEAR THIS BIT
*           WE CANNOT TEST FOR AN INITIAL CONDITION
*           AS THIS BIT IS UNDEFINED UPON POWER UP AND
*           INIT DOESN'T AFFECT IT.
*           THE (-FD) JUMPER MUST BE IN FOR THIS TEST TO WORK.
*****
TST5:  SCOPE
      MOV  #10,$TIMES      ;;DO 10 ITERATIONS
      MOV  #5,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                          IF #FRFD NOTSETIN $USWR THEN
      BIT  #FRFD,$USWR
      BNE  $20
                          EXIT TST
      MOV  #1,$TIMES
      BR   TST6           ;;:EXIT THIS TEST
                          ENDIF
$20:
      ; TRY TO CLEAR DTR BIT
      BGNSUB
      MOV  #64,$LPERR
      BIC  #DTR,@RCSR
      LET  @RCSR := @RCSR CLR.BY #DTR
      ; STUCK TO 0
      IF  #DTR SETIN @RCSR THEN
      ; DTR DID NOT CLEAR IN RCSR
      ERRHRD 16,,DIDNOT
      ENDIF
$21:
      ENDSUB
      ; TRY TO SET DTR
      BGNSUB
      MOV  #64,$LPERR
      BIS  #DTR,@RCSR
      LET  @RCSR := @RCSR SET.BY #DTR
      IF  #DTR NOTSETIN @RCSR THEN
      ; DTR DID NOT SET IN RCSR
      ERRHRD 17,,DIDNOT
      ENDIF
$22:
      ENDSUB
      ; TRY TO CLEAR IT AGAIN
      BGNSUB
      MOV  #64,$LPERR
      BIC  #DTR,@RCSR
      LET  @RCSR :- @RCSR CLR.BY #DTR

```

```
1551  
1552 003054  
1553 003054 032777 000002 176176 BIT #DTR,@RCSR  
1554 003062 001401 BEQ $23  
1555  
1556 003064 ; DTR DID NOT CLEAR IN RCSR  
1557 003064 104020 ERROR 20 ERRHRD 20,,DIDNOT  
1558 003066  
1559 003066 $23: ENDF  
1560 003066 ENDSUB  
1561 003066 ENDTST  
1562  
1563  
1564  
1565 ;:*****
```



```

1566
1567
1568
1569
1570 003066 000004
1571 003070 012767 000010 176062
1572 003076 012767 000006 176074
1573 003104
1574 003104 032767 040000 176106
1575 003112 001004
1576 003114
1577 003114 012767 000001 176036
1578 003122 000452
1579 003124
1580 003124
1581
1582
1583 003124
1584 003124 012767 003132 175756
1585
1586 003132
1587 003132 032777 000004 176120
1588 003140 001401
1589
1590 003142
1591 003142 104021
1592 003144
1593 003144
1594 003144
1595
1596
1597 003144
1598 003144 012767 003152 175736
1599 003152
1600 003152 052777 000004 176100
1601
1602 003160
1603 003160 032777 000004 176072
1604 003166 001001
1605
1606 003170
1607 003170 104022
1608 003172
1609 003172
1610 003172
1611
1612
1613 003172
1614 003172 012767 003200 175710
1615
1616 003200
1617 003200 042777 000004 176052
1618
1619 003206
1620 003206 032777 000004 176044
1621 003214 001401
  
```

```

*****
*TEST 6          REQSEND - RCSR2          SET, CLEAR, RESET
*                THIS TEST ASSUMES THAT JUMPER -(FR) IS IN
*****
TST6:  SCOPE
      MOV      #10,$TIMES          ;;DO 10 ITERATIONS
      MOV      #6,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
                                      IF #FRFD NOTSETIN $USWR THEN
      BIT      #FRFD,$USWR
      BNE     $24
                                      EXIT TST
      MOV      #1,$TIMES
      BR       TST7                ;;EXIT THIS TEST
                                      ENDIF
$24:
                                      ; SEE IF IT IS CLEAR
                                      BGNSUB
      MOV      #64,$LPERR
                                      IF      #REQSEND SETIN @RCSR THEN
      BIT      #REQSEND,@RCSR
      BEQ     $25
                                      ; REQSEND DID NOT RESET IN RCSR
                                      ERRHRD 21,,DIDNOT
      ERROR   21
                                      ENDIF
$25:
                                      ENDSUB
                                      ; TRY TO SET REQSEND BIT
                                      BGNSUB
      MOV      #64,$LPERR
      BIS     #REQSEND,@RCSR
      LET     @RCSR := @RCSR SET.BY #REQSEND
                                      ; STUCK TO 0
                                      IF      #REQSEND NOTSETIN @RCSR THEN
      BIT      #REQSEND,@RCSR
      BNE     $26
                                      ; REQSEND DID NOT SET IN RCSR
                                      ERRHRD 22,,DIDNOT
      ERROR   22
                                      ENDIF
$26:
                                      ENDSUB
                                      ; TRY TO CLEAR A SET BIT
                                      BGNSUB
      MOV      #64,$LPERR
      BIC     #REQSEND,@RCSR
      LET     @RCSR := @RCSR CLR.BY #REQSEND
                                      ; SHOULD HAVE CLEARED
                                      IF      #REQSEND SETIN @RCSR THEN
      BIT      #REQSEND,@RCSR
      BEQ     $27
  
```

```
1622 ; REQSEND DID NOT CLEAR IN RCSR  
1623 003216 ERRHRD 23,,DIDNOT  
1624 003216 104023 ERROR 23  
1625 003220 ENDIF  
1626 003220 $27:  
1627 003220 ENDSUB  
1628 ; NOW SEE IF RESET CLEARS IT  
1629 BGNSUB  
1630 003220  
1631 003220 012767 003226 175662 MOV #64$, $LPERR  
1632  
1633 003226 LET @RCSR := @RCSR SET.BY #REQSEND  
1634 003226 052777 000004 176024 BIS #REQSEND, @RCSR  
1635 ; ISSUE BUS RESET  
1636 003234 BRESËT  
1637 003234 000005 RESET  
1638 003236 IF #REQSEND SETIN @RCSR THEN  
1639 003236 032777 000004 176014 BIT #REQSEND, @RCSR  
1640 003244 001401 BEJ $30  
1641 ; REQSEND DID NOT RESET IN RCSR  
1642 003246 ERRHRD 24,,DIDNOT  
1643 003246 104024 ERROR 24  
1644 003250 ENDIF  
1645 003250 $30:  
1646 003250 ENDSUB  
1647 003250 ENDTST  
1648  
1649  
1650  
1651 ;*****
```

```

1652
1653
1654
1655 003250 000004
1656 003252 012767 000010 175700
1657 003260 012767 000007 175712
1658
1659 003266
1660 003266 012767 003274 175614
1661
1662 003274
1663 003274 032777 000010 175756
1664 003302 001401
1665
1666 003304
1667 003304 104025
1668 003306
1669 003306
1670 003306
1671
1672
1673 003306
1674 003306 012767 003314 175574
1675 003314
1676 003314 052777 000010 175736
1677
1678 003322
1679 003322 032777 000010 175730
1680 003330 001001
1681
1682 003332
1683 003332 104026
1684 003334
1685 003334
1686 003334
1687
1688
1689 003334
1690 003334 012767 003342 175546
1691
1692 003342
1693 003342 042777 000010 175710
1694
1695 003350
1696 003350 032777 000010 175702
1697 003356 001401
1698
1699 003360
1700 003360 104027
1701 003362
1702 003362
1703 003362
1704
1705 003362
1706 003362 012767 003370 175520
1707
  
```

```

*****
: *TEST 7          SECXMIT - RCSR3          SET, CLEAR, RESET
*****
TST7:  SCOPE
      MOV #10,$TIMES          ;;DO 10 ITERATIONS
      MOV #7,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
      ; SEE IF IT IS CLEAR
      BGNSUB
      IF #SECXMIT SETIN @RCSR THEN
      BIT #SECXMIT,@RCSR
      BEQ $31
      ; SECXMIT DID NOT RESET IN RCSR
      ERRHRD 25,,DIDNOT
      ENDIF
      $31:
      ENDSUB
      ; TRY TO SET SECXMIT BIT
      BGNSUB
      MOV #64,$LPERR
      BIS #SECXMIT,@RCSR
      LET @RCSR := @RCSR SET.BY #SECXMIT
      ; STUCK TO 0
      IF #SECXMIT NOTSETIN @RCSR THEN
      BIT #SECXMIT,@RCSR
      BNE $32
      ; SECXMIT DID NOT SET IN RCSR
      ERRHRD 26,,DIDNOT
      ENDIF
      $32:
      ENDSUB
      ; TRY TO CLEAR A SET BIT
      BGNSUB
      MOV #64,$LPERR
      BIC #SECXMIT,@RCSR
      LET @RCSR := @RCSR CLR.BY #SECXMIT
      ; SHOULD HAVE CLEARED
      IF #SECXMIT SETIN @RCSR THEN
      BIT #SECXMIT,@RCSR
      BEQ $33
      ; SECXMIT DID NOT CLEAR IN RCSR
      ERRHRD 27,,DIDNOT
      ENDIF
      $33:
      ENDSUB
      BGNSUB
      MOV #C'$',LPERR
      ; NOW SEE IF RESET CLEARS IT
  
```

```
1708  
1709 003370 LET @RCSR := @RCSR SET.BY #SECXMIT  
1710 003370 052777 000010 175662 BIS #SECXMIT,@RCSR ; ISSUE BUS RESET  
1711 ;  
1712 003376 BRESÉT  
1713 003376 000005 RESET  
1714 003400 IF #SECXMIT SET IN @RCSR THEN  
1715 003400 032777 000010 175652 BIT #SECXMIT,@RCSR  
1716 003406 001401 BEQ $34 ; SECXMIT DID NOT RESET IN RCSR  
1717 ; ERRHRD 30,,DIDNOT  
1718 003410  
1719 003410 104030 ERROR 30  
1720 003412  
1721 003412 $34: ENDIF  
1722 003412 ENDSUB  
1723 003412 ENDTST  
1724  
1725  
1726  
1727 ;:*****
```

```

1728 ::*****
1729 :*TEST 10 DATAIE - RCSR5 SET, CLEAR, RESET
1730 :*****
1731 003412 000004 TST10: SCOPE
1732 003414 012767 000010 175536 MOV #10,$TIMES ;;DO 10 ITERATIONS
1733 003422 012767 000010 175550 MOV #10,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1734 ; SEE IF IT IS CLEAR
1735 003430 BGNSUB
1736 003430 012767 003436 175452 MOV #64$,$LPERR
1737
1738 003436 IF #DATAIE SET IN @RCSR THEN
1739 003436 032777 000040 175614 BIT #DATAIE,@RCSR
1740 003444 001401 BEQ $35
1741 ; DATAIE DID NOT FESET IN RCSR
1742 003446 ERRHRD 31,,DIDNOT
1743 003446 104031 ERROR 31
1744 003450
1745 003450 $35: ENDIF
1746 003450 ENDSUB
1747
1748 ; TRY TO SET DATAIE BIT
1749 003450 BGNSUB
1750 003450 012767 003456 175432 MOV #64$,$LPERR
1751 003456 LET @RCSR := @RCSR SET.BY #DATAIE
1752 003456 052777 000040 175574 BIS #DATAIE,@RCSR
1753 ; STUCK TO 0
1754 003464 IF #DATAIE NOTSET IN @RCSR THEN
1755 003464 032777 000040 175566 BIT #DATAIE,@RCSR
1756 003472 001001 BNE $36
1757 ; DATAIE DID NOT SET IN RCSR
1758 003474 ERRHRD 32,,DIDNOT
1759 003474 104032 ERROR 32
1760 003476
1761 003476 $36: ENDIF
1762 003476 ENDSUB
1763
1764 ; TRY TO CLEAR A SET BIT
1765 003476 BGNSUB
1766 003476 012767 003504 175404 MOV #64$,$LPERR
1767
1768 003504 LET @RCSR := @RCSR CLR.BY #DATAIE
1769 003504 042777 000040 175546 BIC #DATAIE,@RCSR
1770 ; SHOULD HAVE CLEARED
1771 003512 IF #DATAIE SET IN @RCSR THEN
1772 003512 032777 000040 175540 BIT #DATAIE,@RCSR
1773 003520 001401 BEQ $37
1774 ; DATAIE DID NOT CLEAR IN RCSR
1775 003522 ERRHRD 33,,DIDNOT
1776 003522 104033 ERROR 33
1777 003524
1778 003524 $37: ENDIF
1779 003524 ENDSUB
1780
1781 ; NOW SEE IF RESET CLEARS IT
1782 003524 BGNSUB
1783 003524 012767 003532 175356 MOV #64$,$LPERR

```

```
1784  
1785 003532  
1786 003532 052777 000040 175520 BIS #DATAIE,@RCSR LET @RCSR := @RCSR SET.BY #DATAIE  
1787 ; ISSUE BUS RESET  
1788 003540 BRESÉT  
1789 003540 000005 RESET  
1790 003542  
1791 003542 032777 000040 175510 BIT #DATAIE,@RCSR IF #DATAIE SET IN @RCSR THEN  
1792 003550 001401 BEQ $40 ; DATAIE DID NOT RESET IN RCSR  
1793 ; ERRHRD 34,,DIDNOT  
1794 003552  
1795 003552 104034 ERROR 34  
1796 003554  
1797 003554 $40: ENDIF  
1798 003554  
1799 003554 ENDSUB  
1800  
1801  
1802  
1803  
:;*****
```

```
1804 :*****
1805 :*TEST 11 RCVRIE - RCSR6 SET, CLEAR, RESET
1806 :*****
1807 TST11: SCOPE
1808 MOV #10,$TIMES ;;DO 10 ITERATIONS
1809 MOV #11,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1810 ; SEE IF IT IS CLEAR
1811 BGNSUB
1812 MOV #64,$LPERR
1813
1814 IF #RCVRIE SETIN @RCSR THEN
1815 BIT #RCVRIE,@RCSR
1816 BEQ $41
1817 ; RCVRIE DID NOT RESET IN RCSR
1818 ERRHRD 35,,DIDNOT
1819 ERROR 35
1820 ENDIF
1821 $41:
1822 ENDSUB
1823
1824 ; TRY TO SET RCVRIE BIT
1825 BGNSUB
1826 MOV #64,$LPERR
1827
1828 BLS #RCVRIE,@RCSR
1829 LET @RCSR : @RCSR SET.BY #RCVRIE
1830 ; STUCK TO 0
1831 IF #RCVRIE NOTSETIN @RCSR THEN
1832 BIT #RCVRIE,@RCSR
1833 BNE $42
1834 ; RCVRIE DID NOT SET IN RCSR
1835 ERRHRD 36,,DIDNOT
1836 ERROR 36
1837 ENDIF
1838 $42:
1839 ENDSUB
1840
1841 ; TRY TO CLEAR A SET BIT
1842 BGNSUB
1843 MOV #64,$LPERR
1844
1845 BIC #RCVRIE,@RCSR
1846 LET @RCSR := @RCSR CLR.BY #RCVRIE
1847 ; SHOULD HAVE CLEARED
1848 IF #RCVRIE SETIN @RCSR THEN
1849 BIT #RCVRIE,@RCSR
1850 BEQ $43
1851 ; RCVRIE DID NOT CLEAR IN RCSR
1852 ERRHRD 37,,DIDNOT
1853 ERROR 37
1854 ENDIF
1855 $43:
1856 ENDSUB
1857
1858 ; NOW SEE IF RESET CLEARS IT
1859 BGNSUB
1859 MOV #64,$LPERR
```

```
1860  
1861 003674  
1862 003674 052777 000100 175356 BIS #RCVRIE,@RCSR LET @RCSR := @RCSR SET.BY #RCVRIE  
1863  
1864 003702 ; ISSUE BUS RESET  
1865 003702 000005 RESET BRESÉT  
1866 003704  
1867 003704 032777 000100 175346 BIT #RCVRIE,@RCSR IF #RCVRIE SET IN @RCSR THEN  
1868 003712 001401 BEQ $44 ; RCVRIE DID NOT RESET IN RCSR  
1869  
1870 003714 ERRHRD 40,,DIDNOT  
1871 003714 104040 ERROR 40  
1872 003716  
1873 003716 $44:  
1874 003716  
1875 003716  
1876 003716 CKLOOP  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886
```

```
::*****  
:* THE FOLLOWING 4 TESTS VERIFY  
:* THAT RESET (INIT) INITIALIZES READ ONLY BITS.  
:*****
```


1887
 1888
 1889
 1890
 1891
 1892
 1893
 1894
 1895
 1896
 1897
 1898
 1899
 1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1908
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1918
 1919

003716 000004
 003720 012767 000010 175232
 003726 012767 000012 175244

 003734
 003734 012767 003742 175146
 003742
 003742 032777 000200 175310
 003750 001402

 003752
 003752 104041

 003754
 003754 000005
 003756
 003756
 003756
 003756

```

*****
*TEST 12          TEST THAT RCVRDONE - RCSR 7 - IS CLEARED BY INIT
*****
TST12:  SCOPE
        MOV      #10,$TIMES      ;;DO 10 ITERATIONS
        MOV      #12,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
  

        MOV      #64,$SLPERR     BGNSUB
        IF      #RCVRDONE SETIN @RCSR THEN
        BIT      #RCVRDONE,@RCSR
        BEQ     $45
  

        ERROR   41
  

        RESET
  

        $45:
  

        ENDIF
  

        ;;ALLOW LOOPING AFTER ERROR
        CKLOOP
        ENDSUB
    ENDTST
  
```

;RCVRDONE SHOULD HAVE CLEARED BY INIT
 ; RCVRDONE DID NOT CLEAR IN RCSR
 ERRHRD 41,HRESET, DIDNOT
 ;REISSUE RESET
 BRESET

1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1928
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952

003756 000004
 003760 012767 000010 175172
 003766 012767 000013 175204
 003774
 003774 012767 004002 175106
 004002
 004002 032777 000200 175254
 004010 001002
 004012
 004012 104042
 004014
 004014 000005
 004016
 004016
 004016
 004016
 004016
 004016

```

*****
*TEST 13 TEST THAT XMITRDY - TCSR 7 - IS SET BY INIT
*****
TST13: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #13,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

BGNSUB
MOV #64,$LPERR
IF #XMITRDY NOTSET IN @TCSR THEN
BI #XMITRDY,@TCSR
BNE $46
;RESET SHOULD HAVE SET BIT.
;XMITRDY DID NOT SET IN TCSR (AFTER RESE
ERRHRD 42,HRESET,DIDNOT
;ISSUE ANOTHER RESET
BRESET
ENDIF
;ALLOW LOOPING ON ERROR
CKLOOP
ENDSUB
ENDTST
  
```

\$46:

```

1953
1954
1955
1956 004016 000004
1957 004020 012767 000010 175132
1958 004026 012767 000014 175144
1959
1960
1961
1962
1963 004034
1964 004034 012767 004042 175046
1965 004042
1966 004042 032777 100000 175210
1967 004050 001402
1968
1969 004052
1970 004052 104043
1971
1972
1973
1974
1975
1976 004054
1977 004054 000005
1978 004056
1979 004056
1980 004056
1981 004056
1982 004056
1983
1984
1985
1986

;*****
;*TEST 14 TEST THAT DATAINT - RCSR 15 - IS CLEARED BY INIT.
;*****
TST14: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #14,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

BGNSUB
IF #DATAINT SETIN @RCSR THEN
BIT #DATAINT,@RCSR
BEQ $47

ERRHRD 43, HRESET, DIDNOT

;TESTING EFFECT OF RESET ON BIT
;DATAINT DID NOT CLEAR IN RCSR
;ALLOW A FRESH START
BRESET

RESET

ENDIF

CKLOOP
ENDSUB
ENDTST

$47:

;*****

```

```

1987
1988
1989
1990 004056 000004
1991 004060 012767 000010 175072
1992 004066 012767 000015 175104
1993
1994
1995 004074
1996 004074 032767 020000 175116
1997 004102 001004
1998
1999 004104
2000 004104 012767 000001 175046
2001 004112 000411
2002 004114
2003 004114
2004
2005
2006
2007 004114
2008 004114 012767 004122 174766
2009
2010 004122
2011 004122 032777 004000 175130
2012 004130 001402
2013
2014
2015 004132
2016 004132 104044
2017
2018
2019
2020
2021
2022
2023 004134
2024 004134 000005
2025 004136
2026 004136
2027
2028 004136
2029 004136
2030 004136
2031

;*****
;*TEST 15 TEST THAT RCVRCT - RCSR 11 - 15 CLEARED BY INIT
;*****
TST15: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #15,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

IF #CABLE NOTSETIN $USWR THEN
BIT #CABLE,$USWR
BNE $50
; CAN'T TEST WITHOUT BERG OR H315.
EXIT TST
MOV #1,$TIMES
BR TST16 :::EXIT THIS TEST
ENDIF

$50:
MOV #64,$$LPERR BGNSUB
IF #RCVRCT SETIN @RCSR THEN
BIT #RCVRCT,@RCSR
BEQ $51
;RESET SHOULD HAVE CLEARED RCVRCT
ERRHRD 44, HRESET, DIDNOT
ERROR 44
; TESTING EFFECT OF RESET ON BIT
;RCVRCT DID NOT CLEAR IN RCSR
;ALLOW ANOTHER TRY
BRESET
ENDIF
$51:
CKLOOP ;ALLOW LOOPING ON ERROR
ENDSUB
ENDTST
  
```

2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045 004136 000004
2046 004140 012767 000010 175012
2047 004146 012767 000016 175024
2048
2049 004154
2050 004154 032767 060000 175036
2051 004162 001004
2052
2053
2054
2055 004164
2056 004164 012767 000001 174766
2057 004172 000441
2058 004174
2059 004174
2060
2061
2062
2063
2064
2065
2066
2067 004174
2068 004174 012767 004202 174706
2069
2070
2071 004202
2072 004202 042777 000002 175050
2073
2074 004210
2075 004210 032777 010000 175042
2076 004216 001401
2077
2078 004220
2079 004220 104045
2080
2081
2082 004222
2083 004222
2084 004222
2085
2086
2087 004222

```

:*****
:* THE FOLLOWING 4 TESTS VERIFY
:* THAT THE EIA SIGNALS CAN BE TRANSMITTED
:* AND RECEIVED THROUGH THE CABLE
:*****

:*****
:*TEST 16      TEST THAT CARDET SETS AND CLEARS
:*           AS DTR  SETS AND CLEARS
:*           THE (-FD) JUMPER MUST BE IN FOR THIS TEST.
:*****
TST16: SCOPE
      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
      MOV      #16,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                                ; CAN WE USE THE WRAPAROUND??
                                ; #CABLE+FRFD NOTSETIN $USWR THEN
      BIT      #CABLE+FRFD,$USWR
      BNE      $52
                                ; CAN'T TEST WITHOUT BERG OR H315
                                ; OR WITH (-FD) JUMPER OUT.
                                ; OR WITH (-FR) JUMPER OUT.
      EXIT TST
      MOV      #1,$TIMES
      BR       TST17          ;;:EXIT THIS TEST
                                ENDIF
$52:
                                ; DTR AND
                                ; CARDET ARE CONNECTED
                                ; BY THE H315 OR EQUIV.
                                ; CLEAR
                                BGNSUB
      MOV      #64,$, $LPERR
                                ; CLEAR DTR
      LET      @RCSR := @RCSR CLR.BY #DTR
                                ; CARDET SHOULD FOLLOW
      IF      #CARDET SETIN @RCSR THEN
                                ; CARDET DID NOT
      ERRHRD 45,,FORCE
                                ; CLEAR WITH DTR
      ENDIF
$53:
      ENDSUB
      ; SET
      BGNSUB

```

2088	004222	012767	004230	174660	MOV	#64\$, \$LPERR	
2089							
2090							
2091	004230						: SET DTR
2092	004230	052777	000002	175022	BIS	#DTR, @RCSR	LET @RCSR := @RCSR SET.BY #DTR
2093							
2094	004236						: CARDET SHOULD FOLLOW
2095	004236	032777	010000	175014	BIT	#CARDET, @RCSR	IF #CARDET NOTSETIN @RCSR THEN
2096	004244	001001			BNE	\$54	
2097							: CARDET DID NOT SET
2098	004246						ERRHRD 46,,FORCE
2099	004246	104046			ERROR	46	
2100							
2101							: WITH DTR
2102	004250						ENDIF
2103	004250					\$54:	
2104	004250						ENDSUB
2105							
2106							: CLEAR
2107	004250						BGNSUB
2108	004250	012767	004256	174632	MOV	#64\$, \$LPERR	
2109							
2110							
2111	004256						: CLEAR DTR
2112	004256	042777	000002	174774	BIC	#DTR, @RCSR	LET @RCSR := @RCSR CLR.BY #DTR
2113							
2114	004264						: CARDET SHOULD FOLLOW
2115	004264	032777	010000	174766	BIT	#CARDET, @RCSR	IF #CARDET SETIN @RCSR THEN
2116	004272	001401			BEQ	\$55	
2117							: CARDET DID NOT
2118	004274						ERRHRD 47,,FORCE
2119	004274	104047			ERROR	47	
2120							
2121							: CLEAR WITH DTR
2122	004276						ENDIF
2123	004276					\$55:	
2124	004276						ENDSUB
2125	004276						ENDTST
2126							
2127							
2128							
2129							

```

2130
2131
2132
2133
2134
2135
2136 004276 000004
2137 004300 012767 000010 174652
2138 004306 012767 000017 174664
2139
2140 004314
2141 004314 032767 060000 174676
2142 004322 001004
2143
2144 004324
2145 004324 012767 000001 174626
2146 004332 000441
2147 004334
2148 004334
2149
2150
2151
2152
2153
2154
2155
2156 004334
2157 004334 012767 004342 174546
2158
2159
2160 004342
2161 004342 042777 000002 174710
2162
2163 004350
2164 004350 032777 020000 174702
2165 004356 001401
2166
2167 004360
2168 004360 104050
2169
2170
2171 004362
2172 004362
2173 004362
2174
2175
2176 004362
2177 004362 012767 004370 174520
2178
2179
2180 004370
2181 004370 052777 000002 174662
2182
2183 004376
2184 004376 032777 020000 174654
2185 004404 001001
  
```

```

:*****
:*****
:TEST 17 TEST THAT CLRSEND SETS AND CLEARS
: AS DTR SETS AND CLEARS
: (-FD) JUMPER MUST BE IN FOR THIS TEST TO WORK
:*****
TST17: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #17,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
; CAN WE USE THE WRAPAROUND??
IF #CABLE+FRFD NOTSETIN $USWR THEN
; CAN'T TEST WITHOUT BERG OR H315
EXIT TST
MOV #1,$TIMES
BR TST20 :::EXIT THIS EST
ENDIF
$56:
; DTR AND
; CLRSEND ARE CONNECTED
; BY THE H315 OR EQUIV.
; CLEAR
BGNSUB
MOV #64,$LPERR
LET ; CLEAR DTR
@RCSR := @RCSR CLR.BY #DTR
; CLRSEND SHOULD FOLLOW
IF #CLRSEND SETIN @RCSR THEN
; CLRSEND DID NOT
ERRHRD 50,,FORCE
; CLEAR WITH DTR
ENDIF
$57:
ENDSUB
; SET
BGNSUB
MOV #64,$LPERR
LET ; SET DTR
@RCSR := @RCSR SET.BY #DTR
; CLRSEND SHOULD FOLLOW
IF #CLRSEND NOTSETIN @RCSR THEN
  
```

```

2186                                     ; CLRSEND DID NOT SET
2187 004406                                     ERRHRD 51,,FORCE
2188 004406 104051          ERROR 51
2189
2190                                     ; WITH DTR
2191 004410          ENDIF
2192 004410          $60:
2193 004410          ENDSUB
2194
2195                                     ; CLEAR
2196 004410          BGNSUB
2197 004410 012767 004416 174472      MOV #64$, $LPERR
2198
2199
2200 004416          ; CLEAR DTR
2201 004416 042777 000002 174634      BIC #DTR, @RCSR
2202                                     LET @RCSR := @RCSR CLR.BY #DTR
2203 004424          ; CLRSEND SHOULD FOLLOW
2204 004424 032777 020000 174626      BIT #CLRSEND, @RCSR
2205 004432 001401      BEQ $61
2206                                     ; CLRSEND DID NOT
2207 004434          ERRHRD 52,,FORCE
2208 004434 104052          ERROR 52
2209
2210                                     ; CLEAR WITH DTR
2211 004436          ENDIF
2212 004436          $61:
2213 004436          ENDSUB
2214 004436          ENDTST
2215
2216
2217
2218
  
```



```

2219
2220
2221
2222
2223
2224
2225 004436 000004
2226 004440 012767 000010 174512
2227 004446 012767 000020 174524
2228
2229 004454
2230 004454 032767 060000 174536
2231 004462 001004
2232
2233
2234 004464
2235 004464 012767 000001 174466
2236 004472 000441
2237 004474
2238 004474
2239
2240
2241
2242
2243
2244
2245
2246 004474
2247 004474 012767 004502 174406
2248
2249
2250 004502
2251 004502 042777 000004 174550
2252
2253 004510
2254 004510 032777 040000 174542
2255 004516 001401
2256
2257 004520
2258 004520 104053
2259
2260
2261 004522
2262 004522
2263 004522
2264
2265
2266 004522
2267 004522 012767 004530 174360
2268
2269
2270 004530
2271 004530 052777 000004 174522
2272
2273 004536
2274 004536 032777 040000 174514

```

```

*****
*****
*TEST 20 TEST THAT RING SETS AND CLEARS
* AS REQSEND SETS AND CLEARS
* THE (-FR) JUMPER MUST BE IN FOR THIS TEST.
*****
TST20: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #20,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
; CAN WE USE THE WRAPAROUND??
IF #CABLE+FRFD,$USWR THEN #CABLE+FRFD NOTSETIN $USWR THEN
BIT #CABLE+FRFD,$USWR
BNE $62
; CAN'T TEST WITHOUT BERG OR H315
; OR WITH (-FR) JUMPER OUT.
EXIT TST
MOV #1,$TIMES
BR TST21 ;;EXIT THIS TEST
ENDIF
$62:
; REQSEND AND
; RING ARE CONNECTED
; BY THE H315 OR EQUIV.
; CLEAR
BGNSUB
MOV #64,$$,$LPERR
; CLEAR REQSEND
LET @RCSR := @RCSR CLR.BY #REQSEND
; RING SHOULD FOLLOW
IF #RING SETIN @RCSR THEN
; RING DID NOT
ERRHRD 53,,FORCE
; CLEAR WITH REQSEND
ENDIF
ENDSUB
$63:
; SET
BGNSUB
MOV #64,$$,$LPERR
; SET REQSEND
LET @RCSR := @RCSR SET.BY #REQSEND
; RING SHOULD FOLLOW
IF #RING NOTSETIN @RCSR THEN

```

```

2275 004544 001001          BNE      $64
2276                                ; RING DID NOT SET
2277 004546                ERRHRD 54,,FORCE
2278 004546 104054          ERROR    54
2279
2280                                ; WITH REQSEND
2281 004550                ENDIF
2282 004550                $64:
2283 004550                ENDSUB
2284
2285                                ; CLEAR
2286 004550                BGNSUB
2287 004550 012767 004556 174332  MOV     #64$, $LPERR
2288
2289                                ; CLEAR REQSEND
2290 004556                LET     @RCSR :- @RCSR CLR.BY #REQSEND
2291 004556 042777 000004 174474  BIC     #REQSEND, @RCSR
2292                                ; RING SHOULD FOLLOW
2293 004564                IF     #RING SETIN @RCSR THEN
2294 004564 032777 040000 174466  BIT     #RING, @RCSR
2295 004572 001401          BEQ     $65
2296                                ; RING DID NOT
2297 004574                ERRHRD 55,,FORCE
2298 004574 104055          ERROR    55
2299
2300                                ; CLEAR WITH REQSEND
2301 004576                ENDIF
2302 004576                $65:
2303 004576                ENDSUB
2304 004576                ENDTST
2305
2306
2307
2308
  
```



```
2365 004706 ERRHRD 57,,FORCE
2366 004706 104057 ERROR 57
2367
2368
2369 004710 ; WITH SECXMIT
2370 004710 $70: ENDIF
2371 004710 ENDSUB
2372
2373 ; CLEAR
2374 004710 BGNSUB
2375 004710 012767 004716 174172 MOV . #64$, $LPERR
2376
2377 ; CLEAR SECXMIT
2378 004716 LET @RCSR := @RCSR CLR.BY #SECXMIT
2379 004716 042777 000010 174334 BIC #SECXMIT, @RCSR
2380 ; SECRC SHOULD FOLLOW
2381 004724 IF #SECRC SET IN @RCSR THEN
2382 004724 032777 002000 174326 BIT #SECRC, @RCSR
2383 004732 001401 BEQ $71
2384 ; SECRC DID NOT
2385 004734 ERRHRD 60,,FORCE
2386 004734 104060 ERROR 60
2387
2388 ; CLEAR WITH SECXMIT
2389 004736 ENDIF
2390 004736 $71:
2391 004736 ENDSUB
2392 004736 ENDTST
2393
2394
2395
2396
```

```

2397
2398
2399
2400
2401
2402
2403
2404
2405 004736 000004
2406 004740 012767 000010 174212
2407 004746 012767 000022 174224
2408
2409 004754
2410 004754 032767 060000 174236
2411 004762 001004
2412
2413
2414 004764
2415 004764 012767 000001 174166
2416 004772 000463
2417 004774
2418 004774
2419
2420
2421 004774 012746 000340
2422 005000 012746 005006
2423 005004 000002
2424 005006
2425
2426
2427 005006
2428 005006 012767 005014 174074
2429
2430 005014
2431 005014 042777 000002 174236
2432
2433 005022
2434 005022 010546
2435 005024 012745 000001
2436 005030 004767 004712
2437 005034 012605
2438
2439 005036
2440 005036 017703 174216
2441
2442 005042
2443 005042 032777 100000 174210
2444 005050 001401
2445
2446 005052
2447 005052 104061
2448 005054
2449 005054
2450
2451 005054
2452
  
```

```

*****
*****
*TEST 22 TEST THAT DATAINT (RCSR-15) SETS
* WHEN DTR CHANGES STATE
* AND THAT DATAINT IS CLEARED AFTER READING RCSR
* NOTE DTR IS TIED TO BOTH CARDET AND CLSEND BY THE H315
* THE (-FD) JUMPER MUST BE IN FOR THIS TEST.
*****
TST22: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #22,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
IF #CABLE+FRFD,$USWR ; CAN WE USE THE WRAPAROUND??
; #CABLE+FRFD NOTSETIN $USWR THEN
; CAN'T TEST WITHOUT BERG OR H315
; OR WITH (-FD) JUMPER OUT.
EXIT TST
MOV #1,$TIMES
BR TST23 ;;EXIT THIS TEST
ENDIF
$72:
MOV #PR7,-(SP) ;MAKE SURE NOTHING UNEXPECTED HAPPENS
MOV #64,-(SP) ;;PUT NEW PS ON STACK
RTI ;;PUT NEW PC ON STACK
;;POP NEW PC AND PS
64$:
;READ TWICE - CLEARS
BGNSUB
MOV #65,$LPERR
; CLEAR DTR
LET @RCSR := @RCSR CLR.BY #DTR
;WAIT 1 MILLI-SEC FOR CABLE
WAITMS 1
MOV R5,-(SP)
MOV #1,-(R5)
JSR PC,WAIT
MOV (SP)+,R5
; READ RCSR - TO CLEAR DATAINT
LET R3 := @RCSR
; READ RCSR AGAIN
IF #DATAINT SETIN @RCSR THEN
; READING RCSR DID NOT CLEAR DATAINT
ERRHRD 61,EDATAINT
ENDIF
$73:
ERROR 61
ENDSUB
  
```


2499
 2500
 2501
 2502
 2503
 2504
 2505
 2506
 2507
 2508
 2509
 2510
 2511
 2512
 2513
 2514
 2515
 2516
 2517
 2518
 2519
 2520
 2521
 2522
 2523
 2524
 2525
 2526
 2527
 2528
 2529
 2530
 2531
 2532
 2533
 2534
 2535
 2536
 2537
 2538
 2539
 2540
 2541
 2542
 2543
 2544
 2545
 2546
 2547
 2548
 2549
 2550
 2551
 2552
 2553
 2554

005142 000004
 005144 012767 000010 174006
 005152 012767 000023 174020
 005160
 005160 032767 060000 174032
 005166 001004
 005170
 005170 012767 000001 173762
 005176 000473
 005200
 005200
 005200 012746 000340
 005204 012746 005212
 005210 000002
 005212
 005212
 005212 012767 005220 173670
 005220
 005220 042777 000004 174032
 005226
 005226 010546
 005230 012745 000001
 005234 004767 004506
 005240 012605
 005242
 005242 017703 174012
 005246
 005246 032777 100000 174004
 005254 001401
 005256
 005256 104065
 005260
 005260
 005260

```

*****
*****
*TEST 23      TEST THAT DATAINT SETS WHEN RING SETS
*              AND THAT DATAINT DOES NOT SET WHEN RING CLEARS
*              THE (-FR) JUMPER MUST BE IN FOR THIS TEST.
*****
TST23: SCOPE
        MOV      #10,$TIMES      ;;DO 10 ITERATIONS
        MOV      #23,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                                   ; CAN WE USE THE WRAPAROUND??
                                   ; #CABLE+FRFD NOTSETIN $USWR THEN
        BIT      #CABLE+FRFD,$USWR
        BNE      $77
                                   ; CAN'T TEST WITHOUT BERG OR H315
                                   ; OR WITH (-FR) JUMPER OUT.
                                   EXIT TST
        MOV      #1,$TIMES
        BR       TST24          ;;EXIT THIS TEST
                                   ENDF
$77:
                                   ;NO INTERRUPTS
        MOV      #PR7,-(SP)     ;;PUT NEW PS ON STACK
        MOV      #64$,-(SP)    ;;PUT NEW PC ON STACK
        RTI      ;;POP NEW PC AND PS
64$:
                                   ;START OFF WITH EVERYTHING CLEAR
                                   BGNSUB
        MOV      #65$,$LPERR
                                   ;CLEAR RING
        LET      @RCSR := @RCSR CLR.BY #REQSEND
                                   ;WAIT 1 MILLI-SEC FOR CABLE
        WAITMS  1
        MOV      R5,-(SP)
        MOV      #1,-(R5)
        JSR      PC,WAIT
        MOV      (SP)+,R5
                                   ;READ ONCE
        LET      R3 := @RCSR
                                   ;READ TWICE
        IF      #DATAINT SETIN @RCSR THEN
                                   ;READING RCSR DID NOT CLEAR DATAINT
                                   ERRHRD 65, EDATAINT
        ERROR   65
                                   ENDF
$100:
                                   ENDSUB
        ;      SET RING --> SET DATAINT
                                   BGNSUB
  
```



```

2605
2606
2607
2608
2609 005366 000004
2610 005370 012767 000010 173562
2611 005376 012767 000024 173574
2612
2613 005404
2614 005404 032767 020000 173606
2615 005412 001004
2616
2617 005414
2618 005414 012767 000001 173536
2619 005422 000454
2620 005424
2621 005424
2622
2623
2624 005424 012746 000340
2625 005430 012746 005436
2626 005434 000002
2627 005436
2628
2629
2630
2631
2632 005436
2633 005436 042777 000010 173614
2634 005444
2635 005444 017703 173610
2636
2637
2638 005450
2639 005450 012767 005456 173432
2640
2641
2642 005456
2643 005456 052777 000010 173574
2644
2645 005464
2646 005464 07546
2647 005466 02745 000001
2648 005472 004767 004250
2649 005476 012605
2650 005500
2651 005500 032777 100000 173552
2652 005506 001001
2653
2654 005510
2655 005510 104124
2656 005512
2657 005512
2658 005512
2659
2660
  
```

```

:*****
:*****
:TEST 24 TEST THAT DATAINT SETS WHEN SECREC CHANGES STATE
:*****
TST24: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #24,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
; CAN WE USE THE WRAPAROUND??
IF #CABLE NOTSETIN $USWR THEN
; CAN'T TEST WITHOUT BERG OR H315.
EXIT TST
MOV #1,$TIMES
BR TST25 ;;EXIT THIS TEST
ENDIF
$103:
MOV #PR7,-(SP) ;NO INTERRUPTS
MOV #64$,-(SP) ;;PUT NEW PS ON STACK
RTI ;;PUT NEW PC ON STACK
;;POP NEW PC AND PS
;START FRESH
;CLEAR SECREC
LET @RCSR := @RCSR CLR.BY #SECMIT
LET R3 := @RCSR
;SET SECREC --> DATAINT SET
BGNSUB
MOV #65$,$LPERR
;SET SECREC
LET @RCSR := @RCSR SET.BY #SECMIT
;WAIT 1 MILLI-SEC FOR CABLE
WAITMS 1
MOV R5,-(SP)
MOV #1,-(R5)
JSR PC,WAIT
MOV (SP)+,R5
IF #DATAINT NOTSETIN @RCSR THEN
;SETTING SECREC DID NOT SET DATAINT
ERRHRD 124,, E2DATA
ERROR 124
ENDIF
$104:
ENDSUB
;CLEAR SECREC --> DATAINT SET
  
```

```

2661 005512
2662 005512 012767 005520 173370      MOV      #64$, $LPERR      BGNSUB
2663
2664 005520
2665 005520 042777 000010 173532      BIC      #SECXMIT, @RCSR  ;CLEAR SECRC
2666
2667 005526
2668 005526 010546
2669 005530 012745 000001      MOV      #1, -(R5)      LET      @RCSR := @RCSR CLR.BY #SECXMIT
2670 005534 004767 004206      JSR      PC, WAIT      ;WAIT 1 MILLI-SEC FOR CABLE
2671 005540 012605      MOV      (SP)+, R5      WAITMS 1
2672 005542
2673 005542 032777 100000 173510      BIT      #DATAINT, @RCSR  IF      #DATAINT NOTSETIN @RCSR THEN
2674 005550 001001      BNE      $105
2675
2676 005552
2677 005552 104125      ERROR   125      ;CLEARING SECRC DID NOT SET DATAINT
2678 005554
2679 005554      $105:
2680 005554
2681 005554
2682
2683
2684
  
```

ENDSUB
 ENDTST


```

2741
2742
2743 005666
2744 005666 105077 173376 CLR8 @TBUF
2745 005672 000240 NOP
2746
2747
2748 005674
2749 005674 032777 000200 173362 BIT #XMITRDY,@TCSR
2750 005702 001401 BEQ $110
2751
2752 005704
2753 005704 104067 ERROR 67
2754 005706
2755 005706 $110:
2756
2757
2758
2759
2760 005706
2761 005706 010546 MOV R5,-(SP)
2762 005710 012745 177777 MOV #SET,-(R5)
2763 005714 016745 173344 MOV TCSR,-(R5)
2764 005720 012745 000200 MOV #XMITRDY,-(R5)
2765 005724 012745 000500 MOV #500,-(R5)
2766 005730 004767 003534 JSR PC,TIMER
2767 005734 012605 MOV (SP)+,R5
2768 005736
2769 005736 103001 BCC $111
2770
2771 005740
2772 005740 104070 ERROR 70
2773 005742
2774 005742 $111:
2775 005742
2776 005742
  
```

```

;SEND SECOND CHARACTER
LET @TBUF :B= #0
; GIVE IT TIME TO CLEAR
; XMITRDY SHOULD HAVE CLEARED UPON
; RECEIPT OF A CHARACTER
IF #XMITRDY SET IN @TCSR THEN
; XMITRDY DID NOT CLEAR IN TCSR
ERRHRD 67,,DIDNOT
ENDIF

;WAIT A MAXIMUM
;OF 50 MSEC FOR
;XMIT RDY TO SET IN TCSR
CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>

IF.ERROR THEN
;XMIT RDY DID NOT SET IN TCSR
ERRHRD 70,,DIDNOT
ENDIF

ENDSUB
ENDTST
  
```

```

2777
2778
2779
2780
2781
2782
2783 005742 000004
2784 005744 012767 000010 173206
2785 005752 012767 000026 173220
2786
2787
2788 005760
2789 005760 052777 000004 173276
2790
2791 005766
2792 005766 012767 005774 173114
2793
2794
2795 005774
2796 005774 105077 173270
2797
2798
2799
2800
2801 006000
2802 006000 010546
2803 006002 012745 177777
2804 006006 016745 173246
2805 006012 012745 000200
2806 006016 012745 000500
2807 006022 004767 003442
2808 006026 012605
2809
2810
2811 006030
2812 006030 103001
2813
2814 006032
2815 006032 104071
2816 006034
2817 006034
2818
2819 006034
2820
2821 006034
2822 006034 012767 006042 173046
2823
2824
2825 006042
2826 006042 000005
2827
2828 006044
2829 006044 032777 000200 173206
2830 006052 001401
2831
2832 006054

```

```

*****
*****
*TEST 26      TEST THAT OUTPUTTING A CHAR FROM TBUF (WITH MAINT SET)
*              RESULTS IN RCVRDONE SETTING WITHIN A REASONABLE AMOUNT OF TIME
*              AND THAT RESET CLEARS THE BIT.
*****
TST26: SCOPE
        MOV     #10,$TIMES      ;;DO 10 ITERATIONS
        MOV     #26,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                                ; SET THE MAINTENANCE BIT
                                LET @TCSR := @TCSR SET.BY #MAINT
        BIS     #MAINT,@TCSR
                                BGNSUB
        MOV     #64,$LPERR
                                ; SEND A CHARACTER AND LET IT WRAP AROUND
        CLRB   @TBUF           LET @TBUF :B- #0
                                ; WAIT A MAXIMUM OF 50 MSEC
                                ; FOR RCVR DONE TO SET IN
                                ; RCSR
                                CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
        MOV     R5,-(SP)
        MOV     #SET,-(R5)
        MOV     RCSR,-(R5)
        MOV     #RCVRDONE,-(R5)
        MOV     #500,-(R5)
        JSR    PC,TIMER
        MOV     (SP)+,R5
                                ;DIDN'T SET IN TIME
                                IF .ERROR THEN
                                ; RCVRDONE DID NOT SET IN RCSR
                                ERRHRD 71,,I'DNOT
                                ENDIF
$112:
                                ENDSUB
                                BGNSUB
        MOV     #64,$LPERR
                                ; NOW THAT IT IS SET SEE IF IT CAN BE RESE
                                ; THIS ALSO WILL CLEAR THE MAINT. BIT
        RESET
                                BRESET
                                IF #RCVRDONE SET IN @RCSR THEN
        BIT     #RCVRDONE,@RCSR
        BEQ    $113
                                ; RCVRDONE DID NOT RESET IN RCSR.
                                ERRHRD 72,,DI'NOT

```

2833 006054 104072
2834 006056
2835 006056
2836 006056
2837 006056

ERROR 72

\$113:

ENDIF
ENDSUB
ENDTST

```

2838
2839
2840
2841
2842 006056 000004
2843 006060 012767 000010 173072
2844 006066 012767 000027 173104
2845
2846
2847 006074
2848 006074 052777 000004 173162
2849 006102
2850 006102 012767 006110 173000
2851
2852
2853
2854
2855 006110
2856 006110 105077 173154
2857
2858
2859
2860 006114
2861 006114 010546
2862 006116 012745 177777
2863 006122 016 45 173132
2864 006126 012745 000200
2865 006132 012745 000500
2866 006136 004767 003326
2867 006142 012605
2868
2869 006144
2870 006144 103001
2871
2872 006146
2873 006146 104073
2874 006150
2875 006150
2876 006150
2877
2878
2879
2880
2881
2882 006150
2883 006150 117700 173106
2884
2885 006154
2886 006154 032777 000200 173076
2887 006162 001401
2888
2889 006164
2890 006164 104074
2891 006166
2892 006166
2893 006166

```

```

*****
*****
*TEST 27 TEST THAT RCVRDONE IS CLEARED BY READING RBUF
*****
TST27: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #27,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

; SET MAINT. BIT
LET @TCSR := @TCSR SET.BY #MAINT
BGNSUB

MOV #64,$$LPERR
; OUTPUT A CHARACTER WITH MAINTENANCE
; SET, AND WAIT FOR XMITRDY TO SET.

; OUTPUT A CHARACTER
LET @TBUF :B= #0

; WAIT MAXIMUM OF 500 MSEC
; FOR RCVRDONE TO SET IN
; RCSR
CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>

MOV R5,-(SP)
MOV #SET,-(R5)
MOV RCSR,-(R5)
MOV #RCVRDONE,-(R5)
MOV #500,-(R5)
JSR PC,TIMER
MOV (SP)+,R5

; DID IT BECAME READY?
IF.ERROR THEN

;RCVRDONE DID NOT SET IN RCSR
ERRHRD 7;,, DIDNOT

ENDIF

ENDSUB

; NOW THAT IT IS SET LETS SEE IF READING THE
; BUFFER CLEARS RCVRDONE.

;READ BUFFER
LET R0 :B= @RBUF

IF #RCVRDONE SETIN @RCSR THEN

;RCVRDONE DID NOT CLEAR IN RCSR
ERRHRD 74,DIDNOT

ENDIF

ENDTST

```

```

2894
2895
2896
2897
2898
2899
2900 006166 000004
2901 006170 012767 000010 172762
2902 006176 012767 000030 172774
2903
2904
2905 006204
2906 006204 032767 000001 173002
2907 006212 001404
2908 006214
2909 006214 012767 000001 172736
2910 006222 000500
2911 006224
2912 006224
2913 006224
2914 006224 052777 000004 173032
2915 006232
2916 006232 012700 000000
2917 006236
2918 006236 005001
2919
2920
2921
2922
2923 006240
2924 006240 105077 173024
2925 006244
2926 006244
2927 006244
2928 006244 032777 004000 173006
2929 006252 001403
2930 006254
2931 006254 012700 177777
2932 006260
2933 006260 000401
2934 006262
2935 006262
2936 006262 005201
2937 006264
2938 006264
2939 006264
2940 006264 020027 177777
2941 006270 001403
2942 006272 020167 000124
2943 006276 101762
2944 006300
2945 006300
2946 006300 020167 000116
2947 006304 101407
2948
2949

:*****
:*****
:TEST 30 TEST THAT RCVRACT - RCSR 11 - SETS
:* WHEN A START BIT IS RECEIVED AND
:* CLEARS WHEN RCVRDONE - RCSR 7 - SETS
:*****
TST30: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #30,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

; THIS TEST IS 'BREAK OR HALT' SINSATIVE.
; IF #APTENV SETIN $ENV THEN
BIT #APTENV,$ENV
BEQ $116
EXIT TEST
MOV #1,$TIMES
BR TST31
;;EXIT THIS TEST
ENDIF
$116:
LET @TCSR := @TCSR SET.BY #MAINT
LET R0 := #CLR
LET R1 := #0
;LOAD A CHARACTER INTO TBUF
;WAIT FOR RCVRACT TO SET
;SEND A CHARACTER
LET @TBUF :B- #0
REPEAT
IF #RCVRACT SETIN @RCSR THEN
LET R0 := #SET
ELSE
LET R1 := R1 + #1
ENDIF
UNTIL R0 EQ #SET OR R1 HI MAX
$121:
CMP R0,#SET
BEQ $122
CMP R1,MAX
BLOS $117
$122:
IF R1 HI MAX THEN
;IT NEVER SET
;RCVRACT DID NOT SET IN RCSR.
  
```



```

2950 006306
2951 006306 104075
2952 006310
2953 006310 017700 172746
2954 006314
2955 006314 012767 000001 172636
2956 006322 000440
2957 006324
2958 006324 $123:
2959
2960
2961
2962
2963
2964
2965 006324
2966 006324 $124:
2967 006324 032777 004000 172726
2968 006332 001416
2969
2970 006334
2971 006334 032777 000200 172716
2972 006342 001411
2973 006344
2974 006344 032777 004000 172706
2975 006352 001405
2976
2977
2978 006354
2979 006354 104076
2980
2981 006356
2982 006356 012767 000001 172574
2983 006364 000417
2984 006366
2985 006366 $127:
2986 006366
2987 006366 $126:
2988 006366
2989 006366 000756
2990 006370 $125:
2991
2992
2993 006370
2994 006370 032777 000200 172662
2995 006376 001001
2996
2997 006400
2998 006400 104077
2999
3000 006402
3001 006402 $130:
3002
3003
3004
3005

```

ERRHRD 75,, DIDNOT
 LET R0 := @RBUF ; CLEAR BUFFER
 EXIT TEST
 MOV @RBUF,R0
 MOV #1,\$TIMES
 BR TST31
 ;:EXIT THIS TEST
 ENDIF
 ;CHECK FOR TIMING OF RCVRACT. CLEARING
 ;VS RCVRDONE SETTING
 WHILE #RCVRACT SETIN @RCSR DO
 BIT #RCVRACT,@RCSR
 BEQ \$125
 IF #RCVRDONE SETIN @RCSR THEN
 BIT #RCVRDONE,@RCSR
 BEQ \$126
 IF #RCVRACT SETIN @RCSR THEN
 BIT #RCVRACT,@RCSR
 BEQ \$127
 ;RCVRDONE AND RCVRACT
 ;BOTH SET
 ERRHRD 76, DONEACT
 ;NO USE CONTINUING
 EXIT TST
 ;:EXIT THIS TEST
 ENDIF
 ENDIF
 ENDDO
 BR \$124
 ;RCVRACT - 0 NOW.
 IF #RCVRDONE NOTSETIN @RCSR THEN
 BIT #RCVRDONE,@RCSR
 BNE \$130
 ;RCVRDONE DID NOT SET IN RCSR
 ERRHRD 77,,DIDNOT
 ;SET IT BACK.
 ENDIF
 ;TEST THAT READING THE RECEIVER
 ;BUFFER CLEARS RCVRDONE

```
3006  
3007 006402  
3008 006402 017700 172654      MOV    @RBUF,R0      ;READ CHAR.  
3009                                     LET R0 := @RBUF  
3010 006406  
3011 006406 032777 000200 172644  BIT    #RCVRDONE,@RCSR  IF #RCVRDONE SET IN @RCSR THEN  
3012 006414 007401      BEQ    $131  
3013                                     ;RCVRDONE DID NOT CLEAR IN RCSR  
3014 006416                                     ERRHRD 100,,DIDNOT  
3015 006416 104100      ERROR  100  
3016 006420  
3017 006420      $131:      ENDIF  
3018  
3019 006420  
3020 006420 000401      BR     TST31          EXIT  
3021 006422 070000      MAX:70000          ;::EXIT THIS TEST  
3022  
3023 006424  
3024  
                                ENDTST
```

```

3025
3026
3027
3028
3029
3030 006424 000004
3031 006426 012767 000010 172524
3032 006434 012767 000031 172536
3033
3034 006442
3035 006442 012767 006450 172440
3036
3037
3038
3039
3040
3041 006450
3042 006450 105077 172614
3043
3044 006454
3045 006454 010546
3046 006456 012745 000310
3047 006462 004767 003260
3048 006466 012605
3049
3050
3051 006470
3052 006470 105077 172574
3053
3054 006474
3055 006474 010546
3056 006476 012745 000310
3057 006502 004767 003240
3058 006506 012605
3059
3060
3061 006510
3062 006510 017704 172546
3063
3064
3065 006514
3066 006514 032704 040000
3067 006520 001005
3068
3069 006522
3070 006522 104101
3071
3072
3073 006524
3074 006524 012767 000001 172426
3075 006532 000456
3076 006534
3077 006534
3078 006534
3079
3080

```

```

*****
*****
*TEST 31 TEST THE OVERRUN BIT - RBUF 14
*****
TST31: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #31,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
BGNSUB
MOV #64,$LPERR
;OUTPUT 2 CHARACTERS WITH
;AMPLE DELAYS BETWEEN FOR RECEPTION.
;THIS SHOULD AN CAUSE OVERRUN ERROR.
;OUTPUT 1 CHARACTER
LET @TBUF :B= #0
;GO AWAY FOR 200. M SEC
WAITMS 200.
MOV R5, -(SP)
MOV #200, -(R5)
JSR PC, WAIT
MOV (SP)+, R5
;OUTPUT 2ND CHARACTER
LET @TBUF :B= #0
;LET OVERRUN HAPPEN
WAITMS 200.
MOV R5, -(SP)
MOV #200, -(R5)
JSR PC, WAIT
MOV (SP)+, R5
;READ BUFFER AND ERROR BITS
LET R4 := @RBUF
;IT DIDN'T SET
IF #ORERR NOTSET IN R4 THEN
;ORERR DID NOT SET IN RBUF
ERRHRD 101,,DIDNOT
ERROR 101
;NO USE COMPOUNDING ERRORS
EXIT TST
;:::EXIT THIS TEST
ENDIF
ENDSUB
;NOW SEE IF ERROR BIT SET WITH OVERRUN ERROR:

```

\$132:

```

3081 006534                                BGNSUB
3082 006534 012767 006542 172346        MOV    #64$, $LPERR
3083 006542                                IF #ERROR NOTSET IN R4 THEN
3084 006542 032704 100000                BIT    #ERROR, R4
3085 006546 001005                        BNE    $133
3086
3087
3088
3089 006550                                ;ERROR DID NOT SET IN RBUF
006550 104102                            ERRHRD 102,,DIDNOT
3090
3091
3092
3093
3094 006552                                ;--WHEN ORERR SET.
006552 012767 000001 172400            MOV    #1, $TIMES
3095 006560 000443                        BR     TST32
3096 006562                                ;;;EXIT THIS TEST
3097 006562                                ENDIF
3098 006562                                $133:
3099
3100
3101 006562                                BGNSUB
006562 012767 006570 172320            MOV    #64$, $LPERR
3102
3103
3104
3105 006570                                ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
006570 032777 040000 172464            BIT    #ORERR, @RBUF
3106 006576 001002                        BNE    $134
3107
3108
3109
3110 006600                                ;READING RBUF CLEARED ORERR.
006600 104103                            ERRHRD 103,ITCLRED
3111
3112
3113 006602                                ;SKIP REST OF TEST
006602 000432                            BR     TST32
3114 006604                                EXIT
3115 006604                                ;;;EXIT THIS TEST
3116 006604                                ENDIF
3117
3118
3119 006604                                BGNSUB
006604 012767 006612 172276            MOV    #64$, $LPERR
3120
3121
3122
3123
3124 006612                                ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVED
006612 105077 172452                            CLR    @TBUF
3125
3126
3127 006616                                ;SEND A CHARACTER AROUND.
006616 010546                                LET    @TBUF :B= #0
3128 006620 012745 000310                MOV    R5, -(SP)
3129 006624 004767 003116                MOV    #200, -(R5)
3130 006630 012605                        JSR    PC, WAIT
3131
3132
3133
3134 006632                                ;LET IT CIRCULATE
006632 032777 040000 172422            BIT    #ORERR, @RBUF
3135 006640 001405                        WAITMS 200.
3136 006642                                ;ORERR DID NOT CLEAR IN RBUF

```

ERRHRD 104,,DIDNOT

```
3137 006642 104104          ERROR 104
3138
3139
3140
3141 006644
3142 006644 012767 000001 172306  MOV #1,$TIMES
3143 006652 000406          BR    TST32          :::EXIT THIS TEST
3144 006654
3145 006654          $135:
3146
3147 006654
3148 006654 032777 100000 172400  BIT #ERROR,@RBUF
3149 006662 001401          BEQ  $136
3150
3151 006664
3152 006664 104105          ERROR 105          ;-AFTER RECEIVING ANOTHER CHAR
3153
3154 006666
3155 006666          $136:
3156 006666
3157 006666
3158 006666 000400          BR    TST32          :::EXIT THIS TEST
3159
3160 006670
3161
```

IF #ERROR SET IN @RBUF THEN
:ERROR DID NOT CLEAR IN RBUF
ERRHRD 105,,DIDNOT
ENDIF
ENDSUB
EXIT
.EVEN
ENDTST

```

3162
3163
3164
3165
3166
3167
3168
3169
3170 006670 000004
3171 006672 012767 000010 172260
3172 006700 012767 000032 172272
3173 006706
3174 006706 032767 000200 172304
3175 006714 001004
3176 006716
3177 006716 012767 000001 172234
3178 006724 000552
3179 006726
3180 006726
3181
3182 006726
3183 006726 032767 000001 172260
3184 006734 001404
3185 006736
3186 006736 012767 000001 172214
3187 006744 000542
3188 006746
3189 006746
3190
3191 006746
3192 006746 012767 177777 000272
3193 006754
3194 006754 012767 177777 000266
3195 006762
3196 006762 052777 000004 172274
3197
3198 006770
3199 006770 005003
3200 006772 000401
3201 006774
3202 006774 005203
3203 006776
3204 006776 020327 000017
3205 007002 003060
3206 007004
3207 007004 017700 172252
3208
3209 007010
3210 007010 116377 007170 172250
3211
3212 007016
3213 007016 005002
3214
3215 007020
3216 007020 005077 172244
3217

```

```

*****
*****
* EST 32 PROGRAMMABLE BAUD RATE TEST
* TEST AT ALL SPEEDS AVAILABLE
* A COMPARISON WILL BE MADE TO SEE
* IF NEW TIME IS LESS THAN PREVIOUS.
*****
TST32: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #32,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
IF #PBR NOTSETIN $USWR THEN
EXIT TST
MOV #1,$TIMES
BR TST33 ;;EXIT THIS TEST
ENDIF
$137:
; THIS TEST IS 'BREAK OR HALT' SINSATIVE.
IF #APTENV SETIN $ENV THEN
BIT #APTENV,$ENV
BEQ $140
EXIT TEST
MOV #1,$TIMES
BR TST33 ;;EXIT THIS TEST
ENDIF
$140:
LET OLD := #-1
LET OLD+2 := #-1
LET @TCSR := @TCSR SET.BY #MAINT
;EACH BAUD RATE
INCR R3 FROM #0 TO #15. BY #1
$142:
CLR R3
BR $141
$141:
INC R3
CMP R3,#15.
BGT $143
LET R0 := @RBUF
;CHANGE BAUDE RA c
LET @TCSRHI :=RATES(R3)
;FLAG
LET BIT := #0
;OUTPUT THE CHARACTER
LET @TBUF := #0
;INITIALIZE COUNTER

```

3218	007024						LET NEW := #0
3219	007024	005067	000212		CLR	NEW	LET NEW+2 := #0
3220	007030						WHILE BIT EQ #0 DO
3221	007030	005067	000210		CLR	NEW+2	
3222	007034						
3223	007034			\$144:			
3224	007034	005702			TST	BIT	
3225	007036	001014			BNE	\$145	
3226	007040						IF #RCVRDONE SETIN @RCSR THEN
3227	007040	032777	000200	172212	BIT	#RCVRDONE,@RCSR	
3228	007046	001403			BEQ	\$146	
3229							:DONE - ITS READY
3230	007050						LET BIT :- #1
3231	007050	012702	000001		MOV	#1,BIT	
3232	007054						ELSE
3233	007054	000404			BR	\$147	
3234	007056			\$146:			:OTHERWISE-INCREMENT TIME
3235							LET NEW :- NEW + #1
3236	007056						LET NEW+2 := NEW+2 + CARRY
3237	007056	005267	000160		INC	NEW	
3238	007062						ENDIF
3239	007062	005567	000156		ADC	NEW+2	
3240	007066						:SIGNALS DONE
3241	007066			\$147:			ENDDO
3242							
3243	007066						
3244	007066	000762			BR	\$144	
3245	007070			\$145:			
3246							
3247	007070						IF NEW+2 LO OLD+2 THEN
3248	007070	026767	000150	000152	CMP	NEW+2,OLD+2	
3249	007076	103001			BHIS	\$150	
3250							: OK
3251	007100						ELSE
3252	007100	000412			BR	\$151	
3253	007102			\$150:			: NEW+2 >= OLD+2
3254							IF NEW+2 EQ OLD+2 AND NEW LO OLD THEN
3255	007102						
3256	007102	026767	000136	000140	CMP	NEW+2,OLD+2	
3257	007110	001005			BNE	\$152	
3258	007112	026767	000124	000126	CMP	NEW,OLD	
3259	007120	103001			BHIS	\$152	
3260							:OK
3261	007122						ELSE
3262	007122	000401			BR	\$153	
3263	007124			\$152:			:NEW+2 > OLD+2 OR
3264							:(NEW+2 = OLD+2 AND
3265							: NEW >= OLD)
3266							:BAUD RATE DIDN'T CHANGE
3267							ERRHRD 126, BAUDRATE
3268	007124						
3269	007124	104126			ERROR	126	
3270	007126						ENDIF
3271	007126			\$153:			
3272	007126						ENDIF
3273	007126			\$151:			

```

3274
3275 007126 ;UPDATE OLD TIME
3276 007126 016767 000110 000112 MOV NEW,OLD LET OLD := NEW
3277 007134 LET OLD+2 := NEW+2
3278 007134 016767 000104 000106 MOV NEW+2,OLD+2
3279
3280 007142 ENDINC ;BAUD RATE
3281 007142 000714 BR $142
3282 007144 $143:
3283 007144 LET R3 :B= $USWR+1 AND #17 ; PUT BAUD BACK
3284 007144 116703 172051 MOVB $USWR+1,R3
3285 007150 110346 MOVB R3,-(SP)
3286 007152 142716 000017 BICB #17,(SP)
3287 007156 142603 BICB (SP)+,R3
3288 007160 LET @TCSRHI :B= RATES(R3) ; LIKE HE WANTED IT
3289 007160 116377 007170 172100 MOVB RATES(R3),@TCSRHI
3290
3291 007166 EXIT ;SKIP TABLE
3292 007166 000431 BR TST33 ;:EXIT THIS TEST
3293
3294 007170

```

RATES: ;A TABLE OF THE ACTUAL BYTES TO MOVE INTO THE
 ;UPPER BYTE OF XCSR FOR EACH BAUD RATE
 ;** NOTE: : THE VALUE INDICATED IN THE COLUMN 'OFFSET
 ;** INTO TABLE' CAN BE PLACED INTO BITS<11:8>
 ;** OF LOCATION '\$USWR' TO CAUSE THE CORROSPONDING
 ;** BAUD TO BE SELECTED IN THE DLV11-E UPON
 ;** COMPLETION OF THIS TEST.

					BAUD	OFFSET INTO IABLE
3301						
3302						
3303	007170	010	R0050:	.BYTE 010	50	0
3304	007171	030	R0070:	.BYTE 030	70	1
3305	007172	050	R0110:	.BYTE 050	110	2
3306	007173	070	R0135:	.BYTE 070	135	3
3307	007174	110	R0150:	.BYTE 110	150	4
3308	007175	130	R0300:	.BYTE 130	300	5
3309	007176	150	R0600:	.BYTE 150	600	6
3310	007177	170	R0200:	.BYTE 170	1200	7
3311	007200	210	R1800:	.BYTE 210	1800	10
3312	007201	230	R2000:	.BYTE 230	2000	11
3313	007202	250	R2400:	.BYTE 250	2400	12
3314	007203	270	R3600:	.BYTE 270	3600	13
3315	007204	310	R4800:	.BYTE 310	4800	14
3316	007205	330	R7200:	.BYTE 330	7200	15
3317	007206	350	R9600:	.BYTE 350	9600	16
3318	007207	370	R10000:	.BYTE 370	19200	17
3319						

```

3320 007210 040502 042125 051040 BAUDRATE: .ASCIZ /BAUD RATE DIDN'T CHANGE./
3321 007216 052101 020105 044504
3322 007224 047104 052047 041440
3323 007232 040510 043516 027105
3324 007240 000
3325 007242 .EVEN
3326 007242 000000 000000 NEW: 0.0
3327 007246 000000 000000 OLD: 0.0
3328 007252
3329

```

ENDTST

3330
3331

```

3332
3333
3334
3335
3336
3337
3338
3339
3340
3341 007252 000004
3342 007254 012767 000010 171676
3343 007262 012767 000033 171710
3344
3345 007270
3346 007270 005067 002540
3347
3348
3349 007274
3350 007274 016703 171756
3351
3352 007300
3353 007300 062703 000004
3354
3355 007304
3356 007304 010146
3357 007306 010301
3358 007310 012721 012026
3359 007314 012711 000340
3360 007320 012601
3361 007322
3362 007322 012767 007330 171560
3363
3364 007330
3365 007330 042777 000100 171726
3366
3367
3368 007336 012746 000000
3369 007342 012746 007350
3370 007346 000002
3371 007350
3372
3373
3374 007350
3375 007350 052777 000100 171706
3376
3377
3378 007356
3379 007356 010546
3380 007360 012745 000310
3381 007364 014767 002356
3382 007370 012605
3383
3384
3385 007372
3386 007372 026127 002436 000001
3387 007400 001406

```

```

*****
*****
*TEST 33 TRANSMITTER INTERRUPT LOGIC TEST
* LOGICALLY THIS IS 4 SEPARATE TESTS
* A) DOES TRANSMITTER INTERRUPT LOGIC WORK
* B) AT PRIORITY OF 0
* C) AND ONLY ONCE
* D) BUT NOT WITH INTERRUPT ENABLE CLEAR
*****
TST33: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #33,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
;;CLEAR 'INTERRUPT OCCURED' FLAG
LET INTFLAG := #0
;;GET VECTOR ADDRESS
LET R3 := DLVEC
;;FOR THE TRANSMITTER
LET R3 := R3 + #4
SETVEC R3, #INTSRV, #PR7
MOV R1,-(SP)
MOV R3,R1
MOV #INTSRV,(R1)+
MOV #PR7,(R1)
MOV (SP)+,R1
BGNSUB
;;CLEAR INTERRUPT ENABLE
LET @TCSR := @TCSR CLR.BY #XMITIE
;;SET IT TO 0
MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
MOV #65,-(SP) ;;PUT NEW PC ON STACK
RTI ;;POP NEW PC AND PS
65$:
;;NOW SET I.E. BIT
LET @TCSR := @TCSR SET.BY #XMITIE
;;LET INTERRUPT HAVE TIME TO OCCUR
WAITMS 200.
MOV R5,-(SP)
MOV #200,-(R5)
JSR PC,WAIT
MOV (SP)+,R5
;;DID EXACTLY 1 INTERRUPT OCCUR
IF INTFLAG NE #1 THEN
CMP INTFLAG,#1
BEQ $154

```

```
3388 ;NO - WAS IT 0 OR MORE THAN ONCE
3389 007402 IF INTFLAG EQ #0 THEN
3390 007402 005767 002426 TST INTFLAG
3391 007406 001002 BNE $155
3392 ;TRANSMITTER DID NOT INTERRUPT IN TIME
3393 007410 ERRHRD 106,,DIDNOT
3394 007410 104106 ERROR 106
3395 007412 ELSE
3396 007412 000401 BR $156
3397 007414 $155:
3398 ;TWICE
3399 ;TRANSMITTER INTERRUPTED TWICE
3400 007414 ERRHRD 107,,TWICE
3401 007414 104107 ERROR 107
3402 007416
3403 007416 $156:
3404 007416
3405 007416 $154:
3406 007416
3407 ENDSUB
3408 ;INTERRUPT WITHOUT INTERRUPT ENABLE SET
3409 007416 012767 007424 171464 MOV #64$,$LPERR BGNSUB
3410 ;CLEAR 'INTERRUPT OCCURED' FLAG
3411 007424 LET INTFLAG := #0
3412 007424 005067 002404 CLR INTFLAG
3413 ;CLEAR INTERRUPT ENABLE
3414 007430 LET @TCSR := @TCSR CLR.BY #XMITIE
3415 007430 042777 000100 171626 BIC #XMITIE,@TCSR
3416 ;NO INTERRUPTS SHOULD OCCUR.
3417 007436 012746 000000 MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
3418 007442 012746 007450 MOV #65$,-(SP) ;;PUT NEW PC ON STACK
3419 007446 000002 RTI ;;POP NEW PC AND PS
3420 007450 65$:
3421 ;DARE IT TO HAPPEN
3422 007450 WAITMS 2
3423 007450 010546 MOV R5,-(SP)
3424 007452 012745 000002 MOV #2,-(R5)
3425 007456 004767 002264 JSR PC,WAIT
3426 007462 012605 MOV (SP)+,R5
3427 007464 IF INTFLAG NE #0 THEN
3428 007464 005767 002344 TST INTFLAG
3429 007470 001401 BEQ $157
3430 ;INTERRUPT OCCURED WITH I E CLEARED
3431 007472 ERRHRD 110,NOTENAB
3432 007472 104110 ERROR 110
3433 007474 ENDIF
3434 007474 $157:
3435 007474 BRESET
3436 007474 000005 RESET
3437 007476 ENDSUB
3438 ;RESTORE VECTOR AREA
3439 007476 CLRVEC R3
3440 007476 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
3441 007500 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
3442 007502 012701 000003 MOV #R1,R1
3443 007506 010102 MOV R1,R2
```



```

3457
3458
3459
3460
3461
3462
3463
3464 007524 000004
3465 007526 012767 000010 171424
3466 007534 012767 000034 171436
3467
3468
3469 007542
3470 007542 010146
3471 007544 016701 171506
3472 007550 012721 012026
3473 007554 012711 000340
3474 007560 012601
3475
3476 007562
3477 007562 012767 007570 171320
3478 007570
3479 007570 005067 002240
3480
3481 007574
3482 007574 052777 000004 171462
3483
3484 007602
3485 007602 042777 000100 171450
3486
3487
3488 007610 012746 000000
3489 007614 012746 007622
3490 007620 000002
3491 007622
3492
3493
3494 007622
3495 007622 105077 171442
3496
3497
3498
3499 007626
3500 007626 010546
3501 007630 012745 177777
3502 007634 016745 171420
3503 007640 012745 000200
3504 007644 012745 000500
3505 007650 004767 001614
3506 007654 012605
3507
3508 007656
3509 007656 052777 000100 171374
3510
3511 007664
3512 007664 010546

```

```

*****
*****
*TEST 34 RECIIVER INTERRUPT LOGIC TEST
* THIS TEST COVERS ALL OF THE RECEIVER
* SIDE OF THE INTERRUPT LOGIC, BOTH DATASET
* AND CHARACTER MODES.
*****
TST34: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #34,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
;;CLEAR INTERRUPT OCCURED FLAG
;;SET UP RECEIVER INTER.VECTOR
SETVEC DLVEC,#INTSRV,#PR7
MOV R1,-(SP)
MOV DLVEC,R1
MOV #INTSRV,(R1)+
MOV #PR7,(R1)
MOV (SP)+,R1
;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-RCVRIE
BGNSUB
MOV #64,$$LPERR LET INTFLAG := #0
CLR INTFLAG
;SET MAINT. BIT
LET @TCSR := @TCSR SET.BY #MAINT
;CLEAR INTERRUPTS
LET @RCSR := @RCSR CLR.BY #RCVRIE
;CHANGE PRIORITY
;..TO 0
MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
MOV #65,-(SP) ;;PUT NEW PC ON STACK
RTI ;;POP NEW PC AND PS
65$:
;SEND A CHARACTER
LET @TBUF :B= #0
;WAIT A MAXIMUM
;OF 500 MSEC FOR
;RCVR RDY TO SET IN RCSR
CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
MOV R5,-(SP)
MOV #SET,-(R5)
MOV RCSR,-(R5)
MOV #RCVRDONE,-(R5)
MOV #500,-(R5)
JSR PC,TIMER
MOV (SP)+,R5
;SET INTERRUPT ENABLE
LET @RCSR := @RCSR SET.BY #RCVRIE
;LET IT COME IN.
WAITMS 1
MOV R5,-(SP)

```

```

MAINDEC-ZZ-CVDVA-B          MACY11 30A(1052) 23-AUG-78 15:14 PAGE 87 H 7
CVDVAC.P11 23-AUG-78 15:13 T34 RECEIVER INTERRUPT LOGIC TEST                               SEQ 0085

3513 007666 012745 000001      MOV      #1,-(R5)
3514 007672 004767 002050      JSR      PC,WAIT
3515 *007676 012605      MOV      (SP)+,R5
3516 007700
3517 007700 017700 171356      MOV      @RBUF,R0
3518
3519
3520 007704
3521 007704 026727 002124 000001      LMP      INTFLAG,#1
3522 007712 001406      BEQ      $160
3523
3524 007714
3525 007714 005767 002114      TST      INTFLAG
3526 007720 001002      BNE      $161
3527
3528 007722
3529 007722 104111      ERROR    111
3530
3531 007724
3532 007724 000401      BR       $162
3533 007726
3534
3535 007726
3536 007726 104112      ERROR    112
3537 007730
3538 007730
3539 007730
3540 007730
3541
3542 007730
3543 007730 042777 000004 171326      BIC      #MAINT,@TCR
3544
3545 007736
3546 007736 042777 000100 171314      BIC      #RCVRIE,@RCR
3547 007744
3548
3549
3550
3551
3552
3553
3554
3555
3556 007744
3557 007744 012767 007752 171136      MOV      #64,$SLPERR
3558 007752
3559 007752 032767 020000 171200      BIT      #CABLE,$USWR
3560 007760 001004      BNE      $163
3561
3562 007762
3563 007762 012767 000001 171170      MOV      #1,$TIMES
3564 007770 .000466      BR       TST35
3565 007772
3566 007772
3567
3568 007772

;RECEIVER INTERRUPT LOGIC TEST
;DID HE DO IT RIGHT?
IF INTFLAG NE #1 THEN
;NONE OCCURED
IF INTFLAG EQ #0 THEN
;RECEIVER DID NOT INTERRUPT IN TIME
ERRHRD 111,,DIDNOT
;TWICE OR MORE
ELSE
;RECEIVER INTERRUPTED TWICE
ERRHRD 112,,TWICE
ENDIF
ENDIF
;RESET MAINT. BIT.
LET @TCR := @TCR CLR.BY #MAINT
; CLEAR INTERRUPT ENABLE
LET @RCR := @RCR CLR.BY #RCVRIE
ENDSUB

;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-DATAIE
BGNSUB
IF #CABLE NOTSETIN $USWR THEN
;CAN'T TEST WITHOUT A CABLE
EXIT TST
;;;EXIT THIS TEST
ENDIF
; CLEAR 'INTFLAG'
LET INTFLAG := #0

```

```

3569 007772 005067 002036 CLR INTFLAG
3570 ;CLEAR INTERRUPTS
3571 007776 LET @RCSR := @RCSR CLR.BY #DATAIE
3572 007776 042777 000040 171254 BIC #DATAIE,@RCSR
3573 ;CHANGE PRIORITY
3574 ;...TO 0
3575 010004 012746 000000 MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
3576 010010 012746 010016 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
3577 010014 000002 RTI ;;POP NEW PC AND PS
3578 010016 64$:
3579 010016 LET @RCSR := @RCSR CLR.BY #REQSEND
3580 010016 042777 000004 171234 BIC #REQSEND,@RCSR
3581 ;SET INTERRUPT ENABLE
3582 010024 LET @RCSR := @RCSR SET.BY #DATAIE
3583 010024 052777 000040 171226 BIS #DATAIF,@RCSR
3584 010032 LET @RCSR := @RCSR SET.BY #REQSEND
3585 010032 052777 000004 171220 BIS #REQSEND,@RCSR
3586 ;LET IT COME IN.
3587 010040 WAITMS 1
3588 010040
3589 010040 010546 MOV R5,-(SP)
3590 010046 012745 000001 MOV #1,-(R5)
3591 010052 004767 001674 JSR PC,WAIT
3592 MOV (SP)+,R5
3593
3594 010054 ; DID IT DO IT RIGHT?
3595 010054 026727 001754 000001 CMP INTFLAG,#1
3596 010062 001406 BEQ $164
3597
3598 010064 ;NONE OCCURED
3599 010064 005767 001744 TST INTFLAG
3600 010070 001002 BNE $165
3601
3602 010072 ;DATAINT DID NOT INTERRUPT IN TIME
3603 010072 104113 ERROR 113 ; ERRHRD 113,,DIDNOT
3604 ;TWICE OR MORE
3605 010074 ELSE
3606 010074 000401 BR $166
3607 010076 $165:
3608
3609 010076 ; DATAINT INTERRUPTED TWICE
3610 010076 104114 ERROR 114 ; ERRHRD 114,,TWICE
3611 010100
3612 010100 $166:
3613 010100
3614 010100 $164:
3615 010100
3616 010100 042777 000040 171152 BIC #DATAIE,@RCSR
3617 010106 LET @RCSR := @RCSR CLR.BY #DATAIE
3618 010106 042777 000004 171144 BIC #REQSEND,@RCSR
3619 010114 LET @RCSR := @RCSR CLR.BY #REQSEND
3620 ENDSUB
3621 010114 LET R4 := @DLVEC
3622 010114 017704 171136 MOV @DLVEC,R4
3623 010120 CLRVEC R4
3624 010120 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
    
```

3625	010122	010246		MOV	R2,-(SP)	::PUSH R2 ON STACK
3626	010124	012701	000004	MOV	#R4,R1	
3627	010130	010102		MOV	R1,R2	
3628	010132	062702	000002	ADD	#2,R2	
3629	010136	010221		MOV	R2,(R1)+	
3630	010140	005011		CLR	(R1)	
3631	010142	012602		MOV	(SP)+,R2	::POP STACK INTO R2
3632	010144	012601		MOV	(SP)+,R1	::POP STACK INTO R1
3633	010146					ENDTST


```

3634
3635
3636
3637
3638
3639 010146 000004
3640 010150 012767 000001 171002
3641 010156 012767 000035 171014
3642
3643 010164
3644 010164 052777 000004 171072
3645
3646
3647
3648 010172 012746 000000
3649 010176 012746 010204
3650 010202 000002
3651 010204
3652
3653 010204
3654 010204 162705 000002
3655 010210 004767 001432
3656 010214 012501
3657
3658 010216
3659 010216 017700 171040
3660
3661
3662 010222
3663 010222 005002
3664 010224 000401
3665 010226
3666 010226 005202
3667 010230
3668 010230 020227 000377
3669 010234 003047
3670
3671
3672
3673
3674 010236
3675 010236 010546
3676 010240 012745 177777
3677 010244 016745 171014
3678 010250 012745 000200
3679 010254 012745 000500
3680 010260 004767 001204
3681 010264 012605
3682
3683
3684 010266
3685 010266 110277 170776
3686
3687 010272
3688 010272 010546
3689 010274 012745 177777

```

```

*****
*****
*TEST 35 TEST ACTUAL DATA TRANSFERED
* NON-INTERRUPT MAINTENANCE BIT SET
*****
TST35: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #35,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
;;SET MAINT. BIT
LET @TCSR := @TCSR SET.BY #MAINT
;;CHANGE PRIORITY
;;TO 0
MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
MOV #64$,-(SP) ;;PUT NEW PC ON STACK
RTI ;;POP NEW PC AND PS
64$:
;;GET DATA MASK.
CALL DATLNG OUT <R1>
SUB #1*2,R5
JSR PC,DATLNG
MOV (R5)+,R1
;; START CLEAN
LET R0 := @RBUF
;;ALL BINARY CHAR.
INCR R2 FROM #0 TO #377 BY #1
$170:
BR R2 $167
$167:
INC R2
CMP R2,#377
BGT $171
;;TRANSMIT CHAR IN R2
CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
MOV R5,-(SP)
MOV #SET,-(R5)
MOV TCSR,-(R5)
MOV #XMITRDY,-(R5)
MOV #500,-(R5)
JSR PC,TIMER
MOV (SP)+,R5
;;TRANSMIT IT
LET @TBUF :B R2
CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
MOV R5,-(SP)
MOV #SET,-(R5)

```

```

3690 010300 016745 170754      MOV      RCSP,-(R5)
3691 010304 012745 000200      MOV      #RCVRDNF,-(R5)
3692 010310 012745 000500      MOV      #500,-(R5)
3693 010314 004767 001150      JSR      PC,TIMER
3694 010320 012605              MOV      (SP)+,R5
3695
3696 010322              ;AND SAVE IT
3697 010322 017703 170734      MOV      @RBUF,R3      LET R3 := @RBUF
3698
3699
3700              ;COMPARE TO SEE IF WE RECEIVED IT ALL
3701
3702              ;CLEAN OFF NON-DATA BITS
3703              ;ON BOTH TRANSMITTED AND
3704 010326              LET R4 := R2 CLR.BY R1
3705 010326 010204      MOV      R2,R4
3706 010330 040104      BIC      R1,R4
3707 010332              LET R3 := R3 CLR.BY R1
3708 010332 040103      BIC      R1,R3
3709
3710              ;RECEIVED DATA
3711 010334              IF R4 NE R3 THEN
3712 010334 020403      CMP      R4,R3
3713 010336 001405      BEQ     $172
3714
3715 010340              ;DATA COMPARE ERROR
3716 010340 104116      ERROR   116          ERRHRD 116,COMP,SBWAS
3717 010342              EXIT TST ; ON ERROR
3718 010342 012767 000001 170610  MOV      #1,$TIMES
3719 010350 000404      BR       TST36      :::EXIT THIS TEST
3720 010352
3721 010352              $172:
3722 010352
3723 010352 000725      BR       $170
3724 010354              $171:
3725
3726
3727 010354              ;RESET MAINT. BIT.
3728 010354 042777 000004 170702  BIC      #MAINT,@TCSR      LET @TCSR := @TCSR CLR.BY #MAINT
3729 010362
3730
3731
3732
    
```

ENDTST

```

3733
3734
3735
3736
3737 010362 000004
3738 010364 012767 000001 170566
3739 010372 012767 000036 170600
3740 010400
3741 010400 032767 020000 170612
3742 010406 001004
3743
3744 010410
3745 010410 012767 000001 170542
3746 010416 000474
3747 010420
3748 010420
3749
3750 010420
3751 010420 042777 000004 170636
3752
3753
3754 010426 012746 000000
3755 010432 012746 010440
3756 010436 000002
3757 010440
3758
3759 010440
3760 010440 162705 000002
3761 010444 004767 001176
3762 010450 012501
3763 010452
3764 010452 017700 170604
3765
3766 010456
3767 010456 005002
3768 010460 000401
3769 010462
3770 010462 005202
3771 010464
3772 010464 020227 000377
3773 010470 003047
3774
3775
3776
3777
3778 010472
3779 010472 010546
3780 010474 012745 177777
3781 010500 016745 170560
3782 010504 012745 000200
3783 010510 012745 000500
3784 010514 004767 000750
3785 010520 012605
3786
3787
3788 010522

```

```

*****
*****
*TEST 36 TEST DATA THROUGH CABLE
*****
TST36: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #36,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
;;IF #CABLE NOTSETIN $USWR THEN
BIT #CABLE,$USWR
BNE $173 ;CAN'T TEST WITHOUT A CABLE
EXIT TST
MOV #1,$TIMES
BR TST37 ;;EXIT THIS TEST
ENDIF
$173:
;;DON'T USE MAINT.
LET @TCSR := @TCSR CLR.BY #MAINT
;;CHANGE PRIORITY
;;TO 0
MOV #PRO,-(SP) ;;PUT NEW PS ON STACK
MOV #64$,-(SP) ;;PUT NEW PC ON STACK
RTI ;;POP NEW PC AND PS
$64$:
;;GET DATA MASK
CALL DATLNG OUT <R1>
SUB #1*2,R5
JSR PC,DATLNG
MOV (R5)+,R1
LET R0 := @RBUF ; START CLEAN
;;BINARY COUNT PATTERN
INCR R2 FROM #0 TO #377 BY #1
$175:
BR $174
$174:
INC R2
CMP R2,#377
BGT $176
$176:
;;TRANSMIT THE CHAR. IN R2.
CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
MOV R5,-(SP)
MOV #SET,-(R5)
MOV TCSR,-(R5)
MOV #XMITRDY,-(R5)
MOV #500,-(R5)
JSR PC,TIMER
MOV (SP)+,R5
;;START IT ON ITS WAY
LET @TBUF :B= R2

```

```

3789 010522 110277 170542      MOVB   R2,@TBUF
3790 010526
3791 010526 010546      MOV    R5,-(SP)
3792 010530 012745 177777      MOV    #SET,-(R5)
3793 010534 016745 170520      MOV    RCSR,-(R5)
3794 010540 012745 000200      MOV    #RCVRDONE,-(R5)
3795 010544 012745 000500      MOV    #500,-(R5)
3796 010550 004767 000714      JSR   PC,TIMER
3797 010554 012605      MOV    (SP)+,R5
3798
3799
3800 010556
3801 010556 017703 170500      MOV    @RBUF,R3
3802
3803
3804 010562
3805 010562 010204      MOV    R2,R4
3806 010564 040'04      BIC   R1,R4
3807 010566
3808 010566 040103      BIC   R1,R3
3809
3810
3811 010570
3812 010570 020403      CMP   R4,R3
3813 010572 001405      BEQ  $177
3814
3815 010574
3816 010574 104117      ERROR 117
3817 010576
3818 010576 012767 000001 170354      MOV   #1,$TIMES
3819 010604 000401
3820 010606
3821 010606      $177:
3822
3823 010606
3824 010606 000725      $176: BR   $175
3825 010610
3826
3827
3828
3829 010610
3830
3831
3832
3833
  
```

CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
 :RETRIEVE
 LET R3 := @RBUF
 :STRIP OFF JUNK ON BOTH
 LET R4 := R2 CLR.BY R1
 LET R3 := R3 CLR.BY R1
 :WE HAVE TROUBLE
 IF R4 NE R3 THEN
 :DATA COMPARE ERROR
 ERRHRD 117,COMP,SBWAS
 EXIT TST ; ON ERROR
 :::EXIT THIS TEST
 ENDIF
 ENDINC ; R2
 ENDTST

3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889

010610 000004
010612 012767 000001 170340
010620 012767 000037 170352

010626
010626 032767 000001 170360
010634 C01404
010636
010636 012767 000001 170314
010644 000550
010646
010646
010646 162705 000002
010652 004767 000770
010656 012503

010660 012746 000000
010664 012746 010672
010670 000002
010672
010672 016701 170360

010676
010676 012721 011072
010702
010702 012721 000340

010706
010706 012721 011030
010712

```
*****  
*****  
*TEST 37 FULL DATA TRANSFER WITH INTERRUPTS  
* AND MAINTENANCE MODE.  
*****  
TST37: SCOPE  
MOV #1,$TIMES ;;DO 1 ITERATION  
MOV #37,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
  
; THIS TEST IS 'BREAK OR HALT' SENSITIVE.  
IF #APTENV SETIN $ENV THEN  
BIT #APTENV,$ENV  
BEQ $200  
EXIT TEST  
MOV #1,$TIMES  
BR TST40  
;;EXIT THIS TEST  
ENDIF  
$200:  
;GET DATA MASK  
CALL DATLNG OUT <R3>  
SUB #1*2,R5  
JSR PC,DATLNG  
MOV (R5)+,R3  
  
; THIS TEST WILL RUN BOTH TRANSMITTER AND  
; RECIEVER AT FULL SPEED TESTING  
; THE ABILITY OF THE MODULE  
; TO HANDLE INTERRUPTS FROM BOTH SIDES  
; AT ONCE. ALSO, THE DOUBLE BUFFERING LOGIC  
; OF THE UART WILL BE FULLY TESTED.  
; THIS TEST WILL TRANSFER A MAXIMUM OF 400(8)  
; CHARACTERS THROUGH THE MODULE, BUT IF AN ERROR  
; IS DETECTED BY THE TEST A PREMATURE SHUTDOWN OCCURS.  
  
;CHANGE PRIORITY  
;..TO 0  
MOV #PRO,-(SP) ;;PUT NEW PS ON STACK  
MOV #64$,-(SP) ;;PUT NEW PC ON STACK  
RTI ;;POP NEW PC AND PS  
64$:  
;GET VECTOR ADDRESS  
LET R1 := DLVEC  
;RCVR VECTOR  
LET (R1)+ := #REC  
LET (R1)+ := #PR7  
;POINT TO TRANSMITTER VECTOR  
;AND SET IT UP ALSO  
LET (R1)+ := #TRAN  
LET (R1) := #PR7
```

```

3890 010712 012711 000340      MOV      #PR7,(R1)
3891
3892
3893 010716                      ; INITIALIZE COUNTERS
3894 010716 012701 177777      MOV      #-1,R1      LET R1 := #-1
3895                                ;RECEIVER STORAGE
3896 010722                      LET R2 := #0
3897 010722 005002              CLR      R2
3898                                ;# OF RECEIVED CHAR. COUNT.
3899 010724                      LET R4 := #-1
3900 010724 012704 177777      MOV      #-1,R4
3901
3902                                ; CLEAR ERROR COUNT.
3903 010730                      LET ERRCNT := #0
3904 010730 005067 000066      CLR      ERRCNT
3905
3906 010734                      BRESET   ;SET UP ALL REGISTERS
3907 010734 000005              RESET
3908                                ;SET UP MAINTENANCE
3909 010736                      LET @TCSR := @TCSR SET.BY #MAINT
3910 010736 052777 000004 170320  BIS      #MAINT,@TCSR
3911
3912                                ;SET I.E. IN TRANSMITTER
3913 010744                      LET @TCSR := @TCSR SET.BY #XMITIE
3914 010744 052777 000100 170312  BIS      #XMITIE,@TCSR
3915                                ;AND RECEIVER
3916 010752                      LET @RCSR := @RCSR SET.BY #RCVRIE
3917 010752 052777 000100 170300  BIS      #RCVRIE,@RCSR
3918
3919                                ;NOW WE WAIT UNTIL R4 COUNT (RECEIVED) IS EQUAL
3920                                REPEAT
3921 010760                                UNTIL R4 EQ NUMBER OR ERRCNT GT #0
3922 010760                                $201:
3923 010760                                CMP      R4,NUMBER
3924 010760 020467 000040      BEQ      $202
3925 010764 001403              TST     ERRCNT
3926 010766 005767 000030      BLE     $201
3927 010772 003772              $202:
3928 010774
3929
3930                                ;DATA COMPARE ERRORS.
3931 010774                      IF ERRCNT NE #0 THEN
3932 010774 005767 000022      TST     ERRCNT
3933 011000 001401              BEQ     $203
3934                                ;DATA COMPARE ERROR
3935 011002                      ERRHRD 120,COMP,FIRST
3936 011002 104120              ERROR   120
3937 011004                      ENDIF
3938 011004                      $203:
3939
3940 011004                      LFT @TCSR := @TCSR CLR.BY #XMITIE
3941 011004 042777 000100 170252  BIC     #XMITIE,@TCSR
3942 011012                      LET @RCSR := @RCSR CLR.BY #XMITIE
3943 011012 042777 000100 170240  BIC     #XMITIE,@RCSR
3944
3945 011020                      EXIT    ;SKIP OVER SUPPORT ROUTINES & STORAGE
  
```

```

3946 011020 000462          BR      TST40          :::EXIT THIS TEST
3947
3948 011022 000000          ERRCNT: 0
3949 011024 000400          NUMBER: 400
3950 011026      000          SB:      .BYTE 0
3951 011027      000          WAS:     .BYTE 0
3952
3953
3954
3955          ;*****
3956          ;TRANSMIT INTERRUPT HANDLER
3957
3958 011030          BGNSRV  TRAN
3959 011030
3960
3961          ;*****
3962          ;INCREMENT CHAR COUNT
3963 011030          LET R1 := R1 + #1
3964 011030 005201          INC      R1
3965
3966 011032          ;SET UP FOR TRANSFER
3967 011032 010167 000030          MOV     R1,HOLD
3968 011036 040367 000024          BIC     R3,HOLD
3969
3970 011042          ;AND SEND.
3971 011042 016777 000020 170220          MOV     HOLD,@TBUF
3972
3973 011050          ;ALL DONE
3974 011050 020167 177750          CMP     R1,NUMBER
3975 011054 001003          BNE     $204
3976
3977 011056          ;STOP INTERRUPT PROCESSING
3978 011056 042777 000100 170200          BIC     #XMITIE,@TCSR
3979 011064
3980 011064          LET @TCSR := @TCSR CLR.BY #XMITIE
3981          $204:
3982 011064 000401          BR      ZZZ
3983          ; EXIT SRV
3984 011066 000000          HOLD:0
3985
3986 011070          ZZZ:
3987 011070 000002          RTI
3988
3989
3990          ;*****
3991          ;RECEIVER INTERRUPT HANDLER
3992          BGNSRV  REC
3993 011072
3994 011072
3995          REC:
3996          ;*****
3997          ;COUNT THIS CHAR.
3998 011072          LET R4 := R4 + #1
3999 011072 005204          INC      R4
4000
4001 011074          ;GET CHAR IN + MASK IT
          LET R2 := @RBUF CLR.BY R3
  
```

```

4002 011074 017702 170162      MOV  @RBUF,R2
4003 011100 040302      BIC  R3,R2
4004
4005 011102
4006 011102 010467 000054      MOV  R4,RHLD
4007 011106 040367 000050      BIC  R3,RHLD
4008
4009
4010 011112
4011 011112 020267 000044      CMPL R2,RHLD
4012 011116 001412      BEQ  $205
4013
4014 011120
4015 011120 005767 177676      IST  ERRCNT
4016 011124 001005      BNE  $206
4017
4018 011126
4019 011126 116767 000030 177672      MOVB RHLD,SB
4020 011134
4021 011134 110267 177667      MOVB R2,WAS
4022 011140
4023 011140      $206:
4024
4025 011140
4026 011140 005267 177656      INC  ERRCNT
4027 011144
4028 011144      $205:
4029
4030
4031 011144
4032 011144 020467 177654      CMP  R4,NUMBER
4033 011150 001003      BNE  $207
4034
4035 011152
4036 011152 042777 000100 170100      BIC  #RCVRIE,@RCSR
4037
4038 011160
4039 011160      $207:
4040
4041
4042 011160 000401      BR   ZZZZ
4043
4044 011162 000000
4045 011164      ZZZZ:
4046 011164
4047 011164 000002      RTI
4048
4049 011166
4050
4051
4052
  
```

;RHL D WILL CONTAIN EXPECTED INPUT
 LET RHL D := R4 CLR.BY R3

;DO THEY COMPARE
 IF R2 NE RHL D THEN

;FIRST ERROR
 IF ERRCNT EQ #0 THEN

;SAVE RECORD OF FIRST MISS
 LET SB :B= RHL D

LET WAS :B= R2

ENDIF

;COUNT IT.
 LET ERRCNT := ERRCNT + #1

ENDIF

;ALL DONE?
 IF R4 EQ NUMBER THEN

;STOP RECEIVER INTERRUPTS
 LET @RCSR := @RCSR CLR.BY #RCVRIE

;MAIN REPEAT LOOP IS CHECKING
 ENDIF

;FOR 'R4 - NUMBER' ALSO
 ; EXIT SRV

RHL D:0

ENDSRV

ENDTST


```

4053      :*****
4054      :*****
4055      :*TEST 40      TEST BREAK GENERATION LOGIC
4056      :*          TRANSMIT KNOWN CHAR WITH BREAK SET
4057      :*          AND COMPARE RECEIVED WITH 0.
4058      :*****
4059 011166 000004      TST40: SCOPE
4060 011170 012767 000010 167762      MOV #10,$TIMES      ;;DO 10 ITERATIONS
4061 011176 012767 000040 167774      MOV #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4062 011204          BIT #BRK,$USWR      IF #BRK NOTSET IN $USWR THEN
4063 011204 032767 010000 170006      BNE $210
4064 011212 001004          EXIT TST
4065 011214          MOV #1,$TIMES
4066 011214 012767 000001 167736      BR TST41      ;;EXIT THIS TEST
4067 011222 000452          ENDIF
4068 011224          $210:
4069 011224          ;SET MAINTENANCE BIT
4070          LET @TCSR := @TCSR SET.BY #MAINT
4071 011224          ; CLEAR RCVRDONE JUST IN CASE
4072 011224 052777 000004 170032      BIS #MAINT,@TCSR      LET RO := @RBUF
4073          ;
4074          ;
4075 011232          ;SET BREAK BIT
4076 011232 017700 170024      MOV @RBUF,RO      LET @TCSR := @TCSR SET.BY #BREAK
4077          ;
4078          ;
4079          ;NON-ZERO CHAR. '*'
4080 011236          LET @TBUF := #252
4081 011236 052777 000001 170020      BIS #BREAK,@TCSR      (ALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
4082          ;
4083 011244          MOV R5,-(SP)
4084 011244 012777 000252 170016      MOV #252,@TBUF
4085 011252          MOV R5,-(R5)
4086 011252 010546          MOV RCSR,-(R5)
4087 011254 012745 177777      MOV #RCVRDONE,-(R5)
4088 011260 016745 167774      MOV #500,-(R5)
4089 011264 012745 000200      JSR PC,TIMER
4090 011270 012745 000500      MOV (SP)+,R5
4091 011274 004767 000170          IF.ERROR THEN
4092 011300 012605          ; RECIEVER DONE DID NOT SET
4093 011302          FRRHRD 115
4094 011302 103001          ERROR 115
4095          ;
4096 011304          ENDIF
4097 011304 104115          $211:
4098 011306          IFB @RBUF NE #0 THEN
4099 011306          ; BREAK DID NOT EQUAL 0
4100          ERRHRD 121 ,BADBRK
4101 011306          $212:
4102 011306 105777 167750      TSTB @RBUF
4103 011312 001401          BEQ $212
4104          ;
4105 011314          ;
4106 011314 104121          ERROR 121
4107 011316          ;
4108 011316          ;
  
```

```
4109 011316  
4110 011316 000005  
4111 011320  
4112 011320 000413  
4113 011322 051102 040505 020113  
4114 011330 044504 020104 047516  
4115 011336 020124 050505 040525  
4116 011344 020114 000060  
4117  
4118 011350  
  
RESET  
BR TST41  
BRESET ;CLEAN UP  
EXIT  
:::EXIT THIS TEST  
BADBRK: .ASCIZ /BREAK DID NOT EQUAL 0/  
  
ENDTST
```

```

4119
4120
4121
4122
4123 011350 000004
4124 011352 012767 000001 167600
4125 011360 104401 011366
4126 011364 000404
4127
4128 011376
4129 011376 016746 167652
4130 011402 104402
4131 011404 104401 011412
4132 011410 000405
4133
4134 011424
4135 011424 016746 167626
4136 011430 104402
4137 011432 104401 011440
4138 011436 000405
4139
4140 011452
4141 011452 016746 167434
4142 011456 104405
4143 011460 005067 167426
4144 011464 000167 170254

::*****
:*TEST 41 NOT A TEST - SEND BACK TO LOOP
::*****
TST41: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
TYPE ,65$ ;;TYPE ASCIZ STRING
BR ,64$ ;;GET OVER THE ASCIZ
::65$: .ASCIZ <CRLF>*CSR: *
64$: MOV DLADD,-(SP) ;;SAVE DLADD FOR TYPEOUT
TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE ,07$ ;;TYPE ASCIZ STRING
BR ,66$ ;;GET OVER THE ASCIZ
::67$: .ASCIZ *,VECTOR: *
66$: MOV DLVEC,-(SP) ;;SAVE DLVEC FOR TYPEOUT
TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE ,69$ ;;TYPE ASCIZ STRING
BR ,68$ ;;GET OVER THE ASCIZ
::69$: .ASCIZ *,ERRORS: *
68$: MOV $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
CLR $ERTTL ; RESFT FOR NEXT DEVICE/PASS
JMP LC ? ; BACK UP TO THE BEGINNING
  
```

4145
 4146
 4147
 4148 011470
 4149 011470
 4150
 4151
 4152
 4153
 4154
 4155
 4156
 4157
 4158
 4159
 4160
 4161
 4162
 4163
 4164
 4165
 4166
 4167
 4168
 4169
 4170
 4171
 4172
 4173 011470
 4174 011470 016567 000004 000136
 4175 011476
 4176 011476 016567 000000 000132
 4177 011504
 4178 011504 112767 000000 000126
 4179
 4180
 4181
 4182
 4183 011512
 4184 011512
 4185
 4186 011512
 4187 011512 036577 000002 000114
 4188 011520 001004
 4189 011522
 4190 011522 112767 000000 000111
 4191 011530
 4192 011530 000403
 4193 011532
 4194 011532
 4195 011532 112767 177777 000101
 4196 011540
 4197 011540
 4198
 4199
 4200 011540

```

        ;:BGNMOD          SUBS
        ;:*****
        ROUTINE TIMER <HOWLONG,WHICHBIT,REG,SETCLR>
TIMER:
;* ROUTINE:TIMER
;* THIS ROUTINE IS USED TO TEST THE STATUS OF ANY BIT
;* IN ANY REGISTER.
;* INPUT:
;*     HOWLONG    THE MAXIMUM AMOUNT OF TIME TO SPEND IN
;*                 THIS ROUTINE.
;*     WHICHBIT   A MASK WITH THE BIT(S) SET THAT ARE
;*                 TO BE CHECKED.
;*     REG        A POINTER TO THE REGISTER TO BE CHECKED
;*     SETCLR     THE DESIRED RESULTS
;*                 EITHER #SET OR #CLEAR
;* OUTPUT:
;*     THE 'C' BIT IS SET TO INDICATE AN ERROR
;*     BUT IT IS TESTED BY THE IF.ERROR STATEMENT
;*
;*     NOTE:: THE USE OF (R5) IS PART OF THE LINKAGE
;*             MECHANISM BETWEEN THE CALLER AND THE CALLED
        ;:*****
    
```

```

000001      TRUE= 1
000000      FALSE= 0
016567 000004 000136      MOV     REG(R5),REGSAV      LET     REGSAV :- REG(R5) ; GET POINTER TO REGIST
016567 000000 000132      MOV     HOWLONG(R5),TIMSAV    LET     TIMSAV :- HOWLONG(R5) ; 'SAVE HOWLONG FOR
112767 000000 000126      MOVB   #FALSE,FLAG      LET     FLAG :B #FALSE ; INITIALIZE THE EXIT FLA
;           ; START OF AN INFINITE LOOP
LOOP
;           ; TEST TO SEE IF WHICHBIT IS SET
IF         WHICHBIT(R5) NOTSETIN @REGSAV THEN
036577 000002 000114      BIT     WHICHBIT(R5),@REGSAV
001004      BNE     $217
112767 000000 000111      MOVB   #CLR,HOLDSC      LET     HOLDSC :B #CLR
ELSE
000403      BR     $220
112767 177777 000101      MOVB   #SET,HOLDSC     LET     HOLDSC :B #SET ; REMEMBER THIS
ENDIF
;           ; NOW SEE IF THAT WAS WHAT WE WANTED
011540      IFB     HOLDSC EQ SETCLR(R5) THEN
    
```

```

4201 011540 126765 000075 000006      CMPB   HOLDSC,SETCLR(R5)
4202 011546 001003                    BNE    $221
4203                                     ; JUST THE THING WE NEEDED
4204 011550                                     LET    FLAG :B= #TRUE
4205 011550 112767 000001 000062      MOVB   #TRUE,FLAG
4206 011556                                     ENDIF
4207 011556      $221:
4208
4209 011556                                     EXIFB  FLAG EQ #TRUE OR TIMSAV LE #0
4210 011556 126727 000056 000001      CMPB   FLAG,#TRUE
4211 011564 001414                    BEQ    $216
4212 011566 005767 000044                    TST   TIMSAV
4213 011572 003411                    BLE    $216
4214                                     ; ONE WAY OR THE OTHER, WE ARE DONE
4215                                     ; IF WE ARE STILL HERE THEN HANG AROUND A WHILE
4216
4217 011574                                     WAITMS 1          ;WAIT FOR 10 MILLI-SECONDS
4218 011574 010546                    MOV    R5,-(SP)
4219 011576 012745 000001                    MOV    #1,-(R5)
4220 011602 004767 000140                    JSR   PC,WAIT
4221 011606 012605                    MOV    (SP)+,R5
4222 011610
4223 011610 005367 000022                    DEC   TIMSAV
4224 011614
4225 011614 000736                    BR     $215
4226 011616      $216:
4227
4228                                     ; ONLY 2 WAYS TO GET HERE
4229                                     ; 1). WE RAN OUT OF TIME---ERROR
4230                                     ; 2). THE BIT IS IN THE CORRECT CONDITION--GOOD . .
4231
4232 011616                                     IFB   FLAG EQ #TRUE THEN
4233 011616 126727 000016 000001      CMPB   FLAG,#TRUE
4234 011624 001001                    BNE    $222
4235                                     RETURN NO.ERROR      ; GOOD
4236 011626 000405                    BR     $213
4237 011630
4238 011630      $222:
4239 011630
4240 011630 000261                    SEC
4241 011632 000404                    BR     $214
4242
4243 011634 000000                                     REGSAV: .WORD 0
4244 011636 000000                                     TIMSAV: .WORD 0
4245 011640 000                                     FLAG:   .BYTE 0
4246 011641 000                                     HOLDSC: .BYTE 0
4247
4248                                     ; WE ARE DONE GO BACK HOME
4249 011642      $213:                    ENDRTN
4250 011642 000241
4251 011644      $214:
4252 011644 000207                    RTS   PC
    
```

4253
4254
4255 011646
4256 011646
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270 011646
4271 011646 005065 000000
4272 011652
4273 011652 016767 167342 000062
4274 011660 016746 000056
4275 011664 042716 000017
4276 011670 042667 000046
4277
4278 011674
4279 011674 012767 000001 167370
4280 011702 000402
4281 011704
4282 011704 005267 167362
4283 011710
4284 011710 026767 167356 000024
4285 011716 003006
4286 011720
4287 011720 006365 000000
4288 011724
4289 011724 052765 000001 000000
4290 011732
4291 011732 000764
4292 011734
4293 011734
4294 011734 005165 000000
4295 011740
4296 011740 000401
4297 011742 000000
4298 011744
4299 011744
4300 011744
4301 011744 000207

ROUTINE DATLNG <MASK>

DATLNG:

* ROUTINE:DATLNG

* THIS ROUTINE SETS UP A MASK FOR DATA, WITH
* INPUT - NOTHING IS PASSED TO THIS ROUTINE
* BUT GLOBAL INFORMATION IS ASSUMED TO EXIST:
* \$USWR-- THE WORD FOR SOFTWARE PARAMETERS
* DATA-- A MASK FOR THE LOCATION OF THE OCTAL
* NUMBER OF DATA BITS

* OUTPUT----

* MASK-- A MASK OF BINARY ONES RIGHT-JUSTIFIED
* THE NUMBER OF WHICH IS DEFINED IN \$USWR WORD.

LET MASK(R5) := #0 ; START

LET NUMBR := \$USWR AND #DATA

CLR MASK(R5)
MOV \$USWR, NUMBR
MOV NUMBR, -(SP)
BIC #DATA, (SP)
BIC (SP)+, NUMBR

INCR I FROM #1 TO NUMBR BY #1

MOV #1, I
BR \$225

\$226:

INC I

\$225:

CMP I, NUMBR
BGT \$227

LET MASK(R5) := MASK(R5) SHIFT #1

LET MASK(R5) := MASK(R5) SET.BY #1

ENDINC

BIS #1, MASK(R5)

BR \$226

\$227:

LET MASK(R5) := COMP MASK(R5)

COM MASK(R5)

RETURN

BR \$223
NUMBR:0

ENDRTN

\$223:

\$224:

RTS PC

4302
 4303
 4304 011746
 4305 011746
 4306
 4307
 4308
 4309
 4310
 4311
 4312
 4313 011746 010146
 4314 011750 010246
 4315 011752 010346
 4316 011754
 4317 011754 016501 000000
 4318 011760
 4319 011760 012702 000001
 4320 011764 000402
 4321 011766
 4322 011766 062702 000001
 4323 011772
 4324 011772 020201
 4325 011774 101010
 4326 011776
 4327 011776 005003
 4328 012000 000401
 4329 012002
 4330 012002 005203
 4331 012004
 4332 012004 020327 000100
 4333 012010 003001
 4334 012012
 4335 012012 000773
 4336 012014
 4337 012014
 4338 012014 000764
 4339 012016
 4340 012016 012603
 4341 012020 012602
 4342 012022 012601
 4343 012024
 4344 012024
 4345 012024
 4346 012024 000207

```

*****
ROUTINE WAIT <TIME>
WAIT:
;* ROUTINE:WAIT
;* THIS ROUTINE IS USED TO DELAY EXECUTION OF THE
;* MAIN PROGRAM FOR A SPECIFIED AMOUNT OF TIME.
;* THIS IS ACCOMPLISHED BY INCREMENTING A
;* REGISTER UP TO A LIMIT. THE INNER LOOP IS SET
;* TO APPROXIMATE 1 MILLI SEC.
*****
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
LET R1 := TIME(R5)
MOV TIME(R5),R1
INCRU R2 FROM #1 TO R1 BY #1
MOV #1,R2
BR $232
$233: ADD #01,R2
$232: CMP R2,R1
BHI $234
INCR R3 FROM #0 TO #100 BY #1
CLR R3
BR $235
$236: INC R3
$235: CMP R3,#100
BGT $237
ENDINC
BR $236
$237: ENDINC
BR $233
$234: MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
ENDRTN
$230:
$231: RTS PC
    
```

4347
4348
4349
4350 012026
4351
4352
4353
4354
4355
4356
4357
4358
4359 012026
4360 012026 005267 000002
4361 012032
4362 012032 000002
4363 012034 000000

```
.SBTTL INTSRV INTERRUPT SERVICE ROUTINE  
:*****  
INTSRV:  
:* SERVICE ROUTINE: INTSRV  
:* THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT  
:* 'INTFLAG' EACH TIME IT IS CALLED. IT ASSUMES  
:* THAT THE MAIN CALLING ROUTINE WILL KNOW WHAT  
:* TO LOOK FOR.  
:*****  
;ADD 1 TO 'INTERRUPT OCCURED' FLAG  
LET INTFLAG :- INTFLAG + #1  
INC INTFLAG  
ENDSRV ;THAT'S ALL  
RTI  
.NTFLAG: 0
```



```

4364
4365 012036 ROUTINE MYTYPE
4366 012036 MYTYPE:
4367 ::*****
4368 012036 104401 012044 TYPE ,65$ ::TYPE ASCIZ STRING
4369 012042 000405 BR ,64$ ::GET OVER THE ASCIZ
4370 ::65$: .ASCIZ <CRLF>*TEST # *
4371 012056 64$:
4372 012056 016746 167116 MOV $TESTN,-(SP) ::SAVE $TESTN FOR TYPEOUT
4373 012062 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4374 012064 104401 012072 TYPE ,67$ ::TYPE ASCIZ STRING
4375 012070 000405 BR ,66$ ::GET OVER THE ASCIZ
4376 ::67$: .ASCIZ *,ERROR # *
4377 012104 66$:
4378 012104 116767 167004 167064 MOVB $ITEMB,$FATAL ; APT FATAL ERROR NUMBER
4379 012112 016746 167060 MOV $FATAL,-(SP) ::SAVE $FATAL FOR TYPEOUT
4380 012116 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4381 012120 104401 012126 TYPE ,69$ ::TYPE ASCIZ STRING
4382 012124 000404 BR ,68$ ::GET OVER THE ASCIZ
4383 ::69$: .ASCIZ *,PC = *
4384 012136 68$:
4385 012136 016746 166754 MOV $ERRPC,-(SP) ::SAVE $ERRPC FOR TYPEOUT
4386 012142 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4387 012144 104401 012152 TYPE ,71$ ::TYPE ASCIZ STRING
4388 012150 000404 BR ,70$ ::GET OVER THE ASCIZ
4389 ::70$: .ASCIZ *,CSR: *
4390 012162 70$:
4391 012162 016746 167066 MOV DLADD,-(SP) ::SAVE DLADD FOR TYPEOUT
4392 012166 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4393 012170 104401 012176 TYPE ,73$ ::TYPE ASCIZ STRING
4394 012174 000405 BR ,72$ ::GET OVER THE ASCIZ
4395 ::73$: .ASCIZ *,VECTOR: *
4396 012210 72$:
4397 012210 016746 167042 MOV DLVEC,-(SP) ::SAVE DLVEC FOR TYPEOUT
4398 012214 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4399 012216
4400 012216 ENDRTN
4401 012216 $240:
4402 012216 000207 $241: RTS PC
  
```

4403 012220
4404 012220
4405
4406
4407
4408
4409
4410
4411
4412
4413 012220
4414 012220
4415 012220
4416 012220 005767 000122
4417 012224 001027
4418 012226
4419 012226 026727 000116 000001
4420 012234 001003
4421 012236
4422 012236 005067 000106
4423 012242
4424 012242 000403
4425 012244
4426 012244
4427 012244 004767 000110
4428
4429 012250
4430 012250
4431 012250 012600
4432 012252
4433 012252
4434 012252
4435 012252 012767 000001 000066
4436 012260
4437 012260 012767 000001 166716
4438 012266
4439 012266 016767 166756 000056
4440 012274
4441 012274 016767 166744 000052
4442 012302
4443 012302 000410
4444 012304
4445 012304
4446 012304 012704 000010
4447 012310
4448 012310 006167 000032
4449 012314
4450 012314 060467 000032
4451 012320
4452 012320 060467 000030
4453 012324
4454 012324
4455 012324
4456 012324 036767 000016 166720
4457 012332 001732
4458

```
ROUTINE CYCLE
CYCLE:
:*****
* ROUTINE:    CYCLE
*   THIS ROUTINE CAUSES ADRS TO POINT TO THE
*   ADDRESS OF DLV11-E UNDER TEST, ADRS +2 TO
*   POINT TO THE VECTOR OF THE DLV11-E UNDER TEST.
*   IT KEEPS TRACK OF THE CURRENT DEVICE AND BIT
*   MASKS.
:*****
REPEAT
$244:
      IF BITMASK EQ #0 THEN
            IF INITFLAG EQ #1 THEN
                  LET INITFLAG := #0
            ELSE
                  CALL $EOP ; AS A SUBROUTINE
            ; BECAUSE $EOP RETURNS AS A JUMP
            LET R0 := POP
      ENDIF
      LET BITMASK := #1
      LET $DEVCT := #1
      LET ADDRESS := $BASE
      LET VECTOR := $VECT1
    ELSE
      LET R4 : #10
      LET BITMASK := BITMASK ROTATE 1
      LET ADDRESS := ADDRESS + R4
      LET VECTOR := VECTOR + R4
    ENDIF
  UNTIL BITMASK SET IN $DEVCT
$245:
      MOV #1,BITMASK
      MOV #1,$DEVCT
      MOV $BASE,ADDRESS
      MOV $VECT1,VECTOR
      BR $250
$246:
      JSR PC,$EOP
      MOV (SP)+,R0
$250:
      BIT BITMASK,$DEVCT
      BEQ $244
```

```
4459 012334
4460 012334 012701 012352
4461 012340
4462 012340 005267 166640
4463 012344
4464 012344 000404
4465 012346 000000
4466 012350 000001
4467 012352 000000
4468 012354 000000
4469
4470 012356
4471 012356
4472 012356
4473 012356 000207
4474
4475
```

LET ADRS := #ADDRESS
MOV #ADDRESS,ADRS
LET \$DEVCT := \$DEVCT + #1
INC \$DEVCT
BR \$242
RETURN
ENDRTN
\$242:
\$243:
RTS PC

BITMASK: 0
INITFLAG: 1
ADDRESS: 0
VECTOR: 0

```

4476 .SBTTL END OF PASS ROUTINE
4477
4478 ::*****
4479 ::*INCREMENT THE PASS NUMBER ($PASS)
4480 ::*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
4481 ::*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
4482 ::*IF THERES A MONITOR GO TO IT
4483 ::*IF THERE ISN'T JUMP TO SPECIALADDRESS
4484
4485 $EOP:
4486 012360 000004          CLR      $STNUM      ;;ZERO THE TEST NUMBER
4487 012362 005067 166514 CLR      $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
4488 012366 005067 166566 CLR      $PASS       ;;INCREMENT THE PASS NUMBER
4489 012372 005267 166604 INC      $PASS       ;;DON'T ALLOW A NEG. NUMBER
4490 012376 042767 100000 166576 BIC     #100000,$PASS ;;LOOP?
4491 012404 005327          DEC     (PC)+
4492 012406 000001          SEOPCT: .WORD 1
4493 012410 003022          BGT     $DOAGN      ;;YES
4494 012412 012737          MOV     (PC)+,@(PC)+ ;;RESTORE COUNTER
4495 012414 000001          $ENDCT: .WORD 1
4496 012416 012406          $EOPCT
4497 012420 104401 012465  TYPE     ,SENDMG      ;;TYPE 'END PASS #'
4498 012424 016746 166552  MOV     $PASS,-(SP)  ;;SAVE $PASS FOR TYPEOUT
4499 012430 104405          TYPDS   ;;GO TYPE--DECIMAL ASCII WITH SIGN
4500 012432 104401 012462  TYPE     ,SENULL     ;;TYPE A NULL CHARACTER
4501 012436 013700 000042  $GET42: MOV     @#42,R0 ;;GET MONITOR ADDRESS
4502 012442 001405          BEQ     $DOAGN      ;;BRANCH IF NO MONITOR
4503 012444 000005          RESET  ;;CLEAR THE WORLD
4504 012446 004710          $ENDAD: JSR    PC,(R0) ;;GO TO MONITOR
4505 012450 000240          NOP     ;;SAVE ROOM
4506 012452 000240          NOP     ;;FOR
4507 012454 000240          NOP     ;;ACT11
4508 012456
4509 012456 000137          $DOAGN: JMP     @(PC)+      ;;RETURN
4510 012460 012250          $RTNAD: .WORD   SPECIALADDRESS
4511 012462 377 377 000  $FNUL: .BYTE  -1,-1,0 ;;NULL CHARACTER STRING
4512 012465 015 042412 0421'6 $ENDMG: .ASCIZ <15><12>/END PASS #/
4513 012472 050040 051501 02012'
4514 012500 000043
    
```

```

4515          .SBTTL  POWER DOWN AND UP ROUTINES
4516
4517          ;:*****
4518          :POWER DOWN ROUTINE
4519 012502 012737 012646 000024 $PWRDN: MOV    # $ILLUP,@#PWRVEC ;:SET FOR FAST UP
4520 012510 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;:PRIO:7
4521 012516 010046      MOV    R0,-(SP) ;:PUSH R0 ON STACK
4522 012520 010146      MOV    R1,-(SP) ;:PUSH R1 ON STACK
4523 012522 010246      MOV    R2,-(SP) ;:PUSH R2 ON STACK
4524 012524 010346      MOV    R3,-(SP) ;:PUSH R3 ON STACK
4525 012526 010446      MOV    R4,-(SP) ;:PUSH R4 ON STACK
4526 012530 010546      MOV    R5,-(SP) ;:PUSH R5 ON STACK
4527 012532 017746 166402      MOV    @SWR,-(SP) ;:PUSH @SWR ON STACK
4528 012536 010667 000110      MOV    SP,$SAVR6 ;:SAVE SP
4529 012542 012737 012554 000024      MOV    # $PWRUP,@#PWRVEC ;:SET UP VECTOR
4530 012550 000000      HALT
4531 012552 000776      BR     .-2 ;:HANG UP
4532
4533          ;:*****
4534          :POWER UP ROUTINE
4535 012554 012737 012646 000024 $PWRUP: MOV    # $ILLUP,@#PWRVEC ;:SET FOR FAST DOWN
4536 012562 016706 000064      MOV    $SAVR6,SP ;:GET SP
4537 012566 005067 000060      CLR    $SAVR6 ;:WAIT LOOP FOR THE TTY
4538 012572 005267 000054      1$: INC    $SAVR6 ;:WAIT FOR THE INC
4539 012576 001375      BNE    1$ ;:OF WORD
4540 012600 012677 166334      MOV    (SP)+,@SWR ;:POP STACK INTO @SWR
4541 012604 012605      MOV    (SP)+,R5 ;:POP STACK INTO R5
4542 012606 012604      MOV    (SP)+,R4 ;:POP STACK INTO R4
4543 012610 012603      MOV    (SP)+,R3 ;:POP STACK INTO R3
4544 012612 012602      MOV    (SP)+,R2 ;:POP STACK INTO R2
4545 012614 012601      MOV    (SP)+,R1 ;:POP STACK INTO R1
4546 012616 012600      MOV    (SP)+,R0 ;:POP STACK INTO R0
4547 012620 012737 012502 000024      MOV    # $PWRDN,@#PWRVEC ;:SET UP THE POWER DOWN VECTOR
4548 012626 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;:PRIO:7
4549 012634 104401      TYPE ;:REPORT THE POWER FAILURE
4550 012636 012654      $PWRMG: .WORD $POWER ;:POWER FAIL MESSAGE POINTER
4551 012640 012716      MOV    (PC)+,(SP) ;:RESTART AT START
4552 012642 001336      $PWRAD: .WORD START ;:RESTART ADDRESS
4553 012644 000002      RT!
4554 012646 000000      $ILLUP: HALT ;:THE POWER UP SEQUENCE WAS STARTED
4555 012650 000776      BR     .-2 ;:BEFORE THE POWER DOWN WAS COMPLETE
4556 012652 000000      $SAVR6: 0 ;:PUT THE SP HERE
4557 012654 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
4558 012662 000122
4559          .EVEN

```

```

4560 .SBTTL TYPE ROUTINE
4561
4562 :*****
4563 :*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4564 :*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4565 :*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4566 :*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4567 :*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4568 :*
4569 :*CALL:
4570 :*1) USING A TRAP INSTRUCTION
4571 :* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4572 :*OR
4573 :* TYPE
4574 :* MESADR
4575 :*
4576
4577 012664 105767 166267 $TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
4578 012670 100002 BPL 1$ ;;BR IF YES
4579 012672 000000 HALT ;;HALT HERE IF NO TERMINAL
4580 012674 000430 BR 3$ ;;LEAVE
4581 012676 010046 1$: MOV RO,-(SP) ;;SAVE RO
4582 012700 017600 000002 MOV @2(SP),RO ;;GET ADDRESS OF ASCIZ STRING
4583 012704 122767 000001 166302 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
4584 012712 001011 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
4585 012714 132767 000100 166273 BITB #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
4586 012722 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
4587 012724 010067 000004 MOV RO,61$ ;;SETUP MESSAGE ADDRESS FOR APT
4588 012730 004767 000774 JSR PC,$ATY3 ;;SPOOL MESSAGE TO APT
4589 012734 000000 61$: .WORD 0 ;;MESSAGE ADDRESS
4590 012736 132767 000040 166251 62$: BITB #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
4591 012744 001003 BNE 60$ ;;YES,SKIP TYPE OUT
4592 012746 112046 2$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4593 012750 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
4594 012752 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
4595 012754 012600 60$: MOV (SP)+,RO ;;RESTORE RO
4596 012756 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
4597 012762 000002 RTI ;;RETURN
4598 012764 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
4599 012770 001430 BEQ 8$
4600 012772 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
4601 012776 001006 BNE 5$
4602 013000 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
4603 013002 104401 TYPE ;;TYPE A CR AND LF
4604 013004 001171 $CRLF
4605 013006 105067 000130 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
4606 013012 000755 BR 2$ ;;GET NEXT CHARACTER
4607 013014 004767 000056 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
4608 013020 126726 166132 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
4609 013024 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
4610 013026 016746 166122 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
4611 ;;AND THE NULL CHAR.
4612 013032 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
4613 013036 002770 BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
4614 013040 004767 000032 JSR PC,$TYPEL ;;GO TYPE A NULL
4615 013044 105367 000072 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
    
```

```

4616 013050 000770          BR      7$          ;;LOOP
4617
4618          :HORIZONTAL TAB PROCESSOR
4619
4620 013052 112716 000040      8$:      MOVB   #' (SP)          ;;REPLACE TAB WITH SPACE
4621 013056 004767 000014      9$:      JSR    PC,$TYPEC          ;;TYPE A SPACE
4622 013062 132767 000007 000052      BITB   #7,$CHARCNT          ;;BRANCH IF NOT AT
4623 013070 001372          BNE    9$          ;;TAB STOP
4624 013072 005726          TST    (SP)+          ;;POP SPACE OFF STACK
4625 013074 000724          BR     2$          ;;GET NEXT CHARACTER
4626 013076 105777 166046      $TYPEC: TSTB   @2$TPS          ;;WAIT UNTIL PRINTER IS READY
4627 013102 100375          BPL    $TYPEC
4628 013104 116677 000002 166040      MOVB   2(SP),@2$TPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4629 013112 122766 000015 000002      CMPB   #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
4630 013120 001003          BNE    1$          ;;BRANCH IF NO
4631 013122 105067 000014      CLRB   $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
4632 013126 000406          BR     $TYPEX          ;;EXIT
4633 013130 122766 000012 000002      1$:      CMPB   #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
4634 013136 001402          BEQ    $TYPEX          ;;BRANCH IF YES
4635 013140 105227          INCB   (PC)+          ;;COUNT THE CHARACTER
4636 013142 000000      $CHARCNT: .WORD 0          ;;CHARACTER COUNT STORAGE
4637 013144 000207      $TYPEX: RTS    PC
4638
    
```

```

4639 .SBTTL TTY INPUT ROUTINE
4640
4641 ::*****
4642 .ENABL LSB
4643
4644 ::*****
4645 ::*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4646 ::*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4647 ::*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
4648 ::*WHEN OPERATING IN TTY FLAG MODE.
4649 013146 022767 000176 165764 $CKSWR: CMP #SWREG,SWR ::IS THE SOFT-SWR SELECTED?
4650 013154 001074 BNE 15$ ::BRANCH IF NO
4651 013156 105777 165762 TSTB @STKS ::CHAR THERE?
4652 013162 100071 BPL 15$ ::IF NO, DON'T WAIT AROUND
4653 013164 117746 165756 MOVB @STKB,-(SP) ::SAVE THE CHAR
4654 013170 042716 177600 BIC #^C177,(SP) ::STRIP-OFF THE ASCII
4655 013174 022726 000007 CMP #7,(SP)+ ::IS IT A CONTROL G?
4656 013200 001062 BNE 15$ ::NO, RETURN TO USER
4657 013202 126727 165726 000001 CMPB $AJTOB,#1 ::ARE WE RUNNING IN AUTO-MODE?
4658 013210 001456 BEQ 15$ ::BRANCH IF YES
4659
4660 013212 104401 013673 $GTSWR: TYPE .SCNTLG ::ECHO THE CONTROL-G (^G)
4661 013216 104401 013700 TYPE .SMSWR ::TYPE CURRENT CONTENTS
4662 013222 016746 164750 MOV SWREG,-(SP) ::SAVE SWREG FOR TYPEOUT
4663 013226 104402 TYPDC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4664 013230 104401 013711 TYPE .SMNEW ::PROMPT FOR NEW SWR
4665 013234 005046 19$: CLR -(SP) ::CLEAR COUNTER
4666 013236 005046 CLR -(SP) ::THE NEW SWR
4667 013240 105777 165700 7$: TSTB @STKS ::CHAR THERE?
4668 013244 100375 BPL 7$ ::IF NOT TRY AGAIN
4669
4670 013246 117746 165674 MOVB @STKB,-(SP) ::PICK UP CHAR
4671 013252 042716 177600 BIC #^C177,(SP) ::MAKE IT 7-BIT ASCII
4672
4673
4674
4675 013256 021627 000025 9$: CMP (SP),#25 ::IS IT A CONTROL-U?
4676 013262 001005 BNE 10$ ::BRANCH IF NOT
4677 013264 104401 013666 TYPE .SCNTLU ::YES, ECHO CONTROL-U (^U)
4678 013270 062706 000006 20$: ADD #6,SP ::IGNORE PREVIOUS INPUT
4679 013274 000757 BR 19$ ::LET'S TRY IT AGAIN
4680
4681
4682 013276 021627 000015 10$: CMP (SP),#15 ::IS IT A <CR>?
4683 013302 001022 BNE 16$ ::BRANCH IF NO
4684 013304 005766 000004 TST 4(SP) ::YES, IS IT THE FIRST CHAR?
4685 013310 001403 BEQ 11$ ::BRANCH IF YES
4686 013312 016677 000002 165620 MOV 2(SP),@SWR ::SAVE NEW SWR
4687 013320 062706 000006 11$: ADD #6,SP ::CLEAR UP STACK
4688 013324 104401 001171 14$: TYPE .SCRLF ::ECHO <CR> AND <LF>
4689 013330 126727 165601 000001 CMPB $INTAG,#1 ::RE-ENABLE TTY KBD INTERRUPTS?
4690 013336 001003 BNE 15$ ::BRANCH IF NOT
4691 013340 012777 000100 165576 MOV #100,@STKS ::RE-ENABLE TTY KBD INTERRUPTS
4692 013346 000002 15$: RTI ::RETURN
4693 013350 004767 177522 16$: JSR PC,$TYPEC ::ECHO CHAR
4694 013354 021627 000060 CMP (SP),#60 ::CHAR < 0?
    
```



```

4695 013360 002420          BLT      18$          ;;BRANCH IF YES
4696 013362 021627 000067    CMP      (SP),#67    ;;CHAR > 7?
4697 013366 003015          BGT      18$          ;;BRANCH IF YES
4698 013370 042726 000060    BIC      #60,(SP)+  ;;STRIP-OFF ASCII
4699 013374 005766 000002    TST      2(SP)      ;;IS THIS THE FIRST CHAR
4700 013400 001403          BEQ      17$          ;;BRANCH IF YES
4701 013402 006316          ASL      (SP)        ;;NO, SHIFT PRESENT
4702 013404 006316          ASL      (SP)        ;;CHAR OVER TO MAKF
4703 013406 006316          ASL      (SP)        ;;ROOM FOR NEW ONE.
4704 013410 005266 000002    7$: INC      2(SP)    ;;KEEP COUNT OF CHAR
4705 013414 056616 177776    BIS      -2(SP),(SP) ;;SET IN NEW CHAR
4706 013420 000707          BR       7$          ;;GET THE NEXT ONE
4707 013422 104401 001170    18$: TYPE   $QUES    ;;TYPE ?<CR><LF>
4708 013426 000720          BR       20$        ;;SIMULATE CONTROL-U
4709          .DSABL  LSB
    
```

*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

*CALL:

```

*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
*      RETURN HERE   ;;CHARACTER IS ON THE STACK
*                  ;;WITH PARITY BIT STRIPPED OFF
    
```

```

4720 013430 011646          $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
4721 013432 016666 000004 000002    MOV      4(SP),2(SP) ;;SAVE THE PS
4722 013440 105777 165500    1$: TSTB   @TKS      ;;WAIT FOR
4723 013444 100375          BPL      1$          ;;A CHARACTER
4724 013446 117766 165474 000004    MOVB   @TKB,4(SP)   ;;READ THE TTY
4725 013454 042766 177600 000004    BIC      #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
4726 013462 026627 000004 000023    CMP      4(SP),#23  ;;IS IT A CONTROL-S?
4727 013470 001013          BNE      3$          ;;BRANCH IF NO
4728 013472 105777 165446    2$: TSTB   @TKS      ;;WAIT FOR A CHARACTER
4729 013476 100375          BPL      2$          ;;LOOP UNTIL ITS THERE
4730 013500 117746 165442    MOVB   @TKB,-(SP)   ;;GET CHARACTER
4731 013504 042716 177600    BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
4732 013510 022627 000021    CMP      (SP)+,#21  ;;IS IT A CONTROL-Q?
4733 013514 001366          BNE      2$          ;;IF NOT DISCARD IT
4734 013516 000750          BR       1$          ;;YES, RESUME
4735 013520 026627 000004 000140    3$: CMP      4(SP),#140 ;;IS IT UPPER CASE?
4736 013526 002407          BLT      4$          ;;BRANCH IF YES
4737 013530 026627 000004 000175    CMP      4(SP),#175 ;;IS IT A SPECIAL CHAR?
4738 013536 003003          BGT      4$          ;;BRANCH IF YES
4739 013540 042766 000040 000004    BIC      #40,4(SP)  ;;MAKE IT UPPER CASE
4740 013546 000002          4$: RTI          ;;GO BACK TO USER
    
```

*THIS ROUTINE WILL INPUT A STRING FROM THE TTY

*CALL:

```

*      RDLIN         ;;INPUT A STRING FROM THE TTY
*      RETURN HERE   ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
*                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
    
```

```

4748 013550 010346          $RD_LIN: MOV      R3,-(SP) ;;SAVE R3
4749 013552 012703 013656    1$: MOV      #TTYIN,R3 ;;GET ADDRESS
4750 013556 022703 013666    2$: CMP      #TTYIN+8.,R3 ;;BUFFER FULL?
    
```

```

4751 013562 101405          BLOS      4$          ;;BR IF YES
4752 013564 104410          RDCHR
4753 013566 112613          MOV      (SP)+,(R3)  ;;GO READ ONE CHARACTER FROM THE TTY
4754 013570 122713 000177 10$:  CMP      #177,(R3)  ;;GET CHARACTER
4755 013574 001003          BNE      3$          ;;IS IT A RUBOUT
4756 013576 104401 001170 4$:  TYPE    ,SQUES  ;;SKIP IF NOT
4757 013602 000763          BR       1$          ;;TYPE A '?'
4758 013604 111367 000044 3$:  MOV      (R3),9$  ;;CLEAR THE BUFFER AND LOOP
4759 013610 104401 013654          TYPE    ,9$          ;;ECHO THE CHARACTER
4760 013614 122723 000015          CMP      #15,(R3)+  ;;CHECK FOR RETURN
4761 013620 001356          BNE      2$          ;;LOOP IF NOT RETURN
4762 013622 105063 177777          CLRB    -1(R3)      ;;CLEAR RETURN (THE 15)
4763 013626 104401 001172          TYPE    ,LF         ;;TYPE A LINE FEED
4764 013632 012603          MOV      (SP)+,R3   ;;RESTORE R3
4765 013634 011646          MOV      (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4766 013636 016666 000004 000002 MOV      4(SP),2(SP) ;;FIRST ASCII CHARACTER ON IT
4767 013644 012766 013656 000004 MOV      #$TTYIN,4(SP)
4768 013652 000002          RTI
4769 013654 000          9$:  .BYTE   0          ;;RETURN
4770 013655 000          .BYTE   0          ;;STORAGE FOR ASCII CHAR. TO TYPE
4771 013656 000010          $ITYIN: .BLKB  8.    ;;TERMINATOR
4772 013666 052536 005015 000  $CNTLU: .ASCIZ  /^U/<15><12> ;;RESERVE 8 BYTES FOR TTY INPUT
4773 013673 0136 006507 000012  $CNTLG: .ASCIZ  /^G/<15><12> ;;CONTROL 'U'
4774 013700 005015 053523 020122 $MSWR:  .ASCIZ  <15><12>/SWR - / ;;CONTROL 'G'
4775 013706 020075 000
4776 013711 040 047040 053505 $MNEW:  .ASCIZ  / NEW /
4777 013716 036440 000040
    
```

```

4778      .SBTTL  APT COMMUNICATIONS ROUTINE
4779
4780      ::*****
4781      013722 112767 000001 000236 $ATY1:  MOVB  #1,$FFLG      ;;TO REPORT FATAL ERROR
4782      013730 112767 000001 000226 $ATY3:  MOVB  #1,$MFLG      ;;TO TYPE A MESSAGE
4783      013736 000403          BR          $ATYC
4784      013740 112767 000001 000220 $ATY4:  MOVB  #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
4785      013746          $ATYC:
4786      013746 010046          MOV    R0,-(SP)      ;;PUSH R0 ON STACK
4787      013750 010146          MOV    R1,-(SP)      ;;PUSH R1 ON STACK
4788      013752 105767 000206          TSTB  $MFLG      ;;SHOULD TYPE A MESSAGE?
4789      013756 001450          BEQ   5$          ;;IF NOT: BR
4790      013760 122767 000001 165226          CMPB  #APTENV,$ENV      ;;OPERATING UNDER APT?
4791      013766 001031          BNE   3$          ;;IF NOT: BR
4792      013770 132767 000100 165217          BITB  #APTSPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?
4793      013776 001425          BEQ   3$          ;;IF NOT: BR
4794      014000 017600 000004          MOV    @4(SP),R0      ;;GET MESSAGE ADDR.
4795      014004 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
4796      014012 005767 165156          1$:  TST   $MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
4797      014016 001375          BNE   1$          ;;IF NOT: WAIT
4798      014020 010067 165164          MOV    R0,$MSGAD      ;;PUT ADDR IN MAILBOX
4799      014024 105720          2$:  TSTB  (R0)+          ;;FIND END OF MESSAGE
4800      014026 001376          BNE   2$
4801      014030 166700 165154          SUB   $MSGAD,R0      ;;SUB START OF MESSAGE
4802      014034 006200          ASR   R0              ;;GET MESSAGE LNTH IN WORDS
4803      014036 010067 165150          MOV    R0,$MSGGLT      ;;PUT LENGTH IN MAILBOX
4804      014042 012767 000004 165124          MOV    #4,$MSGTYPE      ;;TELL APT TO TAKE MSG.
4805      014050 000413          BR    5$
4806      014052 017667 000004 000016 3$:  MOV    @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
4807      014060 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDRESS
4808      014066 016746 163704          MOV    177776,-(SP)   ;;PUSH 177776 ON STACK
4809      014072 004767 176566          JSR   PC,$TYPE      ;;CALL TYPE MACRO
4810      014076 000000          4$:  .WORD 0
4811      014100          5$:
4812      014100 105767 000062          10$: TSTB  $FFLG      ;;SHOULD REPORT FATAL ERROR?
4813      014104 001416          BEQ   12$          ;;IF NOT: BR
4814      014106 005767 165102          TST   $ENV          ;;RUNNING UNDER APT?
4815      014112 001413          BEQ   12$          ;;IF NOT: BR
4816      014114 005767 165054          11$: TST   $MSGTYPE      ;;FINISHED LAST MESSAGE?
4817      014120 001375          BNE   11$          ;;IF NOT: WAIT
4818      014122 017667 000004 165046          MOV    @4(SP),$FATAL  ;;GET ERROR #
4819      014130 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
4820      014136 005267 165032          INC   $MSGTYPE      ;;TELL APT TO TAKE ERROR
4821      014142 105067 000020          12$: CLRB  $FFLG      ;;CLEAR FATAL FLAG
4822      014146 105067 000013          CLRB  $LFLG      ;;CLEAR LOG FLAG
4823      014152 105067 000006          CLRB  $MFLG      ;;CLEAR MESSAGE FLAG
4824      014156 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
4825      014160 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
4826      014162 000207          RTS   PC            ;;RETURN
4827      014164 000          $MFLG: .BYTE 0      ;;MESSG. FLAG
4828      014165 000          $LFLG: .BYTE 0      ;;LOG FLAG
4829      014166 000          $FFLG: .BYTE 0      ;;FATAL FLAG
4830          014170          .EVEN
4831          000200          APTSIZE=200
4832          000001          APTENV=001
4833          000100          APTSPOOL 100

```

4834 000040 APTCSUP=040

4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888

014170
014170 104407
014172 105267 164705
014176 001775
014200 016777 164676 164734
014206 032777 002000 164724
014214 001402
014216 104401 001164
014222 005267 164664
014226 011667 164664
014232 162767 000002 164656
014240 117767 164652 164646
014246 032777 020000 164664
014254 001004
014256 004767 175554
014262 104401 001171
014266
014266 122767 000001 164720
014274 001007
014276 116767 164612 000004
014304 004767 177430
014310 000
014311 000
014312 000777
014314 005777 164620
014320 100002
014322 000000
014324 104407
014326 032777 001000 164604
014334 001402
014336 016716 164546
014342 005767 164614
014346 001402
014350 016716 164606
014354
014354 022737 012446 000042
014362 001001
014364 000000
014366
014366 000002

```

.SBTTL ERROR HANDLER ROUTINE
:*****
:*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
:*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
:*AND GO TO MYTYPE ON ERROR
:*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:*SW15=1 HALT ON ERROR
:*SW13=1 INHIBIT ERROR TYPEOUTS
:*SW10=1 BELL ON ERROR
:*SW09=1 LOOP ON ERROR
:*CALL
:* ERROR N ;;ERROR EMT AND N=ERROR ITEM NUMBER

$ERROR:
'$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;;SET THE ERROR FLAG
BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
MOV $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10,@SWR ;;BELL ON ERROR?
BEQ 1$ ;;NO - SKIP
TYPE $BELL ;;RING BELL
1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
SUB #2,$ERRPC
MOVB @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET
BNE 20$ ;;SKIP TYPEOUTS
JSR PC,MYTYPE ;;GO TO USER ERROR ROUTINE
TYPE $CRLF

20$: CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
BNE 2$ ;;NO,SKIP APT ERROR REPORT
MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT

21$: .BYTE 0
.BYTE 0

22$: BR 22$ ;;APT ERROR LOOP
2$: TST @SWR ;;HALT ON ERROR
BPL 3$ ;;SKIP IF CONTINUE
HALT ;;HALT ON ERROR!
CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?
BEQ 4$ ;;BR IF NO
MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
BEQ 5$ ;;BR IF NONE
MOV $FSCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE

5$: CMP # $ENDAD,@#4? ;;ACT-11 AUTO-ACCEPT?
BNE 6$ ;;BRANCH IF NO
HALT ;;YES

6$: RTI ;;RETURN
    
```

```

4889          .SBTTL  SCOPE HANDLER ROUTINE
4890
4891          ::*****
4892          ::THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4893          ::AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4894          ::AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
4895          ::THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4896          ::*SW14=1      LOOP ON TEST
4897          ::*SW11=1      INHIBIT ITERATIONS
4898          ::*SW09=1      LOOP ON ERROR
4899          ::*SW08=1      LOOP ON TEST IN SWR<7:0>
4900          ::CALL
4901          ::*      SCOPE          ;;SCOPE-IOT
4902
4903          $SCOPE:
4904          014370 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
4905          014372 032777 040000 164540  $:      BIT      #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
4906          014400 001114          BNE      $OVER          ;;YES IF SW14=1
4907          :#####START OF CODE FOR THE XOR TESTER#####
4908          014402 000416          $XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
4909          :THIS INSTRUCTION TO A 'NOP' (NOP=240)
4910          014404 013746 000004          MOV      @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4911          014410 012737 014430 000004          MOV      #5$,@#ERRVEC      ;;SET FOR TIMEOUT
4912          014416 005737 177060          TST      @#177060          ;;TIME OUT ON XOR?
4913          014422 012637 000004          MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
4914          014426 000463          BR      $$VLAD          ;;GO TO THE NEXT TEST
4915          014430 022626          5$:      CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
4916          014432 012637 000004          MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
4917          014436 000423          BR      7$          ;;LOOP ON THE PRESENT TEST
4918          014440          6$:;#####END OF CODE FOR THE XOR TESTER#####
4919          014440 032777 000400 164472          BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
4920          014446 001404          BEQ      2$          ;;BR IF NO
4921          014450 127767 164464 164424          CMPB    @SWR,$TSTNM      ;;ON THE RIGHT TEST? SWR<7:0>
4922          014456 001465          BEQ      $OVER          ;;BR IF YES
4923          014460 105767 164417          2$:      TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
4924          014464 001421          BEQ      3$          ;;BR IF NO
4925          014466 126767 164423 164407          CMPB    $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4926          014474 101015          BHI      3$          ;;BR IF NO
4927          014476 032777 001000 164434          BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
4928          014504 001404          BEQ      4$          ;;BR IF NO
4929          014506 016767 164376 164372  7$:      MOV      $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
4930          014514 000446          BR      $OVER
4931          014516 105067 164361          4$:      CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
4932          014522 005067 164432          CLR      $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
4933          014526 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
4934          014530 032777 004000 164402  3$:      BIT      #BIT11,@SWR      ;;INHIBIT ITERATIONS?
4935          014536 001011          BNE      1$          ;;BR IF YES
4936          014540 005767 164436          TST      $PASS          ;;IF FIRST PASS OF PROGRAM
4937          014544 001406          BEQ      1$          ;;      INHIBIT ITERATIONS
4938          014546 005267 164332          INC      $ICNT          ;;INCREMENT ITERATION COUNT
4939          014552 026767 164402 164324          CMP      $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
4940          014560 002024          BGE      $OVER          ;;BR IF MORE ITERATION REQUIRED
4941          014562 012767 000001 164314  1$:      MOV      #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
4942          014570 016767 000052 164362          MOV      $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
4943          014576 105267 164300          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
4944          014602 116767 164274 164370          MOVB    $TSTNM,$TESTN      ;;SET TEST NUMBER IN APT MAILBOX
  
```

4945	014610	011667	164272		MOV	(SP), \$LPADR	:::SAVE SCOPE LOOP ADDRESS
4946	014614	011667	164270		MOV	(SP), \$LPERR	:::SAVE ERROR LOOP ADDRESS
4947	014620	005067	164336		CLR	\$ESCAPE	:::CLEAR THE ESCAPE FROM ERROR ADDRESS
4948	014624	112767	000001	164263	MOVB	#1, \$ERMAX	:::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4949	014632	016777	164244	164302	\$OVER: MOV	\$TSTNM, @DISPLAY	:::DISPLAY TEST NUMBER
4950	014640	016716	164242		MOV	\$LPADR, (SP)	:::FUDGE RETURN ADDRESS
4951	014644	000002			RTI		:::FIXES PS
4952	014646	003720			\$MXCNT: 2000.		:::MAX. NUMBER OF ITERATIONS

4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008

014650
014650 010046
014652 010146
014654 010246
014656 010346
014660 010546
014662 020200
014666 016605 000020
014672 100004
014674 005405
014676 112766 000055 000001
014704 005000
014706 012703 015064
014712 112723 000040
014716 005002
014720 016001 015054
014724 160105
014726 002402
014730 005202
014732 000774
014734 060105
014736 005702
014740 001002
014742 105716
014744 100407
014746 106316
014750 103003
014752 116663 000001 177777
014760 052702 000060
014764 052702 000040
014770 110223
014772 005720
014774 020027 000010
015000 002746
015002 003002
015004 010502
015006 000764
015010 105726
015012 100003
015014 116663 177777 177776
015022 105013
015024 012605
015026 012603
015030 012602

```
.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
:*****
:*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
:*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
:*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
:*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
:*REPLACED WITH SPACES.
:*CALL:
:*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
:*      TYPDS                    ;;GO TO THE ROUTINE
$TYPDS:
MOV      R0,-(SP)      ;;PUSH R0 ON STACK
MOV      R1,-(SP)      ;;PUSH R1 ON STACK
MOV      R2,-(SP)      ;;PUSH R2 ON STACK
MOV      R3,-(SP)      ;;PUSH R3 ON STACK
MOV      R5,-(SP)      ;;PUSH R5 ON STACK
MOV      #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
MOV      20(SP),R5      ;;GET THE INPUT NUMBER
BPL      1$            ;;BR IF INPUT IS POS.
NEG      R5            ;;MAKE THE BINARY NUMBER POS.
MOVB    #'-,1(SP)      ;;MAK THE ASCII NUMBER NEG.
1$:      CLR      R0            ;;ZERO THE CONSTANTS INDEX
MOV      #SDBLK,R3      ;;SETUP THE OUTPUT POINTER
MOVB    #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
2$:      CLR      R2            ;;CLEAR THE BCD NUMBER
MOV      $DTBL(R0),R1    ;;GET THE CONSTANT
3$:      SUB      R1,R5          ;;FORM THIS BCD DIGIT
BLT     4$            ;;BR IF DONE
INC     R2            ;;INCREASE THE BCD DIGIT BY 1
BR      3$
4$:      ADD      R1,R5          ;;ADD BACK THE CONSTANT
TST     R2            ;;CHECK IF BCD DIGIT=0
BNE     5$            ;;FALL THROUGH IF 0
TSTB   (SP)          ;;STILL DOING LEADING 0'S?
BMI     7$            ;;BR IF YES
5$:      ASLB    (SP)          ;;MSD?
BCC     6$            ;;BR IF NO
MOVB   1(SP),-1(R3)    ;;YES--SET THE SIGN
6$:      BIS     #'0,R2        ;;MAKE THE BCD DIGIT ASCII
7$:      BIS     #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB   R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST    (R0)+          ;;JUST INCREMENTING
CMP    R0,#10        ;;CHECK THE TABLE INDEX
BLT    2$            ;;GO DO THE NEXT DIGIT
BGT    8$            ;;GO TO EXIT
MOV    R5,R2          ;;GET THE LSD
BR     6$            ;;GO CHANGE TO ASCII
8$:      TSTB   (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
BPL    9$            ;;BR IF NO
MOVB  -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
9$:      CLR    (R3)          ;;SET THE TERMINATOR
MOV    (SP)+,R5       ;;POP STACK INTO R5
MOV    (R3)+,R3       ;;POP STACK INTO R3
MOV    (SP)+,R2       ;;POP STACK INTO R2
```



```
5009 015032 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
5010 015034 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
5011 015036 104401 015064  TYPE      $DBLK        ;;NOW TYPE THE NUMBER
5012 015042 016566 000002 000004  MOV      2(SP),4(SP)    ;;ADJUST THE STACK
5013 015050 012616      MOV      (SP)+,(SP)
5014 015052 000002      RTI                    ;;RETURN TO USER
5015 015054 023420      $DTBL: 10000.
5016 015056 001750      1000.
5017 015060 000144      100.
5018 015062 000012      10.
5019 015064 000004      $DBLK: .BLKW 4
```

```
5020 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
5021
5022
5023 :*****
5024 :*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
5025 :*OCTAL (ASCII) NUMBER AND TYPE IT.
5026 :*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
5027 :*CALL:
5028 :*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
5029 :*      TYPOS    ;;CALL FOR TYPEOUT
5030 :*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
5031 :*      .BYTE   M              ;;M=1 OR 0
5032 :*                               ;;1=TYPE LEADING ZEROS
5033 :*                               ;;0=SUPPRESS LEADING ZEROS
5034 :*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
5035 :*$TYPOS OR $TYPOC
5036 :*CALL:
5037 :*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
5038 :*      TYPON    ;;CALL FOR TYPEOUT
5039 :*
5040 :*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5041 :*CALL:
5042 :*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
5043 :*      TYPOC    ;;CALL FOR TYPEOUT
5044
5045 015074 017646 000000 $TYPOS: MOV      @(SP),-(SP)      ;;PICKUP THE MODE
5046 015100 116667 000001 000211 MOVVB    1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
5047 015106 112667 000207 MOVVB    (SP)+,SOMODE+1    ;;NUMBER OF DIGITS TO TYPE
5048 015112 062716 000002 ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
5049 015116 000406 BR      $TYPON
5050 015120 112767 000001 000171 $TYPOC: MOVVB    #1,SOFILL      ;;SET THE ZERO FILL SWITCH
5051 015126 112767 000006 000165 MOVVB    #6,SOMODE+1    ;;SET FOR SIX(6) DIGITS
5052 015134 112767 000005 000154 $TYPON: MOVVB    #5,SOCNT      ;;SET THE ITERATION COUNT
5053 015142 010346 MOV      R3,-(SP)          ;;SAVE R3
5054 015144 010446 MOV      R4,-(SP)          ;;SAVE R4
5055 015146 010546 MOV      R5,-(SP)          ;;SAVE R5
5056 015150 116704 000145 MOVVB    SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
5057 015154 005404 NEG      R4
5058 015156 062704 000006 ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
5059 015162 110467 000132 MOVVB    R4,SOMODE      ;;SAVE IT FOR USE
5060 015166 116704 000125 MOVVB    SOFILL,R4      ;;GET THE ZERO FILL SWITCH
5061 015172 016605 000012 MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
5062 015176 005003 CLR      R3            ;;CLEAR THE OUTPUT WORD
5063 015200 006105 1$: ROL      R5          ;;ROTATE MSB INTO 'C'
5064 015202 000404 BR      3$
5065 015204 006105 2$: ROL      R5          ;;GO DO MSB
5066 015206 006105 ROL      R5          ;;FORM THIS DIGIT
5067 015210 006105 ROL      R5
5068 015212 010503 MOV      R5,R3
5069 015214 006103 3$: ROL      R3          ;;GET LSB OF THIS DIGIT
5070 015216 105367 000076 DECB    SOMODE          ;;TYPE THIS DIGIT?
5071 015222 100016 BPL     7$            ;;BR IF NO
5072 015224 042703 177770 BIC     #177770,R3     ;;GET RID OF JUNK
5073 015230 001002 BNE     4$            ;;TEST FOR 0
5074 015232 005704 TST     R4            ;;SUPPRESS THIS 0?
5075 015234 001403 BEQ     5$            ;;BR IF YES
```

5076	015236	005204		4\$:	INC	R4	::DON'T SUPPRESS ANYMORE 0'S
5077	015240	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
5078	015244	052703	000040	5\$:	BIS	#' R3	::MAKE ASCII IF NOT ALREADY
5079	015250	110367	000040		MOVB	R3,8\$::SAVE FOR TYPING
5080	015254	104401	015314		TYPE	8\$::GO TYPE THIS DIGIT
5081	015260	105367	000032	7\$:	DECB	\$OCNT	::COUNT BY 1
5082	015264	003347			BGT	2\$::BR IF MORE TO DO
5083	015266	002402			BLT	6\$::BR IF DONE
5084	015270	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
5085	015272	000744			BR	2\$::GO DO THE LAST DIGIT
5086	015274	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
5087	015276	012604			MOV	(SP)+,R4	::RESTORE R4
5088	015300	012603			MOV	(SP)+,R3	::RESTORE R3
5089	015302	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
5090	015310	012616			MOV	(SP)+,(SP)	
5091	015312	000002			RTI		::RETURN
5092	015314	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
5093	015315	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
5094	015316	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
5095	015317	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
5096	015320	000000		\$UMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

5097
 5098
 5099
 5100
 5101
 5102
 5103
 5104
 5105
 5106
 5107
 5108
 5109
 5110
 5111
 5112
 5113
 5114
 5115
 5116
 5117
 5118
 5119
 5120
 5121
 5122
 5123
 5124
 5125
 5126
 5127
 5128
 5129
 5130
 5131
 5132
 5133
 5134
 5135
 5136
 5137
 5138
 5139

015322 010046
 015324 016600 000002
 015330 005740
 015332 111000
 015334 006300
 015336 016000 015354
 015342 000200

 015344 011646
 015346 016666 000004 000002
 015354 000002

 015356 015344
 015360 012664
 015362 015120
 015364 015074
 015366 015134
 015370 014650

 015372 013216

 015374 013146
 015376 013430
 015400 013550
 000001

```

.SBTTL TRAP DECODER
:*****
:*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
:*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
:*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
:*GO TO THAT ROUTINE.
$TRAP:  MOV    RO,-(SP)           ;;SAVE RO
        MOV    2(SP),RO         ;;GET TRAP ADDRESS
        TST    -(RO)           ;;BACKUP BY 2
        MOVB   (RO),RO         ;;GET RIGHT BYTE OF TRAP
        ASL    RO              ;;POSITION FOR INDEXING
        MOV    $TRPAD(RO),RO    ;;INDEX TO TABLE
        RTS    RO              ;;GO TO ROUTINE

;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
$TRAP2: MOV    (SP),-(SP)       ;;MOVE THE PC DOWN
        MOV    4(SP),2(SP)     ;;MOVE THE PSW DOWN
        RTI                    ;;RESTORE THE PSW

.SBTTL TRAP TABLE
:*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED,
:*BY THE 'TRAP' INSTRUCTION.
:
: ROUTINE
:-----
$TRPAD: .WORD  $TRAP2
        $TYPE  ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
        $TYPOC ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  

        $GTSWR ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
  

        $CKSWR ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
:END
    
```


BITMAS	012346	4416	4435*	4448*	4456	4465#								
BIT0 =	000001	796#												
BIT00 =	000001	786#	796	865	886	907								
BIT01 =	000002	785#	795	844	864	906								
BIT02 =	000004	784#	794	843	863	884	905							
BIT03 =	000010	783#	793	842	862	904								
BIT04 =	000020	782#	792	861	903									
BIT05 =	000040	781#	791	840	860	902								
BIT06 =	000100	780#	790	839	859	880	901							
BIT07 =	000200	779#	789	838	858	879	900							
BIT08 =	000400	778#	788	4919										
BIT09 =	001000	777#	787	4877	4927									
BIT1 =	000002	795#												
BIT10 =	002000	776#	835	4854										
BIT11 =	004000	775#	834	874	4934									
BIT12 =	010000	774#	833	853	873									
BIT13 =	020000	773#	832	852	872	4861								
BIT14 =	040000	772#	831	851	871	4905								
BIT15 =	100000	771#	830	850	870									
BIT2 =	000004	794#												
BIT3 =	000010	793#												
BIT4 =	000020	792#												
BIT5 =	000040	791#												
BIT6 =	000100	790#												
BIT7 =	000200	789#												
BIT8 =	000400	788#												
BIT9 =	001000	787#												
BPTVEC =	000014	803#												
BREAK	000001	886#	1272	1285	1288	1302	1305	1319	1324	4081				
BRK -	010000	922#	1260	4063										
CABLE -	020000	923#	1996	2050	2141	2230	2319	2410	2510	2614	3559	3741		
CARDET =	010000	833#	2075	2095	2115									
CKSWR -	104407	4850	4876	4904	5136#									
CLR =	000000	824#	2916	4190										
CLRSEN =	020000	832#	2164	2184	2204									
COMSPD =	000100	916#												
CR -	000015	711#	4629	4639										
CRLF =	000200	712#	1152	4128	4371	4600	4639							
CYCLE	012220	1157	4404#											
DATA	000017	913#	4274	4275										
DATAIE =	000040	840#	1739	1752	1755	1769	1772	1786	1791	3572	3583	3616		
DATAIN	100000	830#	1966	2443	2461	2470	2487	2544	2568	2591	2651	2673		
DATLNG	011646	3655	3761	3856	4256#									
DDISP =	177570	718#	998	1116										
DISPLA	001142	998#	1116*	1124*	4853*	4949*								
DISPRE	000174	933#	1124											
DLADD	001254	1078#	1161*	1166	1169	1171	1174	1177	1180	1196	1228	4129	4391	
DLADDR -	175610	820#	1078	1080	1081	1082	1083	1084						
DLVEC	001256	1079#	1164*	3350	3471	3622	3879	4135	4397					
DSWR	177570	717#	997	1115										
DTR =	000002	844#	1518	1521	1535	1537	1550	1553	2072	2092	2112	2161	2181	2201
		2431	2459	2485										
EMTVEC -	000030	806#	1099*	1100*										
ERRCNT	011022	3904*	3926	3932	3948#	4015	4026*							
ERROR	100000	850#	3084	3148										
ERRVEC	000004	799#	1113	1114*	1125*	4910	4911*	4913*	4916*					

EVENOD=	000040	915#												
FALSE =	000000	4171#	4178											
FLAG	011640	4178#	4205*	4210	4233	4245#								
FRERR =	020000	852#												
FRFD =	040000	924#	1507	1574	2050	2141	2230	2410	2510					
GNS =	*****	932	1151	4127	4133	4139	4370	4376	4383	4389	4395	5128	5129	5130
		5131	5132	5134	5136	5137	5138							
GTSWR =	104406	1146	5134#											
HOLD	011066	3967*	3968*	3971	3984#									
HOLDSC	011641	4190*	4195*	4201	4246#									
HOWLON=	000000	4150#	4176											
HT =	000011	709#	4598	4639										
I	001272	1085#	1205*	1226*	1229	1231	4279*	4282*	4284					
ILLMEM=	000004	811#	1200	1236										
INITFL	012350	1154*	4419	4422*	4466#									
INTFLA	012034	1212*	1217	3346*	3386	3390	3412*	3428	3479*	3521	3525	3569*	3595	3599
		4360*	4363#											
INTSRV	012026	1201	3358	3472	4350#									
IOTVEC=	000020	804#	1097*	1098*										
LF =	000012	710#	4633	4639										
LOOP	001744	1155#	4144											
MAINT =	000004	884#	1348	1361	1364	1378	1381	1395	1400	2789	2848	2914	3196	3482
		3543	3644	3728	3751	3910	4072							
MASK -	000000	4257#	4271*	4287*	4289*	4294*								
MAX	006422	2942	2946	3021#										
MYTYPE	012036	4366#	4863											
NEW	007242	3219*	3221*	3237*	3239*	3248	3256	3258	3276	3278	3326#			
NUMBER	011024	3924	3949#	3974	4032									
NUMBER	011742	4273*	4274	4276*	4284	4297#								
OLD	007246	3192*	3194*	3248	3256	3258	3276*	3278*	3327#					
ORERR =	040000	851#	3066	3105	3133									
PARITY=	000020	914#												
PBAUDS=	004000	874#												
PBAUDO-	010C00	873#												
PBAUD1-	020C00	872#												
PBAUD2=	040000	871#												
PBAUD3	100000	870#												
PBR =	000200	917#	3174											
PERR	010000	853#												
PIRQ -	177772	716#												
PIRQVE=	000240	810#												
PRO	000000	733#	3368	3417	3488	3575	3648	3754	3873					
PR1 -	000040	734#												
PR2	000100	735#												
PR3 -	000140	736#												
PR4 -	000200	737#												
PR5	000240	738#												
PR6 -	000300	739#												
PR7	000340	740#	1202	1420	2421	2521	2624	3359	3473	3884	3890			
PS -	177776	713#	714											
PSW	177776	714#												
PWRVEC-	000024	805#	1103*	1104*	4519*	4520*	4529*	4535*	4547*	4548*				
RATES	007170	3210	3289	3294#										
RBUF	001262	1081#	1171*	1172*	2883	2953	3008	3062	3105	3133	3148	3207	3517	3659
		3697	3764	3801	4002	4076	4102							
RCSR	001260	1080#	1169*	1518*	1521	1535*	1537	1550*	1553	1587	1600*	1603	1617*	1620

\$AUTOB	001134	994#	1148*	4657	4778									
\$BASE	001250	1058#	4439											
\$BDADR	001122	989#												
\$BDDAT	001126	991#												
\$BELL	001164	1009#	4856	4889										
\$BGNLE=	177777	1#												
\$CHARC	013142	4605*	4615*	4622	4631*	4636#								
\$CKSWR	013146	4649#	5136											
\$CMTAG	001100	977#	1090	1091	1099	1105	1106	1107						
\$CM3 =	000000	1007#												
\$CNTLG	013673	4660	4773#											
\$CNTLU	013666	4677	4772#											
\$CPUOP	001222	1032#												
\$CRLF	001171	1011#	4604	4639	4688	4772	4864	4889						
\$DBLK	015064	4977	5011	5019#										
\$DEVCT	001204	1023#	4437*	4462*										
\$DEVM	001252	1059#	4456											
\$DOAGN	012456	4493	4502	4508#										
\$DTBL	015054	4980	5015#											
\$ENDAD	012446	943	1136	4504#	4884									
\$ENDCT	012414	1105	4495#											
\$ENDMG	012465	4497	4512#											
\$ENULL	012462	4500	4511#											
\$ENV	001214	1028#	1142	2697	2906	3183	3846	4583	4790	4814	4866			
\$ENVM	001215	1029#	1128	4585	4590	4792								
\$EOP	012360	4427	4485#											
\$EOPCT	012406	1105*	4492#	4496										
\$ERFLG	001103	980#	4851*	4889	4894	4923	4925	4931*	4953					
\$ERMAX	001115	986#	1108*	4925	4948*	4953								
\$ERRFL	000000	1154#	1155	1157#	1161#	1162	1164#	1165	1166#	1167	1169#	1170	1171#	1173
		1174#	1176	1177#	1179	1180#	1182	1183#	1184	1196#	1197	1199#	1200#	1201#
		1202#	1203#	1204	1205#	1206	1209#	1210	1212#	1213	1226#	1227	1228#	1230
		1236#	1237#	1239#	1240#	1241	1263#	1264	1269#	1270	1283#	1284	1285#	1286
		1299#	1300	1302#	1303	1316#	1317	1319#	1320	1345#	1346	1359#	1360	1361#
		1362	1375#	1376	1378#	1379	1392#	1393	1395#	1396	1427#	1428	1441#	1442
		1443#	1444	1457#	1458	1460#	1461	1474#	1475	1477#	1478	1510#	1511	1516#
		1517	1518#	1519	1532#	1533	1535#	1536	1548#	1549	1550#	1551	1577#	1578
		1584#	1585	1598#	1599	1600#	1601	1614#	1615	1617#	1618	1631#	1632	1634#
		1635	1660#	1661	1674#	1675	1676#	1677	1690#	1691	1693#	1694	1706#	1707
		1710#	1711	1736#	1737	1750#	1751	1752#	1753	1766#	1767	1769#	1770	1783#
		1784	1786#	1787	1812#	1813	1826#	1827	1828#	1829	1842#	1843	1845#	1846
		1859#	1860	1862#	1863	1899#	1900	1931#	1932	1964#	1965	2000#	2001	2008#
		2009	2056#	2057	2068#	2069	2072#	2073	2088#	2089	2092#	2093	2108#	2109
		2112#	2113	2145#	2146	2157#	2158	2161#	2162	2177#	2178	2181#	2182	2197#
		2198	2201#	2202	2235#	2236	2247#	2248	2251#	2252	2267#	2268	2271#	2272
		2287#	2288	2291#	2292	2323#	2324	2335#	2336	2339#	2340	2355#	2356	2359#
		2360	2375#	2376	2379#	2380	2415#	2416	2428#	2429	2431#	2432	2434#	2440#
		2441	2455#	2456	2459#	2460	2481#	2482	2485#	2486	2515#	2516	2528#	2529
		2532#	2533	2535#	2541#	2542	2555#	2556	2560#	2561	2563#	2579#	2580	2583#
		2584	2586#	2618#	2619	2633#	2634	2635#	2636	2639#	2640	2643#	2644	2646#
		2662#	2663	2665#	2666	2668#	2700#	2701	2705#	2706	2713#	2714	2718#	2737#
		2738	2744#	2745	2761#	2789#	2790	2792#	2793	2796#	2797	2802#	2822#	2823
		2848#	2849	2850#	2851	2856#	2857	2861#	2883#	2884	2909#	2910	2914#	2915
		2916#	2917	2918#	2919	2924#	2925	2931#	2932	2936#	2937	2953#	2954	2955#
		2956	2982#	2983	3008#	3009	3035#	3036	3042#	3043	3045#	3052#	3053	3055#
		3062#	3063	3074#	3075	3082#	3083	3094#	3095	3101#	3102	3119#	3120	3124#

	3125	3127#	3142#	3143	3177#	3178	3186#	3187	3192#	3193	3194#	3195	3196#	
	3197	3199#	3200	3202	3206	3207#	3208	3210#	3211	3213#	3214	3216#	3217	
	3219#	3220	3221#	3222	3231#	3232	3237#	3238	3239#	3240	3276#	3277	3278#	
	3279	3284#	3288	3289#	3290	3346#	3347	3350#	3351	3353#	3354	3356#	3357#	
	3358#	3359#	3360#	3361	3362#	3363	3365#	3366	3375#	3376	3379#	3409#	3410	
	3412#	3413	3415#	3416	3423#	3442#	3443#	3445#	3446#	3447	3470#	3471#	3472#	
	3473#	3474#	3475	3477#	3478	3479#	3480	3482#	3483	3485#	3486	3495#	3496	
	3500#	3509#	3510	3512#	3517#	3518	3543#	3544	3546#	3547	3557#	3558	3563#	
	3564	3569#	3570	3572#	3573	3580#	3581	3583#	3584	3585#	3586	3588#	3616#	
	3617	3618#	3619	3622#	3623	3626#	3627#	3629#	3630#	3631	3644#	3645	3654#	
	3659#	3660	3663#	3664	3666	3670	3675#	3685#	3686	3688#	3697#	3698	3705#	
	3707	3708#	3709	3718#	3719	3728#	3729	3745#	3746	3751#	3752	3760#	3764#	
	3765	3767#	3768	3770	3774	3779#	3789#	3790	3791#	3801#	3802	3805#	3807	
	3808#	3809	3818#	3819	3849#	3850	3855#	3879#	3880	3882#	3883	3884#	3885	
	3888#	3889	3890#	3891	3894#	3895	3897#	3898	3900#	3901	3904#	3905	3910#	
	3911	3914#	3915	3917#	3918	3941#	3942	3943#	3944	3964#	3965	3967#	3969	
	3971#	3972	3978#	3979	3999#	4000	4002#	4004	4006#	4008	4019#	4020	4021#	
	4022	4026#	4027	4036#	4037	4066#	4067	4072#	4073	4076#	4077	4081#	4082	
	4084#	4085	4086#	4174#	4175	4176#	4177	4178#	4179	4190#	4191	4195#	4196	
	4205#	4206	4218#	4223#	4224	4236#	4237	4240#	4242	4271#	4272	4273#	4277	
	4279#	4280	4282	4286	4287#	4288	4289#	4290	4294#	4295	4317#	4318	4319#	
	4320	4322	4326	4327#	4328	4330	4334	4360#	4361	4422#	4423	4427#	4431#	
	4432	4435#	4436	4437#	4438	4439#	4440	4441#	4442	4446#	4447	4448#	4449	
	4450#	4451	4452#	4453	4460#	4461	4462#	4463						
\$ERROR	014170	1099	4849#											
\$ERRPC	001116	987#	4385	4858*	4859*	4860	4889							
\$ERRTB	001254	1076#												
\$ERTTL	001112	984#	4141	4143*	4857*	4889								
\$ESCAP	001162	1008#	1107*	4880	4882	4889	4947*							
\$ETABL	001214	1027#												
\$ETEND	001254	969	1060#											
\$FATAL	001176	1020#	4378*	4379	4818*									
\$FFLG	014166	4781*	4784*	4812	4821*	4829#								
\$FILLC	001156	1005#	4608	4639										
\$FILLS	001155	1004#	4639											
\$F\$AND	000310	1#	1219	1262	1274	1290	1307	1326	1350	1366	1383	1402	1432	1448
		1465	1484	1509	1523	1539	1555	1576	1589	1605	1622	1641	1665	1681
		1698	1717	1741	1757	1774	1793	1817	1833	1850	1869	1903	1936	1968
		1998	2013	2052	2077	2097	2117	2143	2166	2186	2206	2232	2256	2276
		2296	2321	2344	2364	2384	2412	2445	2463	2472	2489	2512	2546	2570
		2593	2616	2653	2675	2699	2751	2831	2888	2908	2930	2948	2969	2973
		2976	2996	3013	3068	3086	3107	3135	3150	3176	3185	3226	3229	3250
		3258	3388	3392	3430	3523	3527	3561	3597	3601	3714	3743	3814	3848
		3934	3976	4013	4017	4034	4065	4104	4189	4203	4235	4418	4421	
\$F\$RAD	000401	1#	1219	1262	1274	1290	1307	1326	1350	1366	1383	1402	1432	1448
		1465	1484	1509	1523	1539	1555	1576	1589	1605	1622	1641	1665	1681
		1698	1717	1741	1757	1774	1793	1817	1833	1850	1869	1903	1936	1968
		1998	2013	2052	2077	2097	2117	2143	2166	2186	2206	2232	2256	2276
		2296	2321	2344	2364	2384	2412	2445	2463	2472	2489	2512	2546	2570
		2593	2616	2653	2675	2699	2751	2831	2888	2908	2930	2948	2969	2973
		2976	2996	3013	3068	3086	3107	3135	3150	3176	3185	3226	3229	3250
		3260	3388	3392	3430	3523	3527	3561	3597	3601	3714	3743	3814	3848
		3934	3976	4013	4017	4034	4065	4104	4189	4203	4235	4418	4421	
\$F\$BLA-	000170	1#												
\$F\$CAS	000150	1#												
\$F\$DEC	000220	1#												

\$F\$G00= 000400

1#	1217	1260	1272	1288	1305	1324	1348	1364	1381	1400	1430	1446
1463	1487	1507	1521	1537	1553	1574	1587	1603	1620	1639	1663	1679
1696	1715	1739	1755	1772	1791	1815	1831	1848	1867	1901	1934	1966
1996	2011	2050	2075	2095	2115	2141	2164	2184	2204	2230	2254	2274
2294	2319	2342	2362	2382	2410	2443	2461	2470	2487	2510	2544	2568
2591	2614	2651	2673	2697	2749	2829	2886	2906	2928	2946	2967	2971
2928	2946	2966	2967	2971	2974	2994	3011	3066	3084	3105	3133	3148
3174	3183	3223	3224	3227	3248	3256	3386	3390	3428	3521	3525	3559
3595	3599	3712	3741	3812	3846	3932	3974	4011	4015	4032	4063	4094
4102	4187	4201	4233	4416	4419							

\$F\$IF - 000110

1#	1217	1223	1260	1266	1272	1278	1288	1294	1305	1311	1324	1330
1348	1354	1364	1370	1381	1387	1400	1406	1430	1436	1446	1452	1463
1469	1482	1488	1507	1513	1521	1527	1537	1543	1553	1559	1574	1580
1587	1593	1603	1609	1620	1626	1639	1645	1663	1669	1679	1685	1696
1702	1715	1721	1739	1745	1755	1761	1772	1778	1791	1797	1815	1821
1831	1837	1848	1854	1867	1873	1901	1912	1934	1945	1966	1979	1996
2003	2011	2026	2050	2059	2075	2083	2095	2103	2115	2123	2141	2148
2164	2172	2184	2192	2204	2212	2230	2238	2254	2262	2274	2282	2294
2302	2319	2326	2342	2350	2362	2370	2382	2390	2410	2418	2443	2449
2461	2467	2470	2476	2487	2493	2510	2518	2544	2550	2568	2574	2591
2597	2614	2621	2651	2657	2673	2679	2697	2703	2728	2733	2749	2755
2769	2774	2812	2817	2829	2835	2870	2875	2886	2892	2906	2912	2928
2933	2938	2946	2958	2971	2974	2985	2987	2994	3001	3011	3017	3066
3077	3084	3097	3105	3115	3133	3145	3148	3155	3174	3180	3183	3189
3227	3233	3241	3248	3252	3256	3258	3262	3271	3273	3386	3390	3396
3403	3405	3428	3434	3521	3525	3532	3538	3540	3559	3566	3595	3599
3606	3612	3614	3712	3721	3741	3748	3812	3821	3846	3852	3932	3938
3974	3980	4011	4015	4023	4028	4032	4039	4063	4069	4094	4099	4102
4108	4187	4192	4197	4201	4207	4233	4238	4416	4419	4424	4433	4443
4454												

\$F\$INC 000210

1#	3199	3281	3663	3723	3767	3824	4279	4291	4319	4327	4335	4338
----	------	------	------	------	------	------	------	------	------	------	------	------

\$F\$L00= 000200

1#	4184	4210	4225									
----	------	------	------	--	--	--	--	--	--	--	--	--

\$F\$NAM- 000160

1#												
----	--	--	--	--	--	--	--	--	--	--	--	--

\$F\$NO - 000403

1#	1217	1260	1272	1288	1305	1324	1348	1364	1381	1400	1430	1446
1463	1482	1507	1521	1537	1553	1574	1587	1603	1620	1639	1663	1679
1696	1715	1739	1755	1772	1791	1815	1831	1848	1867	1901	1934	1966
1996	2011	2050	2075	2095	2115	2141	2164	2184	2204	2230	2254	2274
2294	2319	2342	2362	2382	2410	2443	2461	2470	2487	2510	2544	2568
2591	2614	2651	2673	2697	2749	2829	2886	2906	2928	2946	2967	2971
2974	2994	3011	3066	3084	3105	3133	3148	3174	3183	3224	3227	3248
3256	3258	3386	3390	3428	3521	3525	3559	3595	3599	3712	3741	3812
3846	3932	3974	4011	4015	4032	4063	4187	4416	4419			

\$F\$OR 000320

1#	1219	1262	1274	1290	1307	1326	1350	1366	1383	1402	1432	1448
1465	1484	1509	1523	1539	1555	1576	1589	1605	1622	1641	1665	1681
1698	1717	1741	1757	1774	1793	1817	1833	1850	1869	1903	1936	1968
1998	2013	2052	2077	2097	2117	2143	2166	2186	2206	2232	2256	2276
2296	2321	2344	2364	2384	2412	2445	2463	2472	2489	2512	2546	2570
2593	2616	2653	2675	2699	2751	2831	2888	2908	2930	2948	2969	2973
2976	2996	3013	3068	3086	3107	3135	3150	3176	3185	3226	3229	3250
3258	3260	3388	3392	3430	3523	3527	3561	3597	3601	3714	3743	3814
3848	3934	3976	4013	4017	4034	4065	4104	4189	4203	4235	4418	4421

\$F\$RTN= 000300

1#	4150	4249	4257	4299	4306	4344	4367	4400	4405	4471		
----	------	------	------	------	------	------	------	------	------	------	--	--

\$F\$SEL- 000140

1#												
----	--	--	--	--	--	--	--	--	--	--	--	--

\$F\$UNT= 000130

1#	1207	1231	2926	2940	3922	3924	4414	4456				
----	------	------	------	------	------	------	------	------	--	--	--	--

\$F\$WHI- 000120

1#	2966	2967	2989	3223	3224	3244	3258					
----	------	------	------	------	------	------	------	--	--	--	--	--

\$F\$YES 000402

1#	1217	1260	1272	1288	1305	1324	1348	1364	1381	1400	1430	1446
----	------	------	------	------	------	------	------	------	------	------	------	------

1463	1482	1507	1521	1537	1553	1574	1587	1603	1620	1639	1663	1679
1696	1715	1739	1755	1772	1791	1815	1831	1848	1867	1901	1934	1966
1996	2011	2050	2075	2095	2115	2141	2164	2184	2204	2230	2254	2274
2294	2309	2342	2362	2382	2410	2443	2461	2470	2487	2510	2544	2568
2591	2614	2651	2673	2697	2749	2829	2886	2906	2928	2946	2967	2971
2974	2994	3011	3066	3084	3105	3133	3148	3174	3183	3224	3227	3248
3256	3258	3386	3390	3428	3521	3525	3559	3595	3599	3712	3741	3812
3846	3932	3974	4011	4015	4032	4063	4102	4187	4201	4233	4416	4419

\$GDADR 001120
 \$GDDAT 001124
 \$GET42 012436
 \$GTSWR 013216
 \$HD = 000000
 \$HIBTS 001000
 \$ICNT 001104
 \$IFLEV= 177777

988#												
990#												
4501#												
4661#	5134											
689												
964#												
981#	4938*	4939	4941*	4952								
1#	1217#	1223#	1260#	1266#	1272#	1278#	1288#	1294#	1305#	1311#	1324#	1330#
1348#	1354#	1364#	1370#	1381#	1387#	1400#	1406#	1430#	1436#	1446#	1452#	1463#
1469#	1482#	1488#	1507#	1513#	1521#	1527#	1537#	1543#	1553#	1559#	1574#	1580#
1587#	1593#	1603#	1609#	1620#	1626#	1639#	1645#	1663#	1669#	1679#	1685#	1696#
1702#	1715#	1721#	1739#	1745#	1755#	1761#	1772#	1778#	1791#	1797#	1815#	1821#
1831#	1837#	1848#	1854#	1867#	1873#	1901#	1912#	1934#	1945#	1966#	1979#	1996#
2003#	2011#	2026#	2050#	2059#	2075#	2083#	2095#	2103#	2115#	2123#	2141#	2148#
2164#	2172#	2184#	2192#	2204#	2212#	2230#	2238#	2254#	2262#	2274#	2282#	2294#
2302#	2319#	2326#	2342#	2350#	2362#	2370#	2382#	2390#	2410#	2418#	2443#	2449#
2461#	2467#	2470#	2476#	2487#	2493#	2510#	2518#	2544#	2550#	2568#	2574#	2591#
2597#	2614#	2621#	2651#	2657#	2673#	2679#	2697#	2703#	2728#	2733#	2749#	2755#
2769#	2774#	2812#	2817#	2829#	2835#	2870#	2875#	2886#	2892#	2906#	2912#	2928#
2938#	2946#	2958#	2971#	2974#	2985#	2987#	2994#	3001#	3011#	3017#	3066#	3077#
3084#	3097#	3105#	3115#	3133#	3145#	3148#	3155#	3174#	3180#	3183#	3189#	3227#
3241#	3248#	3256#	3271#	3273#	3386#	3390#	3403#	3405#	3428#	3434#	3521#	3525#
3538#	3540#	3559#	3566#	3595#	3599#	3612#	3614#	3712#	3721#	3741#	3748#	3812#
3821#	3846#	3852#	3932#	3938#	3974#	3980#	4011#	4015#	4023#	4028#	4032#	4039#
4063#	4069#	4094#	4099#	4102#	4108#	4187#	4197#	4201#	4207#	4233#	4238#	4416#
4419#	4433#	4454#										
4519	4535	4554#										
995#	4689	4778										
1217#	1223	1260#	1266	1272#	1278	1288#	1294	1305#	1311	1324#	1330	1348#
1354	1364#	1370	1381#	1387	1400#	1406	1430#	1436	1446#	1452	1463#	1469
1482#	1488	1507#	1513	1521#	1527	1537#	1543	1553#	1559	1574#	1580	1587#
1593	1603#	1609	1620#	1626	1639#	1645	1663#	1669	1679#	1685	1696#	1702
1715#	1721	1739#	1745	1755#	1761	1772#	1778	1791#	1797	1815#	1821	1831#
1837	1848#	1854	1867#	1873	1901#	1912	1934#	1945	1966#	1979	1996#	2003
2011#	2026	2050#	2059	2075#	2083	2095#	2103	2115#	2123	2141#	2148	2164#
2172	2184#	2192	2204#	2212	2230#	2238	2254#	2262	2274#	2282	2294#	2302
2319#	2326	2342#	2350	2362#	2370	2382#	2390	2410#	2418	2443#	2449	2461#
2467	2470#	2476	2487#	2493	2510#	2518	2544#	2550	2568#	2574	2591#	2597
2614#	2621	2651#	2657	2673#	2679	2697#	2703	2728#	2733	2749#	2755	2769#
2774	2812#	2817	2829#	2835	2870#	2875	2886#	2892	2906#	2912	2928#	2938
2946#	2958	2971#	2987	2994#	3001	3011#	3017	3066#	3077	3084#	3097	3105#
3115	3133#	3145	3148#	3155	3174#	3180	3183#	3189	3227#	3241	3248#	3273
3386#	3405	3428#	3434	3521#	3540	3559#	3566	3595#	3614	3712#	3721	3741#
3748	3812#	3821	3846#	3852	3932#	3938	3974#	3980	4011#	4028	4032#	4039
4063#	4069	4094#	4099	4102#	4108	4187#	4197	4201#	4207	4233#	4238	4416#
4454												
2974#	2985	3256#	3271	3390#	3403	3525#	3538	3599#	3612	4015#	4023	4419#
4433												

\$ILLUP 012646
 \$INTAG 001135
 \$ISK0 000001

\$ISK1 000001

\$ITEMB 001114
\$LF 001172
\$LFLG 014165
\$LOCTA= 177777

985#	4378	4860*	4868	4889										
1012#	4639	4763	4772	4889										
4822*	4828#													
1#	1207	1208	1218	1219	1223	1224	1232	1233	1261	1262	1266	1267		
1273	1274	1278	1279	1289	1290	1294	1295	1306	1307	1311	1312	1325		
1326	1330	1331	1349	1350	1354	1355	1365	1366	1370	1371	1382	1383		
1387	1388	1401	1402	1406	1407	1431	1432	1436	1437	1447	1448	1452		
1453	1464	1465	1469	1470	1483	1484	1488	1489	1508	1509	1513	1514		
1522	1523	1527	1528	1538	1539	1543	1544	1554	1555	1559	1560	1575		
1576	1580	1581	1588	1589	1593	1594	1604	1605	1609	1610	1621	1622		
1626	1627	1640	1641	1645	1646	1664	1665	1669	1670	1680	1681	1685		
1686	1697	1698	1702	1703	1716	1717	1721	1722	1740	1741	1745	1746		
1756	1757	1761	1762	1773	1774	1778	1779	1792	1793	1797	1798	1816		
1817	1821	1822	1832	1833	1837	1838	1849	1850	1854	1855	1868	1869		
1873	1874	1902	1903	1912	1913	1935	1936	1945	1946	1967	1968	1979		
1980	1997	1998	2003	2004	2012	2013	2026	2027	2051	2052	2059	2060		
2076	2077	2083	2084	2096	2097	2103	2104	2116	2117	2123	2124	2142		
2143	2148	2149	2165	2166	2172	2173	2185	2186	2192	2193	2205	2206		
2212	2213	2231	2232	2238	2239	2255	2256	2262	2263	2275	2276	2282		
2283	2295	2296	2302	2303	2320	2321	2326	2327	2343	2344	2350	2351		
2363	2364	2370	2371	2383	2384	2390	2391	2411	2412	2418	2419	2444		
2445	2449	2450	2462	2463	2467	2468	2471	2472	2476	2477	2488	2489		
2493	2494	2511	2512	2518	2519	2545	2546	2550	2551	2569	2570	2574		
2575	2592	2593	2597	2598	2615	2616	2621	2622	2652	2653	2657	2658		
2674	2675	2679	2680	2698	2699	2703	2704	2728	2729	2733	2734	2750		
2751	2755	2756	2769	2770	2774	2775	2812	2813	2817	2818	2830	2831		
2835	2836	2870	2871	2875	2876	2887	2888	2892	2893	2907	2908	2912		
2913	2926	2927	2929	2930	2933	2934	2935	2938	2939	2941	2942	2943		
2944	2945	2947	2948	2958	2959	2966	2967	2968	2969	2972	2973	2975		
2976	2985	2986	2987	2988	2989	2990	2991	2995	2996	3001	3002	3012		
3013	3017	3018	3067	3068	3077	3078	3085	3086	3097	3098	3106	3107		
3115	3116	3134	3135	3145	3146	3149	3150	3155	3156	3175	3176	3180		
3181	3184	3185	3189	3190	3200	3201	3202	3203	3204	3205	3206	3223		
3224	3225	3226	3228	3229	3233	3234	3235	3241	3242	3244	3245	3246		
3249	3250	3252	3253	3254	3257	3258	3259	3260	3262	3263	3264	3271		
3272	3273	3274	3281	3282	3283	3287	3288	3291	3292	3296	3297	3298		
3403	3404	3405	3406	3429	3430	3434	3435	3522	3523	3526	3527	3532		
3533	3534	3538	3539	3540	3541	3560	3561	3566	3567	3596	3597	3600		
3601	3606	3607	3608	3612	3613	3614	3615	3664	3665	3666	3667	3668		
3669	3670	3713	3714	3721	3722	3723	3724	3725	3742	3743	3748	3749		
3768	3769	3770	3771	3772	3773	3774	3813	3814	3821	3822	3824	3825		
3826	3847	3848	3852	3853	3922	3923	3925	3926	3927	3928	3929	3933		
3934	3938	3939	3975	3976	3980	3981	4012	4013	4016	4017	4023	4024		
4028	4029	4033	4034	4039	4040	4064	4065	4069	4070	4094	4095	4099		
4100	4103	4104	4108	4109	4149	4184	4185	4188	4189	4192	4193	4194		
4197	4198	4202	4203	4207	4208	4211	4212	4213	4214	4225	4226	4227		
4234	4235	4236	4237	4238	4239	4241	4242	4249	4250	4251	4252	4256		
4280	4281	4282	4283	4284	4285	4286	4291	4292	4293	4296	4297	4299		
4300	4301	4305	4320	4321	4322	4323	4324	4325	4326	4328	4329	4330		
4331	4332	4333	4334	4335	4336	4337	4338	4339	4340	4344	4345	4346		
4366	4400	4401	4402	4404	4414	4415	4417	4418	4420	4421	4424	4425		
4426	4433	4434	4443	4444	4445	4454	4455	4457	4458	4464	4465	4471		
4472	4473													
\$LPADR 001106	982#	1109*	4929*	4945*	4950	4952								
\$LPERR 001110	983#	1110*	1209*	1269*	1283*	1299*	1316*	1345*	1359*	1375*	1392*	1427*	1441*	
	1457*	1474*	1516*	1532*	1548*	1584*	1598*	1614*	1631*	1660*	1690*	1706*		

SLSTCN- 177777

1736*	1750*	1766*	1783*	1812*	1826*	1842*	1859*	1899*	1931*	1964*	2008*	2068*
2088*	2108*	2157*	2177*	2197*	2247*	2267*	2287*	2335*	2355*	2375*	2428*	2455*
2481*	2528*	2555*	2579*	2639*	2662*	2705*	2737*	2792*	2822*	2850*	3035*	3082*
3101*	3119*	3362*	3409*	3477*	3557*	4879	4929	4946*	4952			
1#	1207	1208	1217	1219	1223	1231	1260	1262	1266	1272	1274	1278
1288	1290	1294	1305	1307	1311	1324	1326	1330	1348	1350	1354	1364
1366	1370	1381	1383	1387	1400	1402	1406	1430	1432	1436	1446	1448
1452	1463	1465	1469	1482	1484	1488	1507	1509	1513	1521	1523	1527
1537	1539	1543	1553	1555	1559	1574	1576	1580	1587	1589	1593	1603
1605	1609	1620	1622	1626	1639	1641	1645	1663	1665	1669	1679	1681
1685	1696	1698	1702	1715	1717	1721	1739	1741	1745	1755	1757	1761
1772	1774	1778	1791	1793	1797	1815	1817	1821	1831	1833	1837	1848
1850	1854	1867	1869	1873	1901	1903	1912	1934	1936	1945	1966	1968
1979	1996	1998	2003	2011	2013	2026	2050	2052	2059	2075	2077	2083
2095	2097	2103	2115	2117	2123	2141	2143	2148	2164	2166	2172	2184
2186	2192	2204	2206	2212	2230	2232	2238	2254	2256	2262	2274	2276
2282	2294	2296	2302	2319	2321	2326	2342	2344	2350	2362	2364	2370
2382	2384	2390	2410	2412	2418	2443	2445	2449	2461	2463	2467	2470
2472	2476	2487	2489	2493	2510	2512	2518	2544	2546	2550	2568	2570
2574	2591	2593	2597	2614	2616	2621	2651	2653	2657	2673	2675	2679
2697	2699	2703	2728	2729	2733	2749	2751	2755	2769	2770	2774	2812
2813	2817	2829	2831	2835	2870	2877	2875	2886	2888	2892	2906	2908
2912	2926	2927	2928	2930	2934	2935	2938	2940	2945	2946	2948	2958
2966	2967	2969	2971	2973	2974	2976	2985	2987	2989	2990	2994	2996
3001	3011	3013	3017	3066	3068	3077	3084	3086	3097	3105	3107	3115
3133	3135	3145	3148	3150	3155	3174	3176	3180	3183	3185	3189	3199
3201	3202	3203	3206	3223	3224	3226	3227	3229	3234	3235	3241	3244
3245	3248	3250	3253	3254	3256	3260	3263	3264	3271	3273	3281	3282
3386	3388	3390	3392	3397	3398	3403	3405	3428	3430	3434	3521	3523
3525	3527	3533	3534	3538	3540	3559	3561	3566	3595	3597	3599	3601
3607	3608	3612	3614	3663	3665	3666	3667	3670	3712	3714	3721	3723
3724	3741	3743	3748	3767	3769	3770	3771	3774	3812	3814	3821	3824
3825	3846	3848	3852	3922	3923	3924	3929	3932	3934	3938	3974	3976
3980	4011	4013	4015	4017	4023	4028	4032	4034	4039	4063	4065	4069
4094	4095	4099	4102	4104	4108	4150	4184	4185	4187	4189	4193	4194
4197	4201	4203	4207	4225	4226	4233	4235	4238	4249	4257	4279	4281
4282	4283	4286	4291	4292	4299	4306	4319	4321	4322	4323	4326	4327
4329	4330	4331	4334	4335	4336	4338	4339	4344	4367	4400	4405	4414
4415	4416	4418	4419	4421	4425	4426	4433	4444	4445	4454	4456	4471
1#	1154	1155	1157	1158	1161	1162	1164	1165	1166	1167	1169	1170
1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
1184	1196	1197	1199	1200	1201	1202	1203	1204	1205	1206	1209	1210
1212	1213	1217	1218	1219	1225	1227	1228	1229	1230	1231	1232	1233
1236	1237	1238	1239	1240	1241	1260	1261	1262	1263	1264	1269	1270
1272	1273	1274	1283	1284	1285	1286	1288	1289	1290	1299	1300	1302
1303	1305	1306	1307	1316	1317	1319	1320	1324	1325	1326	1345	1346
1348	1349	1350	1359	1360	1361	1362	1364	1365	1366	1375	1376	1378
1379	1381	1382	1383	1392	1393	1395	1396	1400	1401	1402	1427	1428
1430	1431	1432	1441	1442	1443	1444	1446	1447	1448	1457	1458	1460
1461	1463	1464	1465	1474	1475	1477	1478	1482	1483	1484	1507	1508
1509	1510	1511	1516	1517	1518	1519	1521	1522	1523	1532	1533	1535
1536	1537	1538	1539	1548	1549	1550	1551	1553	1554	1555	1574	1575
1576	1577	1578	1584	1585	1587	1588	1589	1598	1599	1600	1601	1603
1604	1605	1614	1615	1617	1618	1620	1621	1622	1631	1632	1634	1635
1639	1640	1641	1660	1661	1663	1664	1665	1674	1675	1676	1677	1679
1680	1681	1690	1691	1693	1694	1696	1697	1698	1706	1707	1710	1711

SLSTIN 000000

1715	1716	1717	1736	1737	1739	1740	1741	1750	1751	1752	1753	1755
1756	1757	1766	1767	1769	1770	1772	1773	1774	1783	1784	1786	1787
1791	1792	1793	1812	1813	1815	1816	1817	1826	1827	1828	1829	1831
1832	1833	1842	1843	1845	1846	1848	1849	1850	1859	1860	1862	1863
1867	1868	1869	1899	1900	1901	1902	1903	1931	1932	1934	1935	1936
1964	1965	1966	1967	1968	1996	1997	1998	2000	2001	2008	2009	2011
2012	2013	2050	2051	2052	2056	2057	2068	2069	2072	2073	2075	2076
2077	2088	2089	2092	2093	2095	2096	2097	2108	2109	2112	2113	2115
2116	2117	2141	2142	2143	2145	2146	2157	2158	2161	2162	2164	2165
2166	2177	2178	2181	2182	2184	2185	2186	2197	2198	2201	2202	2204
2205	2206	2230	2231	2232	2235	2236	2247	2248	2251	2252	2254	2255
2256	2267	2268	2271	2272	2274	2275	2276	2287	2288	2291	2292	2294
2295	2296	2319	2320	2321	2323	2324	2335	2336	2339	2340	2342	2343
2344	2355	2356	2359	2360	2362	2363	2364	2375	2376	2379	2380	2382
2383	2384	2410	2411	2412	2415	2416	2428	2429	2431	2432	2434	2435
2436	2437	2438	2440	2441	2443	2444	2445	2455	2456	2459	2460	2461
2462	2463	2470	2471	2472	2481	2482	2485	2486	2487	2488	2489	2510
2511	2512	2515	2516	2528	2529	2532	2533	2535	2536	2537	2538	2539
2541	2542	2544	2545	2546	2555	2556	2560	2561	2563	2564	2565	2566
2567	2568	2569	2570	2579	2580	2583	2584	2586	2587	2588	2589	2590
2591	2592	2593	2614	2615	2616	2618	2619	2633	2634	2635	2636	2639
2640	2643	2644	2646	2647	2648	2649	2650	2651	2652	2653	2662	2663
2665	2666	2668	2669	2670	2671	2672	2673	2674	2675	2697	2698	2699
2700	2701	2705	2706	2713	2714	2718	2719	2720	2721	2722	2723	2724
2725	2728	2729	2737	2738	2744	2745	2749	2750	2751	2761	2762	2763
2764	2765	2766	2767	2768	2769	2770	2789	2790	2792	2793	2796	2797
2802	2803	2804	2805	2806	2807	2808	2809	2812	2813	2822	2823	2829
2830	2831	2848	2849	2850	2851	2856	2857	2861	2862	2863	2864	2865
2866	2867	2868	2870	2871	2883	2884	2886	2887	2888	2906	2907	2908
2909	2910	2914	2915	2916	2917	2918	2919	2924	2925	2928	2929	2930
2931	2932	2933	2934	2936	2937	2940	2941	2942	2943	2944	2946	2947
2948	2953	2954	2955	2956	2967	2968	2969	2971	2972	2973	2974	2975
2976	2982	2983	2989	2990	2994	2995	2996	3008	3009	3011	3012	3013
3035	3036	3042	3043	3045	3046	3047	3048	3049	3052	3053	3055	3056
3057	3058	3059	3062	3063	3066	3067	3068	3074	3075	3082	3083	3084
3085	3086	3094	3095	3101	3102	3105	3106	3107	3119	3120	3124	3125
3127	3128	3129	3130	3131	3133	3134	3135	3142	3143	3148	3149	3150
3174	3175	3176	3177	3178	3183	3184	3185	3186	3187	3192	3193	3194
3195	3196	3197	3199	3200	3201	3202	3203	3204	3205	3206	3207	3208
3210	3211	3213	3214	3216	3217	3219	3220	3221	3222	3224	3225	3226
3227	3228	3229	3231	3232	3233	3234	3237	3238	3239	3240	3244	3245
3248	3249	3250	3252	3253	3256	3257	3258	3259	3260	3262	3263	3276
3277	3278	3279	3281	3282	3284	3285	3286	3287	3288	3289	3290	3346
3347	3350	3351	3353	3354	3356	3357	3358	3359	3360	3361	3362	3363
3365	3366	3375	3376	3379	3380	3381	3382	3383	3386	3387	3388	3390
3391	3392	3396	3397	3409	3410	3412	3413	3415	3416	3423	3424	3425
3426	3427	3428	3429	3430	3442	3443	3444	3445	3446	3447	3470	3471
3472	3473	3474	3475	3477	3478	3479	3480	3482	3483	3485	3486	3495
3496	3500	3501	3502	3503	3504	3505	3506	3507	3509	3510	3512	3513
3514	3515	3516	3517	3518	3521	3522	3523	3525	3526	3527	3532	3533
3543	3544	3546	3547	3557	3558	3559	3560	3561	3563	3564	3569	3570
3572	3573	3580	3581	3583	3584	3585	3586	3588	3589	3590	3591	3592
3595	3596	3597	3599	3600	3601	3606	3607	3616	3617	3618	3619	3622
3623	3626	3627	3628	3629	3630	3631	3644	3645	3654	3655	3656	3657
3659	3660	3663	3664	3665	3666	3667	3668	3669	3670	3675	3676	3677
3678	3679	3680	3681	3682	3685	3686	3688	3689	3690	3691	3692	3693

3694	3695	3697	3698	3705	3706	3707	3708	3709	3712	3713	3714	3718
3719	3723	3724	3728	3729	3741	3742	3743	3745	3746	3751	3752	3760
3761	3762	3763	3764	3765	3767	3768	3769	3770	3771	3772	3773	3774
3779	3780	3781	3782	3783	3784	3785	3786	3789	3790	3791	3792	3793
3794	3795	3796	3797	3798	3801	3802	3805	3806	3807	3808	3809	3812
3813	3814	3818	3819	3824	3825	3846	3847	3848	3849	3850	3855	3856
3857	3858	3879	3880	3882	3883	3884	3885	3888	3889	3890	3891	3894
3895	3897	3898	3900	3901	3904	3905	3910	3911	3914	3915	3917	3918
3924	3925	3926	3927	3928	3932	3933	3934	3941	3942	3943	3944	3964
3965	3967	3968	3969	3971	3972	3974	3975	3976	3978	3979	3999	4000
4002	4003	4004	4006	4007	4008	4011	4012	4013	4015	4016	4017	4019
4020	4021	4022	4026	4027	4032	4033	4034	4036	4037	4063	4064	4065
4066	4067	4072	4073	4076	4077	4081	4082	4084	4085	4086	4087	4088
4089	4090	4091	4092	4093	4094	4095	4102	4103	4104	4174	4175	4176
4177	4178	4179	4187	4188	4189	4190	4191	4192	4193	4195	4196	4201
4202	4203	4205	4206	4210	4211	4212	4213	4214	4218	4219	4220	4221
4222	4223	4224	4225	4226	4233	4234	4235	4236	4237	4240	4241	4242
4250	4251	4252	4253	4271	4272	4273	4274	4275	4276	4277	4279	4280
4281	4282	4283	4284	4285	4286	4287	4288	4289	4290	4291	4292	4294
4295	4296	4297	4301	4302	4317	4318	4319	4320	4321	4322	4323	4324
4325	4326	4327	4328	4329	4330	4331	4332	4333	4334	4335	4336	4338
4339	4346	4347	4360	4361	4402	4403	4416	4417	4418	4419	4420	4421
4422	4423	4424	4425	4427	4428	4431	4432	4435	4436	4437	4438	4439
4440	4441	4442	4443	4444	4446	4447	4448	4449	4450	4451	4452	4453
4456	4457	4458	4460	4461	4462	4463	4464	4465	4473	4474		
	1207	1208	1217	1219	1223	1231	1260	1262	1266	1272	1274	1278
1288	1290	1294	1305	1307	1311	1324	1326	1330	1348	1350	1354	1364
1366	1370	1381	1383	1387	1400	1402	1406	1430	1432	1436	1446	1448
1452	1463	1465	1469	1482	1484	1488	1507	1509	1513	1521	1523	1527
1537	1539	1543	1553	1555	1559	1574	1576	1580	1587	1589	1593	1603
1605	1609	1620	1622	1626	1639	1641	1645	1663	1665	1669	1679	1681
1685	1696	1698	1702	1715	1717	1721	1739	1741	1745	1755	1757	1761
1772	1774	1778	1791	1793	1797	1815	1817	1821	1831	1833	1837	1848
1850	1854	1867	1869	1873	1901	1903	1912	1934	1936	1945	1966	1968
1979	1996	1998	2003	2011	2013	2026	2050	2052	2059	2075	2077	2083
2095	2097	2103	2115	2117	2123	2141	2143	2148	2164	2166	2172	2184
2186	2192	2204	2206	2212	2230	2232	2238	2254	2256	2262	2274	2276
2282	2294	2296	2302	2319	2321	2326	2342	2344	2350	2362	2364	2370
2382	2384	2390	2410	2412	2418	2443	2445	2449	2461	2463	2467	2470
2472	2476	2487	2489	2493	2510	2512	2518	2544	2546	2550	2568	2570
2574	2591	2593	2597	2614	2616	2621	2651	2653	2657	2673	2675	2679
2697	2699	2703	2728	2729	2733	2749	2751	2755	2769	2770	2774	2812
2813	2817	2829	2831	2835	2870	2871	2875	2886	2888	2892	2906	2908
2912	2926	2927	2928	2930	2933	2934	2935	2938	2940	2946	2948	2958
2966	2967	2969	2971	2973	2974	2976	2985	2987	2989	2990	2994	2996
3001	3011	3013	3017	3066	3068	3077	3084	3086	3097	3105	3107	3115
3133	3135	3145	3148	3150	3155	3174	3176	3180	3183	3185	3189	3199
3201	3202	3203	3206	3223	3224	3226	3227	3229	3233	3234	3235	3241
3244	3245	3248	3250	3252	3253	3254	3256	3260	3262	3263	3264	3271
3273	3281	3282	3386	3388	3390	3392	3396	3397	3398	3403	3405	3428
3430	3434	3521	3523	3525	3527	3532	3533	3534	3538	3540	3559	3561
3566	3595	3597	3599	3601	3606	3607	3608	3612	3614	3663	3665	3666
3667	3670	3712	3714	3721	3723	3724	3741	3743	3748	3767	3769	3770
3771	3774	3812	3814	3821	3824	3825	3846	3848	3852	3922	3923	3924
3932	3934	3938	3974	3976	3980	4011	4013	4015	4017	4023	4028	4032
4034	4039	4063	4065	4069	4094	4095	4099	4102	4104	4108	4150	4184

SLSTST- 177777

\$MXCNT 014646
 \$NEST = 177777

4942	4952#	1217#	1223#	1231#	1260#	1266#	1272#	1278#	1288#	1294#	1305#	1311#	
1#	1207#	1217#	1223#	1231#	1260#	1266#	1272#	1278#	1288#	1294#	1305#	1311#	
1324#	1330#	1348#	1354#	1364#	1370#	1381#	1387#	1400#	1406#	1430#	1436#	1446#	
1452#	1463#	1469#	1482#	1488#	1507#	1513#	1521#	1527#	1537#	1543#	1553#	1559#	
1574#	1580#	1587#	1593#	1603#	1609#	1620#	1626#	1639#	1645#	1663#	1669#	1679#	
1685#	1696#	1702#	1715#	1721#	1739#	1745#	1755#	1761#	1772#	1778#	1791#	1797#	
1815#	1821#	1831#	1837#	1848#	1854#	1867#	1873#	1901#	1912#	1934#	1945#	1966#	
1979#	1996#	2003#	2011#	2026#	2050#	2059#	2075#	2083#	2095#	2103#	2115#	2123#	
2141#	2148#	2164#	2172#	2184#	2192#	2204#	2212#	2230#	2238#	2254#	2262#	2274#	
2282#	2294#	2302#	2319#	2326#	2342#	2350#	2362#	2370#	2382#	2390#	2410#	2418#	
2443#	2449#	2461#	2467#	2470#	2476#	2487#	2493#	2510#	2518#	2544#	2550#	2568#	
2574#	2591#	2597#	2614#	2621#	2651#	2657#	2673#	2679#	2697#	2703#	2728#	2733#	
2749#	2755#	2769#	2774#	2812#	2817#	2829#	2835#	2870#	2875#	2886#	2892#	2906#	
2912#	2926#	2928#	2933	2938#	2940#	2946#	2958#	2966#	2971#	2974#	2985#	2987#	
2989#	2994#	3001#	3011#	3017#	3066#	3077#	3084#	3097#	3105#	3115#	3133#	3145#	
3148#	3155#	3174#	3180#	3183#	3189#	3199#	3223#	3227#	3233	3241#	3244#	3248#	
3252	3256#	3262	3271#	3273#	3281#	3386#	3390#	3396	3403#	3405#	3428#	3434#	
3521#	3525#	3532	3538#	3540#	3559#	3566#	3595#	3599#	3606	3612#	3614#	3663#	
3712#	3721#	3723#	3741#	3748#	3767#	3812#	3821#	3824#	3846#	3852#	3922#	3924#	
3932#	3938#	3974#	3980#	4011#	4015#	4023#	4028#	4032#	4039#	4063#	4069#	4094#	
4099#	4102#	4108#	4150#	4184#	4187#	4192	4197#	4201#	4207#	4210	4225#	4233#	
4238#	4249#	4257#	4279#	4291#	4299#	4306#	4319#	4327#	4335#	4338#	4344#	4367#	
4400#	4405#	4414#	4416#	4419#	4424	4433#	4443	4454#	4456#	4471#			
\$NSKJ 000300	1207#	1231	1260#	1266	1272#	1278	1288#	1294	1305#	1311	1324#	1330	1348#
	1354	1364#	1370	1381#	1387	1400#	1406	1430#	1436	1446#	1452	1463#	1469
	1482#	1488	1507#	1513	1521#	1527	1537#	1543	1553#	1559	1574#	1580	1587#
	1593	1603#	1609	1620#	1626	1639#	1645	1663#	1669	1679#	1685	1696#	1702
	1715#	1721	1739#	1745	1755#	1761	1772#	1778	1791#	1797	1815#	1821	1831#
	1837	1848#	1854	1867#	1873	1901#	1912	1934#	1945	1966#	1979	1996#	2003
	2011#	2026	2050#	2059	2075#	2083	2095#	2103	2115#	2123	2141#	2148	2164#
	2172	2184#	2192	2204#	2212	2230#	2238	2254#	2262	2274#	2282	2294#	2302
	2319#	2326	2342#	2350	2362#	2370	2382#	2390	2410#	2418	2443#	2449	2461#
	2467	2470#	2476	2487#	2493	2510#	2518	2544#	2550	2568#	2574	2591#	2597
	2614#	2621	2651#	2657	2673#	2679	2697#	2703	2728#	2733	2749#	2755	2769#
	2774	2812#	2817	2829#	2835	2870#	2875	2886#	2892	2906#	2912	2926#	2940
	2946#	2958	296#	2989	2994#	3001	3011#	3017	3066#	3077	3084#	3097	3105#
	3115	3133#	3140	3148#	3155	3174#	3180	3183#	3189	3199#	3281	3386#	3405
	3428#	3434	3521#	3540	3559#	3566	3595#	3614	3663#	3723	3741#	3748	3767#
	3824	3846#	3852	3922#	3924	3932#	3938	3974#	3980	4011#	4028	4032#	4039
	4063#	4069	4094#	4099	4102#	4108	4150#	4249	4257#	4277	4306#	4344	4367#
	4400	4405#	4471										
\$NSK1 000130	1217#	1223	2928#	2933	2938	2971#	2987	3223#	3244	3248#	3252	3273	3390#
	3396	3403	3525#	3532	3538	3599#	3606	3612	3712#	3721	3812#	3821	4015#
\$NSK2 000110	4023	4184#	4210	4225	4233#	4238	4279#	4291	4319#	4338	4414#	4471	
	2974#	2985	3227#	3233	3241	3256#	3262	3271	4187#	4192	4197	4201#	4207
	4327#	4335	4416#	4443	4454								
\$NSK3 000110	4419#	4424	4433										
\$NULL 001154	1003#	4610	4639										
\$AWTST 000001	1186#	1188	1250#	1252	1336#	1413#	1495#	1497	1566#	1568	1652#	1728#	1804#
	1887#	1920#	1953#	1987#	2040#	2042	2131#	2133	2220#	2222	2310#	2312	2398#
	2400	2500#	2502	2606#	2686#	2688	2778#	2780	2839#	2895#	2897	3027#	3164#
	3166	3333#	3335	3458#	3460	3635#	3637	3734#	3835#	3837	4054#	4056	4120#
\$OCNT 015316	5052*	5081*	5094#										
\$OMODE 015320	5047*	5051*	5056	5059*	5070*	5096#							
\$OVER 014632	4906	4922	4930	4940	4949#								
\$PASS 001202	1022#	1127*	4489*	4490*	4498	4511	4936	4953					

\$OCNT 015316
 \$OMODE 015320
 \$OVER 014632
 \$PASS 001202

\$PASTM	001006	967#												
\$POWER	012654	4550	4557#											
\$PWRAD	012642	4552#												
\$PWRDN	012502	1103	4519#	4547										
\$PWRMG	012636	4550#												
\$PWRUP	012554	4529	4535#											
\$QUES	001170	1010#	4639	4707	4756	4772	4889							
\$RDCHR	013430	4720#	5137											
\$RDDEC=	***** U	5139												
\$RDLIN	013550	4748#	5138											
\$RDOCT=	***** U	5139												
\$RDSZ -	000010	4741#												
\$RTNAD	012460	4510#												
\$R2A =	***** U	5139												
\$SAVLE=	177777	1#	2989#	2990#	3202#	3206#	3244#	3245#	3666#	3670#	3770#	3774#	4225#	4226#
		4282#	4286#	4322#	4326#	4330#	4334#							
\$SAVRE-	***** U	5139												
\$SAVR6	012652	4528*	4536	4537*	4538*	4556#								
\$SCOPE	014370	1097	4903#											
\$SETUP=	000137	1088#	1096	1097	1099	1101	1103	1105	1106	1107	1109	1136	1139	4487
		4644	4778	4850	4876	4884	4904							
\$SSKO =	000236	2989#	2990	3202#	3206	3244#	3245	3666#	3670	3770#	3774	4225#	4226	4282#
		4286	4322#	4326	4330#	4334								
\$STUP =	177777	1088#												
\$SVLAD	014576	4914	4943#											
\$SVPC -	000204	941#	946											
\$SWR	167400	1#	689	693	694	695	696	697	698	699	1007	1008	1009	1106
		1107	1109	1110	1193	1256	1340	1417	1504	1571	1656	1732	1808	1891
		1924	1957	1991	2046	2137	2226	2315	2406	2506	2610	2692	2784	2843
		2901	3031	3171	3342	3465	3640	3738	3840	4060	4124	4482	4488	4505
		4509	4511	4553	4841	4842	4843	4844	4845	4854	4861	4873	4877	4889
		4895	4896	4897	4898	4899	4905	4917	4919	4920	4923	4924	4925	4932
		4933	4934	4946	4949	4952								
\$SWREG	001216	1030#	1130											
\$SWRMK	000000	699	700	4899	4900	4921								
\$TAGLF	177777	1#	1208#	1219#	1223#	1231#	1262#	1266#	1274#	1278#	1290#	1294#	1307#	1311#
		1326#	1330#	1350#	1354#	1366#	1370#	1383#	1387#	1402#	1406#	1432#	1436#	1448#
		1452#	1465#	1469#	1484#	1488#	1509#	1513#	1523#	1527#	1539#	1543#	1555#	1559#
		1576#	1580#	1589#	1593#	1605#	1609#	1622#	1626#	1641#	1645#	1665#	1669#	1681#
		1685#	1698#	1702#	1717#	1721#	1741#	1745#	1757#	1761#	1774#	1778#	1793#	1797#
		1817#	1821#	1833#	1837#	1850#	1854#	1869#	1873#	1903#	1912#	1936#	1945#	1968#
		1979#	1998#	2003#	2013#	2026#	2052#	2059#	2077#	2083#	2097#	2103#	2117#	2123#
		2143#	2148#	2166#	2172#	2186#	2192#	2206#	2212#	2232#	2238#	2256#	2262#	2276#
		2282#	2296#	2302#	2321#	2326#	2344#	2350#	2364#	2370#	2384#	2390#	2412#	2418#
		2445#	2449#	2463#	2467#	2472#	2476#	2489#	2493#	2512#	2518#	2546#	2550#	2570#
		2574#	2593#	2597#	2616#	2621#	2653#	2657#	2675#	2679#	2699#	2703#	2729#	2733#
		2751#	2755#	2770#	2774#	2813#	2817#	2831#	2835#	2871#	2875#	2888#	2892#	2908#
		2912#	2927#	2930#	2934#	2935#	2938#	2940#	2948#	2958#	2967#	2969#	2973#	2976#
		2985#	2987#	2989#	2996#	3001#	3013#	3017#	3068#	3077#	3086#	3097#	3107#	3115#
		3135#	3145#	3150#	3155#	3176#	3180#	3185#	3189#	3201#	3203#	3206#	3224#	3226#
		3229#	3234#	3235#	3241#	3244#	3250#	3253#	3254#	3260#	3263#	3264#	3271#	3273#
		3281#	3282#	3388#	3392#	3397#	3398#	3403#	3405#	3430#	3434#	3523#	3527#	3533#
		3534#	3538#	3540#	3561#	3566#	3597#	3601#	3607#	3608#	3612#	3614#	3665#	3667#
		3670#	3714#	3721#	3723#	3724#	3743#	3748#	3769#	3771#	3774#	3814#	3821#	3824#
		3825#	3848#	3852#	3923#	3924#	3934#	3938#	3976#	3980#	4013#	4017#	4023#	4028#
		4034#	4039#	4065#	4069#	4095#	4099#	4104#	4108#	4185#	4189#	4193#	4194#	4197#

\$TAGNU= 000251

4203#	4207#	4210	4225#	4235#	4238#	4281#	4283#	4286#	4291#	4292#	4321#	4323#
4326#	4329#	4331#	4334#	4335#	4336#	4338#	4339#	4415#	4418#	4421#	4425#	4426#
4433#	4444#	4445#	4454#	4456#								
1#	1207	1208#	1218	1219#	1261	1262#	1273	1274#	1289	1290#	1306	1307#
1325	1326#	1349	1350#	1365	1366#	1382	1383#	1401	1402#	1431	1432#	1447
1448#	1464	1465#	1483	1484#	1508	1509#	1522	1523#	1538	1539#	1554	1555#
1575	1576#	1588	1589#	1604	1605#	1621	1622#	1640	1641#	1664	1665#	1680
1681#	1697	1698#	1716	1717#	1740	1741#	1756	1757#	1773	1774#	1792	1793#
1816	1817#	1832	1833#	1849	1850#	1868	1869#	1902	1903#	1935	1936#	1967
1968#	1997	1998#	2012	2013#	2051	2052#	2076	2077#	2096	2097#	2116	2117#
2142	2143#	2165	2166#	2185	2186#	2205	2206#	2231	2232#	2255	2256#	2275
2276#	2295	2296#	2320	2321#	2343	2344#	2363	2364#	2383	2384#	2411	2412#
2444	2445#	2462	2463#	2471	2472#	2488	2489#	2511	2512#	2545	2546#	2569
2570#	2592	2593#	2615	2616#	2652	2653#	2674	2675#	2698	2699#	2728	2729#
2750	2751#	2769	2770#	2812	2813#	2830	2831#	2870	2871#	2887	2888#	2907
2908#	2926	2927#	2929	2930#	2933	2935#	2941	2944	2945#	2947	2948#	2966
2967#	2968	2969#	2972	2973#	2975	2976#	2995	2996#	3012	3013#	3067	3068#
3085	3086#	3106	3107#	3134	3135#	3149	3150#	3175	3176#	3184	3185#	3200
3201#	3202#	3205	3206#	3223	3224#	3225	3226#	3228	3229#	3233	3235#	3249
3250#	3252	3254#	3257	3259	3260#	3262	3264#	3387	3388#	3391	3392#	3396
3398#	3429	3430#	3522	3523#	3526	3527#	3532	3534#	3560	3561#	3596	3597#
3600	3601#	3606	3608#	3664	3665#	3666#	3669	3670#	3713	3714#	3742	3743#
3768	3769#	3770#	3773	3774#	3813	3814#	3847	3848#	3922	3923#	3925	3928
3929#	3933	3934#	3975	3976#	4012	4013#	4016	4017#	4033	4034#	4064	4065#
4094	4095#	4103	4104#	4150#	4184	4185#	4188	4189#	4192	4194#	4202	4203#
4234	4235#	4257#	4280	4281#	4282#	4285	4286#	4306#	4320	4321#	4322#	4325
4326#	4328	4329#	4330#	4333	4334#	4367#	4405#	4414	4415#	4417	4418#	4420
4421#	4424	4426#	4443	4445#								

\$TEMP - 000300

1154#	1155#	1161#	1162#	1164#	1165#	1166#	1167#	1169#	1170#	1171#	1173#	1174#
1176#	1177#	1179#	1180#	1182#	1183#	1184#	1196#	1197#	1199#	1200#	1201#	1202#
1203#	1204#	1205#	1206#	1209#	1210#	1212#	1213#	1223#	1226#	1227#	1228#	1230#
1231#	1232	1236#	1237#	1239#	1240#	1241#	1263#	1264#	1266#	1269#	1270#	1278#
1283#	1284#	1285#	1286#	1294#	1299#	1300#	1302#	1303#	1311#	1316#	1317#	1319#
1320#	1330#	1345#	1346#	1354#	1359#	1360#	1361#	1362#	1370#	1375#	1376#	1378#
1379#	1387#	1392#	1393#	1395#	1396#	1406#	1427#	1428#	1436#	1441#	1442#	1443#
1444#	1452#	1457#	1458#	1460#	1461#	1469#	1474#	1475#	1477#	1478#	1488#	1510#
1511#	1513#	1516#	1517#	1518#	1519#	1527#	1532#	1533#	1535#	1536#	1543#	1548#
1549#	1550#	1551#	1559#	1577#	1578#	1580#	1584#	1585#	1593#	1598#	1599#	1600#
1601#	1609#	1614#	1615#	1617#	1618#	1626#	1631#	1632#	1634#	1635#	1645#	1660#
1661#	1669#	1674#	1675#	1676#	1677#	1685#	1690#	1691#	1693#	1694#	1702#	1706#
1707#	1710#	1711#	1721#	1736#	1737#	1745#	1750#	1751#	1752#	1753#	1761#	1766#
1767#	1769#	1770#	1778#	1783#	1784#	1786#	1787#	1797#	1812#	1813#	1821#	1826#
1827#	1828#	1829#	1837#	1842#	1843#	1845#	1846#	1854#	1859#	1860#	1862#	1863#
1873#	1899#	1900#	1912#	1931#	1932#	1945#	1964#	1965#	1979#	2000#	2001#	2003#
2008#	2009#	2026#	2056#	2057#	2059#	2068#	2069#	2072#	2073#	2083#	2088#	2089#
2092#	2093#	2103#	2108#	2109#	2112#	2113#	2123#	2145#	2146#	2148#	2157#	2158#
2161#	2162#	2172#	2177#	2178#	2181#	2182#	2192#	2197#	2198#	2201#	2202#	2212#
2235#	2236#	2238#	2247#	2248#	2251#	2252#	2262#	2267#	2268#	2271#	2272#	2282#
2287#	2288#	2291#	2292#	2302#	2323#	2324#	2326#	2335#	2336#	2339#	2340#	2350#
2355#	2356#	2359#	2360#	2370#	2375#	2376#	2379#	2380#	2390#	2415#	2416#	2418#
2428#	2429#	2431#	2432#	2440#	2441#	2449#	2455#	2456#	2459#	2460#	2467#	2476#
2481#	2482#	2485#	2486#	2493#	2515#	2516#	2518#	2528#	2529#	2532#	2533#	2541#
2542#	2550#	2555#	2556#	2560#	2561#	2574#	2579#	2580#	2583#	2584#	2597#	2618#
2619#	2621#	2633#	2634#	2635#	2636#	2639#	2640#	2643#	2644#	2657#	2662#	2663#
2665#	2666#	2679#	2700#	2701#	2703#	2705#	2706#	2713#	2714#	2733#	2737#	2738#
2744#	2745#	2755#	2774#	2789#	2790#	2792#	2793#	2796#	2797#	2817#	2822#	2823#

\$TPFLG	001157	1006#	4577	4639										
\$TPS	001150	1001#	4626	4639										
\$TRAP	015322	1101	5105#											
\$TRAP2	015344	5116#	5127											
\$TRP =	000012	5120#	5129#	5130#	5131#	5132#	5133#	5134	5135#	5136	5137#	5138#	5139#	
\$TRPAD	015356	5110	5127#											
\$TSKO -	000244	1208#	1231	1262#	1266	1274#	1278	1290#	1294	1307#	1311	1326#	1330	1350#
		1354	1366#	1370	1383#	1387	1402#	1406	1432#	1436	1448#	1452	1465#	1469
		1484#	1488	1509#	1513	1523#	1527	1539#	1543	1555#	1559	1576#	1580	1589#
		1593	1605#	1609	1622#	1626	1641#	1645	1665#	1669	1681#	1685	1698#	1702
		1717#	1721	1741#	1745	1757#	1761	1774#	1778	1793#	1797	1817#	1821	1833#
		1837	1850#	1854	1869#	1873	1903#	1912	1936#	1945	1968#	1979	1998#	2003
		2013#	2026	2052#	2059	2077#	2083	2097#	2103	2117#	2123	2143#	2148	2166#
		2172	2186#	2192	2206#	2212	2232#	2238	2256#	2262	2276#	2282	2296#	2302
		2321#	2326	2344#	2350	2364#	2370	2384#	2390	2412#	2418	2445#	2449	2463#
		2467	2472#	2476	2489#	2493	2512#	2518	2546#	2550	2570#	2574	2593#	2597
		2616#	2621	2653#	2657	2675#	2679	2699#	2703	2729#	2733	2751#	2755	2770#
		2774	2813#	2817	2831#	2835	2871#	2875	2888#	2892	2908#	2912	2927#	2940
		2948#	2958	2967#	2989	2996#	3001	3013#	3017	3068#	3077	3086#	3097	3107#
		3115	3135#	3145	3150#	3155	3176#	3180	3185#	3189	3201#	3203	3206#	3282
		3388#	3405	3430#	3434	3523#	3540	3561#	3566	3597#	3614	3665#	3667	3670#
		3724	3743#	3748	3769#	3771	3774#	3825	3848#	3852	3923#	3924	3934#	3938
		3976#	3980	4013#	4028	4034#	4039	4065#	4069	4095#	4099	4104#	4108	4185#
\$TSK1 -	000250	4225	4235#	4238	4281#	4283	4286#	4292	4321#	4323	4326#	4339	4415#	4456
		1219#	1223	2930#	2934	2935#	2938	2969#	2989	3206#	3281	3392#	3397	3398#
		3403	3527#	3533	3534#	3538	3601#	3607	3608#	3612	3670#	3723	3774#	3824
		4017#	4023	4185#	4210	4225	4286#	4291	4326#	4338	4418#	4444	4445#	4454
\$TSK2 -	000247	2973#	2987	3224#	3244	3250#	3253	3254#	3273	3714#	3721	3814#	3821	4189#
		4193	4194#	4197	4203#	4207	4329#	4331	4334#	4336	4421#	4425	4426#	4433
\$TSK3 =	000236	2976#	2985	3226#	3244	3260#	3263	3264#	3271	4334#	4335			
\$TSK4 -	000147	3229#	3234	3235#	3241									
\$TSTM	001004	966#												
\$TSTM	001102	979#	4487*	4853	4889	4894	4921	4943*	4944	4949	4953			
\$TTYIN	013656	4749	4750	4767	4771#									
\$TYPBN=	***** U	5133												
\$TYPDS	014650	4965#	5132											
\$TYPE	012664	4577#	4809	5120	5128									
\$TYPEC	013076	4607	4614	4621	4626#	4627	4693							
\$TYPEX	013144	4632	4634	4637#										
\$TYPOC	015120	5050#	5129											
\$TYPON	015134	5049	5052#	5131										
\$TYPOS	015074	5045#	5130											
\$UNIT	001206	1024#												
\$UNITM	001010	968#												
\$USWR	001220	1031#	1260	1507	1574	1996	2050	2141	2230	2319	2410	2510	2614	3174
		3284	3559	3741	4063	4273								
\$VECT1	001244	1056#	4441											
\$VECT2	001246	1057#												
\$XTSTR	014402	4908#												
\$YESNO	000001	1172#	1173#	1175#	1176#	1178#	1179#	1181#	1182#	1226#	1227#	1229#	1230#	1238#
		1239#	1285#	1286#	1302#	1303#	1319#	1320#	1361#	1362#	1378#	1379#	1395#	1396#
		1443#	1444#	1460#	1461#	1477#	1478#	1518#	1519#	1535#	1536#	1550#	1551#	1600#
		1601#	1617#	1618#	1634#	1635#	1676#	1677#	1693#	1694#	1710#	1711#	1752#	1753#
		1769#	1770#	1786#	1787#	1828#	1829#	1845#	1846#	1862#	1863#	2072#	2073#	2092#
		2093#	2112#	2113#	2161#	2162#	2181#	2182#	2201#	2202#	2251#	2252#	2271#	2272#
		2291#	2292#	2339#	2340#	2359#	2360#	2379#	2380#	2431#	2432#	2459#	2460#	2485#

SSARGC= 000000
SSBYTE= 000403

SSDST = 000067
SSFLAG= 000001

SSFROM= 000000

SSGET4= 000000
SSLLOC = 012332

2486#	2532#	2533#	2560#	2561#	2583#	2584#	2633#	2634#	2643#	2644#	2665#	2666#
2789#	2790#	2848#	2849#	2914#	2915#	2936#	2937#	3196#	3197#	3202#	3203#	3237#
3238#	3239#	3240#	3285#	3288#	3353#	3354#	3365#	3366#	3375#	3376#	3415#	3416#
3444#	3445#	3482#	3483#	3485#	3486#	3509#	3510#	3543#	3544#	3546#	3547#	3572#
3573#	3580#	3581#	3583#	3584#	3585#	3586#	3616#	3617#	3618#	3619#	3628#	3629#
3644#	3645#	3666#	3667#	3706#	3707#	3708#	3709#	3728#	3729#	3751#	3752#	3770#
3771#	3806#	3807#	3808#	3809#	3910#	3911#	3914#	3915#	3917#	3918#	3941#	3942#
3943#	3944#	3964#	3965#	3968#	3969#	3978#	3979#	3999#	4000#	4003#	4004#	4007#
4008#	4026#	4027#	4036#	4037#	4072#	4073#	4081#	4082#	4223#	4224#	4274#	4277#
4282#	4283#	4287#	4288#	4289#	4290#	4294#	4295#	4322#	4323#	4330#	4331#	4360#
4361#	4448#	4449#	4450#	4451#	4452#	4453#	4462#	4463#				
4150#	4257#	4306#	4367#	4405#								
1217#	1260#	1272#	1288#	1305#	1324#	1348#	1364#	1381#	1400#	1430#	1446#	1463#
1482#	1507#	1521#	1537#	1553#	1574#	1587#	1603#	1620#	1639#	1663#	1679#	1696#
1715#	1739#	1755#	1772#	1791#	1815#	1831#	1848#	1867#	1901#	1934#	1966#	1996#
2011#	2050#	2075#	2095#	2115#	2141#	2164#	2184#	2204#	2230#	2254#	2274#	2294#
2319#	2342#	2362#	2382#	2410#	2443#	2461#	2470#	2487#	2510#	2544#	2568#	2591#
2614#	2651#	2673#	2697#	2749#	2829#	2886#	2906#	2928#	2946#	2967#	2971#	2974#
2994#	3011#	3066#	3084#	3105#	3133#	3148#	3174#	3183#	3224#	3227#	3248#	3256#
3258#	3386#	3390#	3428#	3521#	3525#	3559#	3595#	3599#	3712#	3741#	3812#	3846#
3932#	3974#	4011#	4015#	4032#	4063#	4102#	4187#	4201#	4233#	4416#	4419#	
3285#	4274#											
1217#	1219	1223#	1260#	1262	1266#	1272#	1274	1278#	1288#	1290	1294#	1305#
1307	1311#	1324#	1326	1330#	1348#	1350	1354#	1364#	1366	1370#	1381#	1383
1387#	1400#	1402	1406#	1430#	1432	1436#	1446#	1448	1452#	1463#	1465	1469#
1482#	1484	1488#	1507#	1509	1513#	1521#	1523	1527#	1537#	1539	1543#	1553#
1555	1559#	1574#	1576	1580#	1587#	1589	1593#	1603#	1605	1609#	1620#	1622
1626#	1639#	1641	1645#	1663#	1665	1669#	1679#	1681	1685#	1696#	1698	1702#
1715#	1717	1721#	1739#	1741	1745#	1755#	1757	1761#	1772#	1774	1778#	1791#
1793	1797#	1815#	1817	1821#	1831#	1833	1837#	1848#	1850	1854#	1867#	1869
1873#	1901#	1903	1912#	1934#	1936	1945#	1966#	1968	1979#	1996#	1998	2003#
2011#	2013	2026#	2050#	2052	2059#	2075#	2077	2083#	2095#	2097	2103#	2115#
2117	2123#	2141#	2143	2148#	2164#	2166	2172#	2184#	2186	2192#	2204#	2206
2212#	2230#	2232	2238#	2254#	2256	2262#	2274#	2276	2282#	2294#	2296	2302#
2319#	2321	2326#	2342#	2344	2350#	2362#	2364	2370#	2382#	2384	2390#	2410#
2412	2418#	2443#	2445	2449#	2461#	2463	2467#	2470#	2472	2476#	2487#	2489
2493#	2510#	2512	2518#	2544#	2546	2550#	2568#	2570	2574#	2591#	2593	2597#
2614#	2616	2621#	2651#	2653	2657#	2673#	2675	2679#	2697#	2699	2703#	2728#
2733#	2749#	2751	2755#	2769#	2774#	2812#	2817#	2829#	2831	2835#	2870#	2875#
2886#	2888	2892#	2906#	2908	2912#	2928#	2930	2938#	2946#	2948	2958#	2966#
2967#	2969	2971#	2973	2974#	2976	2985#	2987#	2994#	2996	3001#	3011#	3013
3017#	3066#	3068	3077#	3084#	3086	3097#	3105#	3107	3115#	3133#	3135	3145#
3148#	3150	3155#	3174#	3176	3180#	3183#	3185	3189#	3223#	3224#	3226	3227#
3229	3241#	3248#	3250	3256#	3258#	3260	3271#	3273#	3386#	3388	3390#	3392
3403#	3405#	3428#	3430	3434#	3521#	3523	3525#	3527	3538#	3540#	3559#	3561
3566#	3595#	3597	3599#	3601	3612#	3614#	3712#	3714	3721#	3741#	3743	3748#
3812#	3814	3821#	3846#	3848	3852#	3932#	3934	3938#	3974#	3976	3980#	4011#
4013	4015#	4017	4023#	4028#	4032#	4034	4039#	4063#	4065	4069#	4094#	4099#
4107#	4104	4108#	4187#	4189	4197#	4201#	4203	4207#	4233#	4235	4238#	4416#
4418	4419#	4421	4433#	4454#								
1157#	2434#	2535#	2563#	2586#	2646#	2668#	2718#	2761#	2802#	2861#	3045#	3055#
3127#	3379#	3423#	3500#	3512#	3588#	3654#	3675#	3688#	3760#	3779#	3791#	3855#
4086#	4218#	4427#										
4503#												
1218#	1219	1232#	1233	1261#	1262	1273#	1274	1289#	1290	1306#	1307	1325#
1326	1349#	1350	1365#	1366	1382#	1383	1401#	1402	1431#	1432	1447#	1448

\$127	006366	2975	2985#	
\$13	002552	1401	1406#	
\$130	006402	2995	3001#	
\$131	006420	3012	3017#	
\$132	006534	3067	3077#	
\$133	006562	3085	3097#	
\$134	006604	3106	3115#	
\$135	006654	3134	3145#	
\$136	006666	3149	3155#	
\$137	006726	3175	3180#	
\$14	002622	1431	1436#	
\$140	006746	3184	3189#	
\$141	006776	3200	3203#	
\$142	006774	3201#	3281	
\$143	007144	3205	3282#	
\$144	007034	3223#	3244	
\$145	007070	3225	3245#	
\$146	007056	3228	3234#	
\$147	007066	3233	3241#	
\$15	002650	1447	1452#	
\$150	007102	3249	3253#	
\$151	007126	3252	3273#	
\$152	007124	3257	3259	3263#
\$153	007126	3262	3271#	
\$154	007416	3387	3405#	
\$155	007414	3391	3397#	
\$156	007416	3396	3403#	
\$157	007474	3429	3434#	
\$16	002676	1464	1469#	
\$160	007730	3522	3540#	
\$161	007726	3526	3533#	
\$162	007730	3532	3538#	
\$163	007772	3560	3566#	
\$164	010100	3596	3614#	
\$165	010076	3600	3607#	
\$166	010100	3606	3612#	
\$167	010230	3664	3667#	
\$17	002726	1483	1488#	
\$170	010226	3665#	3723	
\$171	010354	3669	3724#	
\$172	010352	3713	3721#	
\$173	010420	3742	3748#	
\$174	010464	3768	3771#	
\$175	010462	3769#	3824	
\$176	010610	3773	3825#	
\$177	010606	3813	3821#	
\$2	002152	1218	1223#	
\$20	002764	1508	1513#	
\$200	010646	3847	3852#	
\$201	010760	3922#	3927	
\$202	010774	3925	3928#	
\$203	011004	3933	3938#	
\$204	011064	3975	3980#	
\$205	011144	4012	4028#	
\$206	011140	4016	4023#	
\$207	011160	4033	4039#	

\$21	003012	1522	1527#	
\$210	011224	4064	4069#	
\$211	011306	4094	4099#	
\$212	011316	4103	4108#	
\$213	011642	4236	4249#	
\$214	011644	4241	4251#	
\$215	011512	4184#	4225	
\$216	011616	4211	4213	4226#
\$217	011532	4188	4193#	
\$22	003040	1538	1543#	
\$220	011540	4192	4197#	
\$221	011556	4202	4207#	
\$222	011630	4234	4238#	
\$223	011744	4296	4299#	
\$224	011744	4300#		
\$225	011710	4280	4283#	
\$226	011704	4281#	4291	
\$227	011734	4285	4292#	
\$23	003066	1554	1559#	
\$230	012024	4344#		
\$231	012024	4345#		
\$232	011772	4320	4323#	
\$233	011766	4321#	4338	
\$234	012016	4325	4339#	
\$235	012004	4328	4331#	
\$236	012002	4329#	4335	
\$237	012014	4333	4336#	
\$24	003124	1575	1580#	
\$240	012216	4400#		
\$241	012216	4401#		
\$242	012356	4464	4471#	
\$243	012356	4472#		
\$244	012220	4414#	4457	
\$245	012304	4417	4444#	
\$246	012244	4420	4425#	
\$247	012252	4424	4433#	
\$25	003144	1588	1593#	
\$250	012324	4443	4454#	
\$26	003172	1604	1609#	
\$27	003220	1621	1626#	
\$3	002264	1261	1266#	
\$30	003250	1640	1645#	
\$31	003306	1664	1669#	
\$32	003334	1680	1685#	
\$33	003362	1697	1702#	
\$34	003412	1716	1721#	
\$35	003450	1740	1745#	
\$36	003476	1756	1761#	
\$37	003524	1773	1778#	
\$4	002304	1273	1278#	
\$40	003554	1792	1797#	
\$40CAT	***** U	4863	4905	
\$41	003612	1816	1821#	
\$42	003640	1832	1837#	
\$43	003666	1849	1854#	
\$44	003716	1868	1873#	

. ABS. 015402 000

ERRORS DETECTED: 0

CVDV.BIN,CVDV.LST/CRF:SYM/SOL/NL:TGC=CVDVAC.MAC,CVDVAC.SML,CVDVAC.P11
RUN-TIME: 107.116.5 SECONDS
RUN-TIME RATIO: 853/229.3.7
CORE USED: 43K 85 PAGES