

DLV11-E

OFF LINE TEST
CNDVAAO

AH-T436A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 82-83
MADE IN USA



Grid of test data tables (fiches) with various columns and rows of numerical and text data.



8213
8214
8215
8216
8217
8218
8219
8220
8221
8222
8223
8224
8225
8226
8227
8228
8229
8230
8231
8232
8233
8234
8235
8236
8237
8238
8239
8240
8241
8242
8243
8244
8245

.REM @

IDENTIFICATION

PRODUCT CODE: AC-T435A-MC
PRODUCT NAME: CNDVAAO DLV11-E OFFLINE TEST
PRODUCT DATE: DECEMBER, 1982
AUTHOR: ODES CHOATE
MAINTAINER: DIAGNOSTIC SERVICES/ISS

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) 1982,1983 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

TABLE OF CONTENTS

8247		
8248		
8249		
8250		
8251		
8252	1.0	GENERAL PROGRAM INFORMATION.
8253		
8254	1.1	PROGRAM PURPOSE (ABSTRACT).
8255	1.2	SYSTEM REQUIREMENTS.
8256	1.3	RELATED DOCUMENTS AND STANDARDS.
8257	1.4	DIAGNOSTIC HIERARCHY PREREQUISITES.
8258	1.5	ASSUMPTIONS.
8259		
8260		
8261	2.0	OPERATING INSTRUCTIONS.
8262		
8263	2.1	LOADING AND STARTING PROCEDURES.
8264	2.2	SPECIAL ENVIRONMENTS.
8265	2.3	OPERATIONAL SWITCH SETTINGS
8266	2.4	PROGRAM OPTIONS.
8267	2.5	EXECUTION TIMES.
8268		
8269		
8270	3.0	ERROR INFORMATION.
8271		
8272	3.1	ERROR REPORTING PROCEDURE.
8273	3.2	ERROR HALTS.
8274		
8275		
8276	4.0	PERFORMANCE AND PROGRESS REPORTS.
8277		
8278	4.1	PERFORMANCE REPORTS.
8279		
8280		
8281	5.0	DEVICE INFORMATION TABLES.
8282		
8283		
8284	6.0	SUMMARY OF TESTS AND SPECIAL SUBROUTINES
8285		NOTE: CVDVAC DIAGNOSTIC WAS MODIFIED TO RUN ON 11/21 PROCESSOR
8286		BY LOWERING PRIORITY 7 TO 6 AND ALSO ADDING A MACRO CALL
8287		TO CNMAC2.SML TO INIT BRKVEC AND LKVEC AND WAS RENAMED
8288		TO CNDVAA.
8289		-----

8291
8292
8293
8294
8295
8296
8297
8298
8299
8300
8301
8302
8303
8304
8305
8306
8307
8308
8309
8310
8311
8312
8313
8314
8315
8316
8317
8318
8319
8320
8321
8322
8323
8324
8325
8326
8327
8328
8329
8330
8331
8332
8333
8334
8335
8336
8337
8338

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-X 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)

8340
8341 1.3 RELATED DOCUMENTS AND STANDARDS.
8342
8343 DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS 175-003-009-02
8344 APT MD-11-DZZMA
8345 ACT AUTOCAT-11-QZAUB
8346 SYSMAC MD-11-DZQAC
8347
8348
8349 1.4 DIAGNOSTIC HIERARCHY PREREQUISITES.
8350 NO SPECIAL DIAGNOSTICS ARE REQUIRED TO RUN BEFORE THIS, BUT
8351 THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY
8352 OPERATIONAL.
8353
8354 1.5 ASSUMPTIONS.
8355
8356 THIS DIAGNOSTIC ASSUMES THAT THE OPERATOR HAS INITIALIZED
8357 LOCATION '\$USWR' AND '\$DEVM' TO THE PROPER VALUES.
8358 THE (H) JUMPER MUST BE REMOVED FROM ALL DLV1!-E'S UNDER TEST.
8359
8360 2.0 OPERATING INSTRUCTIONS.
8361 -----
8362
8363 2.1 LOADING AND STARTING PROCEDURES.
8364
8365 USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED
8366 MEDIA.
8367
8368 THIS DIAGNOSTIC HAS ONLY ONE (1) STARTING ADDRESS. 200 FOR
8369 START AND RESTART.
8370
8371 THE USER CAN SELECT A SPECIFIC TEST TO BE EXECUTED BY SETTING
8372 SWITCH 8 IN THE SWITCH REGISTER AND THE TEST NUMBER (IN OCTAL)
8373 IN THE LOWER BYTE. (NOTE: ALL TESTS PREVIOUS TO THE SELECTED
8374 ONE ARE EXECUTED WITHOUT ITERATIONS.)
8375
8376 2.2 SPECIAL ENVIRONMENTS.
8377
8378 THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING
8379 UDER APT,ACT,XXDP MONITORS, AS DESCRIBED IN THEIR RESPECTIVE
8380 PROCEDURES MANUAL AND SYSMAC PACKAGE.
8381
8382
8383
8384
8385
8386

8388
8389
8390
8391
8392
8393
8394
8395
8396
8397
8398
8399
8400
8401
8402
8403
8404
8405
8406
8407
8408
8409
8410
8411
8412
8413
8414
8415
8416
8417
8418
8419
8420
8421
8422
8423
8424
8425
8426
8427
8428
8429
8430
8431
8432
8433
8434
8435
8436
8437
8438
8439
8440
8441
8442

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)

8444
8445
8446
8447
8448
8449
8450
8451
8452
8453
8454
8455
8456
8457
8458
8459
8460
8461
8462
8463
8464
8465
8466
8467
8468
8469
8470
8471
8472
8473
8474
8475
8476
8477
8478
8479
8480
8481
8482
8483
8484
8485
8486
8487
8488
8489
8490
8491
8492
8493

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!	!UNIT!	!UNIT!
! 15 !	! #1 !	! #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

8495
8496
8497
8498
8499
8500
8501
8502
8503
8504
8505
8506
8507
8508
8509
8510
8511
8512
8513
8514
8515
8516
8517
8518
8519
8520
8521
8522
8523
8524
8525
8526
8527
8528
8529
8530
8531
8532
8533
8534
8535
8536
8537
8538
8539
8540
8541
8542
8543
8544
8545
8546
8547
8548
8549
8550

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:

BIT POSITION	DEFINITION	DEFAULT VALUE
-----	-----	-----
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 PROGRAMABLE BAUD RATE OPTION IS ENABLED THE PROGRAMABLE 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

	TESTS NOT EXECUTED IF BIT =
-----	-----
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 1
	T22, T23, PARTS OF T34, 1
	T36
(-FR) & (-FD) JUMPERS IN (BIT 14 OF \$USWR)	T5, T6, T16, T17, T20, T22, 1
	T23

ALL OTHER TESTS WILL RUN REGARDLESS OF ANY BIT SETTINGS.

8551
8552
8553
8554
8555
8556
8557
8558
8559
8560
8561
8562
8563
8564
8565
8566

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.

8568
8569
8570
8571
8572
8573
8574
8575
8576
8577
8578
8579
8580
8581
8582
8583
8584
8585
8586
8587
8588
8589
8590
8591
8592
8593
8594
8595
8596
8597
8598
8599
8600
8601
8602
8603
8604
8605
8606
8607
8608
8609
8610
8611
8612
8613
8614
8615
8616
8617
8618
8619
8620
8621
8622
8623

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. CONTINUEING 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

8624
8625
8626
8627
8628

THIS IS TYPED AFTER THE DEVICE HAS COMPLETED ITS PASS.
AFTER ALL DEVICES HAVE BEEN EXERCISED AN END PASS STATEMENT IS
TYPED: 'ENDPASS#-----.'

8630
 8631
 8632
 8633
 8634
 8635
 8636
 8637
 8638
 8639
 8640
 8641
 8642
 8643
 8644
 8645
 8646
 8647
 8648
 8649
 8650
 8651
 8652
 8653
 8654
 8655
 8656
 8657
 8658

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									RECEIVED DATA BUFFER			
	R	ERR	ERR	ERR												
TCSR	PROGRAMMABLE BAUD				PBR				XMIT	XMIT				MAIN	BREA	
	RATE SELECT				ENAB				RDY	IE				T	K	
TBUF													TRANSMITTER DATA BUFFER			

NOTE

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

8660
8661
8662
8663
8664
8665
8666
8667
8668
8669
8670
8671
8672
8673
8674
8675
8676
8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715

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 XMITIE - 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 RCVRIE - RCSR6 SET, CLEAR, RESET

MAINDEC-ZZ-CNDVA-A MACY11 30(1046) 16-DEC-82 15:13 PAGE 150-1^{N 1}
CNDVAA.F11 16-DEC-82 15:12

SEQ 0013

8716
8717

---- --

8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737
8738
8739
8740
8741
8742
8743
8744
8745
8746
8747
8748
8749
8750
8751
8752
8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8766
8767
8768
8769
8770
8771
8772
8773
8774

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

8775
8776
8777

---- --

8779
8780
8781
8782
8783
8784
8785
8786
8787
8788
8789
8790
8791
8792
8793
8794
8795
8796
8797
8798
8799
8800
8801
8802
8803
8804
8805
8806
8807
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829
8830
8831

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

8833
8834
8835
8836
8837
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847
8848
8849
8850
8851
8852
8853
8854
8855
8856
8857
8858
8859
8860
8861
8862
8863
8864
8865
8866
8867
8868
8869
8870
8871
8872
8873
8874
8875
8876

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.

8878
8879
8880
8881
8882
8883
8884
8885
8886
8887
8888
8889
8890
8891
8892
8893
8894
8895
8896
8897
8898
8899
8900
8901
8902
8903
8904
8905
8906
8907
8908
8909
8910
8911
8912
8913
8914
8915
8916
8917
8918
8919
8920

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:WAI

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.

8922
8923
8924
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
8925
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
8926
8927
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

170000

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

```
.TITLE MAINDEC-ZZ-CNDVA-A  
.*COPYRIGHT (C) 1982  
.*DIGITAL EQUIPMENT CORP.  
.*MAYNARD, MASS. 01754  
.*  
.*PROGRAM BY ODES CHOATE  
.*  
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC  
.*PACKAGE (MAINDEC-11-DZQAC-C3), 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  
.****** THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED  
ODTST= 170000  
.*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
```



```

(1)          : *PRIORITY LEVEL DEFINITIONS
(1)          000000 PR0= 0          :: PRIORITY LEVEL 0
(1)          000040 PR1= 40         :: PRIORITY LEVEL 1
(1)          000100 PR2= 100        :: PRIORITY LEVEL 2
(1)          000140 PR3= 140        :: PRIORITY LEVEL 3
(1)          000200 PR4= 200        :: PRIORITY LEVEL 4
(1)          000240 PR5= 240        :: PRIORITY LEVEL 5
(1)          000300 PR6= 300        :: PRIORITY LEVEL 6
(1)          000340 PR7= 340        :: PRIORITY LEVEL 7
(1)          :
(1)          : *"SWITCH REGISTER" SWITCH DEFINITIONS
(1)          100000 SW15= 100000
(1)          040000 SW14= 40000
(1)          020000 SW13= 20000
(1)          010000 SW12= 10000
(1)          004000 SW11= 4000
(1)          002000 SW10= 2000
(1)          001000 SW09= 1000
(1)          000400 SW08= 400
(1)          000200 SW07= 200
(1)          000100 SW06= 100
(1)          000040 SW05= 40
(1)          000020 SW04= 20
(1)          000010 SW03= 10
(1)          000004 SW02= 4
(1)          000002 SW01= 2
(1)          000001 SW00= 1
(1)          .EQUIV SW09,SW9
(1)          .EQUIV SW08,SW8
(1)          .EQUIV SW07,SW7
(1)          .EQUIV SW06,SW6
(1)          .EQUIV SW05,SW5
(1)          .EQUIV SW04,SW4
(1)          .EQUIV SW03,SW3
(1)          .EQUIV SW02,SW2
(1)          .EQUIV SW01,SW1
(1)          .EQUIV SW00,SW0
(1)          :
(1)          : *DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)          100000 BIT15= 100000
(1)          040000 BIT14= 40000
(1)          020000 BIT13= 20000
(1)          010000 BIT12= 10000
(1)          004000 BIT11= 4000
(1)          002000 BIT10= 2000
(1)          001000 BIT09= 1000
(1)          000400 BIT08= 400
(1)          000200 BIT07= 200
(1)          000100 BIT06= 100
(1)          000040 BIT05= 40
(1)          000020 BIT04= 20
(1)          000010 BIT03= 10
(1)          000004 BIT02= 4
(1)          000002 BIT01= 2
(1)          000001 BIT00= 1
(1)          .EQUIV BIT09,BIT9
  
```

```

(1) .EQUIV BIT08,BIT8
(1) .EQUIV BIT07,BIT7
(1) .EQUIV BIT06,BIT6
(1) .EQUIV BIT05,BIT5
(1) .EQUIV BIT04,BIT4
(1) .EQUIV BIT03,BIT3
(1) .EQUIV BIT02,BIT2
(1) .EQUIV BIT01,BIT1
(1) .EQUIV BIT00,BIT0

;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1) 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(1) 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
(1) 000014 TBITVEC=14 ;: "T" BIT
(1) 000014 TRTVEC= 14 ;:TRACE TRAP
(1) 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
(1) 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1) 000024 PWRVEC= 24 ;:POWER FAIL
(1) 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
(1) 000034 TRAPVEC=34 ;: "TRAP" TRAP
(1) 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
(1) 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
(1) ;***** THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(1) 000100 LKVEC= 100 ;:LINE CLOCK VECTOR
(1) 000140 BRKVEC= 140 ;:BREAK VECTOR
(1) 000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR

8928 000004 ILLMEM= 4
8929 000001 ADRS= R1
8930 000002 GOOD= R2
8931 000003 BAD= R3
8932 000001 REGISTER=R1
8933 000002 BIT= R2
8934 000003 FUNCT= R3
8935 000002 LEAD= R2
8936 000004 FOLLOW= R4
8937 175610 DLADDR= 175610

8938
8939 ; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
8940 177777 SET= -1
8941 000000 CLR= 0
8942
8943
8944 ;:*****
8945 ; RCSR REGISTER BIT NAMES
8946 ;:*****
8947 100000 DATAINT= BIT15 ; DATASET INTERRUPT
8948 040000 RING= BIT14 ; RINGING SIGNAL INDICATOR
8949 020000 CLRSEND= BIT13 ; CLEAR TO SEND FROM DATASET
8950 010000 CARDET= BIT12 ; CARRIER DETECT
8951 004000 RCVRACT= BIT11 ; RECEIVER ACTIVE INDICATOR
8952 002000 SECREC= BIT10 ; SECONDARY RECEIVE
8953 ; UNUSED BIT09
8954 ; UNUSED BIT08
8955 000200 RCVRDONE= BIT07 ; RECEIVER DONE
8956 000100 RCVRIE= BIT06 ; RECEIVER INTERRUPT ENABLE
8957 000040 DATAIE= BIT05 ; DATASET INTERRUPT ENABLE

```

```

8958                                     : UNUSED          BIT04
8959      000010      SECXMIT=           BIT03          : SECONDARY TRANSMIT DATA
8960      000004      REQSEND=          BIT02          : REQUEST TO SEND
8961      000002      DTR=              BIT01          : DATA TERMINAL READY
8962                                     : UNUSED          BIT00
8963
8964      ;:*****
8965      ; RBUF REGISTER BIT NAMES
8966      ;:*****
8967      100000      ERROR=              BIT15          : ERROR INDICATOR
8968      040000      ORERR=             BIT14          : OVERRUN ERROR
8969      020000      FRERR=             BIT13          : FRAMING ERROR
8970      010000      PERR=             BIT12          : PARITY ERROR
8971                                     : UNUSED          BIT11
8972                                     : UNUSED          BIT10
8973                                     : UNUSED          BIT09
8974                                     : UNUSED          BIT08
8975      000200      RDATA7=           BIT07          : \
8976      000100      RDATA6=           BIT06          : |
8977      000040      RDATA5=           BIT05          : |
8978      000020      RDATA4=           BIT04          : | RECEIVED DATA BITS
8979      000010      RDATA3=           BIT03          : |
8980      000004      RDATA2=           BIT02          : |
8981      000002      RDATA1=           BIT01          : |
8982      000001      RDATA0=           BIT00          : /
8983
8984      ;:*****
8985      ; TCSR REGISTER BIT NAMES
8986      ;:*****
8987      100000      PBAUD3=            BIT15          : \
8988      040000      PBAUD2=            BIT14          : | PROGRAMMABLE BAUD
8989      020000      PBAUD1=            BIT13          : | RATE BITS
8990      010000      PBAUD0=            BIT12          : |
8991      004000      PBAUDSET=          BIT11          : / ENABLE SETTING OF
8992                                     : PROGRAMMABLE BAUDE RATE
8993                                     : UNUSED          BIT10
8994                                     : UNUSED          BIT09
8995                                     : UNUSED          BIT08
8996      000200      XMITRDY=           BIT07          : TRANSMITTER READY
8997      000100      XMITIE=           BIT06          : TRANSMITTER INTERRUPT ENABLE
8998                                     : UNUSED          BIT05
8999                                     : UNUSED          BIT04
9000                                     : UNUSED          BIT03
9001      000004      MAINT=            BIT02          : MAINTENANCE SET BIT
9002                                     : UNUSED          BIT01
9003      000001      BREAK=            BIT00          : SEND BREAK (CONTINUOUS SPACE)
9004
9005
9006      ;:*****
9007      ; TBUF REGISTER BIT NAMES
9008      ;:*****
9009      : UNUSED          BIT15
  
```

```

9011
9012          : UNUSED      BIT14
9013          : UNUSED      BIT13
9014          : UNUSED      BIT12
9015          : UNUSED      BIT11
9016          : UNUSED      BIT10
9017          : UNUSED      BIT09
9018          : UNUSED      BIT08
9019          000200      TDATA7=      BIT07      : \
9020          000100      TDATA6=      BIT06      : |
9021          000040      TDATA5=      BIT05      : |
9022          000020      TDATA4=      BIT04      : | \ TRANSMITTER DATA BUFFER
9023          000010      TDATA3=      BIT03      : |
9024          000004      TDATA2=      BIT02      : |
9025          000002      TDATA1=      BIT01      : |
9026          000001      TDATA0=      BIT00      : /

;*****
; FLAG BITS TO BE USE OR CLEARED IN $USWR.

9032          000017      DATA =      17
9033          000020      PARITY =     20
9034          000040      EVENODD =    40
9035          000100      COMSPD =    100
9036          000200      PBR =       200

; BAUDE MUST BE ON THE UPPER
; BYTE BOUNDRY OF $USWR.--4 BITS
9038          :
9039          :
9040          007400      BAUD =       7400
9041          010000      BRK =       10000
9042          020000      CABLE =     20000
9043          040000      FRFD =     40000

;*****
.SBTTL TRAP CATCHER
9045
(1)
(1)          000000      .=0
; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)          000174      .=174
(1) 000174 000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
(1) 000176 000000      SWREG:  .WORD 0      ;;SOFTWARE SWITCH REGISTER
(1)
(1) 000200 000137 001336 .SBTTL STARTING ADDRESS(ES)
(1)          JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
9046 .SBTTL ACT11 HOOKS
(1)
(2)
(1)
;*****
;HOOKS REQUIRED BY ACT11
(1)          000204      $SVPC=.      ;SAVE PC
(1)          000046      .=46
(1) 000046 012446      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)          000052      .=52
(1) 000052 000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
(1)          000204      .= $SVPC      ;; RESTORE PC
9047          001000      .=1000
  
```


9048
(1)
(2)
(1)
(2)
(1) 001000
(1) 000024
(1) 000024 000200
(1) 000044 000044
(1) 000044 001000
(1) 001000
(2)
(1)
(1)
(1)
(1) 001000
(1) 001000 000000
(1) 001002 001174
(1) 001004 000005
(1) 001006 000055
(1) 001010 000036
(1) 001012 000030

.SBTTL APT PARAMETER BLOCK
:*****
: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:
\$SHIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MBADR: .WORD \$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
\$TSTM: .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)

(2)	001210	000000	\$MSGAD: .WORD	AMSGAD	::MESSAGE ADDRESS
(2)	001212	000000	\$MSGLG: .WORD	AMSGLG	::MESSAGE LENGTH
(2)	001214		\$ETABLE:		::APT ENVIRONMENT TABLE
(2)	001214	000	\$ENV: .BYTE	AENV	::ENVIRONMENT BYTE
(2)	001215	000	\$ENVM: .BYTE	AENVM	
(2)			::ENVIRONMENT	MODE BITS	
(2)	001216	000000	\$SWREG: .WORD	ASWREG	::APT SWITCH REGISTER
(2)	001220	071110	\$USWR: .WORD	AUSWR	::USER SWITCHES
(2)	001222	000000	\$CPUOP: .WORD	ACPUOP	::CPU TYPE,OPTIONS
(2)			::*		BITS 15-11=CPU TYPE
(2)			::*		11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)			::*		11/70=06,PDQ=07,Q=10
(2)			::*		BIT 10=REAL TIME CLOCK
(2)			::*		BIT 9=FLOATING POINT PROCESSOR
(2)			::*		BIT 8=MEMORY MANAGEMENT
(2)	001224	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
(2)	001225	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE,BLK#1
(2)			::*		MEM.TYPE BYTE -- (HIGH BYTE)
(2)			::*		900 NSEC CORE=001
(2)			::*		300 NSEC BIPOLAR=002
(2)			::*		500 NSEC MOS=003
(2)	001226	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS,BLK#1
(2)			::*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2)	001230	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
(2)	001231	000	\$MTYP2: .BYTE	AMTYP2	::MEM.TYPE,BLK#2
(2)	001232	000000	\$MADR2: .WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
(2)	001234	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
(2)	001235	000	\$MTYP3: .BYTE	AMTYP3	::MEM.TYPE,BLK#3
(2)	001236	000000	\$MADR3: .WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
(2)	001240	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
(2)	001241	000	\$MTYP4: .BYTE	AMTYP4	::MEM.TYPE,BLK#4
(2)	001242	000000	\$MADR4: .WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
(2)	001244	000300	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)	001246	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
(2)	001250	175610	\$BASE: .WORD	ABASE	
(2)			::BASE	ADDRESS	OF EQUIPMENT UNDER TEST
(2)	001252	000001	\$DEV: .WORD	ADEV	::DEVICE MAP
(2)	001254		\$ETEND:		
(2)			.MEXIT		


```

(2) 001524 022777 177777 177406      CMP    #-1,@SWR      ;;TRY TO REFERENCE HARDWARE SWR
(2) 001532 001012                      BNE    66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)                                ;;AND THE HARDWARE SWR IS NOT = -1
(2) 001534 000403                      BR     65$           ;;BRANCH IF NO TIMEOUT
(2) 001536 012716 001544 64$:         MOV    #65$,(SP)    ;;SET UP FOR TRAP RETURN
(2) 001542 000002                      RTI
(2) 001544 012767 000176 177366 65$:  MOV    #SWREG,SWR   ;;POINT TO SOFTWARE SWR
(2) 001552 012767 000174 177362      MOV    #DISPREG,DISPLAY
(2) 001560 012637 000004 66$:         MOV    (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
(1)
(2) 001564 005067 177412                      CLR    $PASS        ;;CLEAR PASS COUNT
(2) 001570 132767 000200 177417      BITB   #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2) 001576 001403                      BEQ    67$           ;;YES,USE NON-APT SWITCH
(2) 001600 012767 001216 177332      MOV    #$$SWREG,SWR ;;NO,USE APT SWITCH REGISTER
(2) 001606
9063
(1)                                .SBTTL  TYPE PROGRAM NAME
(1)                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1) 001606 005227 177777                      INC    #-1           ;;FIRST TIME?
(1) 001612 001051                      BNE    68$           ;;BRANCH IF NO
(1) 001614 022737 012446 000042      CMP    #SENDAD,@#42 ;;ACT-11?
(1) 001622 001445                      BEQ    68$           ;;BRANCH IF YES
(1) 001624 104401 001672                      TYPE   ,69$         ;;TYPE ASCIZ STRING
(2)                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
(2) 001630 005737 000042                      TST    @#42         ;;ARE WE RUNNING UNDER XXDP/ACT?
(2) 001634 001012                      BNE    70$           ;;BRANCH IF YES
(2) 001636 126727 177352 000001      CMPB   $ENV,#1      ;;ARE WE RUNNING UNDER APT?
(2) 001644 001406                      BEQ    70$           ;;BRANCH IF YES
(2) 001646 026727 177266 000176      CMP    SWR,#SWREG   ;;SOFTWARE SWITCH REG SELECTED?
(2) 001654 001005                      BNE    71$           ;;BRANCH IF NO
(2) 001656 104406                      GTSWR                ;;GET SOFT-SWR SETTINGS
(2) 001660 000403                      BR     71$
(2) 001662 112767 000001 177244 70$:  MOVB   #1,$AUTOB    ;;SET AUTO-MODE INDICATOR
(2) 001670 71$:
(1) 001670 000422                      BR     68$           ;;GET OVER THE ASCIZ
(1)                                ;;69$: .ASCIZ <CRLF>*ZZ-CNDVA-A DLV11-E OFFLINE TEST*<CRLF>
(1)                                68$:
9068 001736                      LET    INITFLAG := #1
(4) 001736 012767 000001 010404      MOV    #1,INITFLAG
9069 001744                      LOOP:
9070 001744                      CALL   CYCLE        ; NO ARGUMENTS--ADRS -> NEXT ADDRESS
(3) 001744 004767 010250                      JSR    PC,CYCLE
9071
9072
9073 001750                      ; ADDR+2 -> NEXT VECTOR
(4) 001750 012167 177300                      MOV    (ADRS)+,DLADD ;GET UNIT ADDRESS
9074
9075 001754                      ;GET UNIT VECTOR
(4) 001754 011167 177276                      MOV    (ADRS),DLVEC  DLADD := (ADRS)+
9076 001760                      LET    ADRS := DLADD
(4) 001760 016701 177270                      MOV    DLADD,ADRS
9077
9078 001764                      ;RCSR = DLADD + 0
(4) 001764 016767 177264 177266      MOV    DLADD,RCSR   LET    RCSR := DLADD
9079 001772                      LET    RBUF := DLADD + #2
(4) 001772 016767 177256 177262      MOV    DLADD,RBUF
(7) 002000 062767 000002 177254      ADD    #2,RBUF

```

9080	002006						LET	TCSR := DLADD + #4
(4)	002006	016767	177242	177250	MOV	DLADD,TCSR		
(7)	002014	062767	000004	177242	ADD	#4,TCSR		
9081	002022						LET	TCSRHI := DLADD + #5
(4)	002022	016767	177226	177236	MOV	DLADD,TCSRHI		
(7)	002030	062767	000005	177230	ADD	#5,TCSRHI		
9082	002036						LET	TBUF := DLADD + #6
(4)	002036	016767	177212	177224	MOV	DLADD,TBUF		
(7)	002044	062767	000006	177216	ADD	#6,TBUF		
9083	002052						LET	R5 := #RSSTACK
(4)	002052	012705	001334		MOV	#RSSTACK,R5		
9088								::BRESET
(1)	002056	000005			RESET			


```

9099
9100      ::*****
(3)      ::*TEST 1      ADDRESSABILITY
(4)      ::*          THIS TEST VERIFIES THAT THE ADDRESS AS PLACED IN
(4)      ::*          THE HARDWARE P-TABLE TO BE CORRECT AND THE DLV11-E RESPONDS
(4)      ::*          TO THAT ADDRESS SPACE
(3)      ::*****
(2) 002060 000004      TST1:  SCOPE
(2)
(1) 002062 012767 000002 177070      MOV      #2,$TIMES      ;;DO 2 ITERATIONS
(2) 002070 012767 000001 177102      MOV      #1,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
9105 002076                                LET      ADRS := DLADD
(4) 002076 016701 177152      MOV      DLADD,ADRS
9106
9107                                ; SET UP INTERRUPT
(5) 002102                                #ILLMEM,#INTSRV,#PR6
(5) 002102 010146      MOV      R1,-(SP)
(5) 002104 012701 000004      MOV      #ILLMEM,R1
(5) 002110 012721 012026      MOV      #INTSRV,(R1)+
(5) 002114 012711 000300      MOV      #PR6,(R1)
(5) 002120 012601      MOV      (SP)+,R1
9108 002122                                LET      I := #0
(4) 002122 005067 177144      CLR      I
9109 002126                                REPEAT
(3) 002126                                $1:
9110 002126                                BGNSUB
(5) 002126 012767 002134 176754      MOV      #64$,$LPERR
9111                                ;CLEAR FLAG
9112 002134                                LET INTFLAG := #0
(4) 002134 005067 007674      CLR      INTFLAG
9113
9114                                ;READ FLAG
9115 002140 005711      TST @ADRS
9116 002142                                IF INTFLAG NE #0 THEN
(6) 002142 005767 007666      TST      INTFLAG
(9) 002146 001401      BEQ      $2
9117                                ; FATAL ERROR
9118 002150                                ERRDF 1,,NODL
(1) 002150 104001      ERROR  1
9119 002152                                ENDIF
(4) 002152                                $2:
9120 002152                                ENDSUB
9121 002152                                LET      I := I + #2
(7) 002152 062767 000002 177112      ADD      #2,I
9122 002160                                LET      ADRS := DLADD + I
(4) 002160 016701 177070      MOV      DLADD,ADRS
(7) 002164 066701 177102      ADD      I,ADRS
9123 002170                                UNTIL I EQ #8.
(3) 002170 026727 177076 000010      CMP      I,#8.
(6) 002176 001353      BNE      $1
9124 002200                                CLRVEC ILLMEM
(3) 002200 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
(3) 002202 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
(5) 002204 012701 000004      MOV      #ILLMEM,R1
(5) 002210 010102      MOV      R1,R2
(8) 002212 062702 000002      ADD      #2,R2
(5) 002216 010221      MOV      R2,(R1)+

```

(5) 002220 005011 CLR (R1)
(3) 002222 012602 MOV (SP)+,R2 ;:POP STACK INTO R2
(3) 002224 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
9125 ;END OF TEST
9126 002226

ENDTST

9131 :*****
9132 :* THE FOLLOWING 8 TESTS TEST ALL 'READ WRITE' BITS
9133 :*****
9134
9135

```
9141
9142          ::*****
(3)          ::TEST 2      BREAK - TCSRO SET, CLEAR, RESET
(4)          ::          NOTE: THE (H) JUMPER MUST BE REMOVED FOR THIS
(4)          ::          TEST TO FUNCTION PROPERLY.
(3)          ::*****
(2) 002226 000004          TST2:  SCOPE
(2)
(1) 002230 012767 000010 176722          MOV      #10,$TIMES          ;;DO 10 ITERATIONS
(2) 002236 012767 000002 176734          MOV      #2,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9143
9148 002244                                IF #BRK NOTSETIN $USWR THEN
(6) 002244 032767 010000 176746          BIT      #BRK,$USWR
(9) 002252 001004                                BNE      $3
9149 002254                                EXIT TST
(5) 002254 012767 000001 176676          MOV      #1,$TIMES
(3) 002262 000452          BR        TST3          ;;EXIT THIS TEST
9150 002264                                ENDF
(4) 002264          $3:
9151                                ; SEE IF IT IS CLEAR
9152 002264                                BGNSUB
(5) 002264 012767 002272 176616          MOV      #64,$LPERR
9153
9154 002272                                IF      #BREAK SETIN @TCSR THEN
(6) 002272 032777 000001 176764          BIT      #BREAK,@TCSR
(9) 002300 001401                                BEQ      $4
9155                                ; BREAK DID NOT RESET IN TCSR
9156 002302                                ERRHRD 2,,DIDNOT
(1) 002302 104002          ERROR 2
9157 002304                                ENDF
(4) 002304          $4:
9158 002304                                ENDSUB
9159
9160                                ; TRY TO SET BREAK BIT
9161 002304                                BGNSUB
(5) 002304 012767 002312 176576          MOV      #64,$LPERR
9162 002312                                LET      @TCSR := @TCSR SET.BY #BREAK
(7) 002312 052777 000001 176744          BIS      #BREAK,@TCSR
9163                                ; STUCK TO 0
9164 002320                                IF      #BREAK NOTSETIN @TCSR THEN
(6) 002320 032777 000001 176736          BIT      #BREAK,@TCSR
(9) 002326 001001                                BNE      $5
9165                                ; BREAK DID NOT SET IN TCSR
9166 002330                                ERRHRD 3,,DIDNOT
(1) 002330 104003          ERROR 3
9167 002332                                ENDF
(4) 002332          $5:
9168 002332                                ENDSUB
9169
9170                                ; TRY TO CLEAR A SET BIT
9171 002332                                BGNSUB
(5) 002332 012767 002340 176550          MOV      #64,$LPERR
9172
9173 002340                                LET      @TCSR := @TCSR CLR.BY #BREAK
(7) 002340 042777 000001 176716          BIC      #BREAK,@TCSR
9174                                ; SHOULD HAVE CLEARED
```



```

9201
9202          ;*****
(3)          ;*TEST 3          MAINT - TCSR2 SET, CLEAR, RESET
(3)          ;*****
(2) 002410 000004          TST3:  SCOPE
(2)
(1) 002412 012767 000010 176540          MOV  #10,$TIMES          ;;DO 10 ITERATIONS
(2) 002420 012767 000003 176552          MOV  #3,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9207
9208          ; SEE IF IT IS CLEAR
9209 002426          ; SEE IF IT IS CLEAR
(5) 002426 012767 002434 176454          MOV  #64$,$LPERR          BGNSUB
9210
9211 002434          IF  #MAINT SETIN @TCSR THEN
(6) 002434 032777 000004 176622          BIT  #MAINT,@TCSR
(9) 002442 001401          BEQ  $10
9212          ; MAINT DID NOT RESET IN TCSR
9213 002444          ; MAINT DID NOT RESET IN TCSR
(1) 002444 104006          ERROR 6          ERRHRD 6,,DIDNOT
9214 002446          ENDIF
(4) 002446          $10:
9215 002446          ENDSUB
9216
9217          ; TRY TO SET MAINT BIT
9218 002446          ; TRY TO SET MAINT BIT
(5) 002446 012767 002454 176434          MOV  #64$,$LPERR          BGNSUB
9219 002454          LET  @TCSR := @TCSR SET.BY #MAINT
(7) 002454 052777 000004 176602          BIS  #MAINT,@TCSR
9220          ; STUCK TO 0
9221 002462          IF  #MAINT NOTSETIN @TCSR THEN
(6) 002462 032777 000004 176574          BIT  #MAINT,@TCSR
(9) 002470 001001          BNE  $11
9222          ; MAINT DID NOT SET IN TCSR
9223 002472          ; MAINT DID NOT SET IN TCSR
(1) 002472 104007          ERROR 7          ERRHRD 7,,DIDNOT
9224 002474          ENDIF
(4) 002474          $11:
9225 002474          ENDSUB
9226
9227          ; TRY TO CLEAR A SET BIT
9228 002474          ; TRY TO CLEAR A SET BIT
(5) 002474 012767 002502 176406          MOV  #64$,$LPERR          BGNSUB
9229
9230 002502          LET  @TCSR := @TCSR CLR.BY #MAINT
(7) 002502 042777 000004 176554          BIC  #MAINT,@TCSR
9231          ; SHOULD HAVE CLEARED
9232 002510          IF  #MAINT SETIN @TCSR THEN
(6) 002510 032777 000004 176546          BIT  #MAINT,@TCSR
(9) 002516 001401          BEQ  $12
9233          ; MAINT DID NOT CLEAR INTCSR
9234 002520          ; MAINT DID NOT CLEAR INTCSR
(1) 002520 104010          ERROR 10          ERRHRD 10,,DIDNOT
9235 002522          ENDIF
(4) 002522          $12:
9236 002522          ENDSUB
9237

```

```
9238 ; NOW SEE IF RESET CLEARS IT
9239 002522 BGNSUB
(5) 002522 012767 002530 176360 MOV #64$, $LPERR
9240
9241 002530 LET @TCSR := @TCSR SET.BY #MAINT
(7) 002530 052777 000004 176526 BIS #MAINT, @TCSR
9242 ; ISSUE BUS RESET
9243 002536 BRESSET
(1) 002536 000005 RESET
9244 002540 IF #MAINT SET IN @TCSR THEN
(6) 002540 032777 000004 176516 BIT #MAINT, @TCSR
(9) 002546 001401 BEQ $13
9245 ; MAINT DID NOT RESET IN TCSR
9246 002550 ERRHRD 11, , DIDNOT
(1) 002550 104011 ERROR 11
9247 002552 ENDIF
(4) 002552 $13:
9248 002552 ENDSUB
9249 002552 ENDTST
9250
9251
9252
9257 ;*****
```

```

9259
9260          ;*****
(3)          ;*TEST 4          XMITIE - TCSR6 SET, CLEAR, RESET
(3)          ;*****
(2) 002552 000004          TST4:  SCOPE
(2)
(1) 002554 012767 000010 176376          MOV    #10,$TIMES          ;;DO 10 ITERATIONS
(2) 002562 012767 000004 176410          MOV    #4,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9261                                     ; USE PRIORITY OF 6
9266 002570 012746 000300          MOV    #PR6,-(SP)          ;;PUT NEW PS ON STACK
(1) 002574 012746 002602          MOV    #64$,-(SP)          ;;PUT NEW PC ON STACK
(1) 002600 000002          RTI          ;;POP NEW PC AND PS
(1) 002602          64$:
9271                                     ; SEE IF IT IS CLEAR
9276                                     BGNSUB
9277 002602          MOV    #65$,$LPERR
(5) 002602 012767 002610 176300
9278
9279 002610          IF    #XMITIE SETIN @TCSR THEN
(6) 002610 032777 000100 176446          BIT    #XMITIE,@TCSR
(9) 002616 001401          BEQ    $14
9280                                     ; XMITIE DID NOT RESET IN TCSR
9281 002620          ERROR 12          ERRHRD 12,,DIDNOT
(1) 002620 104012
9282 002622          ENDIF
(4) 002622          $14:
9283 002622          ENDSUB
9284
9285                                     ; TRY TO SET XMITIE BIT
9286 002622          MOV    #64$,$LPERR          BGNSUB
(5) 002622 012767 002630 176260
9287 002630          LET    @TCSR := @TCSR SET.BY #XMITIE
(7) 002630 052777 000100 176426          BIS    #XMITIE,@TCSR
9288                                     ; STUCK TO 0
9289 002636          IF    #XMITIE NOTSETIN @TCSR THEN
(6) 002636 032777 000100 176420          BIT    #XMITIE,@TCSR
(9) 002644 001001          BNE    $15
9290                                     ; XMIT DID NOT RESET IN TCSR
9291 002646          ERROR 13          ERRHRD 13,,DIDNOT
(1) 002646 104013
9292 002650          ENDIF
(4) 002650          $15:
9293 002650          ENDSUB
9294
9295                                     ; TRY TO CLEAR A SET BIT
9296 002650          MOV    #64$,$LPERR          BGNSUB
(5) 002650 012767 002656 176232
9297
9298 002656          LET    @TCSR := @TCSR CLR.BY #XMITIE
(7) 002656 042777 000100 176400          BIC    #XMITIE,@TCSR
9299                                     ; SHOULD HAVE CLEARED
9300 002664          IF    #XMITIE SETIN @TCSR THEN
(6) 002664 032777 000100 176372          BIT    #XMITIE,@TCSR
(9) 002672 001401          BEQ    $16
9301                                     ; XMIT DID NOT CLEAR IN TCSR
9302 002674          ERRHRD 14,,DIDNOT

```



```
(1) 002674 104014          ERROR 14
9303 002676
(4) 002676          $16:          ENDIF
9304 002676          ENDSUB
9305
9306          ; NOW SEE IF RESET CLEARS IT
9307 002676 012767 002704 176204  MOV #64$, $LPERR          BGNSUB
(5) 002676
9308
9309 002704          LET @TCSR := @TCSR SET.BY #XMITIE
(7) 002704 052777 000100 176352  BIS #XMITIE, @TCSR          ; ISSUE BUS RESET
9310          BRESÉT
9311 002712          RESET
(1) 002712 000005
9312 002714          IF #XMITIE SET IN @TCSR THEN
(6) 002714 032777 000100 176342  BIT #XMITIE, @TCSR
(9) 002722 001401          BEQ $17          ; XMIT DID NOT RESET IN TCSR
9313          ERRHRD 15, ,DIDNOT
9314 002724          ERROR 15
(1) 002724 104015          ENDIF
9315 002726          $17:
(4) 002726          ENDSUB
9316 002726          ENDTST
9317 002726
9318
9319
9320
9325          ;*****
```

```
9334
9335
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 002726 000004
(2)
(1) 002730 012767 000010 176222
(2) 002736 012767 000005 176234
9340 002744
(6) 002744 032767 040000 176246
(9) 002752 001004
9341 002754
(5) 002754 012767 000001 176176
(3) 002762 000441
9342 002764
(4) 002764
9343
9344 002764
(5) 002764 012767 002772 176116
9345 002772
(7) 002772 042777 000002 176260
9346
9347 003000
(6) 003000 032777 000002 176252
(9) 003006 001401
9348
9349 003010
(1) 003010 104016
9350 003012
(4) 003012
9351 003012
9352
9353
9354 003012
(5) 003012 012767 003020 176070
9355
9356 003020
(7) 003020 052777 000002 176232
9357 003026
(6) 003026 032777 000002 176224
(9) 003034 001001
9358
9359 003036
(1) 003036 104017
9360 003040
(4) 003040
9361 003040
9362
9363
9364 003040
(5) 003040 012767 003046 176042

*****
*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
      LET @RCSR := @RCSR CLR.BY #DTR
      BIC #DTR,@RCSR
      ; 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
      LET @RCSR := @RCSR SET.BY #DTR
      BIS #DTR,@RCSR
      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
```

9365 003046
(7) 003046 042777 000002 176204
9366
9367 003054
(6) 003054 032777 000002 176176
(9) 003062 001401
9368
9369 003064
(1) 003064 104020
9370 003066
(4) 003066
9371 003066
9372 003066
9373
9374
9375
9380

BIC #DTR,@RCSR
BIT #DTR,@RCSR
BEQ \$23
ERROR 20

\$23:

LET @RCSR := @RCSR CLR.BY #DTR
; SHOULD HAVE CLEARED IT
IF #DTR SET IN @RCSR THEN
; DTR DID NOT CLEAR IN RCSR
ERRHRD 20,,DIDNOT
ENDIF
ENDSUB
ENDTST

::*****

```
9385
9386
(3)
(4)
(3)
(2) 003066 000004
(2)
(1) 003070 012767 000010 176062 MOV #10,$TIMES ;;DO 10 ITERATIONS
(2) 003076 012767 000006 176074 MOV #6,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9391 003104 032767 040000 176106 BIT #FRFD,$USWR IF #FRFD NOTSETIN $USWR THEN
(6) 003104 032767 040000 176106 BNE $24
(9) 003112 001004
9392 003114 012767 000001 176036 MOV #1,$TIMES EXIT TST
(5) 003114 012767 000001 176036 BR TST7 ;;EXIT THIS TEST
(3) 003122 000452
9393 003124
(4) 003124 $24:
9394
9395 ; SEE IF IT IS CLEAR
9396 003124 012767 003132 175756 MOV #64,$LPERR BGNSUB
(5) 003124 012767 003132 175756
9397
9398 003132 032777 000004 176120 BIT #REQSEND,@RCSR IF #REQSEND SETIN @RCSR THEN
(6) 003132 032777 000004 176120 BEQ $25
(9) 003140 001401 ; REQSEND DID NOT RESET IN RCSR
9399 ;ERRHRD 21,,DIDNOT
9400 003142 104021 ERROR 21
(1) 003142 104021
9401 003144
(4) 003144 $25:
9402 003144 ENDSUB
9403
9404 ; TRY TO SET REQSEND BIT
9405 003144 012767 003152 175736 MOV #64,$LPERR BGNSUB
(5) 003144 012767 003152 175736
9406 003152 052777 000004 176100 BIS #REQSEND,@RCSR LET @RCSR := @RCSR SET.BY #REQSEND
(7) 003152 052777 000004 176100
9407
9408 003160 032777 000004 176072 BIT #REQSEND,@RCSR IF ; STUCK TO 0
(6) 003160 032777 000004 176072 BNE $26 ; #REQSEND NOTSETIN @RCSR THEN
(9) 003166 001001 ; REQSEND DID NOT SET IN RCSR
9409 ;ERRHRD 22,,DIDNOT
9410 003170 104022 ERROR 22
(1) 003170 104022
9411 003172
(4) 003172 $26:
9412 003172 ENDSUB
9413
9414 ; TRY TO CLEAR A SET BIT
9415 003172 012767 003200 175710 MOV #64,$LPERR BGNSUB
(5) 003172 012767 003200 175710
9416
9417 003200 042777 000004 176052 BIC #REQSEND,@RCSR LET @RCSR := @RCSR CLR.BY #REQSEND
(7) 003200 042777 000004 176052
9418
9419 003206 IF ; SHOULD HAVE CLEARED
; #REQSEND SETIN @RCSR THEN
```

```
(6) 003206 032777 000004 176044 BIT #REQSEND,@RCSR
(9) 003214 001401 BEQ $27
9420 ; REQSEND DID NOT CLEAR IN RCSR
9421 003216 104023 ERROR 23 ERRHRD 23,,DIDNOT
(1) 003216 104023
9422 003220 ENDF
(4) 003220 $27: ENDSUB
9423 003220 ; NOW SEE IF RESET CLEARS IT
9424 BGNSUB
9425
9426 003220 MOV #64$,$LPERR
(5) 003220 012767 003226 175662
9427
9428 003226 LET @RCSR := @RCSR SET.BY #REQSEND
(7) 003226 052777 000004 176024 BIS #REQSEND,@RCSR
9429 ; ISSUE BUS RESET
9430 003234 BRES
(1) 003234 000005 RESET IF #REQSEND SET IN @RCSR THEN
9431 003236
(6) 003236 032777 000004 176014 BIT #REQSEND,@RCSR
(9) 003244 001401 BEQ $30
9432 ; REQSEND DID NOT RESET IN RCSR
9433 003246 ERROR 24 ERRHRD 24,,DIDNOT
(1) 003246 104024 ENDF
9434 003250 $30: ENDSUB
(4) 003250 ENDTST
9435 003250
9436 003250
9437
9438
9439
9444 ;*****
```

```

9446
9447
(3)
(3)
(2) 003250 000004
(2)
(1) 003252 012767 000010 175700
(2) 003260 012767 000007 175712
9452
9453 003266
(5) 003266 012767 003274 175614
9454
9455 003274
(6) 003274 032777 000010 175756
(9) 003302 001401
9456
9457 003304
(1) 003304 104025
9458 003306
(4) 003306
9459 003306
9460
9461
9462 003306
(5) 003306 012767 003314 175574
9463 003314
(7) 003314 052777 000010 175736
9464
9465 003322
(6) 003322 032777 000010 175730
(9) 003330 001001
9466
9467 003332
(1) 003332 104026
9468 003334
(4) 003334
9469 003334
9470
9471
9472 003334
(5) 003334 012767 003342 175546
9473
9474 003342
(7) 003342 042777 000010 175710
9475
9476 003350
(6) 003350 032777 000010 175702
(9) 003356 001401
9477
9478 003360
(1) 003360 104027
9479 003362
(4) 003362
9480 003362
9481
9482 003362
  
```

```

*****
*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
      ENDSUB
      ; TRY TO SET SECXMIT BIT
      BGNSUB
      MOV      #64,$$LPERR
      BIS      #SECXMIT,@RCSR
      LET     @RCSR := @RCSR SET.BY #SECXMIT
      IF      ; STUCK TO 0
      #SECXMIT NOTSETIN @RCSR THEN
      BIT      #SECXMIT,@RCSR
      BNE     $32
      ; SECXMIT DID NOT SET IN RCSR
      ERRHRD 26,,DIDNOT
      ENDIF
      ENDSUB
      ; TRY TO CLEAR A SET BIT
      BGNSUB
      MOV      #64,$$LPERR
      BIC      #SECXMIT,@RCSR
      LET     @RCSR := @RCSR CLR.BY #SECXMIT
      IF      ; SHOULD HAVE CLEARED
      #SECXMIT SETIN @RCSR THEN
      BIT      #SECXMIT,@RCSR
      BEQ     $33
      ; SECXMIT DID NOT CLEAR IN RCSR
      ERRHRD 27,,DIDNOT
      ENDIF
      ENDSUB
      BGNSUB
  
```

```
(5) 003362 012767 003370 175520 MOV #64$, $LPERR ; NOW SEE IF RESET CLEARS IT
9483
9484
9485 003370 LET @RCSR := @RCSR SET.BY #SECXMIT
(7) 003370 052777 000010 175662 BIS #SECXMIT, @RCSR ; ISSUE BUS RESET
9486 BRESÉT
9487 003376 RESET
(1) 003376 000005
9488 003400 IF #SECXMIT SETIN @RCSR THEN
(6) 003400 032777 000010 175652 BIT #SECXMIT, @RCSR
(9) 003406 001401 BEQ $34 ; SECXMIT DID NOT RESET IN RCSR
9489 ERRHRD 30,,DIDNOT
9490 003410 ERROR 30
(1) 003410 104030
9491 003412 ENDIF
(4) 003412 $34:
9492 003412 ENDSUB
9493 003412 ENDTST
9494
9495
9496
9501 ;*****
```

```

9503
9504
(3)
(3)
(2) 003412 000004
(2)
(1) 003414 012767 000010 175536
(2) 003422 012767 000010 175550
9509
9510 003430
(5) 003430 012767 003436 175452
9511
9512 003436
(6) 003436 032777 000040 175614
(9) 003444 001401
9513
9514 003446
(1) 003446 104031
9515 003450
(4) 003450
9516 003450
9517
9518
9519 003450
(5) 003450 012767 003456 175432
9520 003456
(7) 003456 052777 000040 175574
9521
9522 003464
(6) 003464 032777 000040 175566
(9) 003472 001001
9523
9524 003474
(1) 003474 104032
9525 003476
(4) 003476
9526 003476
9527
9528
9529 003476
(5) 003476 012767 003504 175404
9530
9531 003504
(7) 003504 042777 000040 175546
9532
9533 003512
(6) 003512 032777 000040 175540
(9) 003520 001401
9534
9535 003522
(1) 003522 104033
9536 003524
(4) 003524
9537 003524
9538
9539
  
```

```

*****
*TEST 10 DATAIE - RCSR5 SET, CLEAR, RESET
*****
TST10: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV #10,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
; SEE IF IT IS CLEAR
BGNSUB
MOV #64,$LPERR
IF #DATAIE SETIN @RCSR THEN
BIT #DATAIE,@RCSR
BEQ $35
; DATAIE DID NOT RESET IN RCSR
ERRHRD 31,,DIDNOT
ENDIF
ENDSUB
; TRY TO SET DATAIE BIT
BGNSUB
MOV #64,$LPERR
LET @RCSR := @RCSR SET.BY #DATAIE
BIS #DATAIE,@RCSR
; STUCK TO 0
IF #DATAIE NOTSETIN @RCSR THEN
BIT #DATAIE,@RCSR
BNE $36
; DATAIE DID NOT SET IN RCSR
ERRHRD 32,,DIDNOT
ENDIF
ENDSUB
; TRY TO CLEAR A SET BIT
BGNSUB
MOV #64,$LPERR
LET @RCSR := @RCSR CLR.BY #DATAIE
BIC #DATAIE,@RCSR
; SHOULD HAVE CLEARED
IF #DATAIE SETIN @RCSR THEN
BIT #DATAIE,@RCSR
BEQ $37
; DATAIE DID NOT CLEAR IN RCSR
ERRHRD 33,,DIDNOT
ENDIF
ENDSUB
; NOW SEE IF RESET CLEARS IT
  
```



```
9540 003524          BGNSUB
(5) 003524 012767 003532 175356   MOV   #64$, $LPERR
9541
9542 003532          LET   @RCSR := @RCSR SET.BY #DATAIE
(7) 003532 052777 000040 175520   BIS   #DATAIE, @RCSR
9543                                     : ISSUE BUS RESET
9544 003540          BRESÉT
(1) 003540 000005          RESET
9545 003542          IF   #DATAIE SETIN @RCSR THEN
(6) 003542 032777 000040 175510   BIT   #DATAIE, @RCSR
(9) 003550 001401          BEQ   $40
9546                                     : DATAIE DID NOT RESET IN RCSR
9547 003552          ERRHRD 34,, DIDNOT
(1) 003552 104034          ERROR  34
9548 003554          ENDIF
(4) 003554
9549 003554          $40:
9550 003554          ENDSUB
9551          ENDTST
9552
9553
9558          ;:*****
```

```
9560
9561 (3)
9562 (3)
9563 (2) 003554 000004
9564 (2)
9565 (1) 003556 012767 000010 175374
9566 (2) 003564 012767 000011 175406
9567 003572
9568 (5) 003572 012767 003600 175310
9569 003600
9570 (6) 003600 032777 000100 175452
9571 (9) 003606 001401
9572 003610
9573 (1) 003610 104035
9574 003612
9575 (4) 003612
9576 003612
9577 (5) 003612 012767 003620 175270
9578 003620
9579 (7) 003620 052777 000100 175432
9580 003626
9581 (6) 003626 032777 000100 175424
9582 (9) 003634 001001
9583 003636
9584 (1) 003636 104036
9585 003640
9586 (4) 003640
9587 003640
9588 003640
9589 (5) 003640 012767 003646 175242
9590 003646
9591 (7) 003646 042777 000100 175404
9592 003654
9593 (6) 003654 032777 000100 175376
9594 (9) 003662 001401
9595 003664
9596 (1) 003664 104037
9597 003666
9598 (4) 003666
9599 003666
9600 ; NOW SEE IF RESET CLEARS IT
```

```
*****
*TEST 11 RCVRIE - RCSR6 SET, CLEAR, RESET
*****
TST11: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV #11,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
; SEE IF IT IS CLEAR
BGNSUB
MOV #64,$LPERR
IF #RCVRIE SETIN @RCSR THEN
BIT #RCVRIE,@RCSR
BEQ $41
; RCVRIE DID NOT RESET IN RCSR
ERRHRD 35,,DIDNOT
ENDIF
$41:
ENDSUB
; TRY TO SET RCVRIE BIT
BGNSUB
MOV #64,$LPERR
LET @RCSR := @RCSR SET.BY #RCVRIE
BIS #RCVRIE,@RCSR
; STUCK TO 0
IF #RCVRIE NOTSETIN @RCSR THEN
BIT #RCVRIE,@RCSR
BNE $42
; RCVRIE DID NOT SET IN RCSR
ERRHRD 36,,DIDNOT
ENDIF
$42:
ENDSUB
; TRY TO CLEAR A SET BIT
BGNSUB
MOV #64,$LPERR
LET @RCSR := @RCSR CLR.BY #RCVRIE
BIC #RCVRIE,@RCSR
; SHOULD HAVE CLEARED
IF #RCVRIE SETIN @RCSR THEN
BIT #RCVRIE,@RCSR
BEQ $43
; RCVRIE DID NOT CLEAR IN RCSR
ERRHRD 37,,DIDNOT
ENDIF
$43:
ENDSUB
```

```
9597 003666          BGNSUB
(5) 003666 012767 003674 175214      MOV   #64$, $LPERR
9598
9599 003674          LET   @RCSR := @RCSR SET.BY #RCVRIE
(7) 003674 052777 000100 175356      BIS   #RCVRIE, @RCSR
9600                                     ; ISSUE BUS RESET
9601 003702          BRESÉT
(1) 003702 000005      RESET
9602 003704          IF   #RCVRIE SET IN @RCSR THEN
(6) 003704 032777 000100 175346      BIT   #RCVRIE, @RCSR
(9) 003712 001401      BEQ   $44
9603                                     ; RCVRIE DID NOT RESET IN RCSR
9604 003714          ERRHRD 40, , DIDNOT
(1) 003714 104040      ERROR  40
9605 003716          ENDIF
(4) 003716          $44:
9606 003716          CKLOOP
9607 003716          ENDSUB
9608 003716          ENDTST
9609
9610
9611
9612
9617
```

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

9621
9622

```

9624
9625      ::*****
(3)      :*TEST 12      TEST THAT RCVRDONE - RCSR 7 - IS CLEARED BY INIT
(3)      :*****
(2) 003716 000004      TST12: SCOPE
(2)
(1) 003720 012767 000010 175232      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
(2) 003726 012767 000012 175244      MOV      #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9626
9631
9632
9633
9634
9635 003734      BGNSUB
(5) 003734 012767 003742 175146      MOV      #64$,$LPERR
9636 003742      IF      #RCVRDONE SETIN @RCSR THEN
(6) 003742 032777 000200 175310      BIT      #RCVRDONE,@RCSR
(9) 003750 001402      BEQ      $45
9637
9638      ;RCVRDONE SHOULD HAVE CLEARED BY INIT
9639      ; RCVRDONE DID NOT CLEAR IN RCSR
9640 003752      ERRHRD 41,HRESET, DIDNOT
(1) 003752 104041      ERROR 41
9641      ;REISSUE RESET
9642 003754      BRESET
(1) 003754 000005      RESET
9643 003756      ENDIF
(4) 003756      $45:
9644
9645 003756      ;ALLOW LOOPING AFTER ERROR
9646 003756      CKLOOP
9647 003756      ENDSUB
9648
9649      ENDTST
9650

```

9656
9657
(3)
(3)
(2)
(2)
(1)
(2)
9658
9663
9664
9665
9666
(5)
9667
9668
(6)
(9)
9669
9670
9671
9672
(1)
9673
9674
(1)
9675
(4)
9676
9677
9678
9679
9680
9681
9682

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

: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,\$SLPERR

IF #XMITRDY NOTSETIN @TCSR THEN

BIT #XMITRDY,@TCSR
BNE \$46

:RESET SHOULD HAVE SET BIT.
:XMITRDY DID NOT SET IN TCSR (AFTER RESE
ERRHRD 42,HRESET,DIDNOT

ERROR 42

:ISSUE ANOTHER RESET
BRESET

RESET

ENDIF

\$46:

:ALLOW LOOPING ON ERROR

CKLOOP

ENDSUB

ENDTST

```

9688
9689      ;*****
9690      ;*TEST 14      TEST THAT DATAINT - RCSR 15 - IS CLEARED BY INIT.*
9691      ;*****
9692      ;
9693      (3)
9694      (3)
9695      (2) 004016 000004      TST14: SCOPE
9696      (2)
9697      (1) 004020 012767 000010 175132      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
9698      (2) 004026 012767 000014 175144      MOV      #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9699
9700
9701
9702
9703
9704
9705
9706
9707      004034      BGNSUB
9708      (5) 004034 012767 004042 175046      MOV      #64,$$LPERR
9709      004042      IF      #DATAINT SETIN @RCSR THEN
9710      (6) 004042 032777 100000 175210      BIT      #DATAINT,@RCSR
9711      (9) 004050 001402      BEQ      $47
9712
9713
9714
9715
9716
9717      004052      ERRHRD 43, HRESET, DIDNOT
9718      (1) 004052 104043      ERROR 43
9719
9720
9721
9722
9723
9724
9725
9726
9727
9728
9729
9730
9731
9732
9733
9734
9735
9736
9737
9738
9739
9740
9741
9742
9743
9744
9745
9746
9747
9748
9749
9750
9751
9752
9753
9754
9755
9756
9757
9758
9759
9760
9761
9762
9763
9764
9765
9766
9767
9768
9769
9770
9771
9772
9773
9774
9775
9776
9777
9778
9779
9780
9781
9782
9783
9784
9785
9786
9787
9788
9789
9790
9791
9792
9793
9794
9795
9796
9797
9798
9799
9800
9801
9802
9803
9804
9805
9806
9807
9808
9809
9810
9811
9812
9813
9814
9815
9816
9817
9818
9819
9820
9821
9822
9823
9824
9825
9826
9827
9828
9829
9830
9831
9832
9833
9834
9835
9836
9837
9838
9839
9840
9841
9842
9843
9844
9845
9846
9847
9848
9849
9850
9851
9852
9853
9854
9855
9856
9857
9858
9859
9860
9861
9862
9863
9864
9865
9866
9867
9868
9869
9870
9871
9872
9873
9874
9875
9876
9877
9878
9879
9880
9881
9882
9883
9884
9885
9886
9887
9888
9889
9890
9891
9892
9893
9894
9895
9896
9897
9898
9899
9900
9901
9902
9903
9904
9905
9906
9907
9908
9909
9910
9911
9912
9913
9914
9915
9916
9917
9918
9919
9920
9921
9922
9923
9924
9925
9926
9927
9928
9929
9930
9931
9932
9933
9934
9935
9936
9937
9938
9939
9940
9941
9942
9943
9944
9945
9946
9947
9948
9949
9950
9951
9952
9953
9954
9955
9956
9957
9958
9959
9960
9961
9962
9963
9964
9965
9966
9967
9968
9969
9970
9971
9972
9973
9974
9975
9976
9977
9978
9979
9980
9981
9982
9983
9984
9985
9986
9987
9988
9989
9990
9991
9992
9993
9994
9995
9996
9997
9998
9999

```

```
9721
9722          ;*****
(3)          ;*TEST 15      TEST THAT RCVRACT - RCSR 11 - 15 CLEARED BY INIT
(3)          ;*****
(2) 004056 000004          TST15: SCOPE
(2)
(1) 004060 012767 000010 175072          MOV      #10,$TIMES      ;;DO 10 ITERATIONS
(2) 004066 012767 000015 175104          MOV      #15,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
9723
9724
9729 004074          IF      #CABLE NOTSETIN $USWR THEN
(6) 004074 032767 020000 175116          BIT      #CABLE,$USWR
(9) 004102 001004          BNE      $50
9730          ; CAN'T TEST WITHOUT BERG OR H315.
9731 004104          EXIT TST
(5) 004104 012767 000001 175046          MOV      #1,$TIMES
(3) 004112 000411          BR       TST16          ;;EXIT THIS TEST
9732 004114          ;
(4) 004114          $50:          ENDIF
9733
9734
9735
9736 004114          BGNSUB
(5) 004114 012767 004122 174766          MOV      #64,$LPERR
9737
9738 004122          IF      #RCVRACT SETIN @RCSR THEN
(6) 004122 032777 004000 175130          BIT      #RCVRACT,@RCSR
(9) 004130 001402          BEQ      $51
9739
9740          ;RESET SHOULD HAVE CLEARED RCVRACT
9741 004132          ERRHRD 44, HRESET, DIDNOT
(1) 004132 104044          ERROR   44
9742
9743          ;TESTING EFFECT OF RESET ON BIT
9744          ;RCVRACT DID NOT CLEAR IN RCSR
9745          ;ALLOW ANOTHER TRY
9746          BRESET
9747
9748 004134          RESET
(1) 004134 000005
9749 004136          ENDIF
(4) 004136          $51:
9750          ;ALLOW LOOPING ON ERROR
9751 004136          CKLOOP
9752 004136          ENDSUB
9753 004136          ENDTST
9754
```


9838
9843
9848
(3)
(4)
(4)
(3)
(2)
(2)
(1)
(2)
9853
9854
(6)
(9)
9855
9856
(5)
(3)
9857
(4)
9858
9859
9860
9861
9862
9863
9864
9865
(5)
9866
9867
9868
(7)
9869
9870
(6)
(9)
9871
9872
(1)
9873
9874
9875
(4)
9876
9877
9878
9879
(5)
9880
9881
9882
(7)
9883
9884

004276 000004
004300 012767 000010 174652
004306 012767 000017 174664
004314
004314 032767 060000 174676
004322 001004
004324
004324 012767 000001 174626
004332 000441
004334
004334
004334
004334 012767 004342 174546
004342
004342 042777 000002 174710
004350
004350 032777 020000 174702
004356 001401
004360
004360 104050
004362
004362
004362
004362 012767 004370 174520
004370
004370 052777 000002 174662
004376

```
*****  
*****  
*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  
IF ; CAN WE USE THE WRAPAROUND??  
; #CABLE+FRFD NOTSETIN $USWR THEN  
; CAN'T TEST WITHOUT BERG OR H315  
EXIT TST  
MOV #1,$TIMES  
BR TST20 ;;EXIT THIS TEST  
ENDIF  
$56:  
; DTR AND  
; CLRSEND ARE CONNECTED  
; BY THE H315 OR EQUIV.  
; CLEAR  
BGNSUB  
MOV #64,$SLPERR  
LET ; CLEAR DTR  
@RCSR := @RCSR CLR.BY #DTR  
IF ; CLRSEND SHOULD FOLLOW  
; #CLRSEND SETIN @RCSR THEN  
; CLRSEND DID NOT  
ERRHRD 50,,FORCE  
; CLEAR WITH DTR  
ENDIF  
ENDSUB  
; SET  
BGNSUB  
MOV #64,$SLPERR  
LET ; SET DTR  
@RCSR := @RCSR SET.BY #DTR  
IF ; CLRSEND SHOULD FOLLOW  
; #CLRSEND NOTSETIN @RCSR THEN
```

```
(6) 004376 032777 020000 174654 BIT #CLRSEND,@RCSR
(9) 004404 001001 BNE $60
9885 ; CLRSEND DID NOT SET
9886 004406 ; ERRHRD 51,,FORCE
(1) 004406 104051 ERROR 51
9887 ;
9888 ; WITH DTR
9889 004410 ENDIF
(4) 004410 $60:
9890 004410 ENDSUB
9891 ;
9892 ; CLEAR
9893 004410 BGNSUB
(5) 004410 012767 004416 174472 MOV #64$, $LPERR
9894 ;
9895 ; CLEAR DTR
9896 004416 LET @RCSR := @RCSR CLR.BY #DTR
(7) 004416 042777 000002 174634 BIC #DTR,@RCSR
9897 ; CLRSEND SHOULD FOLLOW
9898 004424 IF #CLRSEND SETIN @RCSR THEN
(6) 004424 032777 020000 174626 BIT #CLRSEND,@RCSR
(9) 004432 001401 BEQ $61
9899 ; CLRSEND DID NOT
9900 004434 ERRHRD 52,,FORCE
(1) 004434 104052 ERROR 52
9901 ;
9902 ; CLEAR WITH DTR
9903 004436 ENDIF
(4) 004436 $61:
9904 004436 ENDSUB
9905 004436 ENDTST
9906
9907
9908
9909
```

```

9911
9916
9921
(3)
(4)
(4)
(3)
(2) 004436 000004
(2)
(1) 004440 012767 000010 174512      MOV #10,$TIMES      ;;DO 10 ITERATIONS
(2) 004446 012767 000020 174524      MOV #20,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
9926
9927 004454
(6) 004454 032767 060000 174536      BIT #CABLE+FRFD,$USWR      IF #CABLE+FRFD NOTSETIN $USWR THEN
(9) 004462 001004
9928
9929
9930 004464
(5) 004464 012767 000001 174466      MOV #1,$TIMES
(3) 004472 000441      BR TST21      ;;EXIT THIS TEST
9931 004474
(4) 004474      $62:
9932
9933
9934
9935
9936
9937
9938
9939 004474
(5) 004474 012767 004502 174406      MOV #64,$$LPERR      BGNSUB
9940
9941
9942 004502
(7) 004502 042777 000004 174550      BIC #REQSEND,@RCSR      LET ; CLEAR REQSEND
9943
9944 004510
(6) 004510 032777 040000 174542      BIT #RING,@RCSR      IF ; RING SHOULD FOLLOW
(9) 004516 001401      BEQ $63              #RING SETIN @RCSR THEN
9945
9946 004520
(1) 004520 104053      ERROR 53              ; RING DID NOT
9947
9948
9949 004522
(4) 004522      $63:
9950 004522
9951
9952
9953 004522
(5) 004522 012767 004530 174360      MOV #64,$$LPERR      BGNSUB
9954
9955
9956 004530
(7) 004530 052777 000004 174522      BIS #REQSEND,@RCSR      LET ; SET REQSEND
9957

```

```

9958 004536          IF #RING NOTSETIN @RCSR THEN
(6) 004536 032777 040000 174514      BIT #RING,@RCSR
(9) 004544 001001          BNE $64
9959                                     ; RING DID NOT SET
9960 004546          ; ERRHRD 54,,FORCE
(1) 004546 104054          ERROR 54
9961                                     ;
9962                                     ; WITH REQSEND
9963 004550          ENDIF
(4) 004550          $64:
9964 004550          ENDSUB
9965                                     ;
9966                                     ; CLEAR
9967 004550          BGNSUB
(5) 004550 012767 004556 174332      MOV #64$,$LPERR
9968
9969                                     ; CLEAR REQSEND
9970 004556          LET @RCSR := @RCSR CLR.BY #REQSEND
(7) 004556 042777 000004 174474      BIC #REQSEND,@RCSR
9971                                     ; RING SHOULD FOLLOW
9972 004564          IF #RING SETIN @RCSR THEN
(6) 004564 032777 040000 174466      BIT #RING,@RCSR
(9) 004572 001401          BEQ $65
9973                                     ; RING DID NOT
9974 004574          ; ERRHRD 55,,FORCE
(1) 004574 104055          ERROR 55
9975                                     ;
9976                                     ; CLEAR WITH REQSEND
9977 004576          ENDIF
(4) 004576          $65:
9978 004576          ENDSUB
9979 004576          ENDTST
9980
9981
9982
9983
  
```

```

9985
9990
9994
(3)
(4)
(3)
(2) 004576 000004
(2)
(1) 004600 012767 000010 174352 MOV #10,$TIMES ;:DO 10 ITERATIONS
(2) 004606 012767 000021 174364 MOV #21,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
9999 ; CAN WE USE THE WRAPAROUND??
10000 004614 ; IF #CABLE NOTSETIN $USWR THEN
(6) 004614 032767 020000 174376 BIT #CABLE,$USWR
(9) 004622 001004 BNE $66
10001 ; CAN'T TEST WITHOUT BERG OR H315.
10002 004624 EXIT TST
(5) 004624 012767 000001 174326 MOV #1,$TIMES
(3) 004632 000441 BR TST22 ;:EXIT THIS TEST
10003 004634 $66: ENDIF
(4) 004634
10004
10005 ; SECXMIT AND
10006 ; SECRC ARE CONNECTED
10007
10008 ; BY THE H315 OR EQUIV.
10009
10010 ; CLEAR
10011 004634 BGNSUB
(5) 004634 012767 004642 174246 MOV #64,$LPERR
10012
10013 ; CLEAR SECXMIT
10014 004642 LET @RCSR := @RCSR CLR.BY #SECXMIT
(7) 004642 042777 000010 174410 BIC #SECXMIT,@RCSR
10015 ; SECRC SHOULD FOLLOW
10016 004650 IF #SECRC SETIN @RCSR THEN
(6) 004650 032777 002000 174402 BIT #SECRC,@RCSR
(9) 004656 001401 BEQ $67
10017 ; SECRC DID NOT
10018 004660 ERRHRD 56,,FORCE
(1) 004660 104056 ERROR 56
10019
10020 ; CLEAR WITH SECXMIT
10021 004662 ENDIF
(4) 004662 $67: ENDSUB
10022 004662
10023
10024 ; SET
10025 004662 BGNSUB
(5) 004662 012767 004670 174220 MOV #64,$LPERR
10026
10027 ; SET SECXMIT
10028 004670 LET @RCSR := @RCSR SET.BY #SECXMIT
(7) 004670 052777 000010 174362 BIS #SECXMIT,@RCSR
10029 ; SECRC SHOULD FOLLOW
10030 004676 IF #SECRC NOTSETIN @RCSR THEN
(6) 004676 032777 002000 174354 BIT #SECRC,@RCSR
  
```

```
(9) 004704 001001 BNE $70 ; SECURE DID NOT SET
10031 ; ERRHRD 57,,FORCE
10032 004706
(1) 004706 104057 ERROR 57
10033
10034 ; WITH SECXMIT
10035 004710 ENDIF
(4) 004710 $70: ENDSUB
10036 004710 BGNSUB
10037 ; CLEAR
10038 ; CLEAR
10039 004710 MOV #64$, $LPERR
(5) 004710 012767 004716 174172
10040
10041 ; CLEAR SECXMIT
10042 004716 LET @RCSR := @RCSR CLR.BY #SECXMIT
(7) 004716 042777 000010 174334 BIC #SECXMIT, @RCSR
10043 ; SECURE SHOULD FOLLOW
10044 004724 IF #SECURE SET IN @RCSR THEN
(6) 004724 032777 002000 174326 BIT #SECURE, @RCSR
(9) 004732 001401 BEQ $71
10045 ; SECURE DID NOT
10046 004734 ERRHRD 60,,FORCE
(1) 004734 104060 ERROR 60
10047
10048 ; CLEAR WITH SECXMIT
10049 004736 ENDIF
(4) 004736 $71: ENDSUB
10050 004736 ENDTST
10051 004736
10052
10053
10054
10055
```

```

10057
10062
10069
  (3)
  (4)
  (4)
  (4)
  (4)
  (3)
  (2) 004736 000004
  (2)
  (1) 004740 012767 000010 174212      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
  (2) 004746 012767 000022 174224      MOV    #22,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10070
10075 004754
  (6) 004754 032767 060000 174236      BIT    #CABLE+FRFD,$USWR      IF      #CABLE+FRFD NOTSETIN $USWR THEN
  (9) 004762 001004
10076
10077
10078 004764
  (5) 004764 012767 000001 174166      MOV    #1,$TIMES
  (3) 004772 000463
10079 004774
  (4) 004774
10080
10081
10086 004774 012746 000300
  (1) 005000 012746 005006      MOV    #PR6,-(SP)      ;;PUT NEW PS ON STACK
  (1) 005004 000002      MCV   #64$,-(SP)      ;;PUT NEW PC ON STACK
  (1) 005006
10091
10092
10093 005006
  (5) 005006 012767 005014 174074      MOV    #65$,$LPERR      ;;READ TWICE - CLEARS
10094
10095 005014
  (7) 005014 042777 000002 174236      BIC    #DTR,@RCSR      BGNSUB
10096
10097 005022
  (4) 005022 010546
  (5) 005024 012745 000001
  (4) 005030 004767 004712
  (4) 005034 012605
10098
10099 005036
  (4) 005036 017703 174216      MOV    @RCSR,R3
10100
10101 005042
  (6) 005042 032777 100000 174210      BIT    #DATAINT,@RCSR
  (9) 005050 001401
10102
10103 005052
  (1) 005052 104061
10104 005054
  (4) 005054
10105
  
```

```

*****
*****
*****
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 CLRSEND BY THE H315
              THE (-FD) JUMPER MUST BE IN FOR THIS TEST.
*****
TST22: SCOPE
  
```

```

EXIT TST
;;;EXIT THIS TEST
ENDIF

$72:
      ;;MAKE SURE NOTHING UNEXPECTED HAPPENS
      ;;POP NEW PC AND PS
      ;;CAN WE USE THE WRAPAROUND??
      ;;CAN'T TEST WITHOUT BERG OR H315
      ;;OR WITH (-FD) JUMPER OUT.

$73:
  
```



```

10139
10144
10149
(3)
(4)
(4)
(3)
(2) 005142 000004
(2)
(1) 005144 012767 000010 174006 MOV #10,$TIMES ;;DO 10 ITERATIONS
(2) 005152 012767 000023 174020 MOV #23,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10150 ; CAN WE USE THE WRAPAROUND??
10155 005160 IF #CABLE+FRFD NOTSETIN $USWR THEN
(6) 005160 032767 060000 174032 BIT #CABLE+FRFD,$USWR
(9) 005166 001004 BNE $77
10156 ; CAN'T TEST WITHOUT BERG OR H315
10157 ; OR WITH (-FR) JUMPER OUT.
10158 005170 EXIT TST
(5) 005170 012767 000001 173762 MOV #1,$TIMES
(3) 005176 000473 BR TST24 ;;EXIT THIS TEST
10159 005200 ENDIF
(4) 005200 $77:
10160
10161 ;NO INTERRUPTS
10166 005200 012746 000300 MOV #PR6,-(SP) ;;PUT NEW PS ON STACK
(1) 005204 012746 005212 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
(1) 005210 000002 RTI ;;POP NEW PC AND PS
(1) 005212 64$:
10171
10172 ;START OFF WITH EVERYTHING CLEAR
10173 005212 ; BGNSUB
(5) 005212 012767 005220 173670 MOV #65$,$LPERR
10174
10175 ;CLEAR RING
10176 005220 LET @RCSR := @RCSR CLR.BY #REQSEND
(7) 005220 042777 000004 174032 BIC #REQSEND,@RCSR
10177 ;WAIT 1 MILLI-SEC FOR CABLE
10178 005226 WAITMS 1
(4) 005226 010546 MOV R5,-(SP)
(5) 005230 012745 000001 MOV #1,-(R5)
(4) 005234 004767 004506 JSR PC,WAIT
(4) 005240 012605 MOV (SP)+,R5
10179
10180 005242 ;READ ONCE
(4) 005242 017703 174012 MOV @RCSR,R3 LET R3 := @RCSR
10181 ;READ TWICE
10182 005246 IF #DATAINT SETIN @RCSR THEN
(6) 005246 032777 100000 174004 BIT #DATAINT,@RCSR
(9) 005254 001401 BEQ $100
10183 ;READING RCSR DID NOT CLEAR DATAINT
10184 005256 ERRHRD 65, EDATAINT
(1) 005256 104065 ERROR 65
10185 005260 ENDIF
(4) 005260 $100:
10186 005260 ENDSUB
10187

```

```

10188 ; SET RING --> SET DATAINT
10189 005260 ; BGNSUB
(5) 005260 012767 005266 173622 MOV #64$, $LPERR
10190
10191 ; WE ARE SETTING RING
10192 ; SET RING
10193 005266 LET @RCSR := @RCSR SET.BY #REQSEND
(7) 005266 052777 000004 173764 BIS #REQSEND, @RCSR
10194 ; WAIT 1 MILLI-SEC FOR CABLE
10195 005274 WAITMS 1
(4) 005274 010546 MOV R5, -(SP)
(5) 005276 012745 MOV #1, -(R5)
(4) 005302 004767 000001 JSR PC, WAIT
(4) 005306 012605 MOV (SP)+, R5
10196 IF #DATAINT NOTSETIN @RCSR THEN
(6) 005310 032777 100000 173742 BIT #DATAINT, @RCSR
(9) 005316 001001 BNE $101
10197 ; SETTING RING DID NOT SET DATAINT
10198 005320 ERRHRD 122,, E2DATA
(1) 005320 104122 ERROR 122
10199 005322 ENDIF
(4) 005322 $101:
10200 005322 ENDSUB
10201
10202 ; CLEAR RING CAUSES NO CHANGE IN DATAINT (CLEAR)
10203 005322 ; BGNSUB
(5) 005322 012767 005330 173560 MOV #64$, $LPERR
10204
10205 ; CLEAR RING
10206 005330 LET @RCSR := @RCSR CLR.BY #REQSEND
(7) 005330 042777 000004 173722 BIC #REQSEND, @RCSR
10207 ; WAIT 1 MILLI-SEC FOR CABLE
10208 005336 WAITMS 1
(4) 005336 010546 MOV R5, -(SP)
(5) 005340 012745 MOV #1, -(R5)
(4) 005344 004767 004376 JSR PC, WAIT
(4) 005350 012605 MOV (SP)+, R5
10209 IF #DATAINT SETIN @RCSR THEN
(6) 005352 032777 100000 173700 BIT #DATAINT, @RCSR
(9) 005360 001401 BEQ $102
10210 ; CLEARING RING SET DATAINT
10211 005362 ERRHRD 123, CLRRNG
(1) 005362 104123 ERROR 123
10212 005364 ENDIF
(4) 005364 $102:
10213 005364 ENDSUB
10214 005364 EXIT ; SKIP AROUND MESSAGE
(3) 005364 000400 BR TST24 ; EXIT THIS TEST
10215 005366 ENDTST
10216
10217
10218
  
```

```
10220
10225
10226
(3)
(3)
(2) 005366 000004
(2)
(1) 005370 012767 000010 173562
(2) 005376 012767 000024 173574
10227
10232 005404
(6) 005404 032767 020000 173606
(9) 005412 001004
10233
10234 005414
(5) 005414 012767 000001 173536
(3) 005422 000454
10235 005424
(4) 005424
10236
10237
10242 005424 012746 000300
(1) 005430 012746 005436
(1) 005434 000002
(1) 005436
10243
10248
10249
10250
10251 005436
(7) 005436 042777 000010 173614
10252 005444
(4) 005444 017703 173610
10253
10254
10255 005450
(5) 005450 012767 005456 173432
10256
10257
10258 005456
(7) 005456 052777 000010 173574
10259
10260 005464
(4) 005464 010546
(5) 005466 012745 000001
(4) 005472 004767 004250
(4) 005476 012605
10261 005500
(6) 005500 032777 100000 173552
(9) 005506 001001
10262
10263 005510
(1) 005510 104124
10264 005512
(4) 005512
10265 005512
```

```

*****
*****
*TEST 24 TEST THAT DATAINT SETS WHEN SECRC 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
$103:
MOV #PR6,-(SP) ;NO INTERRUPTS
MOV #64$,-(SP) ;;PUT NEW PS ON STACK
RTI ;;PUT NEW PC ON STACK
;;POP NEW PC AND PS
64$:
;START FRESH
;CLEAR SECRC
LET @RCSR := @RCSR CLR.BY #SECXMIT
LET R3 := @RCSR
;SET SECRC --> DATAINT SET
BGNSUB
MOV #65$,$LPERR
;SET SECRC
LET @RCSR := @RCSR SET.BY #SECXMIT
;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 SECRC DID NOT SET DATAINT
ERRHRD 124,, E2DATA
ENDIF
$104:
ENDSUB
```

```

10266
10267 ;CLEAR SECRC --> DATAINT SET
10268 005512 012767 005520 173370 MOV #64$, $LPERR BGNSUB
(5) 005512
10269 ;CLEAR SECRC
10270 005520 042777 000010 173532 BIC #SECXMIT, @RCSR LET @RCSR := @RCSR CLR.BY #SECXMIT
(7) 005520
10271 ;WAIT 1 MILLI-SEC FOR CABLE
10272 005526 010546 MOV R5, -(SP) WAITMS 1
(4) 005526 012745 000001 MOV #1, -(R5)
(5) 005530 004767 004206 JSR PC, WAIT
(4) 005540 012605 MOV (SP)+, R5
10273 005542 032777 100000 173510 BIT #DATAINT, @RCSR IF #DATAINT NOTSETIN @RCSR THEN
(6) 005542 001001 BNE $105 ;CLEARING SECRC DID NOT SET DATAINT
(9) 005550 ERRHRD 125,, E2DATA
10274
10275 005552 104125 ERROR 125
(1) 005552
10276 005554 $105: ENDF
(4) 005554
10277 005554 ENJSUB
10278 005554 ENDTST
10279
10280
10281
  
```

```
10283
10288
10293
(3)
(4)
(4)
(3)
(2) 005554 000004
(2)
(1) 005556 012767 000001 173374
(2) 005564 012767 000025 173406
10298
10299
10300 005572
(6) 005572 032767 000001 173414
(9) 005600 001404
10301 005602
(5) 005602 012767 000001 173350
(3) 005610 000454
10302 005612
(4) 005612
10303 005612
(5) 005612 012767 005620 173270
10304
10305
10306
10307
10308
10309
10310 005620
(4) 005620 105077 173444
10311
10312
10313
10314 005624
(3) 005624 010546
(7) 005626 012745 177777
(6) 005632 016745 173426
(5) 005636 012745 000200
(4) 005642 012745 000500
(3) 005646 004767 003616
(3) 005652 012605
10315
10316
10317 005654
(6) 005654 103001
10318
10319 005656
(1) 005656 104066
10320 005660
(4) 005660
10321 005660
10322
10323 005660
(5) 005660 012767 005666 173222
10324
```

```
*****
*****
*TEST 25 TEST THAT XMIT RDY - TCSR 7 - CLEARS
* WHFN TBUF IS LOADED WITH A CHARACTER
* AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.
*****
TST25: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #25,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
; THIS TEST IS 'BREAK OR HALT' SINSATIVE.
; IF #APTENV SETIN $ENV THEN
BIT #APTENV,$ENV
BEQ $106
EXIT TEST
MOV #1,$TIMES
BR TST26
;;;EXIT THIS TEST
ENDIF
$106:
BGNSUB
MOV #64,$LPERR
; LOAD TBUF WITH ONE CHARACTER
; WAIT FOR READY TO SET
; (SHOULD BE VERY SHORT WAIT
; SINCE UART DOUBLE BUFFERS ITS INPUT)
; SEND A CHARACTER
LET @TBUF :B= #0
;WAIT A MAXIMUM
;OF 50 MSEC FOR
;XMIT RDY TO SET IN TCSR
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
;TIMER RETURNS AN ERROR IF BIT DID
;NOT MEET CONDITION WITHIN TIME LIMIT
IF.ERROR THEN
;XMIT RDY DID NOT SET IN TCSR
ERRHRD 66,,DIDNOT
ENDIF
ENDSUB
BGNSUB
MOV #64,$LPERR
; LOAD TBUF WITH A SECOND CHARACTER
```

```

10325                                     ; CHECK IMMEDIATELY THAT XMITRDY IS CLEAR
10326                                     ; AND THEN WAIT FOR IT TO SET
10327
10328                                     ; SEND SECOND CHARACTER
10329 005666                               LET @TBUF :B= #0
    (4) 005666 105077 173376             CLRB @TBUF
10330 005672 000240                       NOP
10331                                     ; GIVE IT TIME TO CLEAR
10332                                     ; XMITRDY SHOULD HAVE CLEARED UPON
10333 005674                               ; RECEIPT OF A CHARACTER
    (6) 005674 032777 000200 173362     IF #XMITRDY SET IN @TCSR THEN
    (9) 005702 001401                     BIT #XMITRDY,@TCSR
                                           BEQ $110
10334                                     ; XMITRDY DID NOT CLEAR IN TCSR
10335 005704                               ERRHRD 67,,DIDNOT
    (1) 005704 104067                     ERROR 67
10336 005706                               ENDIF
    (4) 005706                               $110:
10337
10338                                     ; WAIT A MAXIMUM
10339                                     ; OF 50 MSEC FOR
10340                                     ; XMIT RDY TO SET IN TCSR
10341 005706                               CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
    (3) 005706 C10546                       MOV R5,-(SP)
    (7) 005710 012745 177777             MOV #SET,-(R5)
    (6) 005714 016745 173344             MOV TCSR,-(R5)
    (5) 005720 012745 000200             MOV #XMITRDY,-(R5)
    (4) 005724 012745 000500             MOV #500,-(R5)
    (3) 005730 004767 003534             JSR PC,TIMER
    (3) 005734 012605                     MOV (SP)+,R5
10342 005736                               IF.ERROR THEN
    (6) 005736 103001                     BCC $111
10343
10344 005740                               ; XMIT RDY DID NOT SET IN TCSR
    (1) 005740 104070                     ERROR 70
10345 005742                               ENDIF
    (4) 005742                               $111:
10346 005742
10347 005742
                                           ENDSUB
                                           ENDTST
  
```

```

10349
10354
10359
(3)
(4)
(4)
(3)
(2) 005742 000004
(2)
(1) 005744 012767 000010 173206
(2) 005752 012767 000026 173220
10364
10365
10366 005760
(7) 005760 052777 000004 173276
10367
10368 005766
(5) 005766 012767 005774 173114
10369
10370
10371 005774
(4) 005774 105077 173270
10372
10373
10374
10375
10376 006000
(3) 006000 010546
(7) 006002 012745 177777
(6) 006006 016745 173246
(5) 006012 012745 000200
(4) 006016 012745 000500
(3) 006022 004767 003442
(3) 006026 012605
10377
10378
10379 006030
(6) 006030 103001
10380
10381 006032
(1) 006032 104071
10382 006034
(4) 006034
10383
10384 006034
10385
10386 006034
(5) 006034 012767 006042 173046
10387
10388
10389 006042
(1) 006042 000005
10390
10391 006044
(6) 006044 032777 000200 173206
(9) 006052 001401

```

```

*****
*****
*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
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,,DIDNOT
ENDIF
$112:
ENDSUB
BGNSUB
MOV #64,$LPERR
; NOW THAT IT IS SET SEE IF IT CAN BE RESET
; THIS ALSO WILL CLEAR THE MAINT. BIT
BRESET
RESET
IF #RCVRDONE SETIN @RCSR THEN
BIT #RCVRDONE,@RCSR
BEQ $113

```


10392
10393 006054
(1) 006054 104072
10394 006056
(4) 006056
10395 006056
10396 006056

ERROR 72
\$113:

: RCVRDONE DID NOT RESET IN RCSR.
ERRHRD 72,,DIDNOT
ENDIF
ENDSUB
ENDTST

```

10398
10403
10404
(3)
(3)
(2) 006056 000004
(2)
(1) 006060 012767 000010 173072      MOV #10,$TIMES      ;;DO 10 ITERATIONS
(2) 006066 012767 000027 173104      MOV #27,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10405
10410
10411 006074
(7) 006074 052777 000004 173162      BIS #MAINT,@TCSR   ; SET MAINT. BIT
10412 006102
(5) 006102 012767 006110 173000      MOV #64$, $LPERR   LET @TCSR := @TCSR SET.BY #MAINT
10413
10414
10415
10416
10417 006110
(4) 006110 105077 173154      CLRB @TBUF         BGNSUB
10418
10419
10420
10421 006114
(3) 006114 010546
(7) 006116 012745 177777      MOV R5,-(SP)       ; OUTPUT A CHARACTER
(6) 006122 016745 173132      MOV #SET,-(R5)     LET @TBUF :B= #0
(5) 006126 012745 000200      MCV RCSR,-(R5)    ; WAIT MAXIMUM OF 500 MSEC
(4) 006132 012745 000500      MOV #RCVRDONE,-(R5) ; FOR RCVRDONE TO SET IN
(3) 006136 004767 003326      JSR PC,TIMER      ; RCSR
(3) 006142 012605      MOV (SP)+,R5      CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
10422
10423 006144
(6) 006144 103001      BCC $114           ; DID IT BECAME READY?
10424
10425 006146
(1) 006146 104073      ERROR 73          IF .ERROR THEN
10426 006150
(4) 006150
10427 006150
10428
10429
10430
10431
10432
10433 006150
(4) 006150 117700 173106      MOVB @RBUF,R0     ;RCVRDONE DID NOT SET IN RCSR
10434
10435 006154
(6) 006154 032777 000200 173076    BIT #RCVRDONE,@RCSR ;ERRHRD 73,, DIDNOT
(9) 006162 001401
10436
10437 006164
(1) 006164 104074      ERROR 74          ENDIF
10438 006166
  
```

MAINDEC-ZZ-CNDVA-A MACY11 30(1046) 16-DEC-82 15:13 PAGE 179-1^{G 6}
CNDVAA.F11 16-DEC-82 15:12 T27 TEST THAT RCVRDONE IS CLEARED BY READING RBUF

SEQ 0071

(4) 006166
10439 006166

\$115:

ENDTST

```
10441
10446
10451
(3)
(4)
(4)
(3)
(2) 006166 000004
(2)
(1) 006170 012767 000010 172762
(2) 006176 012767 000030 172774
10456
10457
10458 006204
(6) 006204 032767 000001 173002
(9) 006212 001404
10459 006214
(5) 006214 012767 000001 172736
(3) 006222 000500
10460 006224
(4) 006224
10461 006224
(7) 006224 052777 000004 173032
10462 006232
(4) 006232 012700 000000
10463 006236
(4) 006236 005001
10464
10465
10466
10467
10468 006240
(4) 006240 105077 173024
10469 006244
(3) 006244
10470 006244
(6) 006244 032777 004000 173006
(9) 006252 001403
10471 006254
(4) 006254 012700 177777
10472 006260
(4) 006260 000401
(3) 006262
10473 006262
(7) 006262 005201
10474 006264
(4) 006264
10475 006264
(4) 006264 020027 177777
(6) 006270 001403
(4) 006272 020167 000124
(7) 006276 101762
(4) 006300
10476 006300
(6) 006300 020167 000116
(9) 006304 101407

*****
*****
*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
BIS #MAINT,@TCSR
LET RO := #CLR
MOV #CLR,RO
LET R1 := #0
CLR R1
;LOAD A CHARACTER INTO TBUF
;WAIT FOR RCVRACT TO SET
;SEND A CHARACTER
LET @TBUF :B= #0
REPEAT
IF #RCVRACT SETIN @RCSR THEN
LET RO := #SET
ELSE
LET R1 := R1 + #1
ENDIF
UNTIL RO EQ #SET OR R1 HI MAX
$117:
BIT #RCVRACT,@RCSR
BEQ $120
LET RO := #SET
MOV #SET,RO
ELSE
BR $121
$120:
INC R1
ENDIF
$121:
UNTIL RO EQ #SET OR R1 HI MAX
$122:
CMP RO,#SET
BEQ $122
CMP R1,MAX
BLOS $117
$123:
IF R1 HI MAX THEN
CMP R1,MAX
BLOS $123
```

```

10477 ;IT NEVER SET
10478 ;RCVRCT DID NOT SET IN RCSR.
10479 006306 104075 ERROR 75 ERRHRD 75,, DIDNOT
(1) 006306
10480 006310 017700 172746 MOV @RBUF,RO LET RO := @RBUF ; CLEAR BUFFER
(4) 006310
10481 006314 012767 000001 172636 MOV #1,$TIMES EXIT TEST
(5) 006314
(3) 006322 000440 BR TST31 ;;;EXIT THIS TEST
10482 006324 $123: ENDF
(4) 006324
10483
10484 ;CHECK FOR TIMING OF RCVRCT. CLEARING
10485 ;VS RCVRDONE SETTING
10486
10487
10488
10489 006324 $124: WHILE #RCVRCT SETIN @RCSR DO
(4) 006324
(6) 006324 032777 004000 172726 BIT #RCVRCT,@RCSR
(9) 006332 001416 BEQ $125
10490
10491 006334 IF #RCVRDONE SETIN @RCSR THEN
(6) 006334 032777 000200 172716 BIT #RCVRDONE,@RCSR
(9) 006342 001411 BEQ $126
10492 006344 IF #RCVRCT SETIN @RCSR THEN
(6) 006344 032777 004000 172706 BIT #RCVRCT,@RCSR
(9) 006352 001405 BEQ $127
10493 ;RCVRDONE AND RCVRCT
10494 ;BOTH SET
10495 006354 104076 ERROR 76 ERRHRD 76, DONEACT
(1) 006354
10496 ;NO USE CONTINUING
10497 006356 012767 000001 172574 MOV #1,$TIMES EXIT TST
(5) 006356
(3) 006364 000417 BR TST31 ;;;EXIT THIS TEST
10498 006366 $127: ENDF
(4) 006366 $126: ENDF
10499 006366 $126: ENDDO
(4) 006366
10500 006366 000756 $125: BR $124
(4) 006366
(3) 006370
10501
10502 ;RCVRCT = 0 NCW.
10503 006370 032777 000200 172662 BIT #RCVRDONE,@RCSR IF #RCVRDONE NOTSETIN @RCSR THEN
(6) 006370
(9) 006376 001001 BNE $130
10504 ;RCVRDONE DID NOT SET IN RCSR
10505 006400 104077 ERROR 77 ERRHRD 77,,DIDNOT
(1) 006400
10506 ;SET IT BACK.
10507 006402 $130: ENDF
(4) 006402
10508 ;TEST THAT READING THE RECEIVER
10509 ;BUFFER CLEARS RCVRDONE

```

```
10510
10511
10512
10513 006402          :READ CHAR.
      (4) 006402 017700 172654      MOV   @RBUF,R0      LET R0 := @RBUF
10514
10515 006406          IF #RCVRDONE SETIN @RCSR THEN
      (6) 006406 032777 000200 172644 BIT   #RCVRDONE,@RCSR
      (9) 006414 001401          BEQ   $131
10516
10517 006416          :RCVRDONE DID NOT CLEAR IN RCSR
      (1) 006416 104100          ERROR 100      ERRHRD 100,,DIDNOT
10518 006420          ENDIF
      (4) 006420          $131:
10519
10520 006420          EXIT
      (3) 006420 000401          BR    TST31      :::EXIT THIS TEST
10521 006422 070000          MAX:70000
10522
10523 006424          ENDTST
10524
```

```

10526
10531
10532
10533
(3)
(3)
(2) 006424 000004
(2)
(1) 006426 012767 000010 172524
(2) 006434 012767 000031 172536
10538
10539 006442
(5) 006442 012767 006450 172440
10540
10541
10542
10543
10544
10545 006450
(4) 006450 105077 172614
10546
10547 006454
(4) 006454 010546
(5) 006456 012745 000310
(4) 006462 004767 003260
(4) 006466 012605
10548
10549
10550 006470
(4) 006470 105077 172574
10551
10552 006474
(4) 006474 010546
(5) 006476 012745 000310
(4) 006502 004767 003240
(4) 006506 012605
10553
10554
10555 006510
(4) 006510 017704 172546
10556
10557
10558 006514
(6) 006514 032704 040000
(9) 006520 001005
10559
10560 006522
(1) 006522 104101
10561
10562
10563 006524
(5) 006524 012767 000001 172426
(3) 006532 000456
10564 006534
(4) 006534
10565 006534

```

```

*****
*****
*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,$SLPERR
;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
;NO USE COMPOUNDING ERRORS
EXIT TST
;;EXIT THIS TEST
ENDIF
$132:
ENDSUB

```

```
10566
10567 ;NOW SEE IF ERROR BIT SET WITH OVERRUN ERROR:
10568 006534 012767 006542 172346 MOV #64$, $LPERR BGNSUB
(5) 006534
10569 006542 032704 100000 BIT #ERROR, R4 IF #ERROR NOTSET IN R4 THEN
(6) 006542
(9) 006546 001005 BNE $133
10570
10571 ;ERROR DID NOT SET IN RBUF
10572 006550 104102 ERROR 102 ERRHRD 102,,DIDNOT
(1) 006550
10573
10574 ;-WHEN ORERR SET.
10575 ;GET OUT NOW.
10576 006552 012767 000001 172400 MOV #1, $TIMES EXIT TST
(5) 006552
(3) 006560 000443 BR TST32 ;;;EXIT THIS TEST
10577 006562 $133: ENDF
(4) 006562
10578 006562 ENDSUB
10579
10580 006562 012767 006570 172320 MOV #64$, $LPERR BGNSUB
(5) 006562 ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
10581
10582 IF #ORERR NOTSET IN @RBUF THEN
10583 006570 032777 040000 172464 BIT #ORERR, @RBUF
(6) 006570
(9) 006576 001002 BNE $134
10584
10585 ;READING RBUF CLEARED ORERR.
10586 006600 104103 ERROR 103 ERRHRD 103,ITCLRED
(1) 006600
10587 ;SKIP REST OF TEST
10588 006602 000432 BR TST32 EXIT ;;;EXIT THIS TEST
(3) 006602
10589 006604 $134: ENDF
(4) 006604
10590 006604 ENDSUB
10591
10592 006604 012767 006612 172776 MCV #64$, $LPERR BGNSUB
(5) 006604 ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVED
10593
10594 ;SEND A CHARACTER AROUND.
10595 LET @TBUF :B= #0
10596 006612 105077 172452 CLRB @TBUF
(4) 006612
10597 ;LET IT CIRCULATE
10598 006616 010546 MOV R5, -(SP) WAITMS 200.
(4) 006616
(5) 006620 012745 000310 MOV #200, -(R5)
(4) 006624 004767 003116 JSR PC, WAIT
(4) 006630 012605 MOV (SP)+, R5
10599
10600 006632 032777 040000 172422 BIT #ORERR, @RBUF IF #ORERR SET IN @RBUF THEN
(6) 006632
(9) 006640 001405 BEQ $135
```



```

10601                                     ;ORERR DID NOT CLEAR IN RBUF
10602 006642                               ERRHRD 104,,DIDNOT
  (1) 006642 104104                       ERROR 104
10603
10604                                     ;-AFTER RECEIVING ANOTHER CHAR
10605                                     ;SKIP AROUND REST
10606 006644                               EXIT TST
  (5) 006644 012767 000001 172306         MOV #1,$TIMES
  (3) 006652 000406                       BR TST32                               :::EXIT THIS TEST
10607 006654                               ENDIF
  (4) 006654                               $135:
10608
10609 006654                               IF #ERROR SETIN @RBUF THEN
  (6) 006654 032777 100000 172400         BIT #ERROR,@RBUF
  (9) 006662 001401                       BEQ $136
10610                                     ;ERROR DID NOT CLEAR IN RBUF
10611 006664                               ERRHRD 105,,DIDNOT
  (1) 006664 104105                       ERROR 105
10612
10613 006666                               ENDIF
  (4) 006666                               $136:
10614 006666
10615 006666                               ENDSUB
  (3) 006666 000400                       BR TST32                               EXIT
10616                                     :::EXIT THIS TEST
10617 006670                               .EVEN
10618                                     ENDTST
  
```

```

10620
10621
10626
10632
(3)
(4)
(4)
(4)
(3)
(2) 006670 000004
(2)
(1) 006672 012767 000010 172260      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
(2) 006700 012767 000032 172272      MOV    #32,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10637 006706                                IF #PBR NOTSETIN $USWR THEN
(6) 006706 032767 000200 172304      BIT    #PBR,$USWR
(9) 006714 001004                                BNE    $137
10638 006716                                EXIT TST
(5) 006716 012767 000001 172234      MOV    #1,$TIMES
(3) 006724 000552                                BR     TST33          ;;:EXIT THIS TEST
10639 006726                                ENDIF
(4) 006726                                $137:
10640                                ; THIS TEST IS 'BREAK OR HALT' SINSATIVE.
10641 006726                                IF #APTENV SETIN $ENV THEN
(6) 006726 032767 000001 172260      BIT    #APTENV,$ENV
(9) 006734 001404                                BEQ    $140
10642 006736                                EXIT TEST
(5) 006736 012767 000001 172214      MOV    #1,$TIMES
(3) 006744 000542                                BR     TST33          ;;:EXIT THIS TEST
10643 006746                                ENDIF
(4) 006746                                $140:
10644
10645 006746                                LET OLD := #-i
(4) 006746 012767 177777 000272      MOV    #-1,OLD
10646 006754                                LET OLD+2 := #-1
(4) 006754 012767 177777 000266      MOV    #-1,OLD+2
10647 006762                                LET @TCSR := @TCSR SET.BY #MAINT
(7) 006762 052777 000004 172274      BIS    #MAINT,@TCSR
10648
10649 006770                                ;EACH BAUD RATE
(4) 006770 005003                                INCR R3 FROM #0 TO #15. BY #1
(5) 006772 000401                                CLR    R3
(4) 006774                                $142:      BR     $141
(8) 006774 005203                                $141:      INC    R3
(5) 006776                                $143:      CMP    R3,#15.
(7) 007002 003060                                BGT    $143
10650 007004                                LET R0 := @RBUF
(4) 007004 017700 172252                                MOV    @RBUF,R0
10651
10652 007010                                ;CHANGE BAUDE RATE
(4) 007010 116377 007170 172250      MOVB   RATES(R3),@TCSRHI
10653
10654 007016                                ;FLAG
(4) 007016 005002                                CLR    BIT := #0
10655
10656 007020                                ;OUTPUT THE CHARACTER
LET @TBUF := #0

```

```

(4) 007020 005077 172244 CLR @TBUF
10657
10658 007024
(4) 007024 005067 000212 CLR NEW
10659 007030
(4) 007030 005067 000210 CLR NEW+2
10660 007034
(4) 007034 $144:
(6) 007034 005702 TST BIT
(9) 007036 001014 BNE $145
10661 007040
(6) 007040 032777 000200 172212 BIT #RCVRDONE,@RCR
(9) 007046 001403 BEQ $146
10662
10663 007050
(4) 007050 012702 000001 MOV #1,BIT
10664 007054
(4) 007054 000404 BR $147
(3) 007056 $146:
10665
10666 007056
(7) 007056 005267 000160 INC NEW
10667 007062
(7) 007062 005567 000156 ADC NEW+2
10668 007066
(4) 007066 $147:
10669
10670 007066
(4) 007066 000762 BR $144
(3) 007070 $145:
10671
10672 007070
(6) 007070 026767 000150 000152 CMP NEW+2,OLD+2
(9) 007076 103001 BHIS $150
10673
10674 007100
(4) 007100 000412 BR $151
(3) 007102 $150:
10675
10676 007102
(6) 007102 026767 000136 000140 CMP NEW+2,OLD+2
(9) 007110 001005 BNE $152
(6) 007112 026767 000124 000126 CMP NEW,OLD
(9) 007120 103001 BHIS $152
10677
10678 007122
(4) 007122 000401 BR $153
(3) 007124 $152:
10679
10680
10681
10682
10683 007124
(1) 007124 104126 ERROR 126
10684 007126
(4) 007126 $153:

```

```

;INITIALIZE COUNTER
LET NEW := #0
LET NEW+2 := #0
WHILE BIT EQ #0 DO
    IF #RCVRDONE SETIN @RCR THEN
        ;DONE - ITS READY
        LET BIT := #1
    ELSE
        ;OTHERWISE-INCREMENT TIME
        LET NEW := NEW + #1
        LET NEW+2 := NEW+2 + CARRY
    ENDIF
ENDDO
;SIGNALS DONE
IF NEW+2 LO OLD+2 THEN
    ; OK
ELSE
    ; NEW+2 >= OLD+2
    IF NEW+2 EQ OLD+2 AND NEW LO OLD THEN
        ;OK
    ELSE
        ;NEW+2 > OLD+2 OR
        ;(NEW+2 = OLD+2 AND
        ; NEW >= OLD)
        ;BAUD RATE DIDN'T CHANGE
        ERRHR^ 126 BAUDRATE
    ENDIF

```

```

10685 007126                                ENDIF
      (4) 007126                                $151:
10686                                     ;UPDATE OLD TIME
10687 007126 016767 000110 000112           MOV    NEW,OLD
      (4) 007126 016767 000110 000112           LET OLD := NEW
10688 007134 016767 000104 000106           MOV    NEW+2,OLD+2
      (4) 007134 016767 000104 000106           LET OLD+2 := NEW+2
10689
10690 007142                                ENDINC ;BAUD RATE
      (4) 007142 000714                                $143: BR    $142
      (3) 007144
10691 007144 116703 172051                   MOVB   $USWR+1,R3
      (4) 007144 110346 000017                   MOVB   R3,-(SP)
      (7) 007150 142716 000017                   BICB   #17,(SP)
      (7) 007152 142603 000017                   BICB   (SP)+,R3
10692 007160 116377 007170 172100           MOVB   RATES(R3),@TCSRHI
      (4) 007160 116377 007170 172100           LET @TCSRHI := RATES(R3) ; LIKE HE WANTED IT
10693
10694 007166 000431                           BR    TST33
      (3) 007166 000431                           EXIT   ;SKIP TABLE
10695                                     ;:::EXIT THIS TEST
10696 007170
10697
10698
10699
10700
10701
10702
10703
10704
10705 007170 010                               R0050: .BYTE 010
10706 007171 030                               R0070: .BYTE 030
10707 007172 050                               R0110: .BYTE 050
10708 007173 070                               R0135: .BYTE 070
10709 007174 110                               R0150: .BYTE 110
10710 007175 130                               R0300: .BYTE 130
10711 007176 150                               R0600: .BYTE 150
10712 007177 170                               R0200: .BYTE 170
10713 007200 210                               R1800: .BYTE 210
10714 007201 230                               R2000: .BYTE 230
10715 007202 250                               R2400: .BYTE 250
10716 007203 270                               R3600: .BYTE 270
10717 007204 310                               R4800: .BYTE 310
10718 007205 330                               R7200: .BYTE 330
10719 007206 350                               R9600: .BYTE 350
10720 007207 370                               R10000: .BYTE 370
10721
10722 007210 040502 042125 051040           BAUDRATE: .ASCIZ /BAUD RATE DIDN'T CHANGE./
      007216 052101 020105 044504
      007224 047104 052047 041440
      007232 040510 043516 027105
      007240 000
10723 007242
10724 007242 000000 000000           .EVEN
10725 007246 000000 000000           NEW: 0.0
      OLD: 0.0

```

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 CORRESPONDING
 ;** BAUD TO BE SELECTED IN THE DLV11-E UPON
 ;** COMPLETION OF THIS TEST.

	BAUD	OFFSET INTO TABLE
R0050: .BYTE	010	0
R0070: .BYTE	030	1
R0110: .BYTE	050	2
R0135: .BYTE	070	3
R0150: .BYTE	110	4
R0300: .BYTE	130	5
R0600: .BYTE	150	6
R0200: .BYTE	170	7
R1800: .BYTE	210	10
R2000: .BYTE	230	11
R2400: .BYTE	250	12
R3600: .BYTE	270	13
R4800: .BYTE	310	14
R7200: .BYTE	330	15
R9600: .BYTE	350	16
R10000: .BYTE	370	17

10726 007252
10727
10728
10729

ENDTST

```

10731
10736
10744
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 007252 000004
(2)
(1) 007254 012767 000010 171676      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
(2) 007262 012767 000033 171710      MOV    #33,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10749                                     ;;CLEAR 'INTERRUPT OCCURED' FLAG
10750 007270                                     LET INTFLAG := #0
(4) 007270 005067 002540      CLR    INTFLAG
10751
10752                                     ;GET VECTOR ADDRESS
10753 007274                                     LET R3 := DLVEC
(4) 007274 016703 171756      MOV    DLVEC,R3
10754                                     ;FOR THE TRANSMITTER
10755 007300                                     LET R3 := R3 + #4
(7) 007300 062703 000004      ADD    #4,R3
10756                                     ;SET VECTOR TO POINT TO TRANS.SRV AT PRI
10757 007304                                     SETVEC R3, #INTSRV, #PR6
(5) 007304 010146      MOV    R1,-(SP)
(5) 007306 010301      MCV   R3,R1
(5) 007310 012721 012026      MOV    #INTSRV,(R1)+
(5) 007314 012711 000300      MOV    #PR6,(R1)
(5) 007320 012601      MOV    (SP)+,R1
10758 007322                                     BGNSUB
(5) 007322 012767 007330 171560      MOV    #64$,$LPERR
10759
10760 007330                                     ;CLEAR INTERRUPT ENABLE
(7) 007330 042777 000100 171726      BIC    #XMITIE,@TCSR      LET @TCSR := @TCSR CLR.BY #XMITIE
10761
10766                                     ;SET IT TO 0
10767 007336 012746 000000      MOV    #PRO,-(SP)      ;;PUT NEW PS ON STACK
(1) 007342 012746 007350      MOV    #65$,-(SP)     ;;PUT NEW PC ON STACK
(1) 007346 000002      RTI                    ;;POP NEW PC AND PS
(1) 007350
10772
10773                                     ;NOW SET I.E. BIT
10774 007350                                     LET @TCSR := @TCSR SET.BY #XMITIE
(7) 007350 052777 000100 171706      BIS    #XMITIE,@TCSR
10775
10776                                     ;LET INTERRUPT HAVE TIME TO OCCUR
10777 007356                                     WAITMS 200.
(4) 007356 010546      MOV    R5,-(SP)
(5) 007360 012745 000310      MOV    #200,-(R5)
(4) 007364 004767 002356      JSR   PC,WAIT
(4) 007370 012605      MOV    (SP)+,R5
10778
10779
10780 007372                                     ;DID EXACTLY 1 INTERRUPT OCCUR
                                     IF INTFLAG NE #1 THEN

```

```

*****
*****
*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

65\$:

```

(6) 007372 026727 002436 000001      CMP      INTFLAG,#1
(9) 007400 001406                      BEQ      $154
10781                                     ;NO - WAS IT 0 OR MORE THAN ONCE
10782 007402                                IF INTFLAG EQ #0 THEN
(6) 007402 005767 002426      TST      INTFLAG
(9) 007406 001002                      BNE      $155
10783                                     ;TRANSMITTER DID NOT INTERRUPT IN TIME
10784 007410                                ERRHRD 106,,DIDNOT
(1) 007410 104106      ERROR      106
10785 007412                                ELSE
(4) 007412 000401                      BR       $156
(3) 007414                                $155:
10786                                     ;TWICE
10787                                     ;TRANSMITTER INTERRUPTED TWICE
10788 007414                                ERRHRD 107,,TWICE
(1) 007414 104107      ERROR      107
10789 007416                                ENDIF
(4) 007416                                $156:
10790 007416                                $154:
(4) 007416                                $154:
10791 007416                                ENDSUB
10792                                     ;INTERRUPT WITHOUT INTERRUPT ENABLE SET
10793 007416                                BGNSUB
(5) 007416 012767 007424 171464      MOV      #64$,$LPERR
10794                                     ;CLEAR 'INTERRUPT OCCURED' FLAG
10795 007424                                LET INTFLAG := #0
(4) 007424 005067 002404      CLR      INTFLAG
10796                                     ;CLEAR INTERRUPT ENABLE
10797 007430                                LET @TCSR := @TCSR CLR.BY #XMITIE
(7) 007430 042777 000100 171626      BIC      #XMITIE,@TCSR
10798                                     ;NO INTERRUPTS SHOULD OCCUR.
10803 007436 012746 000000      MOV      #PRO,-(SP)      ;;PUT NEW PS ON STACK
(1) 007442 012746 007450      MOV      #65$,-(SP)      ;;PUT NEW PC ON STACK
(1) 007446 000002      RTI      ;;POP NEW PC AND PS
(1) 007450                                65$:
10808                                     ;DARE IT TO HAPPEN
10809 007450                                WAITMS 2
(4) 007450 010546      MOV      R5,-(SP)
(5) 007452 012745 000002      MOV      #2,-(R5)
(4) 007456 004767 002264      JSR      PC,WAIT
(4) 007462 012605      MOV      (SP)+,R5
10810 007464                                IF INTFLAG NE #0 THEN
(6) 007464 005767 002344      TST      INTFLAG
(9) 007470 001401                      BEQ      $157
10811                                     ;INTERRUPT OCCURED WITH I E CLEARED
10812 007472                                ERRHRD 110,NOTENAB
(1) 007472 104110      ERROR      110
10813 007474                                ENDIF
(4) 007474                                $157:
10814 007474                                BRESET
(1) 007474 000005      RESET
10815 007476                                ENDSUB
10816                                     ;RESTORE VECTOR AREA
10817 007476                                CLRVEC R3
(3) 007476 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
(3) 007500 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK

```

```
(5) 007502 012701 000003      MOV    #R3,R1
(5) 007506 010102              MOV    R1,R2
(8) 007510 062702 000002      ADD    #2,R2
(5) 007514 010221              MOV    R2,(R1)+
(5) 007516 005011              CLR    (R1)
(3) 007520 012602              MOV    (SP)+,R2      ;;POP STACK INTO R2
(3) 007522 012601              MOV    (SP)+,R1      ;;POP STACK INTO R1
10818
10819 007524                      ENDTST
10820
10821
10822
10823
10824
10825
```



```

10874 007664                                WAITMS 1
  (4) 007664 010546                                MOV    R5,-(SP)
  (5) 007666 012745 000001                        MOV    #1,-(R5)
  (4) 007672 004767 002050                        JSR    PC,WAIT
  (4) 007676 012605                                MOV    (SP)+,R5
10875 007700                                LET RO := @RBUF ; CLEAR RCVRDONE
  (4) 007700 017700 171356                        MOV    @RBUF,RO
10876
10877
10878 007704                                ;DID HE DO IT RIGHT?
  (6) 007704 026727 002124 000001                IF INTFLAG NE #1 THEN
  (9) 007712 001406                                CMP    INTFLAG,#1
                                                BEQ    $160
10879
10880 007714                                ;NONE OCCURED
  (6) 007714 005767 002114                        IF INTFLAG EQ #0 THEN
  (9) 007720 001002                                TST   INTFLAG
                                                BNE   $161
10881
10882 007722                                ;RECEIVER DID NOT INTERRUPT IN TIME
  (1) 007722 104111                                ERRHRD 111,,DIDNOT
10883
10884 007724                                ;TWICE OR MORE
  (4) 007724 000401                                ELSE
  (3) 007726                                BR     $162
10885
10886 007726                                ;RECEIVER INTERRUPTED TWICE
  (1) 007726 104112                                ERRHRD 112,,TWICE
10887
10888 007730                                ENDIF
  (4) 007730                                $162:
10889 007730                                $160:
10890 007730                                ;RESET MAINT. BIT.
  (7) 007730 042777 000004 171326                LET @TCSR := @TCSR CLR.BY #MAINT
10891
10892 007736                                ; CLEAR INTERRUPT ENABLE
  (7) 007736 042777 000100 171314                LET @RCSR := @RCSR CLR.BY #RCVRIE
10893 007744                                ENDSUB
10894
10895
10896
10897
10898
10899
10900
10901                                ;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-DATAIE
10902 007744                                BGNSUB
  (5) 007744 012767 007752 171136                MOV    #64$, $LPERR
10903 007752                                IF #CABLE NOTSETIN $USWR THEN
  (6) 007752 032767 020000 171240                BIT    #CABLE,$USWR
  (9) 007760 001004                                BNE   $163
10904
10905 007762                                ;CAN'T TEST WITHOUT A CABLE
  (5) 007762 012767 000001 171170                EXIT TST
  (3) 007770 000466                                BR     TST35
10906 007772                                ;:::EXIT THIS TEST
  (4) 007772                                $163:                                ENDF

```

```

10907                                     ; CLEAR 'INTFLAG'
10908 007772                               LET INTFLAG := #0
   (4) 007772 005067 002036                CLR    INTFLAG
10909
10910 007776                               ; CLEAR INTERRUPTS
   (7) 007776 042777 000040 171254        BIC    #DATAIE,@RCSR
10911
10912                                     ; CHANGE PRIORITY
10917 010004 012746 000000                MOV    #PRO,-(SP)      ;; PUT NEW PS ON STACK
   (1) 010010 012746 010016                MOV    #64$,-(SP)    ;; PUT NEW PC ON STACK
   (1) 010014 000002                          RTI                    ;; POP NEW PC AND PS
   (1) 010016
64$:
10922 010016                               LET @RCSR := @RCSR CLR.BY #REQSEND
   (7) 010016 042777 000004 171234        BIC    #REQSEND,@RCSR
10923
10924 010024                               ; SET INTERRUPT ENABLE
   (7) 010024 052777 000040 171226        BIS    #DATAIE,@RCSR
10925 010032                               LET @RCSR := @RCSR SET.BY #DATAIE
   (7) 010032 052777 000004 171220        BIS    #REQSEND,@RCSR
10926
10927 010040                               ; LET IT COME IN.
   (4) 010040 010546                          MOV    R5,-(SP)      WAITMS 1
   (5) 010042 012745 000001                MOV    #1,-(R5)
   (4) 010046 004767 001674                JSR    PC,WAIT
   (4) 010052 012605                MOV    (SP)+,R5
10928
10929
10930 010054                               ; DID IT DO IT RIGHT?
   (6) 010054 026727 001754 000001        CMP    INTFLAG,#1
   (9) 010062 001406                BEQ    $164
10931
10932 010064                               ; NONE OCCURED
   (6) 010064 005767 001744                TST    INTFLAG
   (9) 010070 001002                BNE    $165
10933
10934 010072                               ; DATAINT DID NOT INTERRUPT IN TIME
   (1) 010072 104113                ERROR  113          ERRHRD 113,,DIDNOT
10935
10936 010074                               ; TWICE OR MORE
   (4) 010074 000401                BR     $166          ELSE
   (3) 010076
$165:
10937
10938 010076                               ; DATAINT INTERRUPTED TWICE
   (1) 010076 104114                ERROR  114          ERRHRD 114,,TWICE
10939 010100
   (4) 010100                $166:
10940 010100
   (4) 010100                $164:
10941 010100
   (7) 010100 042777 000040 171152        BIC    #DATAIE,@RCSR
10942 010106
   (7) 010106 042777 000004 171144        BIC    #REQSEND,@RCSR
10943 010114
10944
10945 010114
   (4) 010114 017704 171136                MOV    @DLVEC,R4
  
```

```

; CLEAR 'INTFLAG'
LET INTFLAG := #0

; CLEAR INTERRUPTS
LET @RCSR := @RCSR CLR.BY #DATAIE

; CHANGE PRIORITY
; TO 0
;; PUT NEW PS ON STACK
;; PUT NEW PC ON STACK
;; POP NEW PC AND PS

LET @RCSR := @RCSR CLR.BY #REQSEND

; SET INTERRUPT ENABLE
LET @RCSR := @RCSR SET.BY #DATAIE

LET @RCSR := @RCSR SET.BY #REQSEND

; LET IT COME IN.
WAITMS 1

; DID IT DO IT RIGHT?
IF INTFLAG NE #1 THEN

; NONE OCCURED
IF INTFLAG EQ #0 THEN

; DATAINT DID NOT INTERRUPT IN TIME
ERRHRD 113,,DIDNOT

; TWICE OR MORE
ELSE

; DATAINT INTERRUPTED TWICE
ERRHRD 114,,TWICE

ENDIF

ENDIF

LET @RCSR := @RCSR CLR.BY #DATAIE
LET @RCSR := @RCSR CLR.BY #REQSEND

ENDSUB

LET R4 := @DLVEC
  
```

```

10946 010120
(3) 010120 010146
(3) 010122 010246
(5) 010124 012701 000004
(5) 010130 010102
(8) 010132 062702 000002
(5) 010136 010221
(5) 010140 005011
(3) 010142 012602
(3) 010144 012601
10947 010146

```

```

MOV R1,-(SP)      CLRVEC R4
MOV R2,-(SP)      ;;PUSH R1 ON STACK
MOV #R4,R1        ;;PUSH R2 ON STACK
MOV R1,R2
ADD #2,R2
MOV R2,(R1)+
CLR (R1)
MOV (SP)+,R2      ;;POP STACK INTO R2
MOV (SP)+,R1      ;;POP STACK INTO R1
ENDTST

```

```

10949
10954
10958
(3)
(4)
(3)
(2) 010146 000004
(2)
(1) 010150 012767 000001 171002
(2) 010156 012767 000035 171014
10963
10964 010164
(7) 010164 052777 000004 171072
10965
10966
10967
10972 010172 012746 000000
(1) 010176 012746 010204
(1) 010202 000002
(1) 010204
10977
10978 010204
(4) 010204 162705 000002
(3) 010210 004767 001432
(4) 010214 012501
10979
10980 010216
(4) 010216 017700 171040
10981
10982
10983 010222
(4) 010222 005002
(5) 010224 000401
(4) 010226
(8) 010226 005202
(5) 010230
(5) 010230 020227 000377
(7) 010234 003047
10984
10985
10986
10987
10988 010236
(3) 010236 010546
(7) 010240 012745 177777
(6) 010244 016745 171014
(5) 010250 012745 000200
(4) 010254 012745 000500
(3) 010260 004767 001204
(3) 010264 012605
10989
10990
10991 010266
(4) 010266 110277 170776
10992
10993 010272

```

```

*****
*****
*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
                   ;GET DATA MASK.
                   CALL DATLNG OUT <R1>
                   ; START CLEAN
                   LET R0 := @RBUF
                   ;ALL BINARY CHAR.
                   INCR R2 FROM #0 TO #377 BY #1
                   CLR R2
                   BR $167
                   INC R2
                   $170:
                   $167:
                   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>

```

```

(3) 010272 010546      MOV      R5,-(SP)
(7) 010274 012745 177777  MOV      #SET,-(R5)
(6) 010300 016745 170754  MOV      RCSR,-(R5)
(5) 010304 012745 000200  MOV      #RCVRDONE,-(R5)
(4) 010310 012745 000500  MOV      #500,-(R5)
(3) 010314 004767 001150  JSR      PC,TIMER
(3) 010320 012605      MOV      (SP)+,R5

10994
10995 010322                ;AND SAVE IT
(4) 010322 017703 170734  MOV      @RBUF,R3      LET R3 := @RBUF

10996
10997
10998
10999                ;COMPARE TO SEE IF WE RECEIVED IT ALL
11000                ;CLEAN OFF NON-DATA BITS
11001                ;ON BOTH TRANSMITTED AND
11002 010326                LET R4 := R2 CLR.BY R1
(4) 010326 010204      MOV      R2,R4
(7) 010330 040104      BIC      R1,R4

11003 010332                LET R3 := R3 CLR.BY R1
(7) 010332 040103      BIC      R1,R3

11004
11005                ;RECEIVED DATA
11006 010334                IF R4 NE R3 THEN
(6) 010334 020403      CMP      R4,R3
(9) 010336 001405      BEQ      $172

11007                ;DATA COMPARE ERROR
11008 010340                ERRHRD 116,COMP,SBWAS
(1) 010340 104116      ERROR    116
11009 010342                EXIT TST ; ON ERROR
(5) 010342 012767 000001 170610  MOV      #1,$TIMES
(3) 010350 000404      BR       TST36      :::EXIT THIS TEST
11010 010352                ENDIF
(4) 010352                $172:
11011 010352                ENDINC ; R2
(4) 010352 000725      BR       $170
(3) 010354                $171:

11012
11013
11014 010354                ;RESET MAINT. BIT.
(7) 010354 042777 000004 170702  BIC      #MAINT,@TCSR  LET @TCSR := @TCSR CLR.BY #MAINT
11015 010362                ENDTST
11016
11017
11018

```

11020
11025
11026
(3)
(3)
(2)
(2)
(1)
(2)
11031
(6)
(9)
11032
11033
(5)
(3)
11034
(4)
11035
11036
(7)
11037
11038
11043
(1)
(1)
(1)
11048
11049
(4)
(3)
(4)
11050
(4)
11051
11052
(4)
(5)
(4)
(8)
(5)
(5)
(7)
11053
11054
11055
11056
11057
(3)
(7)
(6)
(5)
(4)
(3)
(3)
11058

010362 000004
010364 012767 000001 170566
010372 012767 000036 170600
010400
010400 032767 020000 170612
010406 001004
010410
010410 012767 000001 170542
010416 000474
010420
010420
010420 042777 000004 170636
010426 012746 000000
010432 012746 010440
010436 000002
010440
010440
010440 162705 000002
010444 004767 001176
010450 012501
010452
010452 017700 170604
010456
010456 005002
010460 000401
010462
010462 005202
010464
010464 020227 000377
010470 003047
010472
010472 010546
010474 012745 177777
010500 016745 170560
010504 012745 000200
010510 012745 000500
010514 004767 000750
010520 012605

```
*****  
*****  
*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  
INC R2  
$174:  
CMP R2,#377  
BGT $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
```

```

11059
11060 010522
(4) 010522 110277 170542      MOVB   R2,@TBUF
11061 010526
(3) 010526 010546      MOV    R5,-(SP)
(7) 010530 012745 177777      MOV    #SET,-(R5)
(6) 010534 016745 170520      MOV    RCSR,-(R5)
(5) 010540 012745 000200      MOV    #RCVRDONE,-(R5)
(4) 010544 012745 000500      MOV    #500,-(R5)
(3) 010550 004767 000714      JSR    PC,TIMER
(3) 010554 012605      MOV    (SP)+,R5

11062
11063
11064 010556
(4) 010556 017703 170500      MOV    @RBUF,R3

11065
11066
11067 010562
(4) 010562 010204      MOV    R2,R4
(7) 010564 040104      BIC    R1,R4
11068 010566
(7) 010566 040103      BIC    R1,R3

11069
11070
11071 010570
(6) 010570 020403      CMP    R4,R3
(9) 010572 001405      BEQ    $177

11072
11073 010574
(1) 010574 104117      ERROR  117
11074 010576
(5) 010576 012767 000001 170354      MOV    #1,$TIMES
(3) 010604 000401      BR     TST37

11075 010606
(4) 010606      $177:

11076
11077 010606
(4) 010606 000725      $176: BR     $175
(3) 010610

11078
11079
11080
11081 010610
11082
11083
11084
11085
  
```

```

:START IT ON ITS WAY
LET @TBUF :B= R2
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
```



```

11087
11092
11096
(3)
(4)
(3)
(2) 010610 000004
(2)
(1) 010612 012767 000001 170340
(2) 010620 012767 000037 170352
11101
11102
11103
11104 010626
(6) 010626 032767 000001 170360
(9) 010634 001404
11105 010636
(5) 010636 012767 000001 170314
(3) 010644 000550
11106 010646
(4) 010646
11107
11108 010646
(4) 010646 162705 000002
(3) 010652 004767 000770
(4) 010656 012503
11109
11110
11111
11112
11113
11114
11115
11116
11117
11118
11119
11120
11121
11122
11123
11128 010660 012746 000000
(1) 010664 012746 010672
(1) 010670 000002
(1) 010672
11133
11134 010672
(4) 010672 016701 170360
11135
11136 010676
(4) 010676 012721 011072
11137 010702
(4) 010702 012721 000300
11138
11139
11140 010706

```

```

*****
*****
*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 SET IN $ENV THEN
BIT #APTENV,$ENV
BEQ $200
MOV #1,$TIMES EXIT TEST
BR TST40 ;;EXIT THIS TEST
ENDIF
$200:
CALL DATLNG OUT <R3> ;GET DATA MASK

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
;FCVR VECTOR
LET (R1)+ := #REC
LET (R1)+ := #PR6
;POINT TO TRANSMITTER VECTOR
;AND SET IT UP ALSO
LET (R1)+ := #TRAN

```

```

(4) 010706 012721 011030      MOV    #TRAN,(R1)+
11141 010712
(4) 010712 012711 000300      MOV    #PR6,(R1)
11142
11143
11144 010716 012701 177777      MOV    #-1,R1
(4) 010716
11145
11146 010722 005002      CLR    R2
(4) 010722
11147
11148 010724 012704 177777      MOV    #-1,R4
(4) 010724
11149
11150
11151 010730 005067 000066      CLR    ERRCNT
(4) 010730
11152
11153 010734 000005      RESET
(1) 010734
11154
11155 010736 052777 000004 170320      BIS    #MAINT,@TCSR
(7) 010736
11156
11157
11158 010744 052777 000100 170312      BIS    #XMITIE,@TCSR
(7) 010744
11159
11160 010752 052777 000100 170300      BIS    #RCVRIE,@RCR
(7) 010752
11161
11162
11163
11164 010760
(3) 010760
11165 010760 020467 000040      CMP    R4,NUMBER
(4) 010760
(6) 010764 001403      BEQ    $202
(4) 010766 005767 000030      TST   ERRCNT
(7) 010772 003772      BLE   $201
(4) 010774
11166
11167
11168 010774 005767 000022      TST   ERRCNT
(6) 010774
(9) 011000 001401      BEQ   $203
11169
11170 011002 104120      ERROR 120
(1) 011002
11171 011004
(4) 011004
11172
11173 011004 042777 000100 170252      BIC   #XMITIE,@TCSR
(7) 011004
11174 011012 042777 000100 170240      BIC   #XMITIE,@RCR
(7) 011012
  
```

```

LET (R1) := #PR6
; INITIALIZE COUNTERS
LET R1 := #-1
; RECEIVER STORAGE
LET R2 := #0
; # OF RECEIVED CHAR. COUNT.
LET R4 := #-1
; CLEAR ERROR COUNT.
LET ERRCNT := #0
BRESET ;SET UP ALL REGISTERS
;SET UP MAINTENANCE
LET @TCSR := @TCSR SET.BY #MAINT
;SET I.E. IN TRANSMITTER
LET @TCSR := @TCSR SET.BY #XMITIE
;AND RECEIVER
LET @RCR := @RCR SET.BY #RCVRIE
;NOW WE WAIT UNTIL R4 COUNT (RECEIVED) IS EQUAL
REPEAT
UNTIL R4 EQ NUMBER OR ERRCNT GT #0
;DATA COMPARE ERRORS.
IF ERRCNT NE #0 THEN
;DATA COMPARE ERROR
ERRHRD 120,COMP,FIRST
ENDIF
LET @TCSR := @TCSR CLR.BY #XMITIE
LET @RCR := @RCR CLR.BY #XMITIE
  
```

```

11175
11176 011020          BR      TST40          EXIT      :SKIP OVER SUPPORT ROUTINES & STORAGE
(3) 011020 000462          :::EXIT THIS TEST
11177
11178 011022 000000          ERRCNT: 0
11179 011024 000400          NUMBER: 400
11180 011026          000          SB:      .BYTE 0
11181 011027          000          WAS:     .BYTE 0
11182
11183
11184
11185 ;*****
11186 ;TRANSMIT INTERRUPT HANDLER
11187
11188 011030          BGNSRV  TRAN
(1) 011030
11189 TRAN:
11190 ;*****
11191 ;INCREMENT CHAR COUNT
11192 011030          LET R1 := R1 + #1
(7) 011030 005201          INC      R1
11193 ;SET UP FOR TRANSFER
11194 011032 010167 000030      MOV     R1,HOLD
(4) 011032 040367 000024      BIC     R3,HOLD
11195 ;AND SEND.
11196 011042          LET @TBUF := HOLD
(4) 011042 016777 000020 170220      MCV     HOLD,@TBUF
11197 ;ALL DONE
11198 011050          IF R1 EQ NUMBER THEN
(6) 011050 020167 177750      CMP     R1,NUMBER
(9) 011054 001003          BNE     $204
11199 ;STOP INTERRUPT PROCESSING
11200 011056          LET @TCSR := @TCSR CLR.BY #XMITIE
(7) 011056 042777 000100 170200      BIC     #XMITIE,@TCSR
11201 011064          ENDIF
(4) 011064          $204:
11202
11203 011064 000401          BR      ZZZ          ; EXIT SRV
11204
11205 011066 000000          HOLD:0
11206
11207 011070          ZZZ:          ENDSRV
(1) 011070 000002          RTI
11208
11209
11210 ;*****
11211 ;RECEIVER INTERRUPT HANDLER
11212          BGNSRV  REC
11213 011072          REC:
(1) 011072
11214 ;*****
11215
11216 ;COUNT THIS CHAR.
11217 011072          LET R4 := R4 + #1
(7) 011072 005204          INC     R4
  
```



```

11254
11259
11264
(3)
(4)
(4)
(3)
(2) 011166 000004
(2)
(1) 011170 012767 000010 167762      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
(2) 011176 012767 000040 167774      MOV    #40,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
11269 011204 032767 010000 170006      BIT    #BRK,$USWR     IF #BRK NOTSETIN $USWR THEN
(6) 011204 032767 010000 170006      BNE    $210
(9) 011212 001004
11270 011214 012767 000001 167736      MOV    #1,$TIMES
(5) 011214 012767 000001 167736      BR     TST41          ;;EXIT THIS TEST
(3) 011222 000452
11271 011224
(4) 011224      $210:
11272
11273 011224 052777 000004 170032      BIS    #MAINT,@TCSR   ;SET MAINTENANCE BIT
(7) 011224 052777 000004 170032      LET @TCSR := @TCSR SET.BY #MAINT
11274
11275
11276 011232 017700 170024      MOV    @RBUF,R0      ; CLEAR RCVRDONE JUST IN CASE
(4) 011232 017700 170024      LET R0 := @RBUF
11277
11278
11279
11280 011236 052777 000001 170020      BIS    #BREAK,@TCSR  ;SET BREAK BIT
(7) 011236 052777 000001 170020      LET @TCSR := @TCSR SET.BY #BREAK
11281
11282 011244 012777 000252 170016      MOV    #252,@TBUF    ;NON-ZERO CHAR. '*'
(4) 011244 012777 000252 170016      LET @TBUF := #252
11283 011252 010546
(3) 011252 010546      MOV    R5,-(SP)
(7) 011254 012745 177777      MOV    #SET,-(R5)
(6) 011260 016745 167774      MOV    RCSR,-(R5)
(5) 011264 012745 000200      MOV    #RCVRDONE,-(R5)
(4) 011270 012745 000500      MOV    #500,-(R5)
(3) 011274 004767 000170      JSR    PC,TIMER
(3) 011300 012605      MOV    (SP)+,R5
11284 011302
(6) 011302 103001      BCC    $211          IF.ERROR THEN
11285
11286 011304 104115      ERROR  115          ; RECIEVER DONE DID NOT SET
(1) 011304 104115      ERRHRD 115
11287 011306
(4) 011306      $211:
11288
11289 011306 105777 167750      TSTB  @RBUF          IFB @RBUF NE #0 THEN
(6) 011306 105777 167750      BEQ   $212
(9) 011312 001401
11290
11291 011314 104121      ERROR  121          ; BREAK DID NOT EQUAL 0
(1) 011314 104121      ERRHRD 121 ,BADBRK
  
```

```

*****
*****
*TEST 40      TEST BREAK GENERATION LOGIC
*              TRANSMIT KNOWN CHAR WITH BREAK SET
*              AND COMPARE RECEIVED WITH 0.
*****
TST40: SCOPE
  
```

```
11292 011316                                ENDIF
      (4) 011316                                $212:
11293 011316                                RESET
      (1) 011316 000005
11294 011320                                BR      TST41
      (3) 011320 000413
11295 011322 051102 040505 020113          :::EXIT THIS TEST
      011330 044504 020104 047516          BADBRK: .ASCIZ /BREAK DID NOT EQUAL 0/
      011336 020124 050505 040525
      011344 020114 000060
11296
11297 011350                                ENDTST
```

1

```

11299
11304
11305
(3)
(3)
(2) 011350 000004
(2)
(1) 011352 012767 000001 167600
11306 011360 104401 011366
(1) 011364 000404
(1)
(1) 011376
11307 011376 016746 167652
(1) 011402 104402
11308 011404 104401 011412
(1) 011410 000405
(1)
(1) 011424
11309 011424 016746 167626
(1) 011430 104402
11310 011432 104401 011440
(1) 011436 000405
(1)
(1) 011452
11311 011452 016746 167434
(1) 011456 104405
11312 011460 005067 167426
11313 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 ,67$ ;;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 ; RESET FOR NEXT DEVICE/PASS
JMP LOOP ; BACK UP TO THE BEGINNING
  
```

11319
11324
11325
11326
11331 011470
(2) 011470
11332
11333
11334
11335
11336
11337
11338
11339
11340
11341
11342
11343
11344
11345
11346
11347
11348
11349
11354
11359
11360
11361
11362
11363 011470
(4) 011470 016567 000004 000136
11364 011476
(4) 011476 016567 000000 000132
11365 011504
(4) 011504 112767 000000 000126
11366
11367
11368
11369
11370 011512
(3) 011512
11371
11372 011512
(6) 011512 036577 000002 000114
(9) 011520 001004
11373 011522
(4) 011522 112767 000000 000111
11374 011530
(4) 011530 000403
(3) 011532
11375 011532
(4) 011532 112767 177777 000101
11376 011540
(4) 011540
11377
11378

```

      ;: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.
* 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
*
* NOTE:: THE USE OF (R5) IS PART OF THE LINKAGE
*        MECHANISM BETWEEN THE CALLER AND THE CALLED
*:*****

```

```

      TRUE= 1
      FALSE= 0
      LET   REGSAV := REG(R5) ; GET POINTER TO REGIST
      LET   TIMSAV := HOWLONG(R5) ; SAVE HOWLONG FOR
      LET   FLAG :B= #FALSE ; INITIALIZE THE EXIT FLA
      ;
      ; START OF AN INFINITE LOOP
      LOOP
$215:
      ; TEST TO SEE IF WHICHBIT IS SET
      IF   WHICHBIT(R5) NC,SETIN @REGSAV THEN
      LET   HOLDSC :B= #CLR
      ELSE
      LET   HOLDSC :B= #SET ; REMEMBER THIS
      ENDIF
      ; NOW SEE IF THAT WAS WHAT WE WANTED
      MOVB  #FALSE,FLAG
      MOV   REG(R5),REGSAV
      LET   TIMSAV := HOWLONG(R5) ; SAVE HOWLONG FOR
      MOV   HOWLONG(R5),TIMSAV
      MOVB  #FALSE,FLAG
      ;
      ; START OF AN INFINITE LOOP
      LOOP
$215:
      BIT   WHICHBIT(R5),@REGSAV
      BNE   $217
      MOVB  #CLR,HOLDSC
      BR   $220
$217:
      MOVB  #SET,HOLDSC
$220:

```



```

11379 011540                                IFB    HOLDSC EQ SETCLR(R5) THEN
(6) 011540 126765 000075 000006          CMPB   HOLDSC,SETCLR(R5)
(9) 011546 001003                                BNE    $221
11380                                ; JUST THE THING WE NEEDED
11381 011550                                LET    FLAG :B= #TRUE
(4) 011550 112767 000001 000062          MOVB   #TRUE,FLAG
11382 011556                                ENDIF
(4) 011556                                $221:
11383
11384 011556                                EXIFB  FLAG EQ #TRUE OR TIMSAV LE #0
(4) 011556 126727 000056 000001          CMPB   FLAG,#TRUE
(6) 011564 001414                                BEQ    $216
(4) 011566 005767 000044                                TST   TIMSAV
(6) 011572 003411                                BLE    $216
11385                                ; ONE WAY OR THE OTHER, WE ARE DONE
11386                                ; IF WE ARE STILL HERE THEN HANG AROUND A WHILE
11387
11388 011574                                }
(4) 011574 010546                                MOV    R5,-(SP)
(5) 011576 012745 000001                                MOV    #1,-(R5)
(4) 011602 004767 000140                                JSR   PC,WAIT
(4) 011606 012605                                MOV    (SP)+,R5
11389 011610                                LET    TIMSAV := TIMSAV - #1 ; COUNTING DOWN
(7) 011610 005367 000022          DEC    TIMSAV
11390 011614                                ENDLOOP                                ; CONTINUED AT THE TOP
(4) 011614 000736                                BR     $215
(3) 011616                                $216:
11391
11392                                ; ONLY 2 WAYS TO GET HERE
11393                                ; 1). WE RAN OUT OF TIME---ERROR !!
11394                                ; 2). THE BIT IS IN THE CORRECT CONDITION--GOOD !!
11395
11396 011616                                IFB    FLAG EQ #TRUE THEN
(6) 011616 126727 000016 000001          CMPB   FLAG,#TRUE
(9) 011624 001001                                BNE    $222
11397 011626                                RETURN NO.ERROR ; GOOD
(4) 011626 000405                                BR     $213
11398 011630                                ENDIF
(4) 011630                                $222:
11399 011630                                RETURN ERROR ; BAD
(2) 011630 000261                                SEC
(4) 011632 000404                                BR     $214
11400
11401 011634 000000                                REGSAV: .WORD 0
11402 011636 000000                                TIMSAV: .WORD 0
11403 011640 000                                FLAG: .BYTE 0
11404 011641 000                                HOLDSC: .BYTE 0
11405                                ; WE ARE DONE GO BACK HOME
11406 011642                                ENDRTN
(3) 011642                                $213:
(2) 011642 000241                                CLC
(3) 011644                                $214:
(2) 011644 000207                                RTS    PC

```

11408
 11409
 11414
 11419 011646
 (2) 011646
 11420
 11421
 11422
 11423
 11424
 11425
 11426
 11427
 11428
 11429
 11430
 11435
 11440
 11441 011646
 (4) 011646 005065 000000
 11442 011652
 (4) 011652 016767 167342 000062
 (7) 011660 016746 000056
 (7) 011664 042716 000017
 (7) 011670 042667 000046
 11443
 11444 011674
 (4) 011674 012767 000001 167370
 (5) 011702 000402
 (4) 011704 \$226:
 (8) 011704 005267 167362 \$225:
 (5) 011710
 (5) 011710 026767 167356 000024
 (7) 011716 003006
 11445 011720
 (8) 011720 006365 000000
 11446 011724
 (7) 011724 052765 000001 000000
 11447 011732
 (4) 011732 000764
 (3) 011734 \$227:
 11448 011734
 (7) 011734 005165 000000
 11449 011740
 (4) 011740 000401
 11450 011742 000000
 11451 011744
 (3) 011744
 (3) 011744
 (2) 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
                                MOV $USWR,NUMBR
                                MOV NUMBR,-(SP)
                                BIC #DATA,(SP)
                                BIC (SP)+,NUMBR
                                INCR I FROM #1 TO NUMBR BY #1
                                MCV #1,I
                                BR $225
                                INC I
                                $225:
                                CMP I,NUMBR
                                BGT $227
                                LET MASK(R5) := MASK(R5) SHIFT #1
                                LET MASK(R5) := MASK(R5) SET.BY #1
                                ENDINC
                                LET MASK(R5) := COMP MASK(R5)
                                RETURN
                                BR $223
                                NUMBR:0
                                $223:
                                $224:
                                RTS PC
                                ENDRTN

```

11453
 11454
 11459
 11464 011746
 (2) 011746
 11465
 11466
 11467
 11468
 11469
 11470
 11475
 11476 011746 010146
 (2) 011750 010246
 (2) 011752 010346
 11481 011754
 (4) 011754 016501 000000
 11482 011760
 (4) 011760 012702 000001
 (5) 011764 000402
 (4) 011766
 (8) 011766 062702 000001
 (5) 011772
 (5) 011772 020201
 (7) 011774 101010
 11483 011776
 (4) 011776 005003
 (5) 012000 000401
 (4) 012002
 (8) 012002 005203
 (5) 012004
 (5) 012004 020327 000100
 (7) 012010 003001
 11484 012012
 (4) 012012 000773
 (3) 012014
 11485 012014
 (4) 012014 000764
 (3) 012016
 11490 012016 012603
 (2) 012020 012602
 (2) 012022 012601
 11495 012024
 (3) 012024
 (3) 012024
 (2) 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
  
```

11497
11502
11503
11504
11505 012026
11506
11507
11508
11509
11510
11515
11520
11521
11522 012026 005267 000002
(7) 012026
11523 012032
(1) 012032 000002
11524 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
RTI
INTFLAG: 0

;THAT'S ALL
```

```
11526
11527
11528 012036          ROUTINE MYTYPE
      (2) 012036      MYTYPE:
11533 *****
11534 012036 104401 012044      TYPE      ,65$      ;;TYPE ASCIZ STRING
      (1) 012042 000405      BR      64$      ;;GET OVER THE ASCIZ
      (1)      ;;65$: .ASCIZ <CRLF>*TEST # *
      (1) 012056      64$:
11535 012056 016746 167116      MOV      $TESTN,-(SP)      ;;SAVE $TESTN FOR TYPEOUT
      (1) 012062 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11536 012064 104401 012072      TYPE      ,67$      ;;TYPE ASCIZ STRING
      (1) 012070 000405      BR      66$      ;;GET OVER THE ASCIZ
      (1)      ;;67$: .ASCIZ *,ERROR # *
      (1) 012104      66$:
11537 012104 116767 167004 167064      MOVB     $ITEMB,$FATAL      ; APT FATAL ERROR NUMBER
11538 012112 016746 167060      MOV      $FATAL,-(SP)      ;;SAVE $FATAL FOR TYPEOUT
      (1) 012116 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11539 012120 104401 012126      TYPE      ,69$      ;;TYPE ASCIZ STRING
      (1) 012124 000404      BR      68$      ;;GET OVER THE ASCIZ
      (1)      ;;69$: .ASCIZ *,PC = *
      (1) 012136      68$:
11540 012136 016746 166754      MOV      $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
      (1) 012142 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11541 012144 104401 012152      TYPE      ,71$      ;;TYPE ASCIZ STRING
      (1) 012150 000404      BR      70$      ;;GET OVER THE ASCIZ
      (1)      ;;71$: .ASCIZ *,CSR: *
      (1) 012162      70$:
11542 012162 016746 167066      MOV      DLADD,-(SP)      ;;SAVE DLADD FOR TYPEOUT
      (1) 012166 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11543 012170 104401 012176      TYPE      ,73$      ;;TYPE ASCIZ STRING
      (1) 012174 000405      BR      72$      ;;GET OVER THE ASCIZ
      (1)      ;;73$: .ASCIZ *,VECTOR: *
      (1) 012210      72$:
11544 012210 016746 167042      MOV      DLVEC,-(SP)      ;;SAVE DLVEC FOR TYPEOUT
      (1) 012214 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11549 012216      ENDRTN
      (3) 012216      $240:
      (3) 012216      $241:
      (2) 012216 000207      RTS      PC
```

```

11551
11552 012220
(2) 012220
11557
11558
11559
11560
11561
11562
11563
11564
11569 012220
(3) 012220
11570 012220
(6) 012220 005767 000122
(9) 012224 001027
11571 012226
(6) 012226 026727 000116 000001
(9) 012234 001003
11572 012236
(4) 012236 005067 000106
11573 012242
(4) 012242 000403
(3) 012244
11574 012244
(3) 012244 004767 000110
11575
11576 012250
11577 012250
(4) 012250 012600
11578 012252
(4) 012252
11579 012252
(4) 012252 012767 000001 000066
11580 012260
(4) 012260 012767 000001 166716
11581 012266
(4) 012266 016767 166756 000056
11582 012274
(4) 012274 016767 166744 000052
11583 012302
(4) 012302 000410
(3) 012304
11584 012304
(4) 012304 012704 000010
11585 012310
(8) 012310 006167 000032
11586 012314
(7) 012314 060467 000032
11587 012320
(7) 012320 060467 000030
11588 012324
(4) 012324
11589 012324
(3) 012324 036767 000016 166720
(6) 012332 001732
  
```

```

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
          TST BITMASK
          BNE $245
          IF INITFLAG EQ #1 THEN
              CMP INITFLAG,#1
              BNE $246
              LET INITFLAG := #0
          ELSE
              CLR INITFLAG
          BR $247
$246:
      JSR PC,$EOP
      SPECIALADDRESS:
          ; BECAUSE $EOP RETURNS AS A JUMP
          LET RO := POP
          MOV (SP)+,RO
          ENDIF
$247:
      MOV #1,BITMASK
      LET $DEVCT := #1
      MOV #1,$DEVCT
      LET ADDRESS := $BASE
      MOV $BASE,ADDRESS
      LET VECTOR := $VECT1
      MOV $VECT1,VECTOR
      ELSE
          BR $250
$245:
      MOV #10,R4
      LET BITMASK := BITMASK ROTATE 1
      ROL BITMASK
      LET ADDRESS := ADDRESS + R4
      ADD R4,ADDRESS
      LET VECTOR := VECTOR + R4
      ENDIF
$250:
      UNTIL BITMASK SET IN $DEVM
      BIT BITMASK,$DEVM
      BEQ $244
  
```

```
11590
11591 012334          LET ADRS := #ADDRESS
(4) 012334 012701 012352      MOV #ADDRESS,ADRS
11592 012340          LET $DEVCT := $DEVCT + #1
(7) 012340 005267 166640      INC $DEVCT
11593 012344          RETURN
(4) 012344 000404      BR $242
11594 012346 000000      BITMASK: 0
11595 012350 000001      INITFLAG: 1
11596 012352 000000      ADDRESS: 0
11597 012354 000000      VECTOR: 0
11598
11599 012356          ENDRTN
(3) 012356
(3) 012356
(2) 012356 000207
11600
11601
```

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

\$242:
\$243:
RTS PC

11610
11611

.SBTTL POWER DOWN AND UP ROUTINES

```
(1)
(2)
(1)
(1) 012502 012737 012646 000024 $PWRDN: MOV #SILLUP,@PWRVEC ;;SET FOR FAST UP
(1) 012510 012737 000300 000026 MOV #PR6,@PWRVEC+2 ;;PRIO:6
(3) 012516 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 012520 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(3) 012522 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
(3) 012524 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
(3) 012526 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
(3) 012530 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
(3) 012532 017746 166402 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
(1) 012536 010667 000110 MOV SP,$SAVR6 ;;SAVE SP
(1) 012542 012737 012554 000024 MOV #SPWRUP,@PWRVEC ;;SET UP VECTOR
(1) 012550 000000 HALT
(1) 012552 000776 BR .-2 ;;HANG UP
(1)
(2)
(1)
(1) 012554 012737 012646 000024 $PWRUP: MOV #SILLUP,@PWRVEC ;;SET FOR FAST DOWN
(1) 012562 016706 000064 MOV $SAVR6,SP ;;GET SP
(1) 012566 005067 000060 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
(1) 012572 005267 000054 1$: INC $SAVR6 ;;WAIT FOR THE INC
(1) 012576 001375 BNE 1$ ;;OF WORD
(3) 012600 012677 166334 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
(3) 012604 012605 MCV (SP)+,R5 ;;POP STACK INTO R5
(3) 012606 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
(3) 012610 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
(3) 012612 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
(3) 012614 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
(3) 012616 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
(1) 012620 012737 012502 000024 MOV #SPWRDN,@PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 012626 012737 000300 000026 MOV #PR6,@PWRVEC+2 ;;PRIO:6
(1) 012634 104401 TYPE ;;REPORT THE POWER FAILURE
(1) 012636 012654 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
(1) 012640 012716 MOV (PC)+,(SP) ;;RESTART AT START
(1) 012642 001336 $PWRAD: .WORD START ;;RESTART ADDRESS
(1) 012644 000002 RTI
(1) 012646 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
(1) 012650 000776 BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
(1) 012652 000000 $SAVR6: 0 ;;PUT THE SP HERE
(1) 012654 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
(1) 012662 000122 .EVEN
```



```

(1) 013044 105367 000072          DECB  $CHARCNT      ;;DO NOT COUNT AS A COUNT
(1) 013050 000770          BR      7$          ;;LOOP
(1)
(1)          ;HORIZONTAL TAB PROCESSOR
(1) 013052 112716 000040      8$:  MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
(1) 013056 004767 000014      9$:  JSR   PC,$TYPEC    ;;TYPE A SPACE
(1) 013062 132767 000007 000052  BITB  #7,$CHARCNT    ;;BRANCH IF NOT AT
(1) 013070 001372          BNE   9$            ;;TAB STOP
(1) 013072 005726          TST   (SP)+        ;;POP SPACE OFF STACK
(1) 013074 000724          BR    2$            ;;GET NEXT CHARACTER
(1) 013076 105777 166046      $TYPEC: TSTB @ $TPS   ;;WAIT UNTIL PRINTER IS READY
(1) 013102 100375          BPL   $TYPEC
(1) 013104 116677 000002 166040  MOVB  2(SP),@$TPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)
(1) 013112 122766 000015 000002  CMPB  #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
(1) 013120 001003          BNE   1$          ;;BRANCH IF NO
(1) 013122 105067 000014          CLRB  $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
(1) 013126 000406          BR    $TYPEX     ;;EXIT
(1) 013130 122766 000012 000002  1$:  CMPB  #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
(1) 013136 001402          BEQ   $TYPEX     ;;BRANCH IF YES
(1) 013140 105227          INCB  (PC)+      ;;COUNT THE CHARACTER
(1) 013142 000000      $CHARCNT: .WORD  0  ;;CHARACTER COUNT STORAGE
(1) 013144 000207      $TYPEX: RTS      PC
(1)
(1)
(1)
  
```

11616
11617

.SBTTL TTY INPUT ROUTINE

::*****

.ENABL LSB

::*****

::SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
::ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
::SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
::WHEN OPERATING IN TTY FLAG MODE.

(1)	013146	022767	000176	165764	\$CKSWR:	CMP	#SWREG,SWR	::IS THE SOFT-SWR SELECTED?
(1)	013154	001074				BNE	15\$::BRANCH IF NO
(1)	013156	105777	165762			TSTB	@\$TKS	::CHAR THERE?
(1)	013162	100071				BPL	15\$::IF NO, DON'T WAIT AROUND
(1)	013164	117746	165756			MOVB	@\$TKB,-(SP)	::SAVE THE CHAR
(1)	013170	042716	177600			BIC	#^C177,(SP)	::STRIP-OFF THE ASCII
(1)	013174	022726	000007			CMP	#7,(SP)+	::IS IT A CONTROL G?
(1)	013200	001062				BNE	15\$::NO, RETURN TO USER
(1)	013202	126727	165726	000001		CMPB	\$AUTOB,#1	::ARE WE RUNNING IN AUTO-MODE?
(1)	013210	001456				BEQ	15\$::BRANCH IF YES
(1)	013212	104401	013673			TYPE	,\$CNTLG	::ECHO THE CONTROL-G (^G)
(1)	013216	104401	013700		\$GTSWR:	TYPE	,\$MSWR	::TYPE CURRENT CONTENTS
(2)	013222	016746	164750			MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
(2)	013226	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)	013230	104401	013711			TYPE	,\$MNEW	::PROMPT FOR NEW SWR
(1)	013234	005046			19\$:	CLR	-(SP)	::CLEAR COUNTER
(1)	013236	005046				CLR	-(SP)	
(1)	013240	105777	165700		7\$:	TSTB	@\$TKS	::CHAR THERE?
(1)	013244	100375				BPL	7\$::IF NOT TRY AGAIN
(1)	013246	117746	165674			MOVB	@\$TKB,-(SP)	::PICK UP CHAR
(1)	013252	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
(1)								
(1)	013256	021627	000025		9\$:	CMP	(SP),#25	::IS IT A CONTROL-U?
(1)	013262	001005				BNE	10\$::BRANCH IF NOT
(1)	013264	104401	013666			TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (^U)
(1)	013270	062706	000006		20\$:	ADD	#6,SP	::IGNORE PREVIOUS INPUT
(1)	013274	000757				BR	19\$::LET'S TRY IT AGAIN
(1)								
(1)	013276	021627	000015		10\$:	CMP	(SP),#15	::IS IT A <CR>?
(1)	013302	001022				BNE	16\$::BRANCH IF NO
(1)	013304	005766	000004			TST	4(SP)	::YES, IS IT THE FIRST CHAR?
(1)	013310	001403				BEQ	11\$::BRANCH IF YES
(1)	013312	016677	000002	165620		MOV	2(SP),@SWR	::SAVE NEW SWR
(1)	013320	062706	000006		11\$:	ADD	#6,SP	::CLEAR UP STACK
(1)	013324	104401	001171		14\$:	TYPE	,\$CRLF	::ECHO <CR> AND <LF>
(1)	013330	126727	165601	000001		CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
(1)	013336	001003				BNE	15\$::BRANCH IF NOT

```
(1) 013340 012777 000100 165576      MOV    #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1) 013346 000002      RTI                    ;;RETURN
(1) 013350 004767 177522      15$: JSR    PC,$TYPEC    ;;ECHO CHAR
(1) 013354 021627 000060      16$: CMP    (SP),#60     ;;CHAR < 0?
(1) 013360 002420      BLT    18$            ;;BRANCH IF YES
(1) 013362 021627 000067      CMP    (SP),#67     ;;CHAR > 7?
(1) 013366 003015      BGT    18$            ;;BRANCH IF YES
(1) 013370 042726 000060      BIC    #60,(SP)+    ;;STRIP-OFF ASCII
(1) 013374 005766 000002      TST    2(SP)        ;;IS THIS THE FIRST CHAR
(1) 013400 001403      BEQ    17$            ;;BRANCH IF YES
(1) 013402 006316      ASL    (SP)         ;;NO, SHIFT PRESENT
(1) 013404 006316      ASL    (SP)         ;;CHAR OVER TO MAKE
(1) 013406 006316      ASL    (SP)         ;;ROOM FOR NEW ONE.
(1) 013410 005266 000002      17$: INC    2(SP)        ;;KEEP COUNT OF CHAR
(1) 013414 056616 177776      BIS    -2(SP),(SP)  ;;SET IN NEW CHAR
(1) 013420 000707      BR     7$            ;;GET THE NEXT ONE
(1) 013422 104401 001170      18$: TYPE  $QUES      ;;TYPE ?<CR><LF>
(1) 013426 000720      BR     20$           ;;SIMULATE CONTROL-U
(1) .DSABL  LSB
```

```
(1) *****
(1) *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1) *CALL:
(1) * RDCHR                    ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1) * RETURN HERE              ;;CHARACTER IS ON THE STACK
(1) *                          ;;WITH PARITY BIT STRIPPED OFF
(1) *
```

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

```
(2) *****
(1) *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1) *CALL:
(1) * RDLIN                    ;;INPUT A STRING FROM THE TTY
(1) * RETURN HERE              ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1) *                          ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1) *
```

```

(1)
(1) 013550 010346 $RDLIN: MOV R3,-(SP) ;;SAVE R3
(1) 013552 012703 013656 1$: MOV #$TTYIN,R3 ;;GET ADDRESS
(1) 013556 022703 013666 2$: CMP #$TTYIN+8.,R3 ;;BUFFER FULL?
(1) 013562 101405 BLOS 4$ ;;BR IF YES
(1) 013564 104410 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
(1) 013566 112613 MOVB (SP)+,(R3) ;;GET CHARACTER
(1) 013570 122713 000177 10$: CMPB #177,(R3) ;;IS IT A RUBOUT
(1) 013574 001003 BNE 3$ ;;SKIP IF NOT
(1) 013576 104401 001170 4$: TYPE ,SQUES ;;TYPE A '?'
(1) 013602 000763 BR 1$ ;;CLEAR THE BUFFER AND LOOP
(1) 013604 111367 000044 3$: MOVB (R3),9$ ;;ECHO THE CHARACTER
(1) 013610 104401 013654 TYPE ,9$
(1) 013614 122723 000015 CMPB #15,(R3)+ ;;CHECK FOR RETURN
(1) 013620 001356 BNE 2$ ;;LOOP IF NOT RETURN
(1) 013622 105063 177777 CLR B -1(R3) ;;CLEAR RETURN (THE 15)
(1) 013626 104401 001172 TYPE ,SLF ;;TYPE A LINE FEED
(1) 013632 012603 MOV (SP)+,R3 ;;RESTORE R3
(1) 013634 011646 MOV (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1) 013636 016666 000004 000002 MOV 4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
(1) 013644 012766 013656 000004 MOV #$TTYIN,4(SP)
(1) 013652 000002 RTI ;;RETURN
(1) 013654 000 9$: .BYTE 0 ;;STORAGE FOR ASCII CHAR. TO TYPE
(1) 013655 000 .BYTE 0 ;;TERMINATOR
(1) 013656 000010 $TTYIN: .BLKB 8. ;;RESERVE 8 BYTES FOR TTY INPUT
(1) 013666 052536 005015 000 $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
(1) 013673 136 006507 000012 $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
(1) 013700 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
(1) 013706 020075 000
(1) 013711 040 047040 053505 $MNEW: .ASCIZ / NEW = /
(1) 013716 036440 000040

```


(1) 000200
(1) 000001
(1) 000100
(1) 000040

APTSIZE=00
APTENV=J01
APTSPOLL=100
APTCSUP=04U


```
11625  
11626  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1) 014370  
(1) 014370 104407  
(1) 014372 032777 040000 164540  
(1) 014400 001114  
(1)  
(1) 014402 000416  
(1)  
(1) 014404 013746 000004  
(1) 014410 012737 014430 000004  
(1) 014416 005737 177060  
(1) 014422 012637 000004  
(1) 014426 000463  
(1) 014430 C22626  
(1) 014432 012637 000004  
(1) 014436 000423  
(1) 014440  
(1) 014440 032777 000400 164472  
(1) 014446 001404  
(1) 014450 127767 164464 164424  
(1) 014456 001465  
(1) 014460 105767 164417  
(1) 014464 001421  
(1) 014466 126767 164423 164407  
(1) 014474 101015  
(1) 014476 032777 001000 164434  
(1) 014504 001404  
(1) 014506 016767 164376 164372  
(1) 014514 000446  
(1) 014516 105067 164361  
(1) 014522 005067 164432  
(1) 014526 000415  
(1) 014530 032777 004000 164402  
(1) 014536 001011  
(1) 014540 005767 164436  
(1) 014544 001406  
(1) 014546 005267 164332  
(1) 014552 026767 164402 164324  
(1) 014560 002024  
(1) 014562 012767 000001 164314  
(1) 014570 016767 000052 164362  
(1) 014576 105267 164300  
(1) 014602 116767 164274 164370  
(1) 014610 011667 164272
```

.SBTTL SCOPE HANDLER ROUTINE STARS
:*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
:*AND LOAD THE TEST NUMBER(\$TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
:*AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>
:*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:*SW14=1 LOOP ON TEST
:*SW11=1 INHIBIT ITERATIONS
:*SW09=1 LOOP ON ERROR
:*SW08=1 LOOP ON TEST IN SWR<7:0>
:*CALL
:* SCOPE ;;SCOPE=IOT

```
SSCOPE:  
CKSWR ;;TEST FOR CHANGE IN SOFT-SWR  
1$: BIT #BIT14,@SWR ;;LOOP ON PRESENT TEST?  
BNE $OVER ;;YES IF SW14=1  
;####START OF CODE FOR THE XOR TESTER####  
$XTSTR: BR 6$ ;;IF RUNNING ON THE 'XOR' TESTER CHANGE  
 ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)  
MOV @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR  
MOV #5$,@#ERRVEC ;;SET FOR TIMEOUT  
TST @#177060 ;;TIME OUT ON XOR?  
MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR  
BR $SVLAD ;;GO TO THE NEXT TEST  
5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT  
MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR  
BR 7$ ;;LOOP ON THE PRESENT TEST  
6$:;#####END OF CODE FOR THE XOR TESTER#####  
BIT #BIT08,@SWR ;;LOOP ON SPEC. TEST?  
BEQ 2$ ;;BR IF NO  
CMPB @SWR,$TSTNM ;;ON THE RIGHT TEST? SWR<7:0>  
BEQ $OVER ;;BR IF YES  
2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?  
BEQ 3$ ;;BR IF NO  
CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?  
BHI 3$ ;;BR IF NO  
BIT #BIT09,@SWR ;;LOOP ON ERROR?  
BEQ 4$ ;;BR IF NO  
7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE  
BR $OVER  
4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG  
CLF $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE  
BR 1$ ;;ESCAPE TO THE NEXT TEST  
3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?  
BNE 1$ ;;BR IF YES  
TST $PASS ;;IF FIRST PASS OF PROGRAM  
BEQ 1$ ;; INHIBIT ITERATIONS  
INC $ICNT ;;INCREMENT ITERATION COUNT  
CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE  
BGE $OVER ;;BR IF MORE ITERATION REQUIRED  
1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER  
MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO  
$SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS  
MOV $TSTNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX  
MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
```

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

11628
11629

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```
(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(3) 014650
(3) 014650 010046
(3) 014652 010146
(3) 014654 010246
(3) 014656 010346
(3) 014660 010546
(1) 014662 012746 020200
(1) 014666 016605 000020
(1) 014672 100004
(1) 014674 005405
(1) 014676 112766 000055 000001
(1) 014704 005000 1$: CLR R0
(1) 014706 012703 015064 MOV #20200,-(SP)
(1) 014712 112723 000040 MOV 20(SP),R5
(1) 014716 005002 2$: CLR R2
(1) 014720 016001 015054 MOV $DTBL(R0),R1
(1) 014724 160105 3$: SUB R1,R5
(1) 014726 002402 BLT 4$
(1) 014730 005202 INC R2
(1) 014732 000774 BR 3$
(1) 014734 060105 4$: ADD R1,R5
(1) 014736 005702 TST R2
(1) 014740 001002 BNE 5$
(1) 014742 105716 TSTB (SP)
(1) 014744 100407 BMI 7$
(1) 014746 106316 5$: ASLB (SP)
(1) 014750 103003 BCC 6$
(1) 014752 116663 000001 177777 MOVB 1(SP),-1(R3)
(1) 014760 052702 000060 6$: BIS #'0,R2
(1) 014764 052702 000040 7$: BIS #' ,R2
(1) 014770 110223 MOVB R2,(R3)+
(1) 014772 005720 TST (R0)+
(1) 014774 020027 000010 CMP R0,#10
(1) 015000 002746 BLT 2$
(1) 015002 003002 BGT 8$
(1) 015004 010502 MOV R5,R2
(1) 015006 000764 BR 6$
(1) 015010 105726 8$: TSTB (SP)+
(1) 015012 100003 BPL 9$
(1) 015014 116663 177777 177776 9$: MOVB -1(SP),-2(R3)
(1) 015022 105013 CLRB (R3)
(3) 015024 012605 MOV (SP)+,R5
(3) 015026 012603 MOV (SP)+,R3
```

```
*****
*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)     ;;MAKE THE ASCII NUMBER NEG.
1$:    CLR     R0      ;;ZERO THE CONSTANTS INDEX
MOV     #$DBLK,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
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
6$:    MOVB   1(SP),-1(R3) ;;YES--SET THE SIGN
BIS    #'0,R2       ;;MAKE THE BCD DIGIT ASCII
7$:    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
9$:    MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
CLRB   (R3)         ;;SET THE TERMINATOR
MOV    (SP)+,R5     ;;POP STACK INTO R5
MOV    (SP)+,R3     ;;POP STACK INTO R3
```

MAINDEC-22-CNDVA-A
CNDVAA.F11 16-DEC-82

MACY11 30(1046)
15:12

16-DEC-82 15:13 PAGE 203-1
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0121

(3)	015030	012602			MOV	(SP)+,R2	::POP STACK INTO R2
(3)	015032	012601			MOV	(SP)+,R1	::POP STACK INTO R1
(3)	015034	012600			MOV	(SP)+,R0	::POP STACK INTO R0
(1)	015036	104401	015064		TYPE	,SDBLK	::NOW TYPE THE NUMBER
(1)	015042	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
(1)	015050	012616			MOV	(SP)+,(SP)	
(1)	015052	000002			RTI		::RETURN TO USER
(1)	015054	023420			\$DTBL:	10000.	
(1)	015056	001750				1000.	
(1)	015060	000144				100.	
(1)	015062	000012				10.	
(1)	015064	000004			\$DBLK:	.BLKW 4	


```

(1) 015234 001403          BEQ      5$          ;;BR IF YES
(1) 015236 005204          4$: INC      R4          ;;DON'T SUPPRESS ANYMORE 0'S
(1) 015240 052703 000060  BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
(1) 015244 052703 000040  5$: BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
(1) 015250 110367 000040  MOVB     R3,8$        ;;SAVE FOR TYPING
(1) 015254 104401 015314  TYPE     ,8$        ;;GO TYPE THIS DIGIT
(1) 015260 105367 000032  7$: DECB   $OCNT      ;;COUNT BY 1
(1) 015264 003347          BGT      2$          ;;BR IF MORE TO DO
(1) 015266 002402          BLT      6$          ;;BR IF DONE
(1) 015270 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
(1) 015272 000744          BR       2$          ;;GO DO THE LAST DIGIT
(1) 015274 012605          6$: MOV     (SP)+,R5    ;;RESTORE R5
(1) 015276 012604          MOV     (SP)+,R4    ;;RESTORE R4
(1) 015300 012603          MOV     (SP)+,R3    ;;RESTORE R3
(1) 015302 016666 000002 000004 MOV     2(SP),4(SP) ;;SET THE STACK FOR RETURNING
(1) 015310 012616          MOV     (SP)+,(SP)
(1) 015312 000002          RTI          ;;RETURN
(1) 015314 000          8$: .BYTE  0          ;;STORAGE FOR ASCII DIGIT
(1) 015315 000          .BYTE  0          ;;TERMINATOR FOR TYPE ROUTINE
(1) 015316 000          $OCNT: .BYTE 0       ;;OCTAL DIGIT COUNTER
(1) 015317 000          $OFILL: .BYTE 0      ;;ZERO FILL SWITCH
(1) 015320 000000          $OMODE: .WORD 0     ;;NUMBER OF DIGITS TO TYPE
  
```

11634
11635

.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.

(1)	015322	010046		\$TRAP: MOV	RO,-(SP)	::SAVE RO	
(1)	015324	016600	000002		MOV	2(SP),RO	::GET TRAP ADDRESS
(1)	015330	005740			TST	-(RO)	::BACKUP BY 2
(1)	015332	111000			MOVB	(RO),RO	::GET RIGHT BYTE OF TRAP
(1)	015334	006300			ASL	RO	::POSITION FOR INDEXING
(1)	015336	016000	015356		MOV	\$TRPAD(RO),RO	::INDEX TO TABLE
(1)	015342	000200			RTS	RO	::GO TO ROUTINE

::THIS IS USE TO HANDLE THE "GETPRI" MACRO

(1)	015344	011646		\$TRAP2: MOV	(SP),-(SP)	::MOVE THE PC DOWN	
(1)	015346	016666	000004		MOV	4(SP),2(SP)	::MOVE THE PSW DOWN
(1)	015354	000002	000002		RTI		::RESTORE THE PSW

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
*BY THE "TRAP" INSTRUCTION.

(3)				:	ROUTINE		
(3)				:	-----		
(3)	015356	015344		\$TRPAD:	.WORD	\$TRAP2	
(3)	015360	012664			\$TYPE	::CALL=TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
(3)	015362	015120			\$TYPOC	::CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)	015364	015074			\$TYPOS	::CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)	015366	015134			\$TYPON	::CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)	015370	014650			\$TYPDS	::CALL=TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
(1)							
(3)	015372	013216		\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
(1)							
(3)	015374	013146		\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
(3)	015376	013430		\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
(3)	015400	013550		\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE

::THE FOLLOWING MACRO CALL TO CNMAC2.SML WAS ADDED TO INIT
::SBC11/21 SPECIFIC VECTORS

11636
11637
11638

(1)		015402		POINT=.		;SAVE POINTER
(1)		000100		.=100		
(1)	000100	015402		\$CLKVEC		;LKVEC HANDLER
(1)	000102	000300		300		;INTERRUPT HANDLER PRI
(1)		000140		.=140		;BRKVEC
(1)	000140	170000		170000		;ODT START ADDRESS
(1)	000142	000300		300		;PRIORITY
(1)		015402		.=POINT		;RESTORE POINTER
(1)	015402	104401	015410	\$CLKVEC:	TYPE,CLKMES	
(1)	015406	000000			HALT	
(1)	015410	005015	045514	042526	CLKMES: .ASCII?	<15><12>/LKVEC INTERRUPT - DISCONNECT LTC /

(1)	015416	020103	047111	042524	
(1)	015424	051122	050125	020124	
(1)	015432	020055	044504	041523	
(1)	015440	047117	042516	052103	
(1)	015446	046040	041524	000040	
11639		000001			.END

ABASE = 175610	7969#	9049					
ACDW1 = 000000	9049						
ACDW2 = 000000	9049						
ACPUOP= 000000	9049						
ADDRES 012352	11581*	11586*	11591	11596*			
ADDW0 = 000000	9049						
ADDW1 = 000000	9049						
ADDW10= 000000	9049						
ADDW11= 000000	9049						
ADDW12= 000000	9049						
ADDW13= 000000	9049						
ADDW14= 000000	9049						
ADDW15= 000000	9049						
ADDW2 = 000000	9049						
ADDW3 = 000000	9049						
ADDW4 = 000000	9049						
ADDW5 = 000000	9049						
ADDW6 = 000000	9049						
ADDW7 = 000000	9049						
ADDW8 = 000000	9049						
ADDW9 = 000000	9049						
ADEVCT= 000000	9049						
ADEVCT= 000001	7970#	9049					
AENV = 000000	9049						
AENVM = 000000	9049						
AFATAL= 000000	9049						
AMADR1= 000000	9049						
AMADR2= 000000	9049						
AMADR3= 000000	9049						
AMADR4= 000000	9049						
AMAMS1= 000000	9049						
AMAMS2= 000000	9049						
AMAMS3= 000000	9049						
AMAMS4= 000000	9049						
AMSGAD= 000000	9049						
AMSGLG= 000000	9049						
AMSGTY= 000000	9049						
AMTYP1= 000000	9049						
AMTYP2= 000000	9049						
AMTYP3= 000000	9049						
AMTYP4= 000000	9049						
APASS = 000000	9049						
APRIOR= 000000	9049						
APTC SU= 000040	11614	11620#					
APTENV= 000001	10300	10458	10641	11104	11614	11620#	11623
APTSIZ= 000200	9062	11620#					
APTSPO= 000100	11614	11620#					
ASWREG= 000000	9049						
ATESTN= 000000	9049						
AUNIT = 000000	9049						
AUSWR = 071110	7972#	9049					
AVECT1= 000300	7971#	9049					
AVECT2= 000000	9049						
BADBRK 011322	11295#						
BAUD = 007400	9040#						
BAUDRA 007210	10722#						

RATES	007170	10652	10692	10696#															
RBUF	001262	9054#	9079*	10433	10480	10513	10555	10583	10600	10609	10650	10875	10980	10995					
RCSR	001260	11050	11064	11219	11276	11289													
		9053#	9078*	9345*	9347	9356*	9357	9365*	9367	9398	9406*	9408	9417*	9419					
		9428*	9431	9455	9463*	9465	9474*	9476	9485*	9488	9512	9520*	9522	9531*					
		9533	9542*	9545	9569	9577*	9579	9588*	9590	9599*	9602	9636	9699	9738					
		9795*	9797	9809*	9811	9823*	9825	9868*	9870	9882*	9884	9896*	9898	9942*					
		9944	9956*	9958	9970*	9972	10014*	10016	10028*	10030	10042*	10044	10095*	10099					
		10101	10112*	10113	10118	10128*	10129	10176*	10180	10182	10193*	10196	10206*	10209					
		10251*	10252	10258*	10261	10270*	10273	10376	10391	10421	10435	10470	10489	10491					
		10492	10503	10515	10661	10852*	10870	10872*	10892*	10910*	10922*	10924*	10925*	10941*					
		10942*	10993	11061	11160*	11174*	11238*	11283											
RCVRAC=	004000	8951#	9738	10470	10489	10492													
RCVRDO=	000200	8955#	9636	10376	10391	10421	10435	10491	10503	10515	10661	10870	10993	11061					
		11283																	
RCVRIE=	000100	8956#	9569	9577	9579	9588	9590	9599	9602	10852	10872	10892	11160	11238					
RDATA0=	000001	8982#																	
RDATA1=	000002	8981#																	
RDATA2=	000004	8980#																	
RDATA3=	000010	8979#																	
RDATA4=	000020	8978#																	
RDATA5=	000040	8977#																	
RDATA6=	000100	8976#																	
RDATA7=	000200	8975#																	
RDCHR =	104410	11617	11635#																
RDLIN =	104411	11635#																	
REC	011072	11136	11213#																
REG =	000004	11331#	11363																
REGSAV	011634	11363*	11372	11401#															
REQSEN=	000004	8960#	9398	9406	9408	9417	9419	9428	9431	9942	9956	9970	10176	10193					
		10206	10922	10925	10942														
RESVEC=	000010	8927#																	
RHLD	011162	11221*	11224	11228	11245#														
RING =	040000	8948#	9944	9958	9972														
R0050	007170	10705#																	
R0070	007171	10706#																	
R0110	007172	10707#																	
R0135	007173	10708#																	
R0150	007174	10709#																	
R0200	007177	10712#																	
R0300	007175	10710#																	
R0600	007176	10711#																	
R10000	007207	10720#																	
R1800	007200	10713#																	
R2000	007201	10714#																	
R2400	007202	10715#																	
R3600	007203	10716#																	
R4800	007204	10717#																	
R5STAC	001334	9060#	9083																
R7200	007205	10718#																	
R9600	007206	10719#																	
SB	011026	11180#	11228*																
SECREC=	002000	8952#	10016	10030	10044														
SECXMI=	000010	8959#	9455	9463	9465	9474	9476	9485	9488	10014	10028	10042	10251	10258					
		10270																	
SET =	177777	8940#	10314	10341	10376	10421	10471	10475	10870	10988	10993	11057	11061	11283					

	11483#	11522#	11572#	11574#	11577#	11579#	11580#	11581#	11582#	11584#	11585#	11586#	11587#
\$ERROR 014170	11591#	11592#											
\$ERRPC 001116	9062	11623#											
\$ERRTB 001254	9049#	11540	11623*										
\$ERTTL 001112	9049#	11311	11312*	11623*									
\$ESCAP 001162	9049#	9062*	11623	11626*									
\$ETABL 001214	9049#												
\$ETEND 001254	9048	9049#											
\$FATAL 001176	9049#	11537*	11538	11620*									
\$FFLG 014166	11620#*												
\$FILLC 001156	9049#	11614											
\$FILLS 001155	9049#	11614											
\$FSAND= 000310	7962#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289
	9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
	9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
	9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
	9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470	10476	10489	10491
	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661	10672
	10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104
	11168	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571	
\$FSBAD= 000401	7962#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289
	9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
	9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
	9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
	9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470	10476	10489	10491
	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661	10672
	10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104
	11168	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571	
\$FSBLA= 000170	7962#												
\$FSCAS= 000150	7962#												
\$FSDEC= 000220	7962#												
\$FSGOO= 000400	7962#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289
	9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
	9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
	9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
	9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
	10209	10232	10261	10273	10300	10317	10333	10342	10379	10391	10423	10435	10458
	10470	10476	10489	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637
	10641	10660	10661	10672	10676	10780	10782	10810	10878	10880	10903	10930	10932
	11006	11031	11071	11104	11168	11198	11224	11226	11236	11269	11284	11289	11372
	11379	11396	11570	11571									
\$FSIF = 000110	7962#	9116	9119	9148	9150	9154	9157	9164	9167	9175	9178	9187	9190
	9211	9214	9221	9224	9232	9235	9244	9247	9279	9282	9289	9292	9300
	9303	9312	9315	9340	9342	9347	9350	9357	9360	9367	9370	9391	9393
	9398	9401	9408	9411	9419	9422	9431	9434	9455	9458	9465	9468	9476
	9479	9488	9491	9512	9515	9522	9525	9533	9536	9545	9548	9569	9572
	9579	9582	9590	9593	9602	9605	9636	9643	9668	9675	9699	9708	9729
	9732	9738	9749	9779	9784	9797	9802	9811	9816	9825	9830	9854	9857
	9870	9875	9884	9889	9898	9903	9927	9931	9944	9949	9958	9963	9972
	9977	10000	10003	10016	10021	10030	10035	10044	10049	10075	10079	10101	10104
	10113	10116	10118	10121	10129	10132	10155	10159	10182	10185	10196	10199	10209
	10212	10232	10235	10261	10264	10273	10275	10300	10302	10317	10320	10333	10336
	10342	10345	10379	10382	10391	10394	10423	10426	10435	10438	10458	10460	10470

MAINDEC-22-CNDVA-A
CNDVAA.F11 16-DEC-82

MACY11 30(1046)
15:12

16-DEC-82 15:13 PAGE 206-8
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0134

SF\$INC= 000210
SF\$LOO= 000200
SF\$NAM= 000160
SF\$NO = 000403

10472	10474	10476	10482	10491	10492	10498	10499	10503	10507	10515	10518	10558
10564	10569	10577	10583	10589	10600	10607	10609	10613	10637	10639	10641	10643
10661	10664	10668	10672	10674	10676	10678	10684	10685	10780	10782	10785	10789
10790	10810	10813	10878	10880	10884	10887	10888	10903	10906	10930	10932	10936
10939	10940	11006	11010	11031	11034	11071	11075	11104	11106	11168	11171	11198
11201	11224	11226	11230	11233	11236	11240	11269	11271	11284	11287	11289	11292
11372	11374	11376	11379	11382	11396	11398	11570	11571	11573	11578	11583	11588
7962#	10649	10690	10983	11011	11052	11077	11444	11447	11482	11483	11484	11485

SF\$OR = 000320

9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
10209	10232	10261	10273	10300	10333	10391	10435	10458	10470	10476	10489	10491
10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661	10672
10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104
11168	11198	11224	11226	11236	11269	11372	11570	11571				

SF\$RTN= 000300
SF\$SEL= 000140
SF\$UNT= 000130
SF\$WHI= 000120
SF\$YES= 000402

7962#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289
9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
10209	10232	10261	10273	10300	10333	10391	10435	10458	10470	10476	10489	10491
10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661	10672
10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104
11168	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571	

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

7962#	9109	9123	10469	10475	11164	11165	11569	11589				
7962#	10489	10500	10660	10670	10676							
7962#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289
9300	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465
9476	9488	9512	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699
9729	9738	9779	9797	9811	9825	9854	9870	9884	9898	9927	9944	9958
9972	10000	10016	10030	10044	10075	10101	10113	10118	10129	10155	10182	10196
10209	10232	10261	10273	10300	10333	10391	10435	10458	10470	10476	10489	10491
10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661	10672
10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104
11168	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571	

9049#												
9049#												
11608#												
11617#	11635											
8924												
9048#												
9049#	11626*											
7962#	9116#	9119#	9148#	9150#	9154#	9157#	9164#	9167#	9175#	9178#	9187#	9190#
9211#	9214#	9221#	9224#	9232#	9235#	9244#	9247#	9279#	9282#	9289#	9292#	9300#
9303#	9312#	9315#	9340#	9342#	9347#	9350#	9357#	9360#	9367#	9370#	9391#	9393#
9398#	9401#	9408#	9411#	9419#	9422#	9431#	9434#	9455#	9458#	9465#	9468#	9476#
9479#	9488#	9491#	9512#	9515#	9522#	9525#	9533#	9536#	9545#	9548#	9569#	9572#
9579#	9582#	9590#	9593#	9602#	9605#	9636#	9643#	9668#	9675#	9699#	9708#	9729#
9732#	9738#	9749#	9779#	9784#	9797#	9802#	9811#	9816#	9825#	9830#	9854#	9857#
9870#	9875#	9884#	9889#	9898#	9903#	9927#	9931#	9944#	9949#	9958#	9963#	9972#

MAINDEC-22-CNDVA-A
CNDVAA.F11

MACY11 30(1046)
16-DEC-82 15:12

16-DEC-82 15:13 PAGE 206-9
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0135

9977#	10000#	10003#	10016#	10021#	10030#	10035#	10044#	10049#	10075#	10079#	10101#	10104#
10113#	10116#	10118#	10121#	10129#	10132#	10155#	10159#	10182#	10185#	10196#	10199#	10209#
10212#	10232#	10235#	10261#	10264#	10273#	10276#	10300#	10302#	10317#	10320#	10333#	10336#
10342#	10345#	10379#	10382#	10391#	10394#	10423#	10426#	10435#	10438#	10458#	10460#	10470#
10474#	10476#	10482#	10491#	10492#	10498#	10499#	10503#	10507#	10515#	10518#	10558#	10564#
10569#	10577#	10583#	10589#	10600#	10607#	10609#	10613#	10637#	10639#	10641#	10643#	10661#
10668#	10672#	10676#	10684#	10685#	10780#	10782#	10789#	10790#	10810#	10813#	10878#	10880#
10887#	10888#	10903#	10906#	10930#	10932#	10939#	10940#	11006#	11010#	11031#	11034#	11071#
11075#	11104#	11106#	11168#	11171#	11198#	11201#	11224#	11226#	11230#	11233#	11236#	11240#
11269#	11271#	11284#	11287#	11289#	11292#	11372#	11376#	11379#	11382#	11396#	11398#	11570#
11571#	11578#	11588#										

\$ILLUP 012646
\$INTAG 001135
\$ISKO = 000001

11611#	9049#	11617										
9116#	9119	9148#	9150	9154#	9157	9164#	9167	9175#	9178	9187#	9190	9211#
9214	9221#	9224	9232#	9235	9244#	9247	9279#	9282	9289#	9292	9300#	9303
9312#	9315	9340#	9342	9347#	9350	9357#	9360	9367#	9370	9391#	9393	9398#
9401	9408#	9411	9419#	9422	9431#	9434	9455#	9458	9465#	9468	9476#	9479
9488#	9491	9512#	9515	9522#	9525	9533#	9536	9545#	9548	9569#	9572	9579#
9582	9590#	9593	9602#	9605	9636#	9643	9668#	9675	9699#	9708	9729#	9732
9738#	9749	9779#	9784	9797#	9802	9811#	9816	9825#	9830	9854#	9857	9870#
9875	9884#	9889	9898#	9903	9927#	9931	9944#	9949	9958#	9963	9972#	9977

10000#	10003	10016#	10021	10030#	10035	10044#	10049	10075#	10079	10101#	10104	10113#
10116	10118#	10121	10129#	10132	10155#	10159	10182#	10185	10196#	10199	10209#	10212
10232#	10235	10261#	10264	10273#	10276	10300#	10302	10317#	10320	10333#	10336	10342#
10345	10379#	10382	10391#	10394	10423#	10426	10435#	10438	10458#	10460	10470#	10474
10476#	10482	10491#	10499	10503#	10507	10515#	10518	10558#	10564	10569#	10577	10583#
10589	10600#	10607	10609#	10613	10637#	10639	10641#	10643	10661#	10668	10672#	10685
10780#	10790	10810#	10813	10878#	10888	10903#	10906	10930#	10940	11006#	11010	11031#
11034	11071#	11075	11104#	11106	11168#	11171	11198#	11201	11224#	11233	11236#	11240
11269#	11271	11284#	11287	11289#	11292	11372#	11376	11379#	11382	11396#	11398	11570#

\$ISK1 = 000001

11588	10492#	10498	10676#	10684	10782#	10789	10880#	10887	10932#	10939	11226#	11230	11571#
11578													

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

9049#	11537	11623*										
9049#	11614	11617	11623									
11620#*												
7962#	9109	9116	9119	9123	9148	9150	9154	9157	9164	9167	9175	9178
9187	9190	9211	9214	9221	9224	9232	9235	9244	9247	9279	9282	9289
9292	9300	9303	9312	9315	9340	9342	9347	9350	9357	9360	9367	9370
9391	9393	9398	9401	9408	9411	9419	9422	9431	9434	9455	9458	9465
9468	9476	9479	9488	9491	9512	9515	9522	9525	9533	9536	9545	9548
9569	9572	9579	9582	9590	9593	9602	9605	9636	9643	9668	9675	9699
9708	9729	9732	9738	9749	9779	9784	9797	9802	9811	9816	9825	9830
9854	9857	9870	9875	9884	9889	9898	9903	9927	9931	9944	9949	9958
9963	9972	9977	10000	10003	10016	10021	10030	10035	10044	10049	10075	10079
10101	10104	10113	10116	10118	10121	10129	10132	10155	10159	10182	10185	10196
10199	10209	10212	10232	10235	10261	10264	10273	10276	10300	10302	10317	10320
10333	10336	10342	10345	10379	10382	10391	10394	10423	10426	10435	10438	10458
10460	10469	10470	10472	10474	10475	10476	10482	10489	10491	10492	10498	10499
10500	10503	10507	10515	10518	10558	10564	10569	10577	10583	10589	10600	10607
10609	10613	10637	10639	10641	10643	10649	10660	10661	10664	10668	10670	10672
10674	10676	10678	10684	10685	10690	10780	10782	10785	10789	10790	10810	10813
10878	10880	10884	10887	10888	10903	10906	10930	10932	10936	10939	10940	10983
11006	11010	11011	11031	11034	11052	11071	11075	11077	11104	11106	11164	11165
11168	11171	11198	11201	11224	11226	11230	11233	11236	11240	11269	11271	11284
11287	11289	11292	11331	11370	11372	11374	11376	11379	11382	11384	11390	11396

	11397	11398	11399	11406	11419	11444	11447	11449	11451	11464	11482	11483	11484
	11485	11495	11528	11549	11552	11569	11570	11571	11573	11578	11583	11588	11589
	11593	11599											
\$LPADR 001106	9049#	9062*	11626*										
\$LPERR 001110	9049#	9062*	9110*	9152*	9161*	9171*	9182*	9209*	9218*	9228*	9239*	9277*	9286*
	9296*	9307*	9344*	9354*	9364*	9396*	9405*	9415*	9426*	9453*	9462*	9472*	9482*
	9510*	9519*	9529*	9540*	9567*	9576*	9586*	9597*	9635*	9666*	9698*	9736*	9792*
	9806*	9820*	9865*	9879*	9893*	9939*	9953*	9967*	10011*	10025*	10039*	10093*	10109*
	10125*	10173*	10189*	10203*	10255*	10268*	10303*	10323*	10368*	10386*	10412*	10539*	10568*
\$LSTCN= 177777	10580*	10592*	10758*	10793*	10847*	10902*	11623	11626*					
	7962#	7965#	9109	9116	9119	9123	9148	9150	9154	9157	9164	9167	9175
	9178	9187	9190	9211	9214	9221	9224	9232	9235	9244	9247	9279	9282
	9289	9292	9300	9303	9312	9315	9340	9342	9347	9350	9357	9360	9367
	9370	9391	9393	9398	9401	9408	9411	9419	9422	9431	9434	9455	9458
	9465	9468	9476	9479	9488	9491	9512	9515	9522	9525	9533	9536	9545
	9548	9569	9572	9579	9582	9590	9593	9602	9605	9636	9643	9668	9675
	9699	9708	9729	9732	9738	9749	9779	9784	9797	9802	9811	9816	9825
	9830	9854	9857	9870	9875	9884	9889	9898	9903	9927	9931	9944	9949
	9958	9963	9972	9977	10000	10003	10016	10021	10030	10035	10044	10049	10075
	10079	10101	10104	10113	10116	10118	10121	10129	10132	10155	10159	10182	10185
	10196	10199	10209	10212	10232	10235	10261	10264	10273	10276	10300	10302	10317
	10320	10333	10336	10342	10345	10379	10382	10391	10394	10423	10426	10435	10438
	10458	10460	10469	10470	10472	10474	10475	10476	10482	10489	10491	10492	10498
	10499	10500	10503	10507	10515	10518	10558	10564	10569	10577	10583	10589	10600
	10607	10609	10613	10637	10639	10641	10643	10649	10660	10661	10664	10668	10670
	10672	10674	10676	10678	10684	10685	10690	10780	10782	10785	10789	10790	10810
	10813	10878	10880	10884	10887	10888	10903	10906	10930	10932	10936	10939	10940
	10983	11006	11010	11011	11031	11034	11052	11071	11075	11077	11104	11106	11164
	11165	11168	11171	11198	11201	11224	11226	11230	11233	11236	11240	11269	11271
	11284	11287	11289	11292	11331	11370	11372	11374	11376	11379	11382	11390	11396
	11398	11406	11419	11444	11447	11451	11464	11482	11483	11484	11485	11495	11528
	11549	11552	11569	11570	11571	11573	11578	11583	11588	11589	11599		
\$LSTIN= 000000	7962#	7963#	9068	9070	9073	9075	9076	9078	9079	9080	9081	9082	9083
	9105	9107	9108	9110	9112	9116	9121	9122	9123	9124	9148	9149	9152
	9154	9161	9162	9164	9171	9173	9175	9182	9184	9187	9209	9211	9218
	9219	9221	9228	9230	9232	9239	9241	9244	9277	9279	9286	9287	9289
	9296	9298	9300	9307	9309	9312	9340	9341	9344	9345	9347	9354	9356
	9357	9364	9365	9367	9391	9392	9396	9398	9405	9406	9408	9415	9417
	9419	9426	9428	9431	9453	9455	9462	9463	9465	9472	9474	9476	9482
	9485	9488	9510	9512	9519	9520	9522	9529	9531	9533	9540	9542	9545
	9567	9569	9576	9577	9579	9586	9588	9590	9597	9599	9602	9635	9636
	9666	9668	9698	9699	9729	9731	9736	9738	9779	9783	9792	9795	9797
	9806	9809	9811	9820	9823	9825	9854	9856	9865	9868	9870	9879	9882
	9884	9893	9896	9898	9927	9930	9939	9942	9944	9953	9956	9958	9967
	9970	9972	10000	10002	10011	10014	10016	10025	10028	10030	10039	10042	10044
	10075	10078	10093	10095	10097	10099	10101	10109	10112	10113	10118	10125	10128
	10129	10155	10158	10173	10176	10178	10180	10182	10189	10193	10195	10196	10203
	10206	10208	10209	10232	10234	10251	10252	10255	10258	10260	10261	10268	10270
	10272	10273	10300	10301	10303	10310	10314	10317	10323	10329	10333	10341	10342
	10366	10368	10371	10376	10379	10386	10391	10411	10412	10417	10421	10423	10433
	10435	10458	10459	10461	10462	10463	10468	10470	10471	10472	10473	10475	10476
	10480	10481	10489	10491	10492	10497	10500	10503	10513	10515	10539	10545	10547
	10550	10552	10555	10558	10563	10568	10569	10576	10580	10583	10592	10596	10598
	10600	10606	10609	10637	10638	10641	10642	10645	10646	10647	10649	10650	10652
	10654	10656	10658	10659	10660	10661	10663	10664	10666	10667	10670	10672	10674
	10676	10678	10687	10688	10690	10691	10692	10750	10753	10755	10757	10758	10760

MAINDEC-ZZ-CNDVA-A
CNDVAA.F11 16-DEC-82

MACY11 30(1046)
15:12

16-DEC-82 15:13 PAGE 206-13
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0139

\$NSK1 = 000130
\$NSK2 = 000110
\$NSK3 = 000110
\$NULL 001154
\$NWTST= 000001

\$SOCNT 015316
\$SOMODE 015320
\$SOVER 014632
\$SPASS 001202
\$SPASTM 001006
\$SPOWER 012654
\$SPWRAD 012642
\$SPWRDN C12502
\$SPWRMG 012636
\$SPWRUP 012554
\$SQUES 001170
\$SRDCHR 013430
\$SRDDEC= ***** U
\$SRDLIN 013550
\$SRDOCT= ***** U
\$SRDSZ = 000010
\$SRTNAD 012460
\$R2A = ***** U
\$SAVLE= 177777
\$SAVRE= ***** U
\$SAVR6 012652
\$SCOPE 014370
\$SETUP= 000137
\$SSKO = 000236
\$STUP = 177777
\$SVLAD 014576
\$SVPC = 000204
\$SWR = 167400

\$SWREG 001216
\$SWRMK= 000000
\$TAGLE= 177777

9116#	9119	10470#	10472	10474	10491#	10499	10660#	10670	10672#	10674	10685	10782#
10785	10789	10880#	10884	10887	10932#	10936	10939	11006#	11010	11071#	11075	11226#
11230	11370#	11384	11390	11396#	11398	11444#	11447	11482#	11485	11569#	11589	
10492#	10498	10661#	10664	10668	10676#	10678	10684	11372#	11374	11376	11379#	11382
11483#	11484	11570#	11583	11588								
11571#	11573	11578										
9049#	11614											
9100#	9142#	9202#	9260#	9335#	9386#	9447#	9504#	9561#	9625#	9657#	9689#	9722#
9773#	9848#	9921#	9994#	10069#	10149#	10226#	10293#	10359#	10404#	10451#	10533#	10632#
10744#	10838#	10958#	11026#	11096#	11264#	11305#						
11632#*												
11632#*												
11626#												
9049#	9062*	11608*	11626									
9048#												
11611#												
11611#												
9062	11611#											
11611#												
11611#												
9049#	11614	11617	11623									
11617#	11635											
11635												
11617#	11635											
11635												
11617#												
11608#												
11635												
7962#	10500#	10649#	10670#	10983#	11052#	11390#	11444#	11482#	11483#			
11635												
11611#*												
9062	11626#											
9061#	9062	9063	11608	11617	11623	11626						
10500#	10649#	10670#	10983#	11052#	11390#	11444#	11482#	11483#				
9061#												
11626#												
9046#												
7967#	8924	8925	9049	9062	9100	9142	9202	9260	9335	9386	9447	9504
9561	9625	9657	9689	9722	9773	9848	9921	9994	10069	10149	10226	10293
10359	10404	10451	10533	10632	10744	10838	10958	11026	11096	11264	11305	11608
11611	11623	11626										
9049#	9062											
8925	11626											
7962#	9109#	9116#	9119#	9123#	9148#	9150#	9154#	9157#	9164#	9167#	9175#	9178#
9187#	9190#	9211#	9214#	9221#	9224#	9232#	9235#	9244#	9247#	9279#	9282#	9289#
9292#	9300#	9303#	9312#	9315#	9340#	9342#	9347#	9350#	9357#	9360#	9367#	9370#
9391#	9393#	9398#	9401#	9408#	9411#	9419#	9422#	9431#	9434#	9455#	9458#	9465#
9468#	9476#	9479#	9488#	9491#	9512#	9515#	9522#	9525#	9533#	9536#	9545#	9548#
9569#	9572#	9579#	9582#	9590#	9593#	9602#	9605#	9636#	9643#	9668#	9675#	9699#
9708#	9729#	9732#	9738#	9749#	9779#	9784#	9797#	9802#	9811#	9816#	9825#	9830#
9854#	9857#	9870#	9875#	9884#	9889#	9898#	9903#	9927#	9931#	9944#	9949#	9958#
9963#	9972#	9977#	10000#	10003#	10016#	10021#	10030#	10035#	10044#	10049#	10075#	10079#
10101#	10104#	10113#	10116#	10118#	10121#	10129#	10132#	10155#	10159#	10182#	10185#	10196#
10199#	10209#	10212#	10232#	10235#	10261#	10264#	10273#	10276#	10300#	10302#	10317#	10320#
10333#	10336#	10342#	10345#	10379#	10382#	10391#	10394#	10423#	10426#	10435#	10438#	10458#
10460#	10469#	10470#	10472#	10474#	10475#	10476#	10482#	10489#	10491#	10492#	10498#	10499#

10500#	10503#	10507#	10515#	10518#	10558#	10564#	10569#	10577#	10583#	10589#	10600#	10607#
10609#	10613#	10637#	10639#	10641#	10643#	10649#	10660#	10661#	10664#	10668#	10670#	10672#
10674#	10676#	10678#	10684#	10685#	10690#	10780#	10782#	10785#	10789#	10790#	10810#	10813#
10878#	10880#	10884#	10887#	10888#	10903#	10906#	10930#	10932#	10936#	10939#	10940#	10983#
11006#	1100J#	11011#	11031#	11034#	11052#	11071#	11075#	11077#	11104#	11106#	11164#	11165#
11168#	11171#	11198#	11201#	11224#	11226#	11230#	11233#	11236#	11240#	11269#	11271#	11284#
11287#	11289#	11292#	11370#	11372#	11374#	11376#	11379#	11382#	11384#	11390#	11396#	11398#
11444#	11447#	11482#	11483#	11484#	11485#	11569#	11570#	11571#	11573#	11578#	11583#	11588#
11589#												
7962#	9109#	9116#	9148#	9154#	9164#	9175#	9187#	9211#	9221#	9232#	9244#	9279#
9289#	9300#	9312#	9340#	9347#	9357#	9367#	9391#	9398#	9408#	9419#	9431#	9455#
9465#	9476#	9488#	9512#	9522#	9533#	9545#	9569#	9579#	9590#	9602#	9636#	9668#
9699#	9729#	9738#	9779#	9797#	9811#	9825#	9854#	9870#	9884#	9898#	9927#	9944#
9958#	9972#	10000#	10016#	10030#	10044#	10075#	10101#	10113#	10118#	10129#	10155#	10182#
10196#	10209#	10232#	10261#	10273#	10300#	10317#	10333#	10342#	10379#	10391#	10423#	10435#
10458#	10469#	10470#	10472#	10475#	10476#	10489#	10491#	10492#	10503#	10515#	10558#	10569#
10583#	10600#	10609#	10637#	10641#	10649#	10660#	10661#	10664#	10672#	10674#	10676#	10678#
10780#	10782#	10785#	10810#	10878#	10880#	10884#	10903#	10930#	10932#	10936#	10983#	11006#
11031#	11052#	11071#	11104#	11164#	11165#	11168#	11198#	11224#	11226#	11236#	11269#	11284#
11289#	11331#	11370#	11372#	11374#	11379#	11396#	11419#	11444#	11464#	11482#	11483#	11528#
11552#	11569#	11570#	11571#	11573#	11583#							
9068#	9073#	9075#	9076#	9078#	9079#	9080#	9081#	9082#	9083#	9105#	9107#	9108#
9110#	9112#	9119#	9121#	9122#	9123#	9124#	9149#	9150#	9152#	9157#	9161#	9162#
9167#	9171#	9173#	9178#	9182#	9184#	9190#	9209#	9214#	9218#	9219#	9224#	9228#
9230#	9235#	9239#	9241#	9247#	9277#	9282#	9286#	9287#	9292#	9296#	9298#	9303#
9307#	9309#	9315#	9341#	9342#	9344#	9345#	9350#	9354#	9356#	9360#	9364#	9365#
9370#	9392#	9393#	9396#	9401#	9405#	9406#	9411#	9415#	9417#	9422#	9426#	9428#
9434#	9453#	9458#	9462#	9463#	9468#	9472#	9474#	9479#	9482#	9485#	9491#	9510#
9515#	9519#	9520#	9525#	9529#	9531#	9536#	9540#	9542#	9548#	9567#	9572#	9576#
9577#	9582#	9586#	9588#	9593#	9597#	9599#	9605#	9635#	9643#	9666#	9675#	9698#
9708#	9731#	9732#	9736#	9749#	9783#	9784#	9792#	9795#	9802#	9806#	9809#	9816#
9820#	9823#	9830#	9856#	9857#	9865#	9868#	9875#	9879#	9882#	9889#	9893#	9896#
9903#	9930#	9931#	9939#	9942#	9949#	9953#	9956#	9963#	9967#	9970#	9977#	10002#
10003#	10011#	10014#	10021#	10025#	10028#	10035#	10039#	10042#	10049#	10078#	10079#	10093#
10095#	10099#	10104#	10109#	10112#	10116#	10121#	10125#	10128#	10132#	10158#	10159#	10173#
10176#	10180#	10185#	10189#	10193#	10199#	10203#	10206#	10212#	10234#	10235#	10251#	10252#
10255#	10258#	10264#	10268#	10270#	10276#	10301#	10302#	10303#	10310#	10320#	10323#	10329#
10336#	10345#	10366#	10368#	10371#	10382#	10386#	10394#	10411#	10412#	10417#	10426#	10433#
10438#	10459#	10460#	10461#	10462#	10463#	10468#	10471#	10472#	10473#	10474#	10475#	10480#
10481#	10482#	10497#	10498#	10499#	10500#	10507#	10513#	10518#	10539#	10545#	10550#	10555#
10563#	10564#	10568#	10576#	10577#	10580#	10589#	10592#	10596#	10606#	10607#	10613#	10638#
10639#	10642#	10643#	10645#	10646#	10647#	10649#	10650#	10652#	10654#	10656#	10658#	10659#
10663#	10664#	10666#	10667#	10668#	10670#	10674#	10678#	10684#	10685#	10687#	10688#	10690#
10691#	10692#	10750#	10753#	10755#	10757#	10758#	10760#	10774#	10785#	10789#	10790#	10793#
10795#	10797#	10813#	10817#	10845#	10847#	10848#	10850#	10852#	10866#	10872#	10875#	10884#
10887#	10888#	10890#	10892#	10902#	10905#	10906#	10908#	10910#	10922#	10924#	10925#	10936#
10939#	10940#	10941#	10942#	10945#	10946#	10964#	10980#	10983#	10991#	10995#	11002#	11003#
11009#	11010#	11011#	11014#	11033#	11034#	11036#	11050#	11052#	11060#	11064#	11067#	11068#
11074#	11075#	11077#	11105#	11106#	11134#	11136#	11137#	11140#	11141#	11144#	11146#	11148#
11151#	11155#	11158#	11160#	11165#	11171#	11173#	11174#	11192#	11194#	11196#	11200#	11201#
11217#	11219#	11221#	11228#	11229#	11230#	11232#	11233#	11238#	11240#	11270#	11271#	11273#
11276#	11280#	11282#	11287#	11292#	11363#	11364#	11365#	11373#	11374#	11375#	11376#	11381#
11382#	11384#	11389#	11390#	11398#	11406#	11441#	11442#	11444#	11445#	11446#	11447#	11448#
11451#	11481#	11482#	11483#	11484#	11485#	11495#	11522#	11549#	11572#	11573#	11577#	11578#
11579#	11580#	11581#	11582#	11583#	11584#	11585#	11586#	11587#	11588#	11589#	11591#	11592#
11599#												

\$TAGNU= 000251

\$TEMP = 000300

\$TYPOC	015120	11632#	11635											
\$TYPON	015134	11632#	11635											
\$TYPOS	015074	11632#	11635											
\$UNIT	001206	9049#												
\$UNITM	001010	9048#												
\$USWR	001220	9049#	9148	9340	9391	9729	9779	9854	9927	10000	10075	10155	10232	10637
		10691	10903	11031	11269	11442								
\$VECT1	001244	9049#	11582											
\$VECT2	001246	9049#												
\$XTSTR	014402	11626#												
\$YESNO=	000001	9079#	9080#	9081#	9082#	9121#	9122#	9124#	9162#	9173#	9184#	9219#	9230#	9241#
		9287#	9298#	9309#	9345#	9356#	9365#	9406#	9417#	9428#	9463#	9474#	9485#	9520#
		9531#	9542#	9577#	9588#	9599#	9795#	9809#	9823#	9868#	9882#	9896#	9942#	9956#
		9970#	10014#	10028#	10042#	10095#	10112#	10128#	10176#	10193#	10206#	10251#	10258#	10270#
		10366#	10411#	10461#	10473#	10647#	10649#	10666#	10667#	10691#	10755#	10760#	10774#	10797#
		10817#	10850#	10852#	10872#	10890#	10892#	10910#	10922#	10924#	10925#	10941#	10942#	10946#
		10964#	10983#	11002#	11003#	11014#	11036#	11052#	11067#	11068#	11155#	11158#	11160#	11173#
		11174#	11192#	11194#	11200#	11217#	11219#	11221#	11232#	11238#	11273#	11280#	11389#	11442#
		11444#	11445#	11446#	11448#	11482#	11483#	11522#	11585#	11586#	11587#	11592#		
		11331#	11419#	11464#	11528#	11552#								
\$SARGC=	000000	9116#	9148#	9154#	9164#	9175#	9187#	9211#	9221#	9232#	9244#	9279#	9289#	9300#
\$SBYTE=	000403	9312#	9340#	9347#	9357#	9367#	9391#	9398#	9408#	9419#	9431#	9455#	9465#	9476#
		9488#	9512#	9522#	9533#	9545#	9569#	9579#	9590#	9602#	9636#	9668#	9699#	9729#
		9738#	9779#	9797#	9811#	9825#	9854#	9870#	9884#	9898#	9927#	9944#	9958#	9972#
		10000#	10016#	10030#	10044#	10075#	10101#	10113#	10118#	10129#	10155#	10182#	10196#	10209#
		10232#	10261#	10273#	10300#	10333#	10391#	10435#	10458#	10470#	10476#	10489#	10491#	10492#
		10503#	10515#	10558#	10569#	10583#	10600#	10609#	10637#	10641#	10660#	10661#	10672#	10676#
		10780#	10782#	10810#	10878#	10880#	10903#	10930#	10932#	11006#	11031#	11071#	11104#	11168#
		11198#	11224#	11226#	11236#	11269#	11289#	11372#	11379#	11396#	11570#	11571#		
\$SDST =	000067	10691#	11442#											
\$SFLAG=	000001	9116#	9119#	9148#	9150#	9154#	9157#	9164#	9167#	9175#	9178#	9187#	9190#	9211#
		9214#	9221#	9224#	9232#	9235#	9244#	9247#	9279#	9282#	9289#	9292#	9300#	9303#
		9312#	9315#	9340#	9342#	9347#	9350#	9357#	9360#	9367#	9370#	9391#	9393#	9398#
		9401#	9408#	9411#	9419#	9422#	9431#	9434#	9455#	9458#	9465#	9468#	9476#	9479#
		9488#	9491#	9512#	9515#	9522#	9525#	9533#	9536#	9545#	9548#	9569#	9572#	9579#
		9582#	9590#	9593#	9602#	9605#	9636#	9643#	9668#	9675#	9699#	9708#	9729#	9732#
		9738#	9749#	9779#	9784#	9797#	9802#	9811#	9816#	9825#	9830#	9854#	9857#	9870#
		9875#	9884#	9889#	9898#	9903#	9927#	9931#	9944#	9949#	9958#	9963#	9972#	9977#
		10000#	10003#	10016#	10021#	10030#	10035#	10044#	10049#	10075#	10079#	10101#	10104#	10113#
		10116#	10118#	10121#	10129#	10132#	10155#	10159#	10182#	10185#	10196#	10199#	10209#	10212#
		10232#	10235#	10261#	10264#	10273#	10276#	10300#	10302#	10317#	10320#	10333#	10336#	10342#
		10345#	10379#	10382#	10391#	10394#	10423#	10426#	10435#	10438#	10458#	10460#	10470#	10474#
		10476#	10482#	10489#	10491#	10492#	10498#	10499#	10503#	10507#	10515#	10518#	10558#	10564#
		10569#	10577#	10583#	10589#	10600#	10607#	10609#	10613#	10637#	10639#	10641#	10643#	10660#
		10661#	10668#	10672#	10676#	10684#	10685#	10780#	10782#	10789#	10790#	10810#	10813#	10878#
		10880#	10887#	10888#	10903#	10906#	10930#	10932#	10939#	10940#	11006#	11010#	11031#	11034#
		11071#	11075#	11104#	11106#	11168#	11171#	11198#	11201#	11224#	11226#	11230#	11233#	11236#
		11240#	11269#	11271#	11284#	11287#	11289#	11292#	11372#	11376#	11379#	11382#	11396#	11398#
		11570#	11571#	11578#	11588#									
\$FROM=	000000	9070#	10097#	10178#	10195#	10208#	10260#	10272#	10314#	10341#	10376#	10421#	10547#	10552#
		10598#	10777#	10809#	10870#	10874#	10927#	10978#	10988#	10993#	11049#	11057#	11061#	11108#
		11283#	11388#	11574#										
\$GET4=	000000	11608#												
\$LOC =	012332	9116#	9123#	9148#	9154#	9164#	9175#	9187#	9211#	9221#	9232#	9244#	9279#	9289#
		9300#	9312#	9340#	9347#	9357#	9367#	9391#	9398#	9408#	9419#	9431#	9455#	9465#
		9476#	9488#	9512#	9522#	9533#	9545#	9569#	9579#	9590#	9602#	9636#	9668#	9699#

\$142	006774	10649#	10690
\$143	007144	10649	10690#
\$144	007034	10660#	10670
\$145	007070	10660	10670#
\$146	007056	10661	10664#
\$147	007066	10664	10668#
\$15	002650	9289	9292#
\$150	007102	10672	10674#
\$151	007126	10674	10685#
\$152	007124	10676	10678#
\$153	007126	10678	10684#
\$154	007416	10780	10790#
\$155	007414	10782	10785#
\$156	007416	10785	10789#
\$157	007474	10810	10813#
\$16	002676	9300	9303#
\$160	007730	10878	10888#
\$161	007726	10880	10884#
\$162	007730	10884	10887#
\$163	007772	10903	10906#
\$164	010100	10930	10940#
\$165	010076	10932	10936#
\$166	010100	10936	10939#
\$167	010230	10983#	
\$17	002726	9312	9315#
\$170	010226	10983#	11011
\$171	010354	10983	11011#
\$172	010352	11006	11010#
\$173	010420	11031	11034#
\$174	010464	11052#	
\$175	010462	11052#	11077
\$176	010610	11052	11077#
\$177	010606	11071	11075#
\$2	002152	9116	9119#
\$20	002764	9340	9342#
\$200	010646	11104	11106#
\$201	010760	11164#	11165
\$202	010774	11165#	
\$203	011004	11168	11171#
\$204	011064	11198	11201#
\$205	011144	11224	11233#
\$206	011140	11226	11230#
\$207	011160	11236	11240#
\$21	003012	9347	9350#
\$210	011224	11269	11271#
\$211	011306	11284	11287#
\$212	011316	11289	11292#
\$213	011642	11397	11406#
\$214	011644	11399	11406#
\$215	011512	11370#	11390
\$216	011616	11384	11390#
\$217	011532	11372	11374#
\$22	003040	9357	9360#
\$220	011540	11374	11376#
\$221	011556	11379	11382#
\$222	011630	11396	11398#

MAINDEC-22-CNDVA-A
CNDVAA.F11 16-DEC-82

MACY11 30(1046)
15:12

16-DEC-82 15:13 PAGE 206-19
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0145

\$223	011744	11449	11451#
\$224	011744	11451#	
\$225	011710	11444#	
\$226	011704	11444#	11447
\$227	011734	11444	11447#
\$23	003066	9367	9370#
\$230	012024	11495#	
\$231	012024	11495#	
\$232	011772	11482#	
\$233	011766	11482#	11485
\$234	012016	11482	11485#
\$235	012004	11483#	
\$236	012002	11483#	11484
\$237	012014	11483	11484#
\$24	003124	9391	9393#
\$240	012216	11549#	
\$241	012216	11549#	
\$242	012356	11593	11599#
\$243	012356	11599#	
\$244	012220	11569#	11589
\$245	012304	11570	11583#
\$246	012244	11571	11573#
\$247	012252	11573	11578#
\$25	003144	9398	9401#
\$250	012324	11583	11588#
\$26	003172	9408	9411#
\$27	003220	9419	9422#
\$3	002264	9148	9150#
\$30	003250	9431	9434#
\$31	003306	9455	9458#
\$32	003334	9465	9468#
\$33	003362	9476	9479#
\$34	003412	9488	9491#
\$35	003450	9512	9515#
\$36	003476	9522	9525#
\$37	003524	9533	9536#
\$4	002304	9154	9157#
\$40	003554	9545	9548#
\$40CAT= ***** U		11623	11626
\$41	003612	9569	9572#
\$42	003640	9579	9582#
\$43	003666	9590	9593#
\$44	003716	9602	9605#
\$45	003756	9636	9643#
\$46	004016	9668	9675#
\$47	004056	9699	9708#
\$5	002332	9164	9167#
\$50	004114	9729	9732#
\$51	004136	9738	9749#
\$52	004174	9779	9784#
\$53	004222	9797	9802#
\$54	004250	9811	9816#
\$55	004276	9825	9830#
\$56	004334	9854	9857#
\$57	004362	9870	9875#
\$6	002360	9175	9178#

BEGIN	7390#														
BGNHRD	7973#														
BGNHW	7976#														
BGNINI	7979#														
BGNMOD	7982#	11325													
BGNMSG	7985#														
BGNSFT	7989#														
BGNSRV	7992#	11188	11213	11505											
BGNSUB	7997#	9110	9152	9161	9171	9182	9209	9218	9228	9239	9277	9286	9296	9307	9344
	9354	9364	9396	9405	9415	9426	9453	9462	9472	9482	9510	9519	9529	9540	9567
	9576	9586	9597	9635	9666	9698	9736	9792	9806	9820	9865	9879	9893	9939	9953
	9967	10011	10025	10039	10093	10109	10125	10173	10189	10203	10255	10268	10303	10323	10368
	10386	10412	10539	10568	10580	10592	10758	10793	10847	10902					
BGNSW	8006#														
BRESET	8009#	9088	9186	9243	9311	9430	9487	9544	9601	9642	9674	9707	9748	10389	10814
	11153	11293													
CALL	7063#	9070	10097	10178	10195	10208	10260	10272	10314	10341	10376	10421	10547	10552	10598
	10777	10809	10870	10874	10927	10978	10988	10993	11049	11057	11061	11108	11283	11388	11574
CASE	7329#														
CKLOOP	8021#	9606	9645	9677	9709	9751									
CLRVEC	8024#	9124	10817	10946											
COMMEN	1566#	8927#													
DECR	6996#														
DECRU	7002#														
DEFAULT	7372#														
DEVREG	8049#														
DEVYTP	8052#														
DISPAT	8055#														
ELSE	6427#	10472	10664	10674	10678	10785	10884	10936	11374	11573	11583				
END	7407#														
ENDCLN	8058#														
ENDCOM	1578#	8927#													
ENDDEC	7007#														
ENDBO	6498#	10500	10670												
ENDHRD	8061#														
ENDHW	8064#														
ENDIF	6445#	9119	9150	9157	9167	9178	9190	9214	9224	9235	9247	9282	9292	9303	9315
	9342	9350	9360	9370	9393	9401	9411	9422	9434	9458	9468	9479	9491	9515	9525
	9536	9548	9572	9582	9593	9605	9643	9675	9708	9732	9749	9784	9802	9816	9830
	9857	9875	9889	9903	9931	9949	9963	9977	10003	10021	10035	10049	10079	10104	10116
	10121	10132	10159	10185	10199	10212	10235	10264	10276	10302	10320	10336	10345	10382	10394
	10426	10438	10460	10474	10482	10498	10499	10507	10518	10564	10577	10589	10607	10613	10639
	10643	10668	10684	10685	10789	10790	10813	10887	10888	10906	10939	10940	11010	11034	11075
	11106	11171	11201	11230	11233	11240	11271	11287	11292	11376	11382	11398	11578	11588	
ENDINC	6983#	10690	11011	11077	11447	11484	11485								
ENDINI	8067#														
ENDLOO	6871#	11390													
ENDMOD	8070#														
ENDMSG	8073#														
ENDRTN	7225#	11406	11451	11495	11549	11599									
ENDSEL	7316#														
ENDSFT	8077#														
ENDSRV	8080#	11207	11247	11523											
ENDSUB	8091#	9120	9158	9168	9179	9191	9215	9225	9236	9248	9283	9293	9304	9316	9351
	9361	9371	9402	9412	9423	9435	9459	9469	9480	9492	9516	9526	9537	9549	9573
	9583	9594	9607	9646	9678	9710	9752	9803	9817	9831	9876	9890	9904	9950	9964

MAINDEC-ZZ-CNDVA-A
CNDVAA.F11

MAC:11 30(1046)
16-DEC-82 15:12

16-DEC-82 15:13 PAGE 207-1
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0148

	9978	10022	10036	10050	10106	10122	10133	10186	10200	10213	10265	10277	10321	10346	10384
	10395	10427	10565	10578	10590	10614	10791	10815	10893	10943					
ENDSW	8094#														
ENDTST	8097#	9126	9152	9249	9317	9372	9436	9493	9550	9608	9647	9679	9711	9753	9832
	9905	9979	10051	10134	10215	10278	10347	10396	10439	10523	10617	10726	10819	10947	11015
	11081	11249	11297												
EQUALS	8100#														
ERRDF	8104#	9118													
ERRHRD	8116#	9156	9166	9177	9189	9213	9223	9234	9246	9281	9291	9302	9314	9349	9359
	9369	9400	9410	9421	9433	9457	9467	9478	9490	9514	9524	9535	9547	9571	9581
	9592	9604	9640	9672	9701	9741	9799	9813	9827	9872	9886	9900	9946	9960	9974
	10018	10032	10046	10103	10115	10120	10131	10184	10198	10211	10263	10275	10319	10335	10344
	10381	10393	10425	10437	10479	10495	10505	10517	10560	10572	10586	10602	10611	10683	10784
	10788	10812	10882	10886	10934	10938	11008	11073	11170	11286	11291				
ERROR	8927#	9118	9156	9166	9177	9189	9213	9223	9234	9246	9281	9291	9302	9314	9349
	9359	9369	9400	9410	9421	9433	9457	9467	9478	9490	9514	9524	9535	9547	9571
	9581	9592	9604	9640	9672	9701	9741	9799	9813	9827	9872	9886	9900	9946	9960
	9974	10018	10032	10046	10103	10115	10120	10131	10184	10198	10211	10263	10275	10319	10335
	10344	10381	10393	10425	10437	10479	10495	10505	10517	10560	10572	10586	10602	10611	10683
	10784	10788	10812	10882	10886	10934	10938	11008	11073	11170	11286	11291			
ESCAPE	1698#	8927#													
EXIF	6829#														
EXIFB	6851#	11384													
EXIT	8128#	9149	9341	9392	9731	9783	9856	9930	10002	10078	10158	10214	10234	10301	10459
	10481	10497	10520	10563	10576	10588	10606	10615	10638	10642	10694	10905	11009	11033	11074
	11105	11176	11270	11294											
GETPRI	1312#	8927#													
GETSWR	1771#	8927#	9063#												
GPHARD	8152#														
GPRMA	8155#														
GPRMD	8158#														
GPRML	8161#														
HEADER	8164#														
IF	6332#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300	9312
	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512	9522
	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811	9825
	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101	10113
	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470
	10476	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10661	10672	10676
	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104	11168	11198	11224
	11226	11236	11269	11372	11570	11571									
IFB	6338#	11289	11379	11396											
IFCOND	6358#														
IF.ERR	6368#	10317	10342	10379	10423	11284									
IF.NO.	6380#														
INCR	6972#	10649	10983	11052	11444	11483									
INCRU	6978#	11482													
INLINE	7946#														
LASTAD	8167#														
LEAVE	7428#														
LET	7922#	9068	9073	9075	9076	9078	9079	9080	9081	9082	9083	9105	9107	9108	9110
	9112	9121	9122	9124	9149	9152	9161	9162	9171	9173	9182	9184	9209	9218	9219
	9228	9230	9239	9241	9277	9286	9287	9296	9298	9307	9309	9341	9344	9345	9354
	9356	9364	9365	9392	9396	9405	9406	9415	9417	9426	9428	9453	9462	9463	9472
	9474	9482	9485	9510	9519	9520	9529	9531	9540	9542	9567	9576	9577	9586	9588
	9597	9599	9635	9666	9698	9731	9736	9783	9792	9795	9806	9809	9820	9823	9856

TYPBIN	2088#	8927#													
TYPDEC	2058#	8927#	11311	11608											
TYPNAM	1826#	8927#	9063												
TYPNUM	2025#	8927#													
TYPOCS	1978#	8927#													
TYPOCT	1941#	8927#	11307	11309	11535	11538	11540	11542	11544	11617					
TYPTXT	1894#	8927#	11306	11308	11310	11534	11536	11539	11541	11543					
UNTIL	6650#	9123	10475	11165	11589										
UNTILB	6671#														
WAITMS	8188#	10097	10178	10195	10208	10260	10272	10547	10552	10598	10777	10809	10874	10927	11388
WHILE	6487#	10489	10660												
WHILEB	6492#														
\$ADDON	5805#	9109	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300
	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512
	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811
	9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101
	10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379
	10391	10423	10435	10458	10469	10470	10472	10475	10476	10489	10491	10492	10500	10503	10515
	10558	10569	10583	10600	10609	10637	10641	10649	10660	10661	10664	10670	10672	10674	10676
	10678	10780	10782	10785	10810	10878	10880	10884	10903	10930	10932	10936	10983	11006	11031
	11052	11071	11104	11164	11165	11168	11198	11224	11226	11236	11269	11284	11289	11331	11370
	11372	11374	11379	11390	11396	11419	11444	11464	11482	11483	11528	11552	11569	11570	11571
	11573	11583													
\$AND	6092#	10676													
\$BRANC	5931#	9116	9123	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300
	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512
	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811
	9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101
	10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379
	10391	10423	10435	10458	10470	10472	10475	10476	10489	10491	10492	10500	10503	10515	10558
	10569	10583	10600	10609	10637	10641	10649	10660	10661	10664	10670	10672	10674	10676	10678
	10690	10780	10782	10785	10810	10878	10880	10884	10903	10930	10932	10936	10983	11006	11011
	11031	11052	11071	11077	11104	11165	11168	11198	11224	11226	11236	11269	11284	11289	11372
	11374	11379	11384	11390	11396	11397	11399	11444	11447	11449	11482	11483	11484	11485	11570
	11571	11573	11583	11589	11593										
\$BRCOD	6230#	10475	10649	10983	11052	11165	11384	11444	11482	11483					
\$CALL	7039#	9070	10097	10178	10195	10208	10260	10272	10314	10341	10376	10421	10547	10552	10598
	10777	10809	10870	10874	10927	10978	10988	10993	11049	11057	11061	11108	11283	11388	11574
\$CHECK	6288#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300	9312
	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512	9522
	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811	9825
	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101	10113
	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470
	10476	10489	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661
	10672	10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104	11168
	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571				
\$CHK1	7725#	9068	9073	9075	9076	9078	9083	9105	9107	9108	9110	9112	9124	9149	9152
	9161	9171	9182	9209	9218	9228	9239	9277	9286	9296	9307	9341	9344	9354	9364
	9392	9396	9405	9415	9426	9453	9462	9472	9482	9510	9519	9529	9540	9567	9576
	9586	9597	9635	9666	9698	9731	9736	9783	9792	9806	9820	9856	9865	9879	9893
	9930	9939	9953	9967	10002	10011	10025	10039	10078	10093	10099	10109	10125	10158	10173
	10180	10189	10203	10234	10252	10255	10268	10301	10303	10310	10323	10329	10368	10371	10386
	10412	10417	10433	10459	10462	10463	10468	10471	10480	10481	10497	10513	10539	10545	10550
	10555	10563	10568	10576	10580	10592	10596	10606	10638	10642	10645	10646	10649	10650	10652
	10654	10656	10658	10659	10663	10687	10688	10692	10750	10753	10757	10758	10793	10795	10817
	10845	10847	10848	10866	10875	10902	10905	10908	10945	10946	10980	10983	10991	10995	11009

MAINDEC-22-CNDVA-A
CNDVAA.P11 16-DEC-82

MACY11 30(1046)
15:12

16-DEC-82 15:13 PAGE 207-4
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0151

	11033	11050	11052	11060	11064	11074	11105	11134	11136	11137	11140	11141	11144	11146	11148
	11151	11196	11228	11229	11270	11276	11282	11363	11364	11365	11373	11375	11381	11441	11444
	11448	11481	11482	11483	11572	11577	11579	11580	11581	11582	11584	11591			
\$CKOP2	7826/	9079	9080	9081	9082	9121	9122	9124	9162	9173	9184	9219	9230	9241	9287
	929F	9309	9345	9356	9365	9406	9417	9428	9463	9474	9485	9520	9531	9542	9577
	958E	9599	9795	9809	9823	9868	9882	9896	9942	9956	9970	10014	10028	10042	10095
	10112	10128	10176	10193	10206	10251	10258	10270	10366	10411	10461	10473	10647	10666	10667
	10691	10755	10760	10774	10797	10817	10850	10852	10872	10890	10892	10910	10922	10924	10925
	10941	10942	10946	10964	11002	11003	11014	11036	11067	11068	11155	11158	11160	11173	11174
	11192	11194	11200	11217	11219	11221	11232	11238	11273	11280	11389	11442	11445	11446	11522
	11585	11586	11587	11592											
\$CKR6	7554#	10691	11442												
\$CMND	6119#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300	9312
	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512	9522
	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811	9825
	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101	10113
	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470
	10476	10489	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10660	10661
	10672	10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104	11168
	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571				
\$COMP	6280#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300	9312
	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512	9522
	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811	9825
	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101	10113
	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379	10391
	10423	10435	10458	10470	10476	10489	10491	10492	10503	10515	10558	10569	10583	10600	10609
	10637	10641	10649	10660	10661	10672	10676	10780	10782	10810	10878	10880	10903	10930	10932
	10983	11006	11031	11052	11071	11104	11168	11198	11224	11226	11236	11269	11284	11289	11372
	11379	11396	11444	11482	11483	11570	11571								
\$COUNT	7032#	9070	10097	10178	10195	10208	10260	10272	10314	10341	10376	10421	10547	10552	10598
	10777	10809	10870	10874	10927	10978	10988	10993	11049	11057	11061	11108	11283	11388	11574
\$DO	6046#	10489	10660												
\$ELSE	6387#														
\$ERRMS	6002#														
\$EXIFA	6705#														
\$EXIFO	6742#														
\$EXIF2	6800#	11384													
\$EXIF3	6779#														
\$GENBR	5924#	9116	9123	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300
	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512
	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811
	9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101
	10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379
	10391	10423	10435	10458	10470	10472	10475	10476	10489	10491	10492	10500	10503	10515	10558
	10569	10583	10600	10609	10637	10641	10649	10660	10661	10664	10670	10672	10674	10676	10678
	10690	10780	10782	10785	10810	10878	10880	10884	10903	10930	10932	10936	10983	11006	11011
	11031	11052	11071	11077	11104	11165	11168	11198	11224	11226	11236	11269	11284	11289	11372
	11374	11379	11384	11390	11396	11397	11399	11444	11447	11449	11482	11483	11484	11485	11570
	11571	11573	11583	11589	11593										
\$GENTA	5900#	9109	9119	9150	9157	9167	9178	9190	9214	9224	9235	9247	9282	9292	9303
	9315	9342	9350	9360	9370	9393	9401	9411	9422	9434	9458	9468	9479	9491	9515
	9525	9536	9548	9572	9582	9593	9605	9643	9675	9708	9732	9749	9784	9802	9816
	9830	9857	9875	9889	9903	9931	9949	9963	9977	10003	10021	10035	10049	10079	10104
	10116	10121	10132	10159	10185	10199	10212	10235	10264	10276	10302	10320	10336	10345	10382
	10394	10426	10438	10460	10469	10472	10474	10475	10482	10489	10498	10499	10500	10507	10518
	10564	10577	10589	10607	10613	10639	10643	10649	10660	10664	10668	10670	10674	10678	10684

J 12

MAINDEC-ZZ-CNDVA-A MACY11 30(1046) 16-DEC-82 15:13 PAGE 207-5

CNDVAA.F11 16-DEC-82 15:12 CROSS REFERENCE TABLE -- MACRO NAMES SEQ 0152

	10685	10690	10785	10789	10790	10813	10884	10887	10888	10906	10936	10939	10940	10983	11010
	11011	11034	11052	11075	11077	11106	11164	11165	11171	11201	11230	11233	11240	11271	11287
	11292	11370	11374	11376	11382	11390	11398	11406	11444	11447	11451	11482	11483	11484	11485
	11495	11549	11569	11573	11578	11583	11588	11599							
\$IF	6306#	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300	9312
	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512	9522
	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811	9825
	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101	10113
	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10333	10391	10435	10458	10470
	10476	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641	10661	10672	10676
	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071	11104	11168	11198	11224
	11226	11236	11269	11289	11372	11379	11396	11570	11571						
\$IFCOD	6178#	9116	9123	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300
	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512
	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811
	9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101
	10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10333	10391	10435	10458
	10470	10475	10476	10489	10491	10492	10503	10515	10558	10569	10583	10600	10609	10637	10641
	10660	10661	10672	10676	10780	10782	10810	10878	10880	10903	10930	10932	11006	11031	11071
	11104	11165	11168	11198	11224	11226	11236	11269	11289	11372	11379	11396	11570	11571	11589
\$IFCON	6345#	10317	10342	10379	10423	11284									
\$IFOPR	5944#	9116	9123	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300
	9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512
	9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811
	9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101
	10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379
	10391	10423	10435	10458	10470	10475	10476	10489	10491	10492	10503	10515	10558	10569	10583
	10600	10609	10637	10641	10660	10661	10672	10676	10780	10782	10810	10878	10880	10903	10930
	10932	11006	11031	11071	11104	11165	11168	11198	11224	11226	11236	11269	11284	11289	11372
	11379	11396	11570	11571	11589										
\$LET	7909#	9068	9073	9075	9076	9078	9079	9080	9081	9082	9083	9105	9107	9108	9110
	9112	9121	9122	9124	9149	9152	9161	9162	9171	9173	9182	9184	9209	9218	9219
	9228	9230	9239	9241	9277	9286	9287	9296	9298	9307	9309	9341	9344	9345	9354
	9356	9364	9365	9392	9396	9405	9406	9415	9417	9426	9428	9453	9462	9463	9472
	9474	9482	9485	9510	9519	9520	9529	9531	9540	9542	9567	9576	9577	9586	9588
	9597	9599	9635	9666	9698	9731	9736	9783	9792	9795	9806	9809	9820	9823	9856
	9865	9868	9879	9882	9893	9896	9930	9939	9942	9953	9956	9967	9970	10002	10011
	10014	10025	10028	10039	10042	10078	10093	10095	10099	10109	10112	10125	10128	10158	10173
	10176	10180	10189	10193	10203	10206	10234	10251	10252	10255	10258	10268	10270	10301	10303
	10310	10323	10329	10366	10368	10371	10386	10411	10412	10417	10433	10459	10461	10462	10463
	10468	10471	10473	10480	10481	10497	10513	10539	10545	10550	10555	10563	10568	10576	10580
	10592	10596	10606	10638	10642	10645	10646	10647	10650	10652	10654	10656	10658	10659	10663
	10666	10667	10687	10688	10691	10692	10750	10753	10755	10757	10758	10760	10774	10793	10795
	10797	10817	10845	10847	10848	10850	10852	10866	10872	10875	10890	10892	10902	10905	10908
	10910	10922	10924	10925	10941	10942	10945	10946	10964	10980	10991	10995	11002	11003	11009
	11014	11033	11036	11050	11060	11064	11067	11068	11074	11105	11134	11136	11137	11140	11141
	11144	11146	11148	11151	11155	11158	11160	11173	11174	11192	11194	11196	11200	11217	11219
	11221	11228	11229	11232	11238	11270	11273	11276	11280	11282	11363	11364	11365	11373	11375
	11381	11389	11441	11442	11445	11446	11448	11481	11522	11572	11577	11579	11580	11581	11582
	11584	11585	11586	11587	11591	11592									
\$LPCNT	6927#	10649	10983	11052	11444	11482	11483								
\$OPADD	7566#	9079	9080	9081	9082	9121	9122	9124	10473	10649	10666	10667	10755	10817	10946
	10983	11052	11192	11217	11232	11444	11482	11483	11522	11586	11587	11592			
\$OPAND	7608#	10691	11442												
\$OPCD1	7649#	9079	9080	9081	9082	9121	9122	9124	9162	9173	9184	9219	9230	9241	9287
	9298	9309	9345	9356	9365	9406	9417	9428	9463	9474	9485	9520	9531	9542	9577

SSUNTL3	6600#		10660																	
SSWHILE	6460#	10489		10660																
SSCMRE	9049#																			
SSCMTM	9049#																			
SSDEFA	7354#																			
SSENDS	7305#																			
SSERRO	6012#																			
SSESCA	1711#	8927#																		
SSGEN	5888#	9109	9119	9150	9157	9167	9178	9190	9214	9224	9235	9247	9282	9292	9303					
		9315	9342	9350	9360	9370	9393	9401	9411	9422	9434	9458	9468	9479	9491	9515				
		9525	9536	9548	9572	9582	9593	9605	9643	9675	9708	9732	9749	9784	9802	9816				
		9830	9857	9875	9889	9903	9931	9949	9963	9977	10003	10021	10035	10049	10079	10104				
		10116	10121	10132	10159	10185	10199	10212	10235	10264	10276	10302	10320	10336	10345	10382				
		10394	10426	10438	10460	10472	10474	10475	10482	10489	10498	10499	10500	10507	10518	10577				
		10564	10577	10589	10607	10613	10639	10643	10649	10660	10664	10668	10670	10674	10678	10684				
		10685	10690	10785	10789	10790	10813	10884	10887	10888	10906	10936	10939	10940	10983	11010				
		11011	11034	11052	11075	11077	11106	11164	11165	11171	11201	11230	11233	11240	11271	11287				
		11292	11331	11370	11374	11376	11382	11390	11398	11406	11419	11444	11447	11451	11464	11482				
		11483	11484	11485	11495	11528	11549	11552	11569	11573	11578	11583	11588	11599						
SSGETS	5851#	9119	9123	9150	9157	9167	9178	9190	9214	9224	9235	9247	9282	9292	9303					
		9315	9342	9350	9360	9370	9393	9401	9411	9422	9434	9458	9468	9479	9491	9515				
		9525	9536	9548	9572	9582	9593	9605	9643	9675	9708	9732	9749	9784	9802	9816				
		9830	9857	9875	9889	9903	9931	9949	9963	9977	10003	10021	10035	10049	10079	10104				
		10116	10121	10132	10159	10185	10199	10212	10235	10264	10276	10302	10320	10336	10345	10382				
		10394	10426	10438	10460	10472	10474	10475	10482	10498	10499	10500	10507	10518	10564	10577				
		10589	10607	10613	10639	10643	10649	10664	10668	10670	10674	10678	10684	10685	10690	10785				
		10789	10790	10813	10884	10887	10888	10906	10936	10939	10940	10983	11010	11011	11034	11052				
		11075	11077	11106	11165	11171	11201	11230	11233	11240	11271	11287	11292	11374	11376	11382				
		11384	11390	11398	11406	11444	11447	11451	11482	11483	11484	11485	11495	11549	11573	11578				
		11583	11588	11589	11599															
SSGETT	5861#	10472	10664	10674	10678	10785	10884	10936	11374	11384	11573	11583								
SSLPCN	6887#	10649	10983	11052	11444	11482	11483													
SSNEWT	1662#	8927#	9100	9142	9202	9260	9335	9386	9447	9504	9561	9625	9657	9689	9722					
		9773	9848	9921	9994	10069	10149	10226	10293	10359	10404	10451	10533	10632	10744	10838				
		10958	11026	11096	11264	11305														
SSPOP	5875#	9119	9123	9150	9157	9167	9178	9190	9214	9224	9235	9247	9282	9292	9303					
		9315	9342	9350	9360	9370	9393	9401	9411	9422	9434	9458	9468	9479	9491	9515				
		9525	9536	9548	9572	9582	9593	9605	9643	9675	9708	9732	9749	9784	9802	9816				
		9830	9857	9875	9889	9903	9931	9949	9963	9977	10003	10021	10035	10049	10079	10104				
		10116	10121	10132	10159	10185	10199	10212	10235	10264	10276	10302	10320	10336	10345	10382				
		10394	10426	10438	10460	10472	10474	10475	10482	10498	10499	10500	10507	10518	10564	10577				
		10589	10607	10613	10639	10643	10649	10664	10668	10670	10674	10678	10684	10685	10690	10785				
		10789	10790	10813	10884	10887	10888	10906	10936	10939	10940	10983	11010	11011	11034	11052				
		11075	11077	11106	11165	11171	11201	11230	11233	11240	11271	11287	11292	11374	11376	11382				
		11390	11398	11406	11444	11447	11451	11482	11483	11484	11485	11495	11549	11573	11578	11583				
		11588	11589	11599																
SSPUSH	5839#	9109	9116	9148	9154	9164	9175	9187	9211	9221	9232	9244	9279	9289	9300					
		9312	9340	9347	9357	9367	9391	9398	9408	9419	9431	9455	9465	9476	9488	9512				
		9522	9533	9545	9569	9579	9590	9602	9636	9668	9699	9729	9738	9779	9797	9811				
		9825	9854	9870	9884	9898	9927	9944	9958	9972	10000	10016	10030	10044	10075	10101				
		10113	10118	10129	10155	10182	10196	10209	10232	10261	10273	10300	10317	10333	10342	10379				
		10391	10423	10435	10458	10469	10470	10472	10476	10489	10491	10492	10500	10503	10515	10558				
		10569	10583	10600	10609	10637	10641	10649	10660	10661	10664	10670	10672	10674	10676	10678				
		10780	10782	10785	10810	10878	10880	10884	10903	10930	10932	10936	10983	11006	11031	11052				
		11071	11104	11164	11168	11198	11224	11226	11236	11269	11284	11289	11331	11370	11372	11374				
		11379	11390	11396	11419	11444	11464	11482	11483	11528	11552	11569	11570	11571	11573	11583				

. ABS. 015454 000

ERRORS DETECTED: 0

CNDVAA,CNDVAA/CRF/NL:TOC=CNMAC2.SML,CNMAC.MAC,CNDVAA.P11
RUN-TIME: 85 90 5 SECONDS
RUN-TIME RATIO: 341/182=1.8
CORE USED: 43K (86 PAGES)