

DZ11

DZ11 ASYNC MUX TST
CZDZAGO

AH-8783G-MC
FICHE 1 OF 1

OCT 1981
COPYRIGHT © 76-81
MADE IN USA



5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669

.REM %

IDENTIFICATION

PRODUCT CODE: AC-8781G-MC
PRODUCT NAME: CZDZAGO DZ11 LN ASYNC MUX TSTS
PRODUCT DATE: OCT 1981
MAINTAINER: MK-DIAGNOSTIC ENGINEERING

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) 1976,1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL PDP UNIBUS MASSBUS
DEC DECUS DECTAPE

5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710

1. ABSTRACT

THE FUNCTION OF THE DZ11 DIAGNOSTICS IS TO VERIFY THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS ALSO VERIFY THAT THE DZ11 OPERATES IN ITS ENVIRONMENT SUCH AS THE SYSTEM IN WHICH IT IS INSTALLED.

PARAMETERS MAY BE SUPPLIED TO THE PROGRAM BY EITHER 'AUTO SIZING' OR INPUT FROM THE USER ON THE CONSOLE BY HAVING SW00=1 AT START TIME. AUTO SIZING WILL BE DONE ONLY THE FIRST TIME THE PROGRAM IS STARTED AND SW07=0 AND SW00=0 AND SW03=0. THE AUTOSIZER IS DESIGNED TO DETECT DZ11 DEVICE ADDRESSES AND VECTORS AND TO DETERMINE WHETHER THE DZ11 THAT IS DETECTED IS AN EIA OR 20MA BOARD. ALL REMAINING PARAMETERS DEFAULT TO CERTAIN VALUES (SEE SEC.8.5). CONSOLE INPUT MAY BE CONTROLLED AT ANY START TIME THROUGH THE USE OF SW00, SW03, SW04, AND SW06 (SEE SEC. 4.1.1 FOR A DETAILED DESCRIPTION OF THESE SWITCHES).

CURRENTLY THERE IS ONE STANDALONE DIAGNOSTIC (CZDZA), ONE SYSTEM FOR DEC X/11 (DZAA), AND AN ONLINE OVERLAY FOR DZITA (ITEP) - DZDZB. (ITEP) - DZDZB.

CZDZA WILL TEST ALL PARTS OF THE DZ11 SUCH AS CABLES, DIST PNL., INTERFACE MODULE ITSELF.

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (WITH MINIMUM 8K MEMORY)
ASR 33 (OR EQUIVALENT FOR CONSOLE)
DZ11 INTERFACE MODULE (M7819(EIA), M7814(20MA))
H3271 STAGGERED TURNAROUND CONNECTOR FOR EIA MODULE.
H3190 STAGGERED TURNAROUND CONNECTOR FOR 20MA MODULE.
H325 CABLE TURNAROUND AND DIST PNL TESTING FOR EIA MODULE.
H315 THIS MAY BE SUBSTITUTED FOR H325.

NOTE: A STAGGERED TURNAROUND CONNECTOR IS NEEDED IN ORDER TO TEST THE PARITY AND BREAK LOGIC.

5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753

2.2 STORAGE

PROGRAM WILL USE ALL 8K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATION 1500 THRU 2000 ARE ESPECIALLY TO BE NOTED AND TO BE UNTOUCHED BY OPERATOR AFTER PARAMETERS HAVE BEEN INPUT FROM CONSOLE (MODE=1); OR AFTER THE 'AUTO SIZING' HAS BEEN DONE. THESE LOCATIONS MAY BE CHANGED IF THE USER UNDERSTANDS THEIR MEANING AND DIFFERENT PARAMETERS ARE REQUIRED.

3. LOADING PROCEEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED USING THE ABSOLUTE LOADER. NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH AS DISK, MAGTAPE, DECTAPE, OR CASSETTE; FOLLOW INSTRUCTIONS FOR THE MONITOR WHICH HAS BEEN PROVIDED ON THAT SPECIFIC MEDIA.

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY * SIZE

4K	17
8K	37
12K	57
16K	77
20K	117
24K	137
28K	157

3.1.1 PLACE ADDRESS OF ABS LOADER INTO SWITCH REGISTER.
(ALSO PLACE 'HALT' SW UP)

3.1.2 DEPRESS 'LOAD ADDRESS' KEY ON CONSOLE AND RELEASE.

3.1.3 DEPRESS 'START KEY' ON CONSOLE AND RELEASE (PROGRAM SHOULD NOW BE LOADING INTO CPU)

5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810

4. STARTING PROCEEDURE

- A. SET SWITCH REGISTER TO 000200
- B. DEPRESS 'LOAD ADDRESS' KEY AND RELEASE
- C. SET SWR TO ZERO FOR 'AUTO SIZING' OR SET SW00=1 FOR USER PARAMETER INPUT FROM CONSOLE TERMINAL. ON FIRST START IF SW07=1 AND SW00=0 THE PROGRAM WILL DEFAULT TO CONSOLE PARAMETER INPUT (SW00=1).
- D. DEPRESS 'START KEY' AND RELEASE, THE PROGRAM WILL TYPE MAINDEC NAME AND PROGRAM NAME (IF THIS WAS THE FIRST START UP OF THE PROGRAM OR PARAMETERS WERE CHANGED BY SW00=1) AND ALSO THE FOLLOWING:

```

'MAP OF DZ11 STATUS'
1500 160100
1502 000300
1504 000005
1506 000377
1510 017070
1512 000000

```

THE ABOVE IS ONLY AN EXAMPLE! THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 1500 IN THE PROGRAM. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.
THE PROGRAM WILL TYPE 'RUNNING' AND PROCEED TO RUN THE DIAGNOSTIC.

4.1 CONTROL SWITCH SETTINGS

NOTE: IF THERE IS NO REAL SWR (177570); SWR MAY BE MODIFIED AT LOC:176 OR BY HITTING CONTROL 'G' <^G> ON CONSOLE TERMINAL.

- SW 15 SET: HALT ON ERROR
- SW 14 SET: LOOP ON CURRENT TEST
- SW 13 SET: INHIBIT ERROR PRINT OUT
- SW 12 SET: INHIBIT **ALL** TYPE OUT/BELL ON ERROR.
- SW 11 SET: INHIBIT ITERATIONS. (QUICK PASS)
- SW 10 SET: ESCAPE TO NEXT TEST
- SW 09 SET: LOOP WITH CURRENT DATA
- SW 08 SET: CATCH ERROR AND LOOP ON IT
- SW 07 SET: NO AUTO SIZE. IF 1ST START OF PROGRAM AFTER LOADING THE OPERATOR MUST INPUT ADDRESS AND VECTOR FROM CONSOLE.
- SW 06 SET: RESELECT DZ11'S DESIRED ACTIVE
- SW 05 SET: RESERVED
- SW 04 SET: SELECT DELAY PARAMETER (SEE SEC. 4.1.1)
- SW 03 SET: EXTRA PARAMETER INPUT (SEE SEC. 4.1.1)
- SW 02 SET: LOCK ON SELECTED TEST
- **SW 01 SET: RESTART PROGRAM AT SELECTED TEST
- *SW 00 SET: GET USERS PARAMETERS FROM CONSOLE

* FOR ECHO OR CABLE TESTS (PROGRAM STARTED AT LOC. 210) THIS SWITCH SET TO 1 ALLOWS THE USER TO TYPE IN THE VECTOR AND THE CSR FOR THE DZ11 UNDER TEST.
** FOR ECHO OR CABLE TEST THIS SWITCH SET TO 1 ALLOWS THE SELECTION OF EITHER THE ECHO OR CABLE TEST, BAUD RATE, AND THE LINE NUMBER UNDER TEST.

5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861

4.1.1 SWITCH REGISTER CONTROL OF PARAMETER INPUT FROM CONSOLE

SW 00 GET USERS PARAMETERS FROM CONSOLE. SETTING THIS SWITCH AT START UP TIME ALLOWS THE USER TO INPUT AT THE CONSOLE TERMINAL THE FOLLOWING PARAMETERS: BASE DEVICE ADDRESS, BASE VECTOR ADDRESS, BUS REQUEST LEVEL, DECLARE EIA OR 20MA MODULE, MODE OF OPERATION (EXTERNAL, INTERNAL, OR STAGGERED), AND THE NUMBER OF DZ11'S THAT ARE RUNNING. USING THIS SWITCH ALONE DEFAULTS THE FOLLOWING PARAMETERS: ALL 8 LINES ARE SET TO BE TESTED ON EACH DZ11, THE DEFAULT BAUD RATE IS SET AT 9600 BAUD, AND THE CHARACTER LENGTH FOR THE MAJORITY OF TESTING IS SET AT EIGHT BITS PER CHARACTER WITH TWO STOP BITS.

SW 03 EXTRA PARAMETER INPUT. SETTING THIS SWITCH AT START UP TIME PROVIDES THE USER WITH THE ABILITY TO SET THE LINES ACTIVE FOR TESTING AND TO SET THE DEFAULT BAUD RATE USED FOR THE MAJORITY OF THE DIAGNOSTIC TESTS. THE DELAY PARAMETER IS AUTOMATICALLY ADJUSTED TO THE BAUD RATE GIVEN BY THE USER.

SW 04 SELECT DELAY PARAMETER. THE DELAY PARAMETER THIS SWITCH CONTROLS DETERMINES THE LENGTH OF TIME THE PROGRAM STALLS WAITING FOR A CHARACTER TO BE COMPLETELY TRANSMITTED OR RECEIVED. THIS DELAY COUNT IS AUTOMATICALLY SET TO PROVIDE ENOUGH DELAY TIME FOR THE DEFAULT BAUD RATE SPECIFIED WHEN RUNNING THE PROGRAM ON AN 11/45 WITH BIPOLAR MEMORY. WHEN RUNNING THIS PROGRAM ON A FASTER PROCESSOR THE DELAY PARAMETER SHOULD BE ADJUSTED PROPORTIONALLY HIGHER THAN THE FOLLOWING DEFAULTED VALUES:

2450	:TIME FOR	50 BAUD
1560	:TIME FOR	75 BAUD
1120	:TIME FOR	110 BAUD
0750	:TIME FOR	134 BAUD
0660	:TIME FOR	150 BAUD
0330	:TIME FOR	300 BAUD
0150	:TIME FOR	600 BAUD
0060	:TIME FOR	1200 BAUD
0040	:TIME FOR	1800 BAUD
0030	:TIME FOR	2000 BAUD
0020	:TIME FOR	2400 BAUD
0010	:TIME FOR	3600 BAUD
0001	:TIME FOR	4800 BAUD
0001	:TIME FOR	7200 BAUD
0001	:TIME FOR	9600 BAUD
0001	:TIME FOR	19.2 KBAUD

*** NOTE ***
19.2K BAUD IS AN UNSUPPORTED BAUD RATE. IT SHOULD NOT NORMALLY BE USED.
9600 BAUD IS THE SPECIFIED MAXIMUM.

5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 06 RESELECT DZ11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO DZ11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS FOUR DZ11S; BITS 00,01,02,03 WILL BE SET IN LOC 'DZACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW06) ALTERS THAT LOCATION; THEREFORE IF FOUR DZ11S ARE IN THE SYSTEM ***DO NOT*** SET SWITCHES GREATER THAN SW 03 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE DZ11S THAN HAS BEEN GIVEN INFORMATION ABOUT IN PARAMETER INPUT (SW00=1)

METHOD: A: LOAD ADDRESS 200
B: START WITH SW 06=1
C: PROGRAM WILL TYPE MESSAGE
D: SET THE BINARY NUMBER OF DZ11S DESIRED ACTIVE EXAMPLE: 1=1 DZ11; 3=2 DZ11; 7=3 DZ11; 17=4 DZ11 37=5 DZ11 ETC/AA PRESS CONTINUE.
E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)
F: SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

SW 01 RESTART PROGRAM AT SELECTED TEST IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A TEST THAT IS NOT IN THE ORDER OF SEQUENCE THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS. NOTE: IF RUNNING MULTIPLE DZ11'S; THE DZ11 YOU DESIRE TO BE UNDER TEST MUST BE SELECTED BY THE USE OF SW06 BEFORE LOCKING ON THE TEST. IN OTHER WORDS; EACH TIME THE PROGRAM IS STARTED; THE FIRST DZ11 WILL BE SELECTED TO BE UNDER TEST UNLESS SW06 IS USED TO SELECT ONLY ONE.

SW 09 LOOP ON CURRENT DATA: THIS SWITCH WILL ONLY WORK IF CALL 'SCOPI' IS IN THAT TEST. THE REASON BEING THAT MOST TESTS DEAL WITH BLOCKS OF DIFFERENT DATA TO BE SENT OR RECEIVED ALL AT ONCE THUS IN BLOCK DATA, ONE PATTERN CAN'T BE SINGLED OUT. THIS SWITCH IS DESIGNED TO PROVIDE AN AID FOR A TRAINED TROUBLE-SHOOTER TO SAMPLE VARIOUS SIGNALS ON THE MODULE AND IS NOT MEANT TO BE USED AS A GENERAL USER CONTROL SWITCH.

SW 04 SELECT DELAY PARAMETER: THIS SWITCH SHOULD BE USED WITH CARE AS TOO SHORT A DELAY WILL CAUSE VALID TESTS TO FAIL ON CERTAIN PROCESSORS. IT IS RECOMMENDED THAT THIS SWITCH ONLY BE USED IN CONJUNCTION WITH SCOPE LOOPS, E.G. SW 14,9,4,1 SET; SW 9,4,2,1 SET. THE SHORTEST PARAMETER IS 1; THE LONGEST ACCEPTED IS 177776. (SEE SEC. 4.1.1)

5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
5951
5952
5953
5954
5955
5956
5957
5958
5959

4.1.3 SWITCH REGISTER PRIORITIES

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST(ON ERROR).
5. SW 10 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOPI'). IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. ON AN ERROR REPORT (EX. *TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS *USUALLY* THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0) IF THE PROGRAM USER IS TECHNICALLY TRAINED TO ELECTRONICALLY ISOLATE SIGNAL PROBLEMS ON THE DZ11 MODULE. IF SW09 IS NOT ENABELED; AND THERE IS A *HARD* ERROR (CONSTANT); SW08 IS BEST.
2. FOR INTERMITTENT ERRORS EITHER START THE PROGRAM WITH SW01 AND SW02 SET WHICH WILL ALLOW THE USER TO LOCK ON A SELECTED TEST, OR ELSE SET SW14 AS AN ERROR IS BEING TYPED OUT ON THE TERMINAL. SW14 WILL CONTINUE TO LOOP ON THAT TEST REGARDLESS OF WHETHER AN ERROR OCCURS.
3. SW 14 LOOP ON CURRENT TEST.

4.2 STARTING ADDRESS

SA 200 - ADDRESS 200 IS FOR NORMAL EXECUTION OF THE DIAGNOSTIC. THIS WILL DO THE MAJOR TESTING NECESSARY FOR VERIFICATION OF HARDWARE.

SA 210 - CABLE/ECHO - TERMINAL TESTS. STARTING AT ADDRESS 210 WILL GIVE THE USER THE OPTION TO VERIFY THE EIA CABLES AT THE DIST PNL OR VERIFY A TRUE LINK TO ANY DEC SUPPORTED TERMINAL SUPPORTED BY THE DZ11.

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY. AFTER *ALL* AVAILABLE DZ11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION FOUR WILL BE PRINTED AND PROGRAM WILL BEGIN RUNNING THE DIAGNOSTIC.

5961
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971
5972
5973
5974
5975
5976
5977
5978
5979
5980
5981
5982
5983
5984
5985
5986
5987
5988
5989
5990
5991
5992
5993
5994
5995
5996
5997
5998
5999

5.1 NORMAL START OF DIAGNOSTIC

ON THE FIRST START OF THE DIAGNOSTIC AT ADDRESS 200; IF AUTO SIZING IS NOT USED OR WHENEVER SW00=1; THE FOLLOWING QUESTIONS ARE ASKED AND MUST BE ANSWERED.

'1ST CSR ADDRESS (160000:163700): ''

YOU MUST TYPE IN THE FIRST DZ11 CSR IN THE SYSTEM YOU WISH TESTING TO BEGIN AT. RANGE: 160000:163700

'1ST VECTOR ADDRESS (300:770): ''

YOU MUST TYPE IN THE VECTOR OF THE FIRST DZ11 IN THE SYSTEM UNDER TEST. RANGE 300:770

'BR LEVEL (4:6): ''

TYPE IN THE PRIORITY LEVEL OF THE DZ11 THAT THE ABOVE INFORMATION HAS BEEN GIVEN ABOUT. RANGE 4 OR 5 OR 6.

'TYPE 'A' FOR EIA MODULE OR 'B' FOR 20MA (A:B): ''

TYPE 'A' IF RUNNING A DZ11-A,B,E (EIA).
TYPE 'B' IF RUNNING A DZ11-C,D,F (20MA).
TYPING A <CR> DEFAULTS TO EIA MODULES.

'MAINTENANCE MODE

[EXTERNAL <H325>-EIA ONLY (E)]
[INTERNAL <DZCSR03=1> (I)]
[STAGGERED <H3271>-EIA ONLY (S)]
[STAGGERED <H3190>-20MA ONLY (S)] :

TYPE 'E' OR 'I' OR 'S' DEPENDING ON WHICH MODE YOU WISH TO RUN IN. IF RUNNING 'EXTERNAL'; ALL SELECTED LINES MUST BE TERMINATED BY AN H325 TEST CONNECTOR.

6001
6002
6003
6004
6005
6006
6007
6008
6009
6010
6011
6012
6013
6014
6015
6016
6017
6018
6019
6020
6021
6022
6023
6024
6025
6026
6027
6028
6029
6030
6031
6032
6033
6034
6035
6036
6037
6038
6039
6040
6041

'# OF DZ11'S <IN OCTAL> (1:20): ''

TYPE TOTAL NUMBER OF DZ11'S TO BE TESTED IN THE SYSTEM. RANGE IS 1 THRU 20 IN OCTAL.

***** IF SW03=1 THEN *****
IF SW03=1 THE FOLLOWING WILL BE PRINTED.

'LINES ACTIVE BY BIT <IN OCTAL> (001:377):''

EACH BIT REPRESENTS A LINE AND ANY COMBINATION OF LINES MAY BE SELECTED (HOWEVER IN STAGGERED MODE TWO ADJACENT LINES MUST BE SELECTED (0-1, 2-3, 4-5, 6-7))..

'DEFAULT BAUD RATE <IN OCTAL> (00:16): ''

THIS GIVES THE USER A CHANCE TO CHANGE THE DEFAULT BAUD RATE USED IN APP. 90 PERCENT OF THE TEST. BAUD RATE CHOICES ARE:
'00''(50 BAUD), '01''(75 BAUD), '02''(110 BAUD), '03''(134 BAUD),
'04''(150 BAUD), '05''(300 BAUD), '06''(600 BAUD), '07''(1200 BAUD),
'10''(1800 BAUD), '11''(2000 BAUD), '12''(2400 BAUD), '13''(3600 BAUD),
'14''(4800 BAUD), '15''(7200 BAUD), '16''(9600 BAUD), '17''(19.2 KBAUD)
LOW DEFAULT BAUD RATES ARE NOT SUGGESTED SINCE THEY LENGTHEN THE TIME TO COMPLETE A PROGRAM PASS DRAMATICALLY.

*** NOTE ***

SPEED SELECT CODE 17 CAN BE USED TO SELECT 19.2K BAUD, BUT THIS SPEED IS NOT SPECIFIED BY DEC, AND SHOULD NOT NORMALLY BE USED.

IT IS IMPORTANT TO NOTE THAT ALL DZ11'S IN THE SYSTEM MUST BE CONTIGIOUS FOR BOTH ADDRESS AND VECTORS. ALSO ALL THE EXTRA PARAMETERS OTHER THAN CSR AND VECTORS ARE GIVEN TO THE EXISTING DZ11'S IN THE SYSTEM. IF NOT ALL DZ11'S ARE SAME PRIORITY OR IF THE MODE OF OPERATION IS DIFFERENT FOR EACH DZ11; THIS MUST BE 'PATCHED' INTO THE CORRECT STATUS MAP ENTRY WHICH IS PRINTED AT START TIME. AN ALTERNATIVE IS TO PUT SW00=1 AT START TIME; ANSWER QUESTIONS ABOUT DZ11 UNDER TEST AND INDICATE ONLY 1 DZ11 IN THE SYSTEM. IF THE STATUS MAP IS TO BE 'PATCHED' IT MUST BE DONE AFTER THE QUESTIONS ARE ANSWERED OR AFTER THE AUTO SIZE.

6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095

5.2 HOW TO RUN THE "CABLE/ECHO" TESTS.

NORMAL STARTING FOR THE FIRST TIME WOULD BE: LOAD ADDRESS 210; START WITH THE SWR EQUAL TO 003.

NOTE: SW00=1 ASKS FOR "VECTOR" AND "CSR"
SW01=1 ASKS FOR "WHICH TEST ECHO OR CABLE", "BAUD RATE", "LINE" UNDER TEST. PROGRAM WILL PRINT OUT:

"VECTOR ADDRESS-"

YOU TYPE VECTOR WITH A <CR>.

"CONTROL REGISTER ADDRESS-"

YOU TYPE IN DZCSR UNDER TEST.

"WHICH TEST ? ECHO OR CABLE (E OR C)"

LETS DO THE CABLE TEST FIRST. **THIS TEST IS ONLY TO BE DONE ON THE EIA VERSION OF THE DZ11 NOT THE 20MA VERSION". TYPE "C" <CR>

"BAUD RATE- "

TYPE EITHER 50, 110, 135, 150, 300, 600, 1200 1800, 2000, 2400, 3600, 4800, 7200, 9600 FOLLOWED BY <CR>

"LINE: "

YOU TYPE THE LINE WHICH HAS THE H325 TEST CONNECTOR. (TYPE EITHER 0, 1, 2, 3, 4, 5, 6, 7) PROGRAM WILL THEN PRINT:

"CABLE TEST"

AND IF EVERYTHING IS WORKING; THE FOLLOWING WILL BE PRINTED:

"PASS DONE."
"PASS DONE."
ETC.

TO CHANGE LINES; HIT ANY PRINTING KEY ON YOUR CONSOLE TERMINAL WHILE THE PROGRAM IS RUNNING AND THE FOLLOWING WILL BE PRINTED:

"LINE: "

NOW CHANGE THE H325 TEST CONNECTOR TO ANOTHER LINE AND TYPE THE NEW LINE. PROGRAM WILL THEN PRINT:

"CABLE TEST"
"PASS DONE."
"PASS DONE."

CONTINUE THIS OPERATION UNTIL ALL LINES ARE TESTED.

6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146

5.3 ECHO TEST

IF PROGRAM HAS ALREADY BEEN STARTED AT 210 AND THE VECTOR AND ADDRESS HAVE BEEN TYPED IN; JUST LOAD ADDRESS 210 AND START WITH SWR EQUAL TO 002. PROGRAM WILL PRINT:

'WHICH TEST ? ECHO OR CABLE (E OR C)''

NOW TYPE AN 'E' TO DO THE ECHO TEST. PROGRAM WILL PRINT:

'BAUD RATE-''

TYPE BAUD RATE AT WHICH THE TERMINAL IS SET THAT IS CONNECTED TO THE DZ11 DIST PNL. BAUD RATE CHOICES ARE: 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600. THE PROGRAM WILL THEN PRINT:

LINE: ''

TYPE THE LINE THE TERMINAL IS CONNECTED TO AT THE DIST PNL THEN THE PROGRAM WILL PRINT:

'TERMINAL ECHO TEST''

*** AT THIS POINT THE MESSAGE:

'THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789''

SHOULD BE PRINTED ON THE TERMINAL CONNECTED TO THE DZ11. IF THIS MESSAGE IS DESIRED TO BE CONTINUOUSLY OUTPUT; SET THE SWR TO 377 (SWR=377) WHILE IT IS BEING OUTPUT OR WHEN THE LINE NO. IS REQUESTED ABOVE. WHEN THIS MESSAGE IS DONE AND THE SWR IS NOT EQUAL TO 377; THE CONSOLE WILL PRINT:

'TYPE A CHAR. ON DZ11 TERMINAL''

ANY PRINTABLE CHAR HIT ON DZ11 TERMINAL SHOULD BE ECHOED BACK ON THE TERMINAL. **IF YOU HIT CNTRL C <^C> ON THE DZ11 TERMINAL THE PROGRAM WILL PRINT:

'PASS DONE. ''

ON THE CONSOLE TERMINAL AND THE 'QUICK BROWN FOX' WILL BE PRINTED ON DZ11 TERMINAL AGAIN AND THE ECHO TEST WILL BE RUNNING. TO CHANGE LINES: TYPE ANY PRINTABLE CHARACTER ON THE CONSOLE TERMINAL (NOT THE DZ11 TERMINAL). THE PROGRAM WILL AGAIN TYPE 'LINE: '' AND WAIT FOR A RESPONSE.

6148
6149
6150
6151
6152
6153
6154
6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
6188
6189
6190

5.4 PROGRAM AND/OR OPERATOR ACTION

THE VARIETY OF PROGRAM CONTROL SWITCHES PROVIDED IN THIS DIAGNOSTIC PACKAGE IS DESIGNED TO PROVIDE THE USER WITH A WIDE RANGE OF TROUBLE-SHOOTING TECHNIQUES. BEFORE THE USER ATTEMPTS TO RUN THIS DIAGNOSTIC HE SHOULD BECOME FAMILIAR WITH THE USE OF THESE CONTROL SWITCHES AND THEIR RESTRICTIONS. (SEE SEC. 4.1, 4.1.1, 4.1.2, 4.1.3)

WHEN THE PROGRAM DETECTS AN ERROR THE TEST NUMBER AND PC WILL BE TYPED OUT AND POSSIBLY AN ERROR MESSAGE (DEPENDING ON THE PARTICULAR ERROR). IF IT IS NECESSARY TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT THEN LOOK IN THE PROGRAM LISTING FOR THAT TEST NUMBER AND THEN NOTE THE PC OF THE ERROR REPORT. THE REASON FOR THE ERROR REPORT WILL BECOME CLEARER WHEN READING THE COMMENTS IN THE PROGRAM LISTING.

6. ERRORS

AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE A TEST NUMBER AND PC TYPED OUT AT THE TIME OF AN ERROR (PROVIDING SW 13=0 AND SW 12=0). IN MOST CASES ADDITIONAL INFORMATION WILL BE SUPPLIED TO THE THE ERROR MESSAGE WHICH IS TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.2 ERROR RECOVERY

IF FOR SOME REASON THE DZ11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR TO REGAIN CONTROL OF CPU. IF THIS SHOULD HAPPEN; LOOK IN LOCATION 'TSTNO' (ADDRESS 1216) FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE DZ11 WAS DOING AT THE TIME OF THE ERROR.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4.1.2
STATUS TABLE SHOULD BE VERIFIED REGARDLESS OF HOW PROGRAM WAS STARTED. ALSO IT IS IMPORTANT TO USE THIS LISTING ALONG WITH THE INFORMATION PRINTED ON THE TTY TO COMPLETELY ISOLATE PROBLEMS.

6192
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224

7.2 OPERATING RESTRICTIONS

PARAMETER MUST BE INPUT FROM USER OR APT IF "AUTO SIZING" IS NOT USED.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DZ11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 2 MIN. THIS IS ASSUMING SW11=1 (INHIBIT ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION. THE ACTUAL EXECUTION TIME DEPENDS GREATLY ON THE PDP11 CPU CONFIGURATION. AN 11/40 WITH CORE MEMORY WILL TAKE AROUND 100 SECONDS TO EXECUTE A PASS WITH NO ITERATIONS AND ABOUT 400 SECONDS TO EXECUTE A FULLY ITERATED PASS. ANY OTHER PDP11 CPU TYPE WILL EXECUTE A PASS IN TIME PROPORTIONAL TO THE EXECUTION SPEED OF THE CPU 'S MEMORY IN RELATION TO THAT OF AN 11/40.

8.2 PASS COMPLETE

NOTE: *EVERY* TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO *HARD* ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTIL ALL DZ11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CZDZA-G CSR: 160010 VEC: 300 PASSES: 000001 ERRORS:

NOTE: THE NUMBERS FOR CSR AND VEC ARE NOT NECESSARILY THE VALUES FOR THE DEVICE. THEY ARE ONLY FOR THIS EXAMPLE.

6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253

8.4 KEY LOCATIONS

\$LPADR (1126)

CONTAINS THE ADDRESS WHERE PROGRAM WILL RETURN WHEN ITERATION COUNT IS REACHED OR IF LOOP ON TEST IS ASSERTED.

NEXT (1360)

CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.

\$TSTNM (1122)

CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.

RUN (1406)

THE BIT IN 'RUN' ALWAYS POINTS ONE PAST THE DZ11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1304/0000000001000000 MEANS THAT DZ11 NO.05 IS THE DZ11 NOW RUNNING.

STATUS MAP
(1500)-(2000)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) DZ11S SEQUENTIALY. THEY CONTAIN THE CSR,VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DZ11.

DZACTV (1404)

EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DZ11 WILL BE TESTED IN TURN. EXAMPLE: (DZACTV) 1300/0000000000011111 MEANS THAT DZ11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DZACTV) 1300/0000000000010001 MEANS THAT DZ11 NO. 00,04 WILL BE TESTED.

\$BASE (1310)

CONTAINS THE RECEIVER CSR OF THE CURRENT DZ11 UNDER TEST.

6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299

8.4A MORE ON THAT 'STATUS TABLE' (1500-2000)

'MAP OF DZ11 STATUS'	
1500	160100
1502	000300
1504	000005
1506	000377
1510	017070
1512	000000

THE ABOVE INFORMATION WILL BE REPEATED FOR EACH OF UP TO 16 DZ11'S IN THE SYSTEM (THESE WILL FOLLOW UNDER THIS TABLE). EXPLANATION:

1500	160100	THIS IS THE SYSTEM CONTROL REGISTER FOR THE 1ST DZ11 IN THE SYSTEM.
1502	000300	THIS IS VECTOR 'A' FOR THE FIRST DZ11 IN THE SYSTEM.
1504	000005	THIS REPRESENTS THE BUS INTERRUPT PRIORITY LEVEL OF THE DZ11. BIT15 OF THIS LOCATION INDICATES EITHER EIA OR 20MA. IF BIT15=0 MODULE SHOULD BE AN M7819, IF BIT15=1 MODULE SHOULD BE AN M7814.
1506	000377	THIS IS THE BINARY REPRESENTATION OF WHAT LINES ARE TO BE TESTED.
1510	017070	THIS IS THE PARAMETER LOCATION USED IN MOST OF THE TESTS. IT INDICATES PARAMETERS OF: RX ON, SPEED SELECT 16 (9600 BAUD) EIGHT BITS PER CHAR, AND TWO STOP BITS. THE USER MAY ALTER THE STOP BITS AND THE SPEED, BUT THE REMAINING PARAMETERS SHOULD BE LEFT ALONE. THIS LOCATION IS USED TO LOAD THE DZ11 LINE PARAMETER REGISTER FOR EACH LINE. THE MEANING OF THE BITS SET IN THIS LOCATION IS THE SAME AS THE FUNCTION OF THE RELATED BITS IN THE DEVICE LINE PARAMETER REGISTER.
1512	000000	THIS LOCATION WILL CONTAIN EITHER ALL ZEROS INDICATING THAT INTERNAL LOOP WAS SELECTED AS MODE OF OPERATION OR IT WILL CONTAIN 10000 INDICATING THAT 'STAGGERED MODE' WAS SELECTED OR IT WILL CONTAIN 000200 INDICATING THAT 'EXTERNAL' WAS THE MODE SELECTED.

THE ABOVE IS REPEATED FOR EACH DZ11 IN THE SYSTEM. THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT PROGRAM AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION.

6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350
6351
6352

8.5 *** METHOD OF AUTO SIZING ***

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE PROGRAM WILL START AT ADDRESS 160000 AND START 'REFERENCING' THE ADDRESS IN THE POINTER. IF A NON-EX MEMORY TRAP OCCURES, THE POINTER (HOLDING 160000) IS UPDATED BY 10 AND THE ABOVE IS REPEATED UNTIL ADDRESS 163700 IS REACHED. IF A 'SLAVE SYNC RESPONSE' WAS ISSUED BY THE DZ11 (OR ANY OTHER DEVICE) (NO NXM TRAP), 'MASTER SCAN ENABLE' IS ATTEMPTED TO BE SET AND THE 'TCR' BIT FOR LINE 7 IS SET. 'TRDY' IS THEN TESTED TO BE SET AND BOTH 'TCR07' AND 'MASTER SCAN ENABLE' ARE TESTED TO BE STILL SET. IF ALL OF THIS WORKED; THEN A 'DEVICE CLEAR' IS ISSUED TESTING THAT THE BIT CAN BE READ BACK AND THAT AFTER SOME TIME IT SELF CLEARS. IF ALL OF THE ABOVE WORKED; THIS DEVICE IS ASSUMED TO BE A DZ11. IF ANY OF THE ABOVE FAILED; UPDATING OF THE POINTER IS DONE AND THE SEQUENCE IS REPEATED.

NOTE: IF THE PROGRAM DOES NOT FIND YOUR DZ11; SOMETHING IS WRONG AND AUTO SIZING SHOULD NOT BE DONE. AFTER IDENTIFYING A DZ11 THE PROGRAM THEN ATTEMPTS TO SET ALL DTR BITS IN DEVICE REGISTER 4. IF ANY DTR BITS DID SET THE MODULE IS ASSUMED TO BE AN EIA MODULE (M7819) OTHERWISE THE STATUS MAP ENTRY IS SET FOR 20MA (M7814).

8.5.2 FINDING THE VECTOR

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '+2' (NEXT ADDRESS). BIT14 AND BIT5 (TX INTERRUPT ENABLE AND MSTSCAN ENABLE) ARE SET INTO THE DZCSR. 'TCR07' IS THEN SET. A DELAY IS MADE AND IF NO INTERRUPT OCCURES (BECAUSE OF A BAD DZ11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED; THE PROGRAM SHOULD BE RE-SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERRUPT OCCURRED; THE ADDRESS TO WHICH THE DZ11 INTERRUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU; THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.5.3 PARAMETER ASSUMPTIONS.

SINCE TOO MUCH HARDWARE WOULD NEED TO BE TURNED ON TO SIZE THE REST OF THE PARAMETERS; THE PROGRAM MUST ASSUME THE REMAINING VARIATIONS. THE RESULT IF NOT TO YOUR SPECIFIC CONFIGURATION MAY BE ALTERED BY HAND (TOGGLE IN) IF DESIRED). IN THIS WAY 95 PERCENT OF THE PARAMETER SETUP WAS DONE BY THE PROGRAM, AND 5 PERCENT BY YOU.

THEREFORE:

- 1) BUS PRIORITY IS SET TO LEVEL5.
- 2) ALL EIGHT LINES ARE ASSUMED TO BE TESTED.
- 3) DEFAULT BAUD RATE IS SET TO 16 (9600 BAUD).
- 4) MODE OF OPERATION IS 'INTERNAL MODE'.

FOR ALL PARAMETER ADJUSTMENTS PLEASE REFER TO SECTION 8.4A FOR GREATER DETAIL.

6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378
6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409

9.0 RUNNING THE DZ11 DIAGNOSTIC UNDER APT

9.1.1 THE APT INTERFACE

CZDZA HAS BEEN REDESIGNED TO BE COMPATIBLE WITH THE APT-AUTOMATED PRODUCT TEST SYSTEM. IT CAN BE RUN AS A STANDALONE DIAGNOSTIC OR IN EITHER OF THE APT MODES. CERTAIN VARIABLES IN THE ORIGINAL APT MODULE WERE REASSIGNED TO THE AREAS SET ASIDE FOR APT INTERFACING. THESE NEW VARIABLES GENERALLY BEGIN WITH A DOLLAR SIGN (\$), E.G., \$DEVN, \$BASE.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

THE DIAGNOSTIC USES SEVERAL VARIABLES IN THE REGION SUBTITLED 'APT MAILBOX-ETABLE'. THESE VARIABLES ARE:

\$SWREG - USED IF A SOFTWARE SWITCH REGISTER IS DESIRED WHILE UNDER APT

\$VECT1 - USED TO SPECIFY THE INTERRUPT LEVEL AND THE FIRST VECTOR ADDRESS

\$BASE - USED TO INDICATE BOTTOM ADDRESS OF DZ11 UNDER TEST

\$DEVN - A BIT MAP REPRESENTING WHICH DZ11'S WILL BE TESTED

\$CDW1 - USED TO INDICATE WHICH LINES TO RUN ON ALL DZ11'S

\$DDW0 - EACH OF THE \$DDW WORDS DESCRIBES THE PARAMETERS (LPR) FOR A PARTICULAR DZ11, GOING UP TO 16 DZ11'S

9.1.3 RUNNING UNDER APT

THE USER SHOULD BE FAMILIAR WITH THE APT SYSTEM. THE APT TIMING PARAMETERS FOR THE DZ11 DIAGNOSTIC WERE BASED ON AN 11/40 PROCESSOR. IT MAY BE NECESSARY TO ADD A FEW MORE SECONDS IF THE DIAGNOSTIC IS OUT ON AN 11/05 PROCESSOR.

ALL OF THE VARIABLES MENTIONED IN SECTION 9.1.2 SHOULD BE SET UP PRIOR TO RUNNING THE DIAGNOSTIC UNDER APT.

NOTE

BE SURE \$BASE POINTS TO THE FIRST DZ11 BEFORE RUNNING

BASED ON THESE VALUES, THE DIAGNOSTIC WILL SET UP THE STATUS

C7DZA-GO
C7DZAG.P11

MACY11 30A(1052) 24-JUN-81 09:46 PAGE 74-1
24-JUN-81 09:45

F 2

SEQ 0018

6410

TABLE. THE USER IS THEN FREE TO MONITOR UNDER APT AS NORMAL.


```
(2) 177774 STKLMT= 177774 ::STACK LIMIT REGISTER
(2) 177772 PIRQ= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
(2) 177570 DSWR= 177570 ::HARDWARE SWITCH REGISTER
(2) 177570 DDISP= 177570 ::HARDWARE DISPLAY REGISTER
(2)
(2) ;*GENERAL PURPOSE REGISTER DEFINITIONS
(2) 000000 R0= %0 ::GENERAL REGISTER
(2) 000001 R1= %1 ::GENERAL REGISTER
(2) 000002 R2= %2 ::GENERAL REGISTER
(2) 000003 R3= %3 ::GENERAL REGISTER
(2) 000004 R4= %4 ::GENERAL REGISTER
(2) 000005 R5= %5 ::GENERAL REGISTER
(2) 000006 R6= %6 ::GENERAL REGISTER
(2) 000007 R7= %7 ::GENERAL REGISTER
(2) 000006 SP= %6 ::STACK POINTER
(2) 000007 PC= %7 ::PROGRAM COUNTER
(2)
(2) ;*PRIORITY LEVEL DEFINITIONS
(2) 000000 PR0= 0 ::PRIORITY LEVEL 0
(2) 000040 PR1= 40 ::PRIORITY LEVEL 1
(2) 000100 PR2= 100 ::PRIORITY LEVEL 2
(2) 000140 PR3= 140 ::PRIORITY LEVEL 3
(2) 000200 PR4= 200 ::PRIORITY LEVEL 4
(2) 000240 PR5= 240 ::PRIORITY LEVEL 5
(2) 000300 PR6= 300 ::PRIORITY LEVEL 6
(2) 000340 PR7= 340 ::PRIORITY LEVEL 7
(2)
(2) ;*'SWITCH REGISTER' SWITCH DEFINITIONS
(2) 100000 SW15= 100000
(2) 040000 SW14= 40000
(2) 020000 SW13= 20000
(2) 010000 SW12= 10000
(2) 004000 SW11= 4000
(2) 002000 SW10= 2000
(2) 001000 SW09= 1000
(2) 000400 SW08= 400
(2) 000200 SW07= 200
(2) 000100 SW06= 100
(2) 000040 SW05= 40
(2) 000020 SW04= 20
(2) 000010 SW03= 10
(2) 000004 SW02= 4
(2) 000002 SW01= 2
(2) 000001 SW00= 1
(2) .EQUIV SW09,SW9
(2) .EQUIV SW08,SW8
(2) .EQUIV SW07,SW7
(2) .EQUIV SW06,SW6
(2) .EQUIV SW05,SW5
(2) .EQUIV SW04,SW4
(2) .EQUIV SW03,SW3
(2) .EQUIV SW02,SW2
(2) .EQUIV SW01,SW1
(2) .EQUIV SW00,SW0
(2)
(2) ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
```

```
(2) 100000 BIT15= 100000
(2) J40000 BIT14= 40000
(2) 020000 BIT13= 20000
(2) 010000 BIT12= 10000
(2) 004000 BIT11= 4000
(2) 002000 BIT10= 2000
(2) 001000 BIT09= 1000
(2) 000400 BIT08= 400
(2) 000200 BIT07= 200
(2) 000100 BIT06= 100
(2) 000040 BIT05= 40
(2) 000020 BIT04= 20
(2) 000010 BIT03= 10
(2) 000004 BIT02= 4
(2) 000002 BIT01= 2
(2) 000001 BIT00= 1
(2) .EQUIV BIT09,BIT9
(2) .EQUIV BIT08,BIT8
(2) .EQUIV BIT07,BIT7
(2) .EQUIV BIT06,BIT6
(2) .EQUIV BIT05,BIT5
(2) .EQUIV BIT04,BIT4
(2) .EQUIV BIT03,BIT3
(2) .EQUIV BIT02,BIT2
(2) .EQUIV BIT01,BIT1
(2) .EQUIV BIT00,BIT0
```

```
(2) ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(2) 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(2) 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
(2) 000014 TBITVEC=14 ;: 'T' BIT
(2) 000014 TRTVEC= 14 ;:TRACE TRAP
(2) 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
(2) 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
(2) 000024 PWRVEC= 24 ;:POWER FAIL
(2) 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
(2) 000034 TRAPVEC=34 ;: 'TRAP' TRAP
(2) 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
(2) 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
(2) 000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR
```

```
(1) ;INSTRUCTION DEFINITIONS
(1) ;-----
```

```
(1) 005746 PUSH1SP=5746 ;:DECREMENT PROCESSOR STACK 1 WORD
(1) 005726 POP1SP=5726 ;:INCREMENT PROCESSOR STACK 1 WORD
(1) 010046 PUSHRO=10046 ;:SAVE R0 ON STACK
(1) 012600 POPRO=12600 ;:RESTORE R0 FROM STACK
(1) 024646 PUSH2SP=24646 ;:DECREMENT STACK TWICE
(1) 022626 PGP2SP=22626 ;:INCREMENT STACK TWICE
```

```
(1) ;DZ11 CONTROL AND STATUS REGISTER DEFINITIONS
(1) ;(DZCSR) BIT DEFINITIONS
(1) ;-----
```

```
(1) 00001C MAINT = BIT3 :MAINTENANCE MODE ENABLE
(1) 000020 DCLR=BIT4 :DEVICE CLEAR
(1) 000040 MSENAB=BIT5 :MASTER SCAN ENABLE
(1) 000100 RIE=BIT6 :RECEIVER INTERRUPT ENABLE
(1) 000200 RDONE=BIT7 :RECEIVER DONE
(1) 010000 SILOEN= BIT12 :SILO ALARM ENABLE
(1) 020000 SILOAL = BIT13 :SILO ALARM
(1) 040000 TIE=BIT14 :TRANSMITTER INTERRUPT ENABLE
(1) 100000 TRDY=BIT15 :TRANSMITTER READY

(1) :DZCSR WORD DEFINITIONS
(1) -----
(1) 000000 TL0=0 :TRANSMIT LINE 0
(1) 000400 TL1=BIT8 :TRANSMIT LINE 1
(1) 001000 TL2=BIT9 :TRANSMIT LINE 2
(1) 001400 TL3=BIT9!BIT8 :TRANSMIT LINE 3
(1) 002000 TL4=BIT10 :TRANSMIT LINE 4
(1) 002400 TL5=BIT10!BIT8 :TRANSMIT LINE 5
(1) 003000 TL6=BIT10!BIT9 :TRANSMIT LINE 6
(1) 003400 TL7=BIT10!BIT9!BIT8 ;TRANSMIT LINE 7

(1) :DZRBUF BIT DEFINITIONS
(1) -----
(1) 010000 PARER=BIT12 :PARITY ERROR
(1) 020000 FRMERR=BIT13 :FRAME ERROR
(1) 040000 OVRRUN=BIT14 :OVERRUN ERROR
(1) 100000 DVALID=BIT15 :DATA VALID

(1) :DZRBUF WORD DEFINITIONS
(1) -----
(1) 000000 RL0=0 :RECEIVER LINE 0
(1) 000400 RL1=BIT8 :RECEIVER LINE 1
(1) 001000 RL2=BIT9 :RECEIVER LINE 2
(1) 001400 RL3=BIT9!BIT8 :RECEIVER LINE 3
(1) 002000 RL4=BIT10 :RECEIVER LINE 4
(1) 002400 RL5=BIT10!BIT8 :RECEIVER LINE 5
(1) 003000 RL6=BIT10!BIT9 :RECEIVER LINE 6
(1) 003400 RL7=BIT10!BIT9!BIT8 ;RECEIVER LINE 7

(1) :DZLPR WORD DEFINITIONS
(1) -----
(1) 000000 LP0=0 :LINE PARAMETER 0
(1) 000001 LP1=BIT0 :LINE PARAMETER 1
(1) 000002 LP2=BIT1 :LINE PARAMETER 2
(1) 000003 LP3=BIT1!BIT0 :LINE PARAMETER 3
(1) 000004 LP4=BIT2 :LINE PARAMETER 4
(1) 000005 LP5=BIT2!BIT0 :LINE PARAMETER 5
(1) 000006 LP6=BIT2!BIT1 :LINE PARAMETER 6
(1) 000007 LP7=BIT2!BIT1!BIT0 ;LINE PARAMETER 7

(1) 000000 FIVE=0 :FIVE BITS/CHAR,1 STOP BIT
(1) 000010 SIX=BIT3 :SIX BITS/CHAR,1 STOP BIT
```

```

(1) 00002C SEVEN=BIT4 ;SEVEN BITS/CHAR,1 STOP BIT
(1) 000030 EIGHT=BIT4!BIT3 ;EIGHT BITS/CHAR,1 STOP BIT
(1) 000040 FIVES=BIT5 ;FIVE BITS/CHAR,2 STOP BITS
(1) 000050 SIXS=BIT5!BIT3 ;SIX BITS/CHAR,2 STOP BITS
(1) 000060 SEVENS=BIT5!BIT4 ;SEVEN BITS/CHAR, 2 STOP BITS
(1) 000070 EIGHTS=BIT5!BIT4!BIT3 ;EIGHT BITS/CHAR, 2 STOP BITS
(1)
(1) 000100 PARITY=BIT6 ;PARITY ENABLED
(1) 000200 ODDPAR=BIT7 ;ODD PARITY ENABLED
(1) 000000 ONESTOP=0 ;ONE STOP BIT ENABLED
(1) 000040 TWOSTOP=BIT5 ;TWO STOP BITS ENABLED
(1) 000000 EVEPAR=0 ;EVEN PARITY ENABLED
(1) 010000 RCVON=BIT12 ;ENABLE RECEIVER (RECEIVER ON)
(1)
(1) 000000 S50=0 ;SPEED 50 BAUD
(1) 000400 S75=BIT8 ;SPEED 75 BAUD
(1) 001000 S110=BIT9 ;SPEED 110 BAUD
(1) 001400 S134=BIT9!BIT8 ;SPEED 134.5 BAUD
(1) 002000 S150=BIT10 ;SPEED 150 BAUD
(1) 002400 S300=BIT10!BIT8 ;SPEED 300 BAUD
(1) 003000 S600=BIT10!BIT9 ;SPEED 600 BAUD
(1) 003400 S1200=BIT10!BIT9!BIT8 ;SPEED 1200 BAUD
(1) 004000 S1800=BIT11 ;SPEED 1800 BAUD
(1) 004400 S2000=BIT11!BIT8 ;SPEED 2000 BAUD
(1) 005000 S2400=BIT11!BIT9 ;SPEED 2400 BAUD
(1) 005400 S3600=BIT11!BIT9!BIT8 ;SPEED 3600 BAUD
(1) 006000 S4800=BIT11!BIT10 ;SPEED 4800 BAUD
(1) 006400 S7200=BIT11!BIT10!BIT8 ;SPEED 7200 BAUD
(1) 007000 S9600=BIT11!BIT10!BIT9 ;SPEED 9600 BAUD
(1) 007400 S19200=BIT11!BIT10!BIT9!BIT8 ;SPEED 19200 BAUD

```

:DZTCR BIT DEFINITIONS

```

(1)
(1)
(1) 000001 TCR0=BIT0 ;TCR0
(1) 000002 TCR1=BIT1 ;TCR1
(1) 000004 TCR2=BIT2 ;TCR2
(1) 000010 TCR3=BIT3 ;TCR3
(1) 000020 TCR4=BIT4 ;TCR4
(1) 000040 TCR5=BIT5 ;TCR5
(1) 000100 TCR6=BIT6 ;TCR6
(1) 000200 TCR7=BIT7 ;TCR7
(1) 000400 DTR0=BIT8 ;DTR0
(1) 001000 DTR1=BIT9 ;DTR1
(1) 002000 DTR2=BIT10 ;DTR2
(1) 004000 DTR3=BIT11 ;DTR3
(1) 010000 DTR4=BIT12 ;DTR4
(1) 020000 DTR5=BIT13 ;DTR5
(1) 040000 DTR6=BIT14 ;DTR6
(1) 100000 DTR7=BIT15 ;DTR7

```

:DZMSR BIT DEFINITIONS

```

(1)
(1)
(1) 000001 RING0=BIT0 ;RING INDICATED ON LINE 0
(1) 000002 RING1=BIT1 ;RING INDICATED ON LINE 1
(1) 000004 RING2=BIT2 ;RING INDICATED ON LINE 2
(1) 000010 RING3=BIT3 ;RING INDICATED ON LINE 3

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-4

GENERAL DEFINITIONS AND EQUIVALENCES

SEQ 0024

(1)	000020	RING4=BIT4	:RING INDICATED ON LINE 4
(1)	000040	RING5=BIT5	:RING INDICATED ON LINE 5
(1)	000100	RING6=BIT6	:RING INDICATED ON LINE 6
(1)	000200	RING7=BIT7	:RING INDICATED ON LINE 7
(1)	000400	C00=BIT8	:CARRIER PRESENT ON LINE 0
(1)	001000	C01=BIT9	:CARRIER PRESENT ON LINE 1
(1)	002000	C02=BIT10	:CARRIER PRESENT ON LINE 2
(1)	004000	C03=BIT11	:CARRIER PRESENT ON LINE 3
(1)	010000	C04=BIT12	:CARRIER PRESENT ON LINE 4
(1)	020000	C05=BIT13	:CARRIER PRESENT ON LINE 5
(1)	040000	C06=BIT14	:CARRIER PRESENT ON LINE 6
(1)	100000	C07=BIT15	:CARRIER PRESENT ON LINE 7

:DZTDR BIT DEFINITIONS

(1)		-----	
(1)		:DZTDR BIT DEFINITIONS	
(1)		-----	
(1)	000400	BRK0=BIT8	:BREAK FOR LINE 0
(1)	001000	BRK1=BIT9	:BREAK FOR LINE 1
(1)	002000	BRK2=BIT10	:BREAK FOR LINE 2
(1)	004000	BRK3=BIT11	:BREAK FOR LINE 3
(1)	010000	BRK4=BIT12	:BREAK FOR LINE 4
(1)	020000	BRK5=BIT13	:BREAK FOR LINE 5
(1)	040000	BRK6=BIT14	:BREAK FOR LINE 6
(1)	100000	BRK7=BIT15	:BREAK FOR LINE 7

:TABLE OF LOOP AROUND FUNCTIONS (H325)

(1)	:	-----	
(1)	:	I	^
(1)	:	V	^
(1)	:	REC	TRANS
(1)	:	DATA	DATA
(1)	:	-----	
(1)	:	I	^
(1)	:	V	^
(1)	:	CO	RTS
(1)	:	-----	
(1)	:	I	^
(1)	:	V	^
(1)	:	RING	DTR
(1)	:		

```

(1) ;:*****
(1) ;-----
(1) ; TRAPCATCHER FOR ILLEGAL INTERRUPTS
(1) ; THE STANDARD 'TRAP CATCHER' IS PLACED
(1) ; BETWEEN ADDRESS 0 TO ADDRESS 776.
(1) ; IT LOOKS LIKE 'PC+2 HALT'.
(1) ;-----
(1) ;:*****
(1)
(1) 000000 . =0
(1) ; STANDARD INTERRUPT VECTORS
(1) ;-----
(1)
(1) 000010 000010 . =10
(1) 000012 011440 SET.PS ; FAKE 'MTPS' INSTRUCTION TRAP
(1) 000012 000340 PR7 ; MAKE SURE PS IS PRIORITY 7
(1)
(1) 000020 000020 . =20
(1) 000020 005122 .SCOPE ; SCOPE LOOP HANDLER
(1) 000022 000340 PR7 ; HANDLE AT PRIORITY 7
(1) 000024 010320 $PWDRN ; POWER FAIL HANDLER
(1) 000026 000340 340 ; SERVICE AT PRIORITY LEVEL 7
(1) 000030 007230 $ERROR ; ERROR HANDLER
(1) 000032 000340 340 ; SERVICE AT PRIORITY LEVEL 7
(1) 000034 007122 .TRPSRV ; GENERAL HANDLER DISPATCH SERVICE
(1) 000036 000340 340 ; SERVICE AT PRIORITY LEVEL 7
(2) .SBTTL ACT11 HOOKS
(2)
(3) ;:*****
(2) ; HOOKS REQUIRED BY ACT11
(2) 000040 $SVPC= ; SAVE PC
(2) 000046 . =46
(2) 000046 005056 $ENDAD ; 1) SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(2) 000052 000052 . =52
(2) 000052 000000 .WORD 0 ; 2) SET LOC.52 TO ZERO
(2) 000052 000040 .=$SVPC ; RESTORE PC
(1)
(1) 000174 000174 . =174
(1) 000174 000000 DISPREG:0 ; SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 11S
(1) 000176 000000 SWREG: 0 ; SOFTWARE SWITCH REGISTER FOR SWITCHLESS 11S
(1) 000200 000200 . =200
(1) 000200 000137 002150 JMP .START ; GO TO START OF PROGRAM
(1) 000210 000210 . =210
(1) 000210 000137 024442 JMP XSTART ; GOTO CABLE TEST/ECHO TEST
(1)
(2)
(2) 001000 001000 . =1000
(2) 001000 005200 055103 055104 MTITLE: .ASCIZ <200><12>/CZDZA-GO/<200>/CZDZAGO DZ11 LN ASYNC MUX TSTS /<200>
(2)

```

```

(3)          .SBTTL COMMON TAGS
(3)
(4)          ::*****
(3)          ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(3)          ::*USED IN THE PROGRAM.
(3)
(3)          001120          .=1120
(3) 001120 001120          $CMTAG:          ::START OF COMMON TAGS
(3) 001120 000000          .WORD          0          ::CONTAINS THE TEST NUMBER
(3) 001122 000          $STNM: .BYTE          0          ::CONTAINS ERROR FLAG
(3) 001123 000          $ERFLG: .BYTE          0          ::CONTAINS SUBTEST ITERATION COUNT
(3) 001124 000000          $ICNT: .WORD          0          ::CONTAINS SCOPE LOOP ADDRESS
(3) 001126 000000          $LPADR: .WORD          0          ::CONTAINS SCOPE RETURN FOR ERRORS
(3) 001130 000000          $LPERR: .WORD          0          ::CONTAINS TOTAL ERRORS DETECTED
(3) 001132 000000          $ERTTL: .WORD          0          ::CONTAINS ITEM CONTROL BYTE
(3) 001134 000          $ITEMB: .BYTE          0          ::CONTAINS MAX. ERRORS PER TEST
(3) 001135 001          $ERMAX: .BYTE          1          ::CONTAINS PC OF LAST ERROR INSTRUCTION
(3) 001136 000000          $ERRPC: .WORD          0          ::CONTAINS ADDRESS OF 'GOOD' DATA
(3) 001140 000000          $GDADR: .WORD          0          ::CONTAINS ADDRESS OF 'BAD' DATA
(3) 001142 000000          $BDADR: .WORD          0          ::CONTAINS 'GOOD' DATA
(3) 001144 000000          $GDDAT: .WORD          0          ::CONTAINS 'BAD' DATA
(3) 001146 000000          $BDDAT: .WORD          0          ::RESERVED--NOT TO BE USED
(3) 001150 000000          .WORD          0
(3) 001152 000000          .WORD          0
(3) 001154 000          $AUTOB: .BYTE          0          ::AUTOMATIC MODE INDICATOR
(3) 001155 000          $INTAG: .BYTE          0          ::INTERRUPT MODE INDICATOR
(3) 001156 000000          .WORD          0
(3) 001160 177570          SWR: .WORD          DSWR          ::ADDRESS OF SWITCH REGISTER
(3) 001162 177570          DISPLAY: .WORD          DDISP          ::ADDRESS OF DISPLAY REGISTER
(3) 001164 177560          $TKS: 177560          ::TTY KBD STATUS
(3) 001166 177562          $TKB: 177562          ::TTY KBD BUFFER
(3) 001170 177564          $TPS: 177564          ::TTY PRINTER STATUS REG. ADDRESS
(3) 001172 177566          $TPB: 177566          ::TTY PRINTER BUFFER REG. ADDRESS
(3) 001174 000          $NULL: .BYTE          0          ::CONTAINS NULL CHARACTER FOR FILLS
(3) 001175 002          $FILLS: .BYTE          2          ::CONTAINS # OF FILLER CHARACTERS REQUIRED
(3) 001176 012          $FILLC: .BYTE          12          ::INSERT FILL CHARS. AFTER A 'LINE FEED'
(3) 001177 000          $TPFLG: .BYTE          0          ::"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(3) 001200 000000          $REGAD: .WORD          0          ::CONTAINS THE ADDRESS FROM
(3)          ::WHICH ($REGO) WAS OBTAINED
(5) 001202 000000          $REG0: .WORD          0          ::CONTAINS (($REGAD)+0)
(5) 001204 000000          $REG1: .WORD          0          ::CONTAINS (($REGAD)+2)
(5) 001206 000000          $REG2: .WORD          0          ::CONTAINS (($REGAD)+4)
(5) 001210 000000          $REG3: .WORD          0          ::CONTAINS (($REGAD)+6)
(5) 001212 000000          $REG4: .WORD          0          ::CONTAINS (($REGAD)+10)
(5) 001214 000000          $REG5: .WORD          0          ::CONTAINS (($REGAD)+12)
(5) 001216 000000          $TMP0: .WORD          0          ::USER DEFINED
(5) 001220 000000          $TMP1: .WORD          0          ::USER DEFINED
(5) 001222 000000          $TMP2: .WORD          0          ::USER DEFINED
(5) 001224 000000          $TMP3: .WORD          0          ::USER DEFINED
(3) 001226 000000          $TIMES: 0          ::MAX. NUMBER OF ITERATIONS
(3) 001230 077          $QUES: .ASCII /?/          ::QUESTION MARK
(3) 001231 015          $CRLF: .ASCII <15>          ::CARRIAGE RETURN
(3) 001232 000012          $LF: .ASCII <12>          ::LINE FEED
(4)          ::*****
(4)          .SBTTL APT MAILBOX-ETABLE
(4)

```



```

(3)          .SBTTL  ERROR POINTER TABLE
(3)
(3)          ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(3)          ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(3)          ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(3)          ;*NOTE1:      IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(3)          ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(3)
(3)          ;*      EM      ;;POINTS TO THE ERROR MESSAGE
(3)          ;*      DH      ;;POINTS TO THE DATA HEADER
(3)          ;*      DT      ;;POINTS TO THE DATA
(3)          ;*      DF      ;;POINTS TO THE DATA FORMAT
(3)
(3)          $ERRTB:
(3)
(2)          ;PROGRAM CONTROL PARAMETERS
(2)          ;-----
(2)          001360 000000      NEXT: 0          ;ADDRESS OF NEXT TEST TO BE EXECUTED
(2)          001362 000000      LOCK: 0          ;ADDRESS FOR LOCK ON CURRENT DATA
(2)
(2)          ;PROGRAM VARIABLES
(2)          ;-----
(2)          001364 000377      LINE: 377          ;DEFAULT ALL EIGHT LINES RUNNING
(2)          001366 017070      PAR: 17070         ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS, 9600 BAUD,NO PARIT
(2)          001370 000000      MODE: 0           ;DEFAULT MAINTENANCE MODE
(2)          001372 000000      SAVLIN: 0          ;LINE NUMBER
(2)          001374 000000      XMTLIN: 0         ;TRANSMISSION LINE NUMBER
(2)          001376 000000      XMTCNT: 0        ;COUNT OF WORDS IN A TRANSMISSION PATTERN
(2)          001400 000000      REGIST: 0         ;DEVICE ADDRESS STORAGE LOCATION
(2)          001402 000000      SAVPC: 0          ;PROGRAM COUNTER STORAGE
(2)          001404 000001      DZACTV: .BLKW 1    ;*DZ11'S SELECTED ACTIVE.
(2)          001406 00J001      RUN: 1          ;*POINTER ONE PAST RUNNING DEVICE.
(2)          001410 000001      DZNUM: .BLKB 1   ;*OCTAL NUMBER OF DZ11'S.
(2)          001411 001          SAVNUM: .BYTE 1    ;*WORKABLE NUMBER.
(2)          .EVEN
(2)          001412 001500      ACTIVE: DZ.MAP ;TABLE POINTER.

```

```

(2)
(2)
(2)
(2)
(2) 001414 000
(2) 001415 000
(2) 001416 000
(2) 001417 000
(2) 001420 000
(2) 001422
(2) 001422 000000
(2) 001424 000000
(2) 001426 000000
(2) 001430 000000
(2) 001432 000000
(2) 001434 000000
(2) 001436 000000
(2) 001440 000000
(2) 001442 000000
(2) 001444 000000
(2) 001446 000000
(2) 001450 000000
(2) 001452 000000
(2) 001454 000000
(2) 001456 000000
(2) 001460 000000
(2) 001462
(2)
(2)
(3)
(2)
(3)
(2) 001462
(2) 000024
(2) 000024 000200
(2) 000044
(2) 000044 001462
(2) 001462
(3)
(2)
(2)
(2)
(2) 001462
(2) 001462 000000
(2) 001464 001234
(2) 001466 000132
(2) 001470 000137
(2) 001472 000137
(2) 001474 000052
(1)
(1)
(1)
(1) 001500 001500
(3)

```

```

;PROGRAM CONTROL FLAGS
;-----
EIAFLG: .BYTE 0 ;0=EIA 100000=20MA
INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
HDRFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
MNTFLG: .BYTE 0 ;MAINTENANCE BIT SET FLAG
DNFLG: .BYTE 0 ;TRANSMISSION COMPLETION FLAG
.EVEN
;DATA VARIABLES
TD0: .WORD 0
TD1: .WORD 0
TD2: .WORD 0
TD3: .WORD 0
TD4: .WORD 0
TD5: .WORD 0
TD6: .WORD 0
TD7: .WORD 0
TR0: .WORD 0
TR1: .WORD 0
TR2: .WORD 0
TR3: .WORD 0
TR4: .WORD 0
TR5: .WORD 0
TR6: .WORD 0
TR7: .WORD 0
STOP:
.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:
$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADDR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM: .WORD 90. ;;RUN TIM OF LONGEST TEST
$PASTM: .WORD 95. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 95. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
;DZ11 STATUS TABLE AND ADDRESS ASSIGNMENTS
;-----

.=1500
DZ.MAP:

```

(3)	001500	000001	DZCR0:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 0
(3)	001502	000001	DZVC0:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 0
(3)	001504	000001	DZLV0:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001506	000001	LINE0:	.BLKW	1	;ALL LINES SELECTED
(3)	001510	000001	PAR0:	.BLKW	1	;PARAMETERS
(3)	001512	000001	MANT0:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001514	000001	DZCR1:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 1
(3)	001516	000001	DZVC1:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 1
(3)	001520	000001	DZLV1:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001522	000001	LINE1:	.BLKW	1	;ALL LINES SELECTED
(3)	001524	000001	PAR1:	.BLKW	1	;PARAMETERS
(3)	001526	000001	MANT1:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001530	000001	DZCR2:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 2
(3)	001532	000001	DZVC2:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 2
(3)	001534	000001	DZLV2:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001536	000001	LINE2:	.BLKW	1	;ALL LINES SELECTED
(3)	001540	000001	PAR2:	.BLKW	1	;PARAMETERS
(3)	001542	000001	MANT2:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001544	000001	DZCR3:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 3
(3)	001546	000001	DZVC3:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 3
(3)	001550	000001	DZLV3:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001552	000001	LINE3:	.BLKW	1	;ALL LINES SELECTED
(3)	001554	000001	PAR3:	.BLKW	1	;PARAMETERS
(3)	001556	000001	MANT3:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001560	000001	DZCR4:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 4
(3)	001562	000001	DZVC4:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 4
(3)	001564	000001	DZLV4:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001566	000001	LINE4:	.BLKW	1	;ALL LINES SELECTED
(3)	001570	000001	PAR4:	.BLKW	1	;PARAMETERS
(3)	001572	000001	MANT4:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001574	000001	DZCR5:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 5
(3)	001576	000001	DZVC5:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 5
(3)	001600	000001	DZLV5:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001602	000001	LINE5:	.BLKW	1	;ALL LINES SELECTED
(3)	001604	000001	PAR5:	.BLKW	1	;PARAMETERS
(3)	001606	000001	MANT5:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001610	000001	DZCR6:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 6
(3)	001612	000001	DZVC6:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 6
(3)	001614	000001	DZLV6:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001616	000001	LINE6:	.BLKW	1	;ALL LINES SELECTED
(3)	001620	000001	PAR6:	.BLKW	1	;PARAMETERS
(3)	001622	000001	MANT6:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						
(3)	001624	000001	DZCR7:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 7
(3)	001626	000001	DZVC7:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 7
(3)	001630	000001	DZLV7:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001632	000001	LINE7:	.BLKW	1	;ALL LINES SELECTED
(3)	001634	000001	PAR7:	.BLKW	1	;PARAMETERS
(3)	001636	000001	MANT7:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
(3)						

(3)	001640	000001	DZCR10: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 10
(3)	001642	000001	DZVC10: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 10
(3)	001644	000001	DZLV10: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001646	000001	LINE10: .BLKW	1	:ALL LINES SELECTED
(3)	001650	000001	PAR10: .BLKW	1	:PARAMETERS
(3)	001652	000001	MANT10: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001654	000001	DZCR11: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 11
(3)	001656	000001	DZVC11: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 11
(3)	001660	000001	DZLV11: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001662	000001	LINE11: .BLKW	1	:ALL LINES SELECTED
(3)	001664	000001	PAR11: .BLKW	1	:PARAMETERS
(3)	001666	000001	MANT11: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001670	000001	DZCR12: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 12
(3)	001672	000001	DZVC12: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 12
(3)	001674	000001	DZLV12: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001676	000001	LINE12: .BLKW	1	:ALL LINES SELECTED
(3)	001700	000001	PAR12: .BLKW	1	:PARAMETERS
(3)	001702	000001	MANT12: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001704	000001	DZCR13: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 13
(3)	001706	000001	DZVC13: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 13
(3)	001710	000001	DZLV13: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001712	000001	LINE13: .BLKW	1	:ALL LINES SELECTED
(3)	001714	000001	PAR13: .BLKW	1	:PARAMETERS
(3)	001716	000001	MANT13: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001720	000001	DZCR14: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 14
(3)	001722	000001	DZVC14: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 14
(3)	001724	000001	DZLV14: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001726	000001	LINE14: .BLKW	1	:ALL LINES SELECTED
(3)	001730	000001	PAR14: .BLKW	1	:PARAMETERS
(3)	001732	000001	MANT14: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001734	000001	DZCR15: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 15
(3)	001736	000001	DZVC15: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 15
(3)	001740	000001	DZLV15: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001742	000001	LINE15: .BLKW	1	:ALL LINES SELECTED
(3)	001744	000001	PAR15: .BLKW	1	:PARAMETERS
(3)	001746	000001	MANT15: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001750	000001	DZCR16: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 16
(3)	001752	000001	DZVC16: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 16
(3)	001754	000001	DZLV16: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001756	000001	LINE16: .BLKW	1	:ALL LINES SELECTED
(3)	001760	000001	PAR16: .BLKW	1	:PARAMETERS
(3)	001762	000001	MANT16: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001764	000001	DZCR17: .BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 17
(3)	001766	000001	DZVC17: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 17
(3)	001770	000001	DZLV17: .BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
(3)	001772	000001	LINE17: .BLKW	1	:ALL LINES SELECTED
(3)	001774	000001	PAR17: .BLKW	1	:PARAMETERS
(3)	001776	000001	MANT17: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(1)					

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-13
APT PARAMETER BLOCK

H 3

SEQ 0033

(1) 002000 177777

DZ.END: 177777

```

(1)                                     ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
(1)                                     ;POINTERS TO SUBROUTINES CAN BE FOUND
(1)                                     ;IN THE TABLE IMMEDIATELY FOLLOWING THE DEFINITIONS
(1)
(1)                                     ;:*****
(1)                                     ;-----
(1) 002002 104400 .TRPTAB:
(3) 002002 007216 ADVANCE=TRAP+0 ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
(3) 104401 .ADVANCE
(2) 002004 005376 SCOP1=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
(3) 104402 .SCOP1
(2) 002006 005422 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
(3) 104403 .TYPE
(2) 002010 006252 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
(3) 104404 .INSTR
(2) 002012 006426 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
(3) 104405 .INSTER
(2) 002014 006446 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
(3) 104406 .PARAM
(2) 002016 011152 SETFLG=TRAP+6 ;CALL TO SET FLAG ROUTINE
(3) 104407 .SETFLG
(2) 002020 006646 SAV05=TRAP+7 ;CALL TO REGISTER SAVE ROUTINE
(3) 104410 .SAV05
(2) 002022 006706 RES05=TRAP+10 ;CALL TO REGISTER RESTORE ROUTINE
(3) 104411 .RES05
(2) 002024 006740 CONVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE
(3) 104412 .CONVRT
(2) 002026 006744 CNVRT=TRAP+12 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
(3) 104413 .CNVRT
(2) 002030 007144 DEVICE.CLR=TRAP+13 ;CALL TO ISSUE A DEVICE CLEAR
(3) 104414 .DEVICE.CLR
(2) 002032 007176 DELAY=TRAP+14 ;CALL TO DELAY FOR FAST CPU'S
(3) 104415 .DELAY
(2) 002034 026170 PARMD=TRAP+15 ;CONVERT DECIMAL STRING TO OCTAL
(3) 104416 .PARMD
(2) 002036 026364 PAWCH=TRAP+16 ;SET FLAG ECHO OR CABLE
(3) 104417 .PAWCH
(2) 002040 007164 DCLASM=TRAP+17 ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(1) .DCLASM
(1)
(1)                                     ;:*****

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-15
APT PARAMETER BLOCK

SEQ 0035

```

(1)                                     ;DZ11 VECTOR AND REGISTER INDIRECT POINTERS
(1)                                     ;WORKING AREA
(1)
(1) 002042 160040 DZCSR: 160040 ;R/W
(1) 002044 160041 HDZCSR: 160041 ;R/W
(1) 002046 160042 DZRBUF: 160042 ;READ ONLY
(1) 002050 160043 HDZRBUF: 160043 ;READ ONLY
(1) 002052 160042 DZLPR: 160042 ;WRITE ONLY
(1) 002054 160043 HDZLPR: 160043 ;WRITE ONLY
(1) 002056 160044 DZTCR: 160044 ;R/W
(1) 002060 160045 HDZTCR: 160045 ;R/W
(1) 002062 160046 DZMSR: 160046 ;READ ONLY
(1) 002064 160047 HDZMSR: 160047 ;READ ONLY
(1) 002066 160046 DZTDR: 160046 ;WRITE ONLY
(1) 002070 160047 HDZTDR: 160047 ;WRITE ONLY
(1)                                     ;DEFAULT DZ VECTORS
(1) 002072 000300 DZRIV: 300 ;REC INTR VECTOR
(1) 002074 000302 DZRIS: 302 ;REC INTR STATUS
(1) 002076 000304 DZTIV: 304 ;XMIT INTR VECTOR
(1) 002100 000306 DZTIS: 306 ;XMIT INTR STATUS
(1)
(1)

```


CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-18

PROGRAM INITIALIZATION AND START UP.

SEQ 0038

```

(1) 002446 000137 004420      JMP      16$      ;GO PRINT DZ STATUS TABLE
(1) 002452 032777 000001 176500 30$: BIT      #SW00,@SWR ;RESELECT ?
(1) 002460 001011              BNE      32$      ;IF YES, GO SET UP THE INFORMATION
(1) 002462 122737 000377 001415  CMPB    #377,INIFLG ;ON 1ST START; MUST ANSWER QUESTION
(1) 002470 001003              BNE      .+10     ;IF NOT ANSWERING QUESTIONS
(1) 002472 105777 176462      TSTB    @SWR      ;ARE U AUTO SIZING?
(1) 002476 100402              BMI      32$      ;NO AUTO SIZE! NO SW00=1 ON 1ST START!
(1) 002500 000137 003244      JMP      73$      ;IF NO, SKIP THE INTERROGATION
(1) 002504 012700 001500 32$: MOV      #DZ.MAP,RO ;POINT TO THE BEGINNING OF THE MAP TABLE
(1) 002510 105037 001416      CLRB    HDRFLG    ;MAKE SURE A MAP GETS PRINTED
(1) 002514 005020 65$: CLR      (RO)+   ;CLEAR A TABLE LOCATION
(1) 002516 020027 002000      CMP      RO,#DZ.END ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1) 002522 001374              BNE      65$      ;IF NOT ,CLEAR THE NEXT LOCATION IN THE TABLE
(1) 002524 105337 001415      DECB    INIFLG    ;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!

```

;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
;TABLE AND SET UP THE DIAGNOSTIC.

;GET THE BASE ADDRESS OF THE DZ11'S

```

(1) 002530 33$:
(2) 002530 104403      INSTR    ;CALL THE STRING INPUT ROUTINE
(2) 002532 003464      66$     ;POINTER TO MESSAGE TO BE PRINTED
(2) 002534 104405      PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 002536 160000      160000  ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002540 163770      163770  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002542 001500      DZCRO    ;POINTER TO MAP LOCATION TO BE FILLED
(2) 002544 007         .BYTE    7     ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 002545 001         .BYTE    1     ;NUMBER OF PARAMETERS TO STORE
(1) 002546 013737 001500 001310  MOV      DZCRO,$BASE ;COPY BASE ADDRESS TO ETABLE

```

;GET THE BASE VECTOR ADDRESS

```

(1) 002554 34$:
(2) 002554 104403      INSTR    ;CALL THE STRING INPUT ROUTINE
(2) 002556 003530      67$     ;POINTER TO MESSAGE TO BE PRINTED
(2) 002560 104405      PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 002562 000300      300     ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002564 000776      776     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002566 001502      DZVCO    ;POINTER TO MAP LOCATION TO BE FILLED
(2) 002570 003         .BYTE    3     ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 002571 001         .BYTE    1     ;NUMBER OF PARAMETERS TO STORE
(1) 002572 013737 001502 001304  MOV      DZVCO,$VECT1 ;COPY VECTOR TO ETABLE

```

;GET THE BUS REQUEST LEVEL

```

(1) 002600
(2) 002600 104403      INSTR    ;CALL THE STRING INPUT ROUTINE
(2) 002602 003571      68$     ;POINTER TO MESSAGE TO BE PRINTED
(2) 002604 104405      PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 002606 000004      4       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002610 000007      7       ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 002612 001504      DZLVO    ;POINTER TO MAP LOCATION TO BE FILLED
(2) 002614 000         .BYTE    0     ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 002615 001         .BYTE    1     ;NUMBER OF PARAMETERS TO STORE
(1) 002616 113737 001504 001305  MOVB    DZLVO,$VECT1+1 ;GET BUS REQUEST LEVEL INTO ETABLE
(1) 002624 106337 001305      ASLB    $VECT1+1 ;ALIGN THE BITS PROPERLY

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-19
PROGRAM INITIALIZATION AND START UP.

SEQ 0039

```

(1) 002630 106337 001305 ASLB $VECT1+1 ;ALIGN THE BITS PROPERLY
(1) 002634 106337 001305 ASLB $VECT1+1 ;ALIGN THE BITS PROPERLY
(1) 002640 106337 001305 ASLB $VECT1+1 ;ALIGN THE BITS PROPERLY
(1) 002644 106337 001305 ASLB $VECT1+1 ;ALIGN THE BITS PROPERLY
(1)
(1) ;FIND OUT IF MODULE IS EIA OR 20 MA.
(1)
(1) 002650 104402 004260 TYPE ,74$ ;PRINT EIA MESSAGE
(1) 002654 005037 001220 CLR $TMP1 ;USE $TMP1
(1) 002660 105777 176300 80$: TSTB @$TKS ;IS KEYBOARD DONE?
(1) 002664 100375 BPL 80$ ;IF NOT, WAIT FOR IT
(1) 002666 017746 176274 MOV @$TKB,-(SP) ;IF YES, PUT CHARACETR ON STACK
(1) 002672 042716 177600 BIC #177600,(SP) ;STRIP DOWN CHARACTER ;:DSH-BHL
(1) 002676 122716 000023 CMPB #$XOFF,(SP) ;IS IT A XOFF? ;:DSH-BHL
(1) 002702 001014 BNE 83$ ;BR IF NOT ;:DSH
(1) 002704 105777 176254 101$: TSTB @$TKS ;WAIT FOR A CHARACTER ;:DSH-BHL
(1) 002710 100375 BPL 101$ ;:DSH-BHL
(1) 002712 117716 176250 MOVB @$TKB,(SP) ;GET CHARACTER ;:DSH-BHL
(1) 002716 042716 177600 BIC #177600,(SP) ;STRIP DOWN CHARACTER ;:DSH-BHL
(1) 002722 122716 000021 CMPB #$XON,(SP) ;WAIT FOR A XON? ;:DSH-BHL
(1) 002726 001366 BNE 101$ ;GET NEXT CHAR IF NOT ;:DSH-BHL
(1) 002730 005726 TST (SP)+ ;POP STACK ;:DSH
(1) 002732 000752 BR 80$ ;WAIT FOR A CHAR ;:DSH
(1) 002734 122716 000021 83$: CMPB #$XON,(SP) ;IS IT A RANDOM XON ;:DSH-BHL
(1) 002740 001002 BNE 102$ ;BR IF NO ;:DSH-BHL
(1) 002742 005726 TST (SP)+ ;ELSE, POP STACK ;:DSH-BHL
(1) 002744 000745 BR 80$ ;GO GET NEXT CHAR ;:DSH-BHL
(1) 002746 122726 000015 102$: CMPB #15,(SP)+ ;IS IT <CR> ?
(1) 002752 001414 BEQ 81$ ;IF SO, GET OUT
(1) 002754 014677 176212 MOV -(SP),@$TPB ;IF NOT, PRINT CHARACTER
(1) 002760 042737 100000 001504 BIC #BIT15,DZLV0 ;CLEAR EIA FLAG
(1) 002766 122726 000102 CMPB #102,(SP)+ ;IS IT A B?
(1) 002772 001332 BNE 80$ ;IF NOT, GO BACK FOR INPUT
(1) 002774 052737 100000 001504 BIS #BIT15,DZLV0 ;IF SQ, SET FLAG
(1) 003002 000726 BR 80$ ;GET MORE INPUT
(1) 003004 81$:
(1) ;GET THE MODE OF OPERATION (E,I,S)
(1)
(1)
(2) 003004 104403 INSTR ;CALL THE STRING INPUT ROUTINE
(2) 003006 004002 72$ ;POINTER TO THE MESSAGE TO BE PRINTED
(2) 003010 104406 SETFLG ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
(2) 003012 001512 MANT0 ;THIS IS THE FLAG BEING SETUP
(1)
(1) ;GET THE NUMBER OF 'DZ11'S RUNNING
(1)
(2) 003014 104403 INSTR ;CALL THE STRING INPUT ROUTINE
(2) 003016 003740 71$ ;POINTER TO MESSAGE TO BE PRINTED
(2) 003020 104405 PARAM ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 003022 000001 1 ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003024 000020 16. ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003026 001220 $TMP1 ;POINTER TO MAP LOCATION TO BE FILLED
(2) 003030 000 .BYTE 0 ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 003031 001 .BYTE 1 ;NUMBER OF PARAMETERS TO STORE
(1)
(1) 003032 012737 000377 001506 MOV #377,LIN0 ;SET UP DEFAULT LINES

```

```

(1) 003040 012737 017070 001510      MOV      #17070,PARO      ;SET UP DEFAULT LPR PARAMETER
(1)                                     ;RECEIVER ON; 9600 BAUD; 2STOP BITS; 8 BIT/CHAR
(1) 003046 012737 000001 007214      MOV      #1,DLYCNT      ;INITIALIZE DELAY COUNT
(1) 003054 032777 000010 176076      BIT      #SW03,@SWR     ;DO YOU WANT PARAMETERS?
(1) 003062 001402                                     BEQ      40$             ;IF NO, SKIP THE PARAMETER CALL
(1) 003064 004737 003274                                     JSR      PC,23$         ;GET PARAMETERS
(1) 003070 012737 000001 001312 40$:  MOV      #1,$DEVN      ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
(1) 003076 113737 001220 001410      MOVVB   $TMP1,DZNUM     ;COPY THE NUMBER OF DEVICES
(1) 003104 113737 001220 001411      MOVVB   $TMP1,SAVNUM    ;COPY A BACKUP NUMBER
(1) 003112 005337 001220 62$:        DEC      $TMP1          ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
(1) 003116 001404                                     BEQ      61$             ;SELECTED DEVICES
(1) 003120 000261                                     SEC                                     ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
(1) 003122 006137 001312      ROL      $DEVN          ;POINT TO THE NEXT DEVICE
(1) 003126 000771      BR      62$             ;GO DO THIS PROCEDURE AGAIN
(1) 003130 013737 001312 001222 61$:  MOV      $DEVN,$TMP2    ;# OF TIMES
(1) 003136 013737 001312 001404      MOV      $DEVN,DZACTV   ;COPY THE ACTIVE DEVICE PARAMETER
(1) 003144 012700 001500      MOV      #DZCR0,R0     ;SET A POINTER TO THE SPECIFIED INFORMATION
(1) 003150 012701 001514      MOV      #DZCR1,R1     ;POINT R1 TO THE REST OF THE MAP TABLE
(1) 003154 012702 001320      MOV      #SDDW0,R2     ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
(1) 003160 000241      CLC                                     ;INITIALIZE THE 'C' BIT FOR A ROTATION
(1) 003162 006037 001222      ROR      $TMP2          ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
(1) 003166 006237 001222 64$:        ASR      $TMP2          ;ISOLATE A SELECTION FLAG IN THE 'C' BIT
(1) 003172 103404      BCS      41$             ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
(1) 003174 012711 177777      MOV      #-1,(R1)      ;TERMINATE THE LIST
(1) 003200 000137 004374      JMP      63$             ;GO TO THE NEXT BLOCK
(1) 003204 012011 41$:        MOV      (R0)+,(R1)     ;ADDRESS
(1) 003206 062721 000010      ADD      #10,(R1)+     ;POINT TO THE NEXT DZ11 ADDRESS VALUE
(1) 003212 012011      MOV      (R0)+,(R1)     ;VECTOR
(1) 003214 062721 000010      ADD      #10,(R1)+     ;POINT TO THE NEXT VECTOR VALUE
(1) 003220 012021      MOV      (R0)+,(R1)+   ;LEVEL
(1) 003222 012021      MOV      (R0)+,(R1)+   ;LINES
(1) 003224 016012 177774      MOV      -4(R0),(R2)   ;GET THE EIA FLAG FROM THE PRIORITY WORD
(1) 003230 042712 077777      BIC      #77777,(R2)   ;ISOLATE THAT FLAG
(1) 003234 051022      BIS      (R0),(R2)+    ;ADD PARAMETERS TO DEVICE DESCRIPTOR WORD
(1) 003236 012021      MOV      (R0)+,(R1)+   ;PARAMETERS
(1) 003240 012021      MOV      (R0)+,(R1)+   ;MAINTENANCE MODE
(1) 003242 000751      BR      64$
(1) 003244 032777 000010 175706 73$:  BIT      #SW03,@SWR     ;ASK PARAMETERS ?
(1) 003252 001002      BNE      42$             ;IF NO, GO DO AUTO SIZING
(1) 003254 000137 004374      JMP      63$             ;GO SET UP FOR AUTO SIZING
(1) 003260 004737 003274 42$:        JSR      PC,23$         ;GO ASK PARAMETERS
(1) 003264 105337 001415      DECB    INIFLG         ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
(1) 003270 000137 004420      JMP      16$             ;GO TO THE NEXT BLOCK
(1)                                     ;GET THE ACTIVE LINES PARAMETER
(1) 23$:
(2) 003274 104403      INSTR                                     ;CALL THE STRING INPUT ROUTINE
(2) 003276 003614      69$                                     ;POINTER TO MESSAGE TO BE PRINTED
(2) 003300 104405      PARAM                                    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 003302 000001      |                                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003304 000377      377                                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003306 001506      LINE0                                    ;POINTER TO MAP LOCATION TO BE FILLED
(2) 003310 000      .BYTE 0                                ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 003311 001      .BYTE 1                                ;NUMBER OF PARAMETERS TO STORE
(1) 003312 105037 001416      CLRB   HDRFLG          ;MAKE SURE THE CHANGES ARE PRINTED

```

```

(1)
(1) ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
(1) ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
(1)
(1) 003316 005737 001512 TST MANTO ;IS STAGGERED THE MODE OF OPERATION?
(1) 003322 100021 BPL 26$ ;IF NOT, SKIP THIS SEGMENT
(1) 003324 013703 001506 MOV LINE0,R3 ;GET A SCRATCH COPY OF THE ACTIVE LINES
(1) 003330 006003 24$: ROR R3 ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
(1) 003332 103410 BCS 25$ ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
(1) 003334 001414 BEQ 26$ ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
(1) 003336 006203 ASR R3 ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
(1) 003340 103373 BCC 24$ ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
(1) 003342 104402 001230 27$: TYPE ,SQUES ;THIS IS AN INCORRECT PARAMETER
(1) 003346 104402 011075 TYPE ,MBADLN ;LET THE USER KNOW ABOUT IT
(1) 003352 000750 BR 23$ ;GO GET THE CORRECT PARAMETER
(1) 003354 001772 25$: BEQ 27$ ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
(1) 003356 006203 ASR R3 ;GET THE NEXT FLAG
(1) 003360 103370 BCC 27$ ;IF IT ISN'T SET, THERE'S AN ERROR
(1) 003362 000241 CLC ;INITIALIZE THE 'C' BIT FOR TESTING OF THE NEXT PAIR
(1) 003364 000761 BR 24$ ;GO TEST THE NEXT PAIR OF FLAGS
(1)
(1) ;GET THE LINE PARAMETER REGISTER ARGUMENT
(1)
(1) 003366 26$:
(2) 003366 104403 INSTR ;CALL THE STRING INPUT ROUTINE
(2) 003370 003670 70$ ;POINTER TO MESSAGE TO BE PRINTED
(2) 003372 104405 PARAM ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 003374 000000 0 ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003376 000017 17 ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 003400 001510 PAR0 ;POINTER TO MAP LOCATION TO BE FILLED
(2) 003402 000 .BYTE 0 ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 003403 001 .BYTE 1 ;NUMBER OF PARAMETERS TO STORE
(1) 003404 012702 001506 MOV #LINE0,R2 ;POINT TO THE LINE SELECTION PARAMETER
(1) 003410 012703 001510 MOV #PAR0,R3 ;POINT TO THE CHOSEN PARAMETERS
(1) 003414 011304 MOV (R3),R4 ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
(1) 003416 006304 ASL R4 ;ALIGN INDEX ON WORD BOUNDARY
(1) 003420 016437 031522 007214 MOV DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
(1) 003426 000313 SWAB (R3) ;PLACE IN HIGH BYTE
(1) 003430 052713 010070 BIS #10070,(R3) ;PLACE EXTRA PARAMETERS INTO LOC
(1) 003434 011262 000014 28$: MOV (R2),14(R2) ;LOAD THE LINES
(1) 003440 011363 000014 MOV (R3),14(R3) ;LOAD THE PARAMETERS
(1) 003444 062702 000014 ADD #14,R2 ;POINT TO THE NEXT SET
(1) 003450 062703 000014 ADD #14,R3 ;.. OF BOTH PARAMETERS
(1) 003454 020327 001774 CMP R3,#PAR17 ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1) 003460 001365 BNE 28$ ;IF NOT, GO LOAD SOME MORE PARAMETERS
(1) 003462 000207 RTS PC ;RETURN TO CALLING BLOCK
(1) 003464 030600 052123 041440 66$: .ASCIZ <200>/1ST CSR ADDRESS (160000:163700): /
(1) 003530 030600 052123 053040 67$: .ASCIZ <200>/1ST VECTOR ADDRESS (300:770): /
(1) 003571 200 051102 046040 68$: .ASCIZ <200>/BR LEVEL (4:6): /
(1) 003614 046200 047111 051505 69$: .ASCIZ <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:377): /
(1) 003670 042200 043105 052501 70$: .ASCIZ <200>/DEFAULT BAUD RATE <IN OCTAL>(00:16): /
(1) 003740 021600 047440 020106 71$: .ASCIZ <200>/# OF DZ11'S <IN OCTAL> (1:20): /
(1) 004002 046600 044501 052116 72$: .ASCII <200>/MAINTENANCE MODE/
(1) 004023 200 055440 054105 .ASCII <200>/ [EXTERNAL <H325>-EIA ONLY (E)]/
(1) 004071 200 055440 047111 .ASCII <200>/ [INTERNAL <DZCSR03=1> (I)]/
(1) 004137 200 055440 052123 .ASCII <200>/ [STAGGERED <H3271>-EIA ONLY (S)]: /

```

```

(1) 004207      200 055440 052123      .ASCIZ <200>/ [STAGGERED <H3190>-20MA ONLY (S)]: /
(1) 004260 052200 050131 020105 74$: .ASCIZ <200>/TYPE 'A' FOR EIA MODULE OR 'B' FOR 20 MA (A:B): /
(1) 004342 042600 052116 051105 75$: .ASCIZ <200>/ENTER DELAY PARAMETER: /
(1) 004374      004374      .EVEN
(1) 004374      122737 000377 001415 63$:
(1) 004402 001006      CMPB #377,INIFLG ;ONLY DO AUTO SIZE ON 1ST START
(1) 004404 032777 000200 174546      BNE 16$ ;
(1) 004412 001002      BIT #BIT7,@SWR ;BIT7=1??
(1) 004414 004737 012264      BNE 16$ ;BR IF NO AUTO SIZE
(1) 004420 105737 001416      JSR PC,AUTO.SIZE ;GO DO THE AUTO SIZE
(1) 004424 001021      TSTB HDRFLG ;HAS THE TABLE BEEN TYPED YET?
(1) 004426 105337 001416      BNE 1$ ;IF SO, DON'T TYPE IT AGAIN
(1) 004432 104402 011050      DECB HDRFLG ;INDICATE THAT THE TABLE WILL BE TYPED
(1) 004436 012700 001500      TYPE ,XHEAD ;TYPE MAP HEADER
(1) 004442 010037 001220      MOV #DZ.MAP,RO ;SET POINTER
(1) 004446 012037 001222      MOV RO,$TMP1 ;POINT TO THE MAP LOCATION
(1) 004452 022737 177777 001222      MOV (RO)+,$TMP2 ;SET DATA
(1) 004460 001403      CMP #-1,$TMP2 ;END OF LIST?
(1) 004462 104411      BEQ 1$ ;BR IF YES
(1) 004464 011140      CONVRT ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
(1) 004466 000765      XSTATQ ;CONVERT THE DATA AT THIS ADDRESS
(1) 004470 005737 000042      BR 5$ ;GO PRINT THE NEXT PARAMETER
(1) 004474 001026      TST @#42 ;IS PROGRAM RUNNING UNDER MONITOR
(1) 004476 032777 000100 174454      BNE 3$ ;YES
(1) 004504 001422      BIT #SW06,@SWR ;DESELECT SPECIFIC DEVICES??
(1) 004506 104402 010771      BEQ 3$ ;BR IF NO.
(1) 004512 005000      TYPE ,MNEW ;TYPE THE MESSAGE.
(1) 004514 000000      CLR RO ;ZERO DATA DISPLAY
(1) 004516 027737 174436 001312      HALT ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
(1) 004524 101404      CMP @SWR,$DEVM ;IS THE NUMBER VALID?
(1) 004526 104402 010643      BLOS 2$ ;BR IF NUMBER IS OK.
(1) 004532 000000      TYPE ,MERR3 ;TELL USER OF INVALID NUMBER.
(1) 004534 000776      HALT ;STOP EVERY THING.
(1) 004536 017737 174416 001404      BR 9$ ;RESTART THE PROGRAM AGAIN.
(1) 004544 013700 001404      MOV @SWR,DZACTV ;GET NEW DEVICE PATTERN
(1) 004550 000000      MOV DZACTV,RO ;SHOW THE USER WHAT HE SELECTED.
(1) 004552 032777 000020 174400      HALT ;CONTINUE DYNAMIC SWITCHES.
(1) 004560 001407      BIT #SW04,@SWR ;CHECK TO SEE IF DELAY COUNT CHANGES
(1) 004562 104403      BEQ 18$ ;IF NOT, GO CLEAR VECTOR AREA
(1) 004564 004342      INSTR ;CALL THE STRING INPUT ROUTINE
(1) 004566 104405      75$ ;POINTER TO MESSAGE TO BE PRINTED
(1) 004570 000001      PARAM ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(1) 004572 177777      1 ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(1) 004574 007214      177777 ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(1) 004576 000      DLYCNT ;POINTER TO MAP LOCATION TO BE FILLED
(1) 004577 001      .BYTE 0 ;MASK OF INVALID BITS FOR THIS PARAMETER
(1) 004600 012700 000300      .BYTE 1 ;NUMBER OF PARAMETERS TO STORE
(1) 004604 012701 000302      MOV #300,RO ;PREPARE TO CLEAR THE FLOATING
(1) 004610 010120      MOV #302,R1 ;VECTOR AREA. 300-776
(1) 004612 005021      MOV R1,(R0)+ ;START PUTTING 'PC+2 - HALT'
(1) 004614 022021      CLR (R1)+ ;IN VECTOR AREA.
(1) 004616 022700 001000      CMP (R0)+,(R1)+ ;POP POINTERS
(1) 004622 001372      CMP #1000,RO ;ALL DONE??
(1) 004622 001372      BNE 4$ ;BR IF NO.
(1) ;TEST START AND RESTART

```

PROGRAM INITIALIZATION AND START UP.

```
(1) ;-----  
(1)  
(1) 004624 012706 001120 .BEGIN: MOV #STACK,SP ;SET UP STACK  
(1) 004630 106427 000340 MTPS #PR7 ;LOCK OUT INTERRUPTS  
(1) 004634 005737 000042 TST @#42 ;IS PROGRAM UNDER MONITOR CONTROL  
(1) 004640 001015 BNE 2$ ;BR IF YES  
(1) 004642 032777 000004 174310 BIT #BIT2,@SWR ;CHECK FOR LOCK ON TEST  
(1) 004650 001406 BEQ 1$ ;BR IF NO LOCK DESIRED.  
(1) 004652 104402 010667 TYPE ,MLOCK ;TYPE LOCK SELECTED.  
(1) 004656 012737 000240 005140 MOV #NOP,TTST ;ADJUST SCOPE ROUTINE.  
(1) 004664 000403 BR 2$ ;CONTINUE ALONG.  
(1) 004666 013737 005372 005140 1$: MOV BRW,TTST ;PREPARE NORMAL SCOPE ROUTINE  
(1) 004674 012737 011542 001126 2$: MOV #CYCLE,$LPADR ;START AT "CYCLE" FIND WHICH DEVICE TO TEST  
(1) 004702 104402 010560 TYPE ,MR ;TYPE "RUNNING"  
(1) 004706 000177 174214 JMP @LPADR ;START TESTING
```

8617

(2)
(2)
(2)
(2)
(3)
(3)
(4)
(3)
(3)
(3)
(3)
(3)
(5) 004712
(5) 004712
(5) 004714
(5) 004720
(5) 004724
(5) 004730
(5) 004734
(5) 004740
(5) 004744
(5) 004750
(5) 004754
(5) 004760
(5) 004764
(5) 004770
(5) 004774
(5) 005000
(5) 005004
(5) 005006
(3) 005014
(3) 005020
(3) 005024
(3) 005032
(3) 005034
(3) 005036
(3) 005040
(3) 005042
(3) 005044
(3) 005046
(3) 005052
(3) 005054
(3) 005056
(3) 005060
(3) 005062
(3) 005064
(3) 005066
(3) 005066
(3) 005070
(2)
(2) 005072
(2) 005074
(2) 005076
(2) 005100
(2) 005102
(2) 005104

000004
005037 001136
105037 001123
104402 010535
104402 010716
104412 005072
104402 010724
104412 005100
005237 001242
104402 010732
104412 005106
005337 001242
104402 010743
104412 005114
105337 001411
001030
113737 001410 001411
005037 001226
005237 001242
042737 100000 001242
000001
003013
012737
000001
005034 000042
013700
001405
000005
004710
000240
000240
000240
000137
011542

000001
006 002
002042
000001
003 002
002072

```

:END OF PASS
:TYPE NAME OF TEST
:UPDATE PASS COUNT
:CHECK FOR EXIT TO ACT-11
:RESTART TEST
.SBTTL END OF PASS ROUTINE

:*****
:*INCREMENT THE PASS NUMBER ($PASS)
:*IF THERES A MONITOR GO TO IT
:*IF THERE ISN'T JUMP TO CYCLE

$EOP:
SCOPE
CLR $ERRPC ;CLEAR LAST ERROR PC
CLR $ERFLG ;CLEAR ERROR FLAG
TYPE ,MEPASS ;TYPE END PASS
TYPE ,MCSRX ;TYPE CSR
CNVRT ,XCSR ;SHOW IT
TYPE ,MVECX ;TYPE VECTOR
CNVRT ,XVEC ;SHOW IT
INC $PASS ;RAISE PASS COUNT
TYPE ,MPASSX ;TYPE PASSES
CNVRT ,XPASS ;SHOW IT
DEC $PASS ;RESTORE PASS COUNT
TYPE ,MERRX ;TYPE ERRORS
CNVRT ,XERR ;SHOW IT
DECB SAVNUM ;ARE ALL DEVICES TESTED?
BNE $DOAGN ;BR IF NO.
MOVB DZNUM,SAVNUM ;RESTORE THE COUNT
CLR $TIMES ;:ZERO THE NUMBER OF ITERATIONS
INC $PASS ;:INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;:DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;:LOOP?

$EOPCT: .WORD 1
BGT $DOAGN ;:YES
MOV (PC)+,a(PC)+ ;:RESTORE COUNTER

$ENDCT: .WORD 1

$GET42: MOV @#42,R0 ;:GET MONITOR ADDRESS
BEQ $DOAGN ;:BRANCH IF NO MONITOR
RESET ;:CLEAR THE WORLD
$ENDAD: JSR PC,(R0) ;:GO TO MONITOR
NOP ;:SAVE ROOM
NOP ;:FOR
NOP ;:ACT11

$DOAGN: JMP a(PC)+ ;:RETURN

$RTNAD: .WORD CYCLE

XCSR: 1
.BYTE 6,2
DZCSR

XVEC: 1
.BYTE 3,2
DZRIV
```

(2) 005106 000001
 (2) 005110 006 002
 (2) 005112 001242
 (2) 005114 000001
 (2) 005116 006 002
 (2) 005120 001132

XPASS: 1
 .BYTE 6,2
 \$PASS
 XERR: 1
 .BYTE 6,2
 \$ERTTL

:SCOPE LOOP AND ITERATION HANDLER
 :-----

.SBTTL SCOPE HANDLER ROUTINE

::*****
 :*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
 :*CALL
 :* SCOPE ;:SCOPE=IOT

(3) 005122
 (5) 005122 004737 007652
 (5) 005126 005037 001136
 (5) 005132 022716 013050
 (5) 005136 001417
 (5) 005140 000412
 (5) 005142 105777 174016
 (5) 005146 100073
 (5) 005150 127727 174012 000021
 (5) 005156 001467
 (5) 005160 017766 174002 177776
 (3) 005166 032777 040000 173764
 (3) 005174 001060
 (3) 005176 000416
 (3) 005200 013746 000004
 (3) 005204 012737 005224 000004
 (3) 005212 005737 177060
 (3) 005216 012637 000004
 (3) 005222 000436
 (3) 005224 022626
 (3) 005226 012637 000004
 (3) 005232 000441
 (3) 005234
 (3) 005234 105737 001123
 (3) 005240 001404
 (3) 005242 105037 001123
 (3) 005246 005037 001226
 (3) 005252 032777 004000 173700
 (3) 005260 001011
 (3) 005262 005737 001242
 (3) 005266 001406
 (3) 005270 005237 001124

\$SCOPE:
 .SCOPE: JSR PC,SERV.G ;FIND OUT IF <^G> WAS HIT
 CLR \$ERRPC ;CLEAR LAST ERROR PC.
 CMP #TST1+2,(SP) ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
 BEQ \$XTSTR ;IF SO, DON'T LOOP ON IT
 TTST: BR 1\$;GOTO 1\$ (IF LOCK SW02=1; THIS LOC =240)
 TSTB @\$TKS ;KEYBOARD DONE?
 BPL \$OVER ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
 CMPB @\$TKB,#\$XON ;IS CHAR A RANDOM XON ? ;:DSH
 BEQ \$OVER ;BR IF YES ;:DSH
 MOV @\$TKB,-2(SP)
 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 \$OVER ;:LOOP ON THE PRESENT TEST
 6\$:;#####END OF CODE FOR THE XOR TESTER#####
 2\$: TSTB \$ERFLG ;:HAS AN ERROR OCCURRED?
 BEQ 3\$;:BR IF NO
 4\$: CLRB \$ERFLG ;:ZERO THE ERROR FLAG
 CLR \$TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
 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


```

(2) 005722          102$:
(2) 005722 005726          TST      (SP)+          ;;FIX STACK          :MJD001
(2) 005724          10$:          BPL      10$          ;;WAIT UNTIL PRINTER IS READY :MJD001
(2) 005724 105777 173240      TSTB     @$TPS          ;;IS CHARACTER A RANDOM XON?  :MJD001
(2) 005730 100375          BPL      10$          ;;BRANCH IF YES              :RAN001
(2) 005732 126627 000002 000021  CMPB     2(SP),#$XON      ;;LOAD CHAR TO BE TYPED INTO DATA REG. :RAN001
(2) 005740 001420          BEQ      $TYPEX          ;;IS CHARACTER A CARRIAGE RETURN?
(2) 005742 116677 000002 173222  MOVB     2(SP),@$TPB      ;;BRANCH IF NO
(2) 005750 122766 000015 000002  CMPB     #CR,2(SP)       ;;YES--CLEAR CHARACTER COUNT
(2) 005756 001003          BNE      1$              ;;EXIT
(2) 005760 105037 006000          CLRB     $CHARCNT       ;;IS CHARACTER A LINE FEED?
(2) 005764 000406          BR       $TYPEX          ;;BRANCH IF YES
(2) 005766 122766 000012 000002 1$:      CMPB     #LF,2(SP)       ;;COUNT THE CHARACTER
(2) 005774 001402          BEQ      $TYPEX          ;;CHARACTER COUNT STORAGE
(2) 005776 105227          INCB     (PC)+
(2) 006000 000000          $CHARCNT: .WORD 0
(2) 006002 000207          $TYPEX: RTS      PC

(2)
(2)
(2)
(3)
(2) 006004 112737 000001 006250  ;;*****
(2) 006012 112737 000001 006246  $ATY1:  MOVB     #1,$FFLG      ;;TO REPORT FATAL ERROR
(2) 006020 000403          $ATY3:  MOVB     #1,$MFLG      ;;TO TYPE A MESSAGE
(2) 006022 112737 000001 006250  BR       $ATYC
(2) 006030          $ATY4:  MOVB     #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
(2) 006030 010046          $ATYC:  MOV      R0,-(SP)       ;;PUSH R0 ON STACK
(4) 006032 010146          MOV      R1,-(SP)       ;;PUSH R1 ON STACK
(2) 006034 105737 006246          TSTB     $MFLG          ;;SHOULD TYPE A MESSAGE?
(2) 006040 001450          BEQ      5$              ;;IF NOT: BR
(2) 006042 122737 000001 001254  CMPB     #APTENV,$ENV      ;;OPERATING UNDER APT?
(2) 006050 001031          BNE      3$              ;;IF NOT: BR
(2) 006052 132737 000100 001255  BITB     #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
(2) 006060 001425          BEQ      3$              ;;IF NOT: BR
(2) 006062 017600 000004          MOV      @4(SP),R0       ;;GET MESSAGE ADDR.
(2) 006066 062766 000002 000004  ADD      #2,4(SP)         ;;BUMP RETURN ADDR.
(2) 006074 005737 001234          1$:      TST      $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(2) 006100 001375          BNE      1$              ;;IF NOT: WAIT
(2) 006102 010037 001250          MOV      R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(2) 006106 105720          2$:      TSTB     (R0)+          ;;FIND END OF MESSAGE
(2) 006110 001376          BNE      2$
(2) 006112 163700 001250          SUB      $MSGAD,R0       ;;SUB START OF MESSAGE
(2) 006116 006200          ASR      R0              ;;GET MESSAGE LNTH IN WORDS
(2) 006120 010037 001252          MOV      R0,$MSGGLT      ;;PUT LENGTH IN MAILBOX
(2) 006124 012737 000004 001234  MOV      #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(2) 006132 000413          BR       5$
(2) 006134 017637 000004 006160  3$:      MOV      @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(2) 006142 062766 000002 000004  ADD      #2,4(SP)         ;;BUMP RETURN ADDRESS
(4) 006150 013746 177776          MOV      177776,-(SP)    ;;PUSH 177776 ON STACK
(2) 006154 004737 005440          JSR     PC,$TYPE        ;;CALL TYPE MACRO
(2) 006160 000000          4$:      .WORD 0
(2) 006162          5$:
(2) 006162 105737 006250          10$:     TSTB     $FFLG          ;;SHOULD REPORT FATAL ERROR?
(2) 006166 001416          BEQ      12$            ;;IF NOT: BR
(2) 006170 005737 001254          TST      $ENV           ;;RUNNING UNDER APT?
(2) 006174 001413          BEQ      12$            ;;IF NOT: BR

```

```

(2) 006176 005737 001234      11$:  TST      $MSGTYPE      ;;FINISHED LAST MESSAGE?
(2) 006202 001375              BNE      11$              ;;IF NOT: WAIT
(2) 006204 017637 000004 001236  MOV      @4(SP), $FATAL  ;;GET ERROR #
(2) 006212 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
(2) 006220 005237 001234      INC      $MSGTYPE      ;;TELL APT TO TAKE ERROR
(2) 006224 105037 006250      12$:  CLRB     $FFLG       ;;CLEAR FATAL FLAG
(2) 006230 105037 006247      CLRB     $LFLG       ;;CLEAR LOG FLAG
(2) 006234 105037 006246      CLRB     $MFLG       ;;CLEAR MESSAGE FLAG
(4) 006240 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
(4) 006242 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
(2) 006244 000207      RTS      PC            ;;RETURN
(2) 006246      000          $MFLG: .BYTE 0        ;;MESSG. FLAG
(2) 006247      000          $LFLG: .BYTE 0        ;;LOG FLAG
(2) 006250      000          $FFLG: .BYTE 0        ;;FATAL FLAG
(2)          006252      .EVEN
(2)          000200      APTSIZE=200
(2)          000001      APTENV=001
(2)          000100      APTSPool=100
(2)          000040      APTCSUP=040

(1)          ;STRING INPUT ROUTINE
(1)          ;-----
(1) 006252 010346      .INSTR: MOV      R3,-(SP)   ;SAVE R3 ON STACK
(1) 006254 010446      MOV      R4,-(SP)   ;SAVE R4 ON STACK
(1) 006256 017637 000004 006274  MOV      @4(SP),.MSG  ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1) 006264 062766 000002 000004  ADD      #2,4(SP)   ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1) 006272 104402      .INST1: TYPE      ;PRINT THE MESSAGE
(1) 006274 000000      .MSG:  0           ;MESSAGE IS POINTED TO FROM HERE
(1) 006276 012704 011272      MOV      #INBUF,R4  ;POINT R4 TO THE INPUT BUFFER
(1) 006302 012703 000007      MOV      #7,R3      ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1) 006306 105777 172652      1$:  TSTB     @ $TKS   ;HAS A CHARACTER BEEN RECEIVED?
(1) 006312 100375      BPL      1$         ;IF NO, KEEP WAITING FOR IT
(1) 006314 117714 172646      MOVB     @ $TKB,(R4) ;IF YES, SAVE IT IN THE INPUT BUFFER
(1) 006320 142714 000200      BICB     #200,(R4)  ;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1) 006324 122714 000023      CMPB     # $XOFF,(R4) ;IS IT A XOFF?
(1) 006330 001014      BNE      83$       ;BR IF NOT
(1) 006332 105777 172626      101$: TSTB     @ $TKS   ;WAIT FOR A CHARACTER
(1) 006336 100375      BPL      101$      ;
(1) 006340 117714 172622      MOVB     @ $TKB,(R4) ;GET CHARACTER
(1) 006344 142714 000200      BICB     #200,(R4)  ;STRIP DOWN CHARACTER
(1) 006350 122714 000021      CMPB     # $XON,(R4) ;WAIT FOR A XON?
(1) 006354 001366      BNE      101$     ;GET NEXT CHAR IF NOT
(1) 006356 105724      TSTB     (R4)+     ;POP STACK
(1) 006360 000752      BR       1$        ;WAIT FOR A CHAR
(1) 006362 122714 000021      83$:  CMPB     # $XON,(R4) ;IS IT A RANDOM XON
(1) 006366 001002      BNE      102$     ;BR IF NO
(1) 006370 105724      TSTB     (R4)+     ;ELSE, POP STACK
(1) 006372 000745      BR       1$        ;GO GET NEXT CHAR
(1) 006374 122724 000015      102$: CMPB     #15,(R4)+ ;IS IT <CR> ?
(1) 006400 001417      BEQ      INSTR2   ;IF SO, TERMINATE THE INPUT SEQUENCE
(1) 006402 105777 172562      2$:  TSTB     @ $TPS   ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1) 006406 100375      BPL      2$        ;IF WE CAN'T, WAIT UNTIL WE CAN
(1) 006410 017777 172552 172554  MOV      @ $TKB,@ $TPB ;ECHO THE CHARACTER BACK
(1) 006416 005303      DEC      R3        ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
(1) 006420 001332      BNE      1$        ;IF WE DON'T HAVE 7, GO GET SOME MORE

```



```

(1) 007014 062703 000060      ADD      #060,R3      ;MAKE AN ASCII CHARACTER OUT OF THEM
(1) 007020 110346      MOV      R3,-(SP)    ;SAVE THAT CHARACTER
(1) 007022 006004      ROR      R4          ;MOVE THE NEXT THREE BITS INTO PLACE
(1) 007024 006204      ASR      R4          ;MOVE THEM AGAIN
(1) 007026 006204      ASR      R4          ;AND FINALLY A THIRD TIME
(1) 007030 005305      DEC      R5          ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
(1)                                ;BUILT?
(1) 007032 001365      BNE      3$          ;IF NO, GO BUILD THE NEXT ONE.
(1) 007034 012703 011376      MOV      #MDATA,R3  ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1) 007040 112623      MOV      (SP)+,(R3)+ 4$: ;STORE THE CHARACTER, STARTING WITH THE MOST
(1) 007042 105337 007116      DECB    CHRCNT      ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1) 007046 001374      BNE      4$          ;IF NO, GO TRANSFER ANOTHER
(1) 007050 105700      TSTB    R0          ;ARE ANY SPACES TO BE PRINTED?
(1) 007052 001404      BEQ      6$          ;IF NO, DON'T SET UP ANY
(1) 007054 112723 000040      MOV      #040,(R3)+ 5$: ;ADD A SPACE TO THE OUTPUT BUFFER
(1) 007060 105300      DECB    R0          ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1) 007062 001374      BNE      5$          ;IF YES, GO ADD ANOTHER SPACE
(1) 007064 105013      CLRB    (R3)        ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
(1) 007066 104402 011376      TYPE    ,MDATA      ;PRINT THE STRING WE JUST BUILT
(1) 007072 005337 007114      DEC      WRDCNT      ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1) 007076 001336      BNE      1$          ;IF YES, GO CONVERT THEM
(1) 007100 012605      MOV      (SP)+,R5    ;RESTORE R5
(1) 007102 012604      MOV      (SP)+,R4    ;RESTORE R4
(1) 007104 012603      MOV      (SP)+,R3    ;RESTORE R3
(1) 007106 012601      MOV      (SP)+,R1    ;RESTORE R1
(1) 007110 012600      MOV      (SP)+,R0    ;RESTORE R0
(1) 007112 000002      RTI                                ;RETURN TO THE MAIN PROGRAM
(1) 007114 000000      WRDCNT: 0
(1) 007116 000          CHRCNT: .BYTE
(1) 007117 000          SPACNT: .BYTE 0      ;NUMBER OF CHARACTERS TO PRINT
(1)                                ;NUMBER OF SPACES TO PRINT
(1) 007120 000000      BINWRD: 0
(1)
(1)
(1)                                ;TRAP DISPATCH SERVICE
(1)                                ;ARGUMENT OF TRAP IS EXTRACTED
(1)                                ;AND USED AS OFFSET TO OBTAIN POINTER
(1)                                ;TO SELECTED SUBROUTINE
(1)
(1) 007122 010046      .TRPSR: MOV      R0,-(SP)  ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
(1) 007124 016600 000002      MOV      2(SP),R0   ;GET TRAP ADDRESS
(1) 007130 005740      TST      -(R0)      ;GET TRAP
(1) 007132 111000      MOV      (R0),R0    ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
(1) 007134 006300      ASL      R0          ;POSITION OFFSET FOR TABLE INDEXING
(1) 007136 016000 002002      MOV      .TRPTAB(R0),R0 ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1) 007142 000200      RTS      R0          ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                                ;DEVICE CLEAR ROUTINE
(1)                                ;ISSUE A DEVICE CLEAR
(1)                                ;-----
(1) 007144      .DEVICE.CLR:
(1) 007144 052777 000020 172670      BIS      #DCLR,@DZCSR ;SET DCLR
(1) 007152 032777 000020 172662 1$: BIT      #DCLR,@DZCSR ;DID IT CLEAR?
(1) 007160 001374      BNE      1$          ;BR IF NO
(1) 007162 000002      RTI                                ;EXIT ROUTINE

```

```

(1)                                     ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                                     ;-----
(1) 007164 104413                       .DCLASM:DEVICE.CLR           ;ISSUE A DEVICE CLEAR
(1) 007166 153777 001417 172646        BISB  MNTFLG,@DZCSR       ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1) 007174 000002                       RTI                       ;RETURN TO CALLING ROUTINE
(1)
(1) 007176                               .DELAY:
(1) 007176 010046                       MOV  R0,-(SP)            ;SAVE R0
(1) 007200 013700 007214                MOV  DLYCNT,R0           ;SET COUNT
(1) 007204 005300                        1$:  DEC  R0              ;DELAY
(1) 007206 001376                        BNE  1$                  ;
(1) 007210 012600                        MOV  (SP)+,R0           ;RESTORE R0
(1) 007212 000002                        RTI                       ;LEAVE ROUTINE
(1) 007214 000001                        DLYCNT: .WORD  1        ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                                     ;ADVANCE TO NEXT TEST HANDLER
(1)                                     ;-----
(1) 007216 013716 001360                .ADVANCE:MOV  NEXT,(SP)  ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
(1) 007222 005037 001362                CLR  LOCK               ;RESET TIGHT LOOP ADDRESS
(1) 007226 000002                        RTI                       ;CHECK TO SEE IF OLD TEST GETS REPEATED
(1)
(1)                                     ;ERROR HANDLER
(1)                                     ;-----
(1) 007230 004737 007652                $ERROR: JSR  PC,SERV.G    ;FIND OUT IF <^G> WAS HIT
(1) 007234 032777 010000 171716        BIT  #SW12,@SWR         ;BELL ON ERROR?
(1) 007242 001406                       BEQ  XBX                 ;BR IF NO BELL
(1) 007244 105777 171720                TSTB @STPS              ;TTY READY.
(1) 007250 100003                       BPL  XBX                 ;DON'T WAIT IF TTY NOT READY.
(1) 007252 112777 000207 171712        MOVB #207,@STPB         ;PUSH A BELL AT THE TTY.
(1) 007260 032777 020000 171672        XBX:  BIT  #SW13,@SWR    ;DELETE ERROR PRINT OUT?
(1) 007266 001113                       BNE  HALTS              ;BR IF NO PRINT OUT WANTED.
(1) 007270 021637 001136                CMP  (SP),$ERRPC        ;WAS THIS ERROR FOUND LAST TIME?
(1) 007274 001404                       BEQ  1$                  ;BR IF YES
(1) 007276 011637 001136                MOV  (SP),$ERRPC        ;RECORD BEING HERE
(1) 007302 105037 001123                CLR  $ERFLG             ;PREPARE HEADER
(1) 007306 104407                        1$:  SAV05              ;SAVE ALL PROC REGISTERS
(1) 007310 011605                       MOV  (SP),R5            ;GET THE PC OF ERROR
(1) 007312 162705 000002                SUB  #2,R5              ;GET ADDRESS OF TRAP CALL
(1) 007316 011504                       MOV  (R5),R4            ;GET ERROR INSTRUCTION
(1) 007320 110437 001134                MOV  R4,$ITEMB          ;COPY TEST NUMBER FOR APT HANDLING
(1) 007324 006304                       ASL  R4                  ;MULT BY TWO
(1) 007326 061504                       ADD  (R5),R4            ;DOUBLE IT
(1) 007330 006304                       ASL  R4                  ;MULT AGAIN
(1) 007332 042704 177001                BIC  #177001,R4         ;CLEAR JUNK
(1) 007336 062704 027536                ADD  #.ERRTAB,R4        ;GET POINTER
(1) 007342 012437 007466                MOV  (R4)+,ERRMSG       ;GET ERROR MESSAGE
(1) 007346 012437 007500                MOV  (R4)+,DATAHD       ;GET DATA HEADRER
(1) 007352 011437 007512                MOV  (R4),DATABP        ;GET DATA TABLE
(1) 007356 105737 001123                TSTB $ERFLG            ;TYPE HEADER
(1) 007362 001403                       BEQ  TYPMSG             ;BR IF YES
(1) 007364 005737 007512                TST  DATABP             ;DOES DATA TABLE EXIST?
(1) 007370 001044                       BNE  TYPDAT             ;BR IF YES.
(1) 007372 104402 001231                TYPMSG: TYPE  $CRLF      ;TYPE A CARRIAGE RETURN
(1) 007376 104402 001231                TYPE  $CRLF             ;AND TYPE ANOTHER

```

```

(1) 007402 005737 001362          TST      LOCK
(1) 007406 001402                   BEQ      1$
(1) 007410 104402 010766          TYPE    ,MASTEK
(1) 007414 104402 010754          1$:    TYPE    ,MTSTN
(1) 007420 104412 007644          CNVRT   ,XTSTN      :SHOW IT
(1) 007424 104402 011043          TYPE    ,MERRPC    :TYPE PC.
(1) 007430 104412 007636          CNVRT   ,ERTABO    :SHOW IT
(1) 007434 104402 010716          TYPE    ,MCSRX
(1) 007440 104412 005072          CNVRT   ,XCSR
(1) 007444 104402 001231          TYPE    ,$CRLF     :GIVE A CR/LF
(1) 007450 112737 177777 001123  MOVB    #-1,$ERFLG  :NO MORE HEADER UNLESS NO DATA TABLE.
(1) 007456 005737 007466          TST     ERRMSG     :IS THERE AN ERROR MESSAGE?
(1) 007462 001402                   BEQ     WTBS.FM    :BR IF NO.
(1) 007464 104402                   TYPE
(1) 007466 000000          ERRMSG: 0         :TYPE
(1) 007470          WTBS.FM:         :      ERROR MESSAGE
(1) 007470 005737 007500          TST     DATAHD   :DATA HEADER?
(1) 007474 001402                   BEQ     TYPDAT    :BR IF NO
(1) 007476 104402                   TYPE
(1) 007500 000000          DATAHD: 0       :      DATA HEADER
(1) 007502 005737 007512          TYPDAT: TST     DATABP   :DATA TABLE?
(1) 007506 001402                   BEQ     RESREG    :BR IF NO.
(1) 007510 104411          CONVRT
(1) 007512 000000          DATABP: 0       :      DATA TABLE
(1) 007514 104410          RESREG: RES05   :RESTORE PROC REGISTERS
(1) 007516 122737 000001 001254  HALTS:  CMPB     #APTENV,$ENV :IS APT RUNNING?
(1) 007524 001007          BNE     2$
(1) 007526 113737 001134 007540  MOVB    $ITEMB,7$ :SKIP APT CALL IF NOT
(1) 007534 004737 006022          JSR     PC,$ATY4  :COPY ERROR NUMBER
(1) 007540 000000          7$:    .WORD    0   :CALL APT SERVICE
(1) 007542 000777          8$:    BR      8$   :ERROR NUMBER STUCK HERE
(1) 007544 022737 005056 000042  2$:    CMP     #$ENDAD,@#42 :LOCK UP HERE
(1) 007552 001403          BEQ     1$
(1) 007554 005777 171400          TST     @SWR
(1) 007560 100004          BPL     EXITER   :CHECK TO SEE IF IN ACT-11 MODE
(1) 007562 016677 000002 171372  1$:    MOV     2(SP),@DISPLAY :IF SO, HANDLE ACCORDINGLY
(1) 007570 000000          HALT
(1) 007572 005237 001132          EXITER: INC     $ERTTL  :HALT ON ERROR?
(1) 007576 032777 000400 171354  BIT     #SW08,@SWR  :BR IF NO HALT ON ERROR
(1) 007604 001007          BNE     1$
(1) 007606 032777 002000 171344  BIT     #SW10,@SWR :SHOW ERROR PC IN DATA DISPLAY
(1) 007614 001407          BEQ     2$
(1) 007616 013737 001360 001126  MOV     NEXT,$LPADR :HALT
(1) 007624 012706 001120          1$:    MOV     #STACK,SP :UPDATE ERROR COUNT
(1) 007630 000177 171272          JMP     @LPADR    :GOTO TOP OF TEST?
(1) 007634 000002          2$:    RTI
(1) 007636 000001          ERTABO: 1
(1) 007640 006 002          .BYTE  6,2
(1) 007642 001402          SAVPC
(1) 007644 000001          XTSTN: 1
(1) 007646 002 002          .BYTE  2,2
(1) 007650 001122          $TSTNM
(1) 007652 022737 177570 001160  SERV.G: CMP     #177570,$SWR :IS THE SWITCH REGISTER HARDWIRED?
(1) 007660 001002          BNE     99$
(1) 007662 000137 010270          JMP     6$
(1) 007666 017746 171274          99$:  MOV     @TKB,-(SP) :IF SO, IGNORE ^G
:OTHERWISE, GET THE LAST CHARACTER TYPED
:DSH-BHL

```

```

(1) 007672 042716 177600          BIC      #177600,(SP)      ;STRIP CHAR          ;;DSH-BHL
(1) 007676 122716 000023          CMPB     #$XOFF,(SP)    ;IS IT A XOFF       ;;DSH-BHL
(1) 007702 001012                   BNE      102$           ;BR IF NO           ;;DSH-BHL
(1)
(1) 007704 105777 171254          101$:   TSTB     @TKS           ;WAIT FOR A CHAR    ;;DSH-BHL
(1) 007710 100375                   BPL      101$           ;                   ;;DSH-BHL
(1) 007712 117716 171250          MOVB     @TKB,(SP)      ;GET THE CHAR       ;;DSH-BHL
(1) 007716 042716 177600          BIC      #177600,(SP)    ;STRIP CHAR         ;;DSH-BHL
(1) 007722 122716 000021          CMPB     #$XON,(SP)     ;IS IT A XON        ;;DSH-BHL
(1) 007726 001366                   BNE      101$           ;BR IF NO           ;;DSH-BHL
(1)
(1) 007730 122716 000021          102$:   CMPB     #$XON,(SP)    ;IS IT RANDOM XON ? ;;DSH
(1) 007734 001002                   BNE      7$             ;BR IF NOT          ;;DSH
(1) 007736 005726                   TST      (SP)+          ;POP STACK          ;;DSH
(1) 007740 000553                   BR       6$             ;IGNORE XON CHAR    ;;DSH
(1)
(1) 007742 122726 000007          7$:     CMPB     #7,(SP)+      ;IS IT ^G?          ;
(1) 007746 001150                   BNE      6$             ;IF NOT, IGNORE INPUT ;
(1) 007750 032777 004000 171206    BIT      #4000,@TKS     ;RX BUSY?           ;
(1) 007756 001335                   BNE      SERV.G         ;BR IF YES          ;
(1) 007760 017737 171174 010312    MOV      @SWR,90$       ;SAVE (SWR).        ;
(1) 007766 013777 010312 171164    1$:     MOV      90$,@SWR      ;                   ;
(1) 007774 104402 010272          TYPE     ,89$           ;                   ;
(1) 010000 104412 010304          CNVRT    ,88$           ;                   ;
(1) 010004 104402 010314          TYPE     ,91$           ;                   ;
(1) 010010 105777 171150          9$:     TSTB     @TKS           ;WAIT FOR DONE.     ;
(1) 010014 100375                   BPL      -4             ;                   ;
(1) 010016 017746 171144          MOV      @TKB,-(SP)     ;                   ;
(1) 010022 042716 177600          BIC      #177600,(SP)    ;STRIP CHAR          ;;DSH-BHL
(1) 010026 122716 000023          CMPB     #$XOFF,(SP)    ;IS IT A XOFF       ;;DSH-BHL
(1) 010032 001012                   BNE      112$          ;BR IF NO           ;;DSH-BHL
(1)
(1) 010034 105777 171124          111$:   TSTB     @TKS           ;WAIT FOR A CHAR    ;;DSH-BHL
(1) 010040 100375                   BPL      111$           ;                   ;;DSH-BHL
(1) 010042 117716 171120          MOVB     @TKB,(SP)      ;GET THE CHAR       ;;DSH-BHL
(1) 010046 042716 177600          BIC      #177600,(SP)    ;STRIP CHAR         ;;DSH-BHL
(1) 010052 122716 000021          CMPB     #$XON,(SP)     ;IS IT A XON        ;;DSH-BHL
(1) 010056 001366                   BNE      111$           ;BR IF NO           ;;DSH-BHL
(1)
(1) 010060 122716 000021          112$:   CMPB     #$XON,(SP)    ;IS IT RANDOM XON ? ;;DSH
(1) 010064 001002                   BNE      8$             ;BR IF NOT          ;;DSH
(1) 010066 005726                   TST      (SP)+          ;POP STACK          ;;DSH
(1) 010070 000747                   BR       9$             ;IGNORE XON CHAR    ;;DSH
(1)
(1) 010072 122726 000015          8$:     CMPB     #15,(SP)+      ;                   ;
(1) 010076 001472                   BEQ      5$             ;                   ;
(1) 010100 005077 171054          CLR      @SWR           ;                   ;
(1) 010104 105777 171060          2$:     TSTB     @TPS           ;                   ;
(1) 010110 100375                   BPL      -4             ;                   ;
(1) 010112 016677 177776 171052    MOV      -2(SP),@TPB    ;                   ;
(1) 010120 000241                   CLC                       ;                   ;
(1) 010122 006177 171032          ROL      @SWR           ;                   ;
(1) 010126 006177 171026          ROL      @SWR           ;                   ;
(1) 010132 006177 171022          ROL      @SWR           ;                   ;
(1) 010136 103713                   BCS     1$              ;ERROR               ;
(1) 010140 026627 177776 000060    CMP      -2(SP),#60     ;                   ;

```

```

(1) 010146 002707          BLT      1$          :
(1) 010150 026627 177776 000067    CMP      -2(SP),#67  :
(1) 010156 003303          BGT      1$          :
(1) 010160 042766 177770 177776    BIC      #^C<7>,-2(SP) :
(1) 010166 056677 177776 170764    BIS      -2(SP),@SWR  :
(1) 010174 105777 170764          12$:    TSTB     @$TKS      :
(1) 010200 100375          BPL      -4          :
(1) 010202 017746 170760          MOV      @$TKB,-(SP)  :
(1) 010206 042716 177600          BIC      #177600,(SP) ;STRIP CHAR          ;;DSH-BHL
(1) 010212 122716 000023          CMPB     #$XOFF,(SP) ;IS IT A XOFF        ;;DSH-BHL
(1) 010216 001012          BNE      122$        ;BR IF NO            ;;DSH-BHL
(1)
(1) 010220 105777 170740          121$:   TSTB     @$TKS      ;WAIT FOR A CHAR      ;;DSH-BHL
(1) 010224 100375          BPL      121$        ;                     ;;DSH-BHL
(1) 010226 117716 170734          MOVB     @$TKB,(SP)  ;GET THE CHAR        ;;DSH-BHL
(1) 010232 042716 177600          BIC      #177600,(SP) ;STRIP CHAR          ;;DSH-BHL
(1) 010236 122716 000021          CMPB     #$XON,(SP)  ;IS IT A XON         ;;DSH-BHL
(1) 010242 001366          BNE      121$        ;BR IF NO            ;;DSH-BHL
(1)
(1) 010244 122716 000021          122$:   CMPB     #$XON,(SP) ;IS IT RANDOM XON ?  ;;DSH
(1) 010250 001002          BNE      10$         ;BR IF NOT           ;;DSH
(1) 010252 005726          TST      (SP)+       ;POP STACK           ;;DSH
(1) 010254 000747          BR       12$         ;IGNORE XON CHAR     ;;DSH
(1)
(1) 010256 122726 000015          10$:    CMPB     #15,(SP)+  :
(1) 010262 001310          BNE      2$          :
(1) 010264 104402 001231          5$:     TYPE     ,$CRLF  :
(1) 010270 000207          6$:     RTS      PC      :
(1)
(1) 010272 020200 051450 051127    89$:    .ASCIZ   <200>? (SWR)=/?
(1) 010300 036451 000057          .EVEN
(1) 010304 000001          88$:    1
(1) 010306 006 000          .BYTE   6,0
(1) 010310 010312          90$:    90$
(1) 010312 000000          90$:    .WORD   0
(1) 010314 036457 000057          91$:    .ASCIZ   ?/=/?
(1)
(2) .EVEN
(2) .SBTTL  POWER DOWN AND UP ROUTINES
(3)
(2)
(2) 010320 012737 010464 000024    $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(2) 010326 012737 000340 000026    MOV     #340,@#PWRVEC+2 ;;PRIO:7
(4) 010334 010046          MOV     R0,-(SP)      ;;PUSH R0 ON STACK
(4) 010336 010146          MOV     R1,-(SP)      ;;PUSH R1 ON STACK
(4) 010340 010246          MOV     R2,-(SP)      ;;PUSH R2 ON STACK
(4) 010342 010346          MOV     R3,-(SP)      ;;PUSH R3 ON STACK
(4) 010344 010446          MOV     R4,-(SP)      ;;PUSH R4 ON STACK
(4) 010346 010546          MOV     R5,-(SP)      ;;PUSH R5 ON STACK
(4) 010350 017746 170604          MOV     @SWR,-(SP)    ;;PUSH @SWR ON STACK
(2) 010354 010637 010470          MOV     SP,$SAVR6    ;;SAVE SP
(2) 010360 012737 010372 000024    MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(2) 010366 000000          HALT
(2) 010370 000776          BR      -2           ;;HANG UP
(2)

```

```

(3)
(2)
(2) 010372 012737 010464 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
(2) 010400 013706 010470          MOV    $SAVR6,SP      ;;GET SP
(2) 010404 005037 010470          CLR    $SAVR6        ;;WAIT LOOP FOR THE TTY
(2) 010410 005237 010470 1$: INC    $SAVR6        ;;WAIT FOR THE INC
(2) 010414 001375          BNE    1$           ;;OF WORD
(4) 010416 012677 170536          MOV    (SP)+,@SWR    ;;POP STACK INTO @SWR
(4) 010422 012605          MOV    (SP)+,R5     ;;POP STACK INTO R5
(4) 010424 012604          MOV    (SP)+,R4     ;;POP STACK INTO R4
(4) 010426 012603          MOV    (SP)+,R3     ;;POP STACK INTO R3
(4) 010430 012602          MOV    (SP)+,R2     ;;POP STACK INTO R2
(4) 010432 012601          MOV    (SP)+,R1     ;;POP STACK INTO R1
(4) 010434 012600          MOV    (SP)+,R0     ;;POP STACK INTO R0
(2) 010436 012737 010320 000024 MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(2) 010444 012737 000340 000026 MOV    #340,@#PWRVEC+2 ;;PRIO:7
(2) 010452 104402          TYPE                                ;;REPORT THE POWER FAILURE
(2) 010454 010472 $PWRMG: .WORD MPFAIL          ;;POWER FAIL MESSAGE POINTER
(2) 010456 012716          MOV    (PC)+,(SP)    ;;RESTART AT RESTART
(2) 010460 012106 $PWRAD: .WORD RESTART        ;;RESTART ADDRESS
(2) 010462 000002          RTI
(2) 010464 000000 $ILLUP: HALT                ;;THE POWER UP SEQUENCE WAS STARTED
(2) 010466 000776          BR    -2             ;;BEFORE THE POWER DOWN WAS COMPLETE
(2) 010470 000000          $SAVR6: 0            ;;PUT THE SP HERE
(2) 010472 050200 051127 043040 MPFAIL: .ASCIZ <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 010535 200 047105 020104 MEPASS: .ASCIZ <200>/END PASS CZDZA-G /
(2) 010560 051200 047125 044516 MR: .ASCIZ <200>/RUNNING /
(2) 010574 050200 047522 051107 MERR2: .ASCIZ <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 010643 200 047111 052523 MERR3: .ASCIZ <200>/INSUFFICIENT DATA!/
(2) 010667 200 047514 045503 MLOCK: .ASCIZ <200>/LOCK ON SELECTED TEST/
(2) 010716 051503 035122 000040 MCSRX: .ASCIZ /CSR: /
(2) 010724 042526 035103 000040 MVECX: .ASCIZ /VEC: /
(2) 010732 040520 051523 051505 MPASSX: .ASCIZ /PASSES: /
(2) 010743 105 051122 051117 MERRX: .ASCIZ /ERRORS: /
(2) 010754 042524 052123 047040 MTSTN: .ASCIZ /TEST NO: /
(2) 010766 020052 000 MASTEK: .ASCIZ /* /
(2) 010771 200 042523 020124 MNEW: .ASCIZ <200>/SET SWITCH REG TO DZ11'S DESIRED ACTIVE./
(2) 011043 120 035103 000040 MERRPC: .ASCIZ /PC: /
(2) 011050 046600 050101 047440 XHEAD: .ASCIZ <200>/MAP OF DZ11 STATUS/<200>
(2) 011075 200 046111 042514 MBADLN: .ASCIZ <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2) 011140 011140          .EVEN
(2) 011140 000002          XSTATQ: 2
(2) 011142 006 003          .BYTE 6,3
(2) 011144 001220          $TMP1
(2) 011146 006 002          .BYTE 6,2
(2) 011150 001222          $TMP2
(1)
(2)
(2)
(2)
(2)
(2)
(2) 011152 017605 000000          .SETFLG:MOV    @(SP),R5    ;;PICK UP ADDRESS OF TAG
(2) 011156 042737 000040 011272 BIC    #40,INBUF        ;;STRIP LOWER CASE
(2) 011164 122737 000105 011272 CMPB   #'E,INBUF        ;;IS IT EXTERNAL LOOP BACK ?
(2) 011172 001005          BNE    4$             ;;NO

```

```

(2) 011174 013715 011264      MOV      1$, (R5)      ;YES STORE INFO
(2) 011200 105037 001417      CLRB     MNTFLG      ;SET MAINT BIT =0
(2) 011204 000422              BR       7$          ;GET OUT
(2) 011206 122737 000111 011272 4$:  CMPB    #'I, INBUF   ;IS IT INTERNAL LOOP BACK ?
(2) 011214 001006              BNE     5$          ;NO
(2) 011216 013715 011266      MOV      2$, (R5)      ;YES STORE INFO
(2) 011222 112737 000010 001417  MOVB    #MAINT, MNTFLG ;SET UP THE MAINTENANCE FLAG LOADER
(2) 011230 000410              BR       7$          ;GET OUT
(2) 011232 122737 000123 011272 5$:  CMPB    #'S, INBUF   ;IS IT STAGGERED LOOP BACK ?
(2) 011240 001007              BNE     6$          ;WHAT ?
(2) 011242 013715 011270      MOV      3$, (R5)      ;YES STORE INFO
(2) 011246 105037 001417      CLRB     MNTFLG      ;ZERO BITS
(2) 011252 062716 000002      ADD     #2, (SP)      ;POP AROUND
(2) 011256 000002              RTI
(2) 011260 104404              6$:  INSTER          ;RETRY
(2) 011262 000733              BR       .SETFLG     ;DITTO
(2) 011264 000200              1$:  .WORD    200      ;EXTERNAL = E
(2) 011266 000000              2$:  .WORD    0        ;INTERNAL = I
(2) 011270 100000              3$:  .WORD    100000   ;STAGGERED = S
(2)
(2)
(2) ;BUFFERS FOR INPUT-OUTPUT
(2) 011272 000000      INBUF:  0
(2)              .=. +40
(2) 011334 000000      TEMP:   0
(2)              .=. +40
(2) 011376 000000      MDATA:  0
(2)              .=. +40
(2)
(2) 011440 011637 011536      SET.PS: MOV      (SP), 3$
(2) 011444 162737 000002 011536  SUB     #2, 3$
(2) 011452 017737 000060 011540  MOV     @3$, 4$
(2) 011460 022737 106427 011540  CMP     #106427, 4$
(2) 011466 001003              BNE     1$
(2) 011470 011637 011536      MOV     (SP), 3$
(2) 011474 000412              BR      2$
(2) 011476 022737 106437 011540 1$:  CMP     #106437, 4$
(2) 011504 001401              BEQ    .+4
(2) 011506 000000              HALT   ;RESERVED INSTRUCTION NOT 'MTPS'
(2) 011510 011637 011536      MOV     (SP), 3$
(2) 011514 017737 000016 011536  MOV     @3$, 3$
(2) 011522 062716 000002      2$:  ADD     #2, (SP)
(2) 011526 017766 000004 000002  MOV     @3$, 2(SP)
(2) 011534 000002              RTI
(2) 011536 000000              3$:  0
(2) 011540 000000              4$:  0

```

```

(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2) 011542 005737 001404      CYCLE: TST      DZACTV      ;ARE ANY DZ11'S TO BE TESTED?
(2) 011546 001004              BNE      1$      ;BR IF OK.
(2) 011550 104402 010574      TYPE      ,MERR2  ;NO DZ11'S SELECTED!!
(2) 011554 000000              HALT      ;STOP THE SHOW.
(2) 011556 000776              BR      -2      ;DISQUALIFY CONT. SW.
(2) 011560 013737 005374 001226 1$: MOV      $MXCNT,$TIMES ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
(2) 011566 033737 001406 001404 BIT      RUN,DZACTV ;IS THIS ONE 'ACTIVE'
(2) 011574 001020              BNE      2$      ;BR IF GOOD ONE FOUND.
(2) 011576 000241              CLC      ;
(2) 011600 006137 001406      ROL      RUN      ;UPDATE POINTER
(2) 011604 005537 001406      ADC      RUN      ;CATCH CARRY FROM RUN
(2) 011610 062737 000014 001412 ADD      #14,ACTIVE ;UPDATE ADDRESS POINTER.
(2) 011616 022737 002000 001412 CMP      #DZ.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
(2) 011624 001355              BNE      1$      ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
(2) 011626 012737 001500 001412 MOV      #DZ.MAP,ACTIVE ;RESET ADDRESS POINTER.
(2) 011634 000751              BR      1$      ;KEEP LOOKING FOR ACTIVE DZ11
(2) 011636 000241              CLC      ;
(2) 011640 006137 001406      ROL      RUN      ;UPDATE POINTER.
(2) 011644 005537 001406      ADC      RUN      ;CATCH CARRY.
(2) 011650 013700 001412      MOV      ACTIVE,R0 ;GET ADDRESS POINTER.
(2) 011654 062737 000014 001412 ADD      #14,ACTIVE ;UPDATE.
(2) 011662 022737 002000 001412 CMP      #DZ.END,ACTIVE ;
(2) 011670 001003              BNE      3$      ;ALL DONE?
(2) 011672 012737 001500 001412 MOV      #DZ.MAP,ACTIVE ;BR IF NO.
(2) 011700 012037 001310      3$: MOV      (R0)+,$BASE ;RESTORE POINTER.
(2) 011704 012037 002072      MOV      (R0)+,DZRIV ;LOAD SYSTEM CTRL. REG
(2) 011710 012037 027532      MOV      (R0)+,DZPRT ;LOAD VECTOR
(2) 011714 113737 027533 001414 MOVB     DZPRT+1,EIAFLG ;LOAD PRIORITY
(2) 011722 042737 100000 027532 BIC      #BIT15,DZPRT ;EIA OR 20MA
(2) 011730 012037 001364      MOV      (R0)+,LINE ;CLEAR FLAG
(2) 011734 012037 001366      MOV      (R0)+,PAR  ;SET UP LINE DZ LINES ACTIVE
(2) 011740 012037 001370      MOV      (R0)+,MODE ;SET UP PARAMETERIZATION
(2) 011744 004737 027324      JSR     PC,DZLEV  ;SET UP MAINTENANCE MODE
(2) 011750 005737 000042      TST     @#42     ;SET UP
(2) 011754 001051              BNE      4$      ;ARE WE UNDER MONITOR CONTROL?
(2) 011756 032777 000002 167174 BIT      #SW01,@SWR ;IF YES, SKIP THIS SETUP
(2) 011764 001445              BEQ     4$      ;IF SW01=1, GET STARTING TEST #
(2) 011766 104402 001231      7$: TYPE      ,SCLRF ;BR IF NO TEST IS TO BE INPUTTED
(3) 011772 104403              INSTR     ;CALL THE STRING INPUT ROUTINE
(3) 011774 010754              MTSTN    ;POINTER TO MESSAGE TO BE PRINTED
(3) 011776 104405              PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(3) 012000 000001              1        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3) 012002 001000              1000     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3) 012004 001122              $STNM    ;POINTER TO MAP LOCATION TO BE FILLED
(3) 012006 000          .BYTE 0    ;MASK OF INVALID BITS FOR THIS PARAMETER
(3) 012007 001          .BYTE 1    ;NUMBER OF PARAMETERS TO STORE

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-40
POWER DOWN AND UP ROUTINES

SEQ 0060

```

(2) 012010 01270C 013046      MOV      #TST1,R0
(2) 012014 022710 000004      5$:     CMP      #4,(R0)
(2) 012020 001020              BNE      6$
(2) 012022 022760 012737 000002  CMP      #12737,2(R0)
(2) 012030 001014              BNE      6$
(2) 012032 023760 001122 000004  CMP      $TSTNM,4(R0)      :IS THIS THE TEST ?
(2) 012040 001010              BNE      6$                :IF NOT, DON'T PROCESS NUMBER
(2) 012042 010037 001126      MOV      R0,$LPADR         :SAVE PC
(2) 012046 062737 000002 001126  ADD      #2,$LPADR         :POP OVER SCOPE
(2) 012054 104402 001231      TYPE    ,$CRLF
(2) 012060 000412              BR       8$
(2) 012062 005720              6$:     TST      (R0)+
(2) 012064 020027 023454      CMP      R0,#TLAST+10
(2) 012070 001351              BNE      5$
(2) 012072 104402 001230      TYPE    ,$QUES
(2) 012076 000733              BR       7$
(2) 012100 012737 013046 001126  4$:     MOV      #TST1,$LPADR      :PREPARE TEST ADDRESS
(2) 012106              8$:
(2) 012106 000177 167014      RESTART:JMP @ $LPADR      :GO START TESTING.***WARNING!***
(2)                               :THIS JUMP IS USED BY POWER UP ROUTINE!!!!
(2)

```

```

(2)                                     ; -ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
(2)                                     ; IF BIT7 IN THE ENVIRONMENT MODE ($ENVM) BYTE IS SET,
(2)                                     ; THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
(2) 012112 012700 001500          SETAPT: MOV    #DZ.MAP,R0      ;POINT TO THE DEVICE MAP TABLE
(2) 012116 013701 001310          MOV    $BASE,R1      ;BUILD DEVICE ADDRESSES IN R1
(2) 012122 013702 001304          MOV    $VECT1,R2     ;BUILD DEVICE VECTORS IN R2
(2) 012126 042702 177007          BIC    #^C<770>,R2   ;STRIP AWAY OTHER INFORMATION
(2)
(2) 012132 113703 001305          MOVB   $VECT1+1,R3    ;LOAD THE INTERRUPT PRIORITY FROM R3
(2) 012136 106003                   RORB   R3             ;ALIGN THE NUMBER
(2) 012140 106003                   RORB   R3             ;ALIGN THE NUMBER
(2) 012142 106003                   RORB   R3             ;ALIGN THE NUMBER
(2) 012144 106003                   RORB   R3             ;ALIGN THE NUMBER
(2) 012146 106003                   RORB   R3             ;ALIGN THE NUMBER
(2) 012150 042703 177770          BIC    #^C<7>,R3     ;REMOVE ALL BUT BUS LEVEL NUMBER
(2) 012154 012704 001320          MOV    #DZ.DW0,R4    ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
(2) 012160 013705 001312          MOV    $DEVN,R5      ;GET THE MAP OF ACTIVE DEVICES
(2) 012164 010537 001404          MOV    R5,DZACTV     ;SAVE THE BIT MAP
(2) 012170 006005          1$:  ROR    R5             ;GET A DEVICE SELECTION BIT
(2) 012172 103407          BCS   3$             ;IF IT IS SELECTED, GO SET UP A MAP
(2) 012174 001425          BEQ   5$             ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
(2) 012176 005724          TST   (R4)+          ;POINT TO NEXT DEVICE DESCRIPTOR
(2) 012200 062701 000010          2$:  ADD   #10,R1      ;SET UP THE NEXT ADDRESS
(2) 012204 062702 000010          ADD   #10,R2      ;SET UP THE NEXT VECTOR GROUP
(2) 012210 000767          BR    1$           ;GO SEE IF MORE DEVICES REMAIN
(2) 012212 010120          3$:  MOV   R1,(R0)+    ;LOAD DEVICE ADDRESS
(2) 012214 010220          MOV   R2,(R0)+    ;LOAD THE VECTOR ADDRESS
(2) 012216 010320          MOV   R3,(R0)+    ;LOAD THE INTERRUPT PRIORITY LEVEL
(2) 012220 013720 001314          MOV   $CDW1,(R0)+ ;GET THE NUMBER OF LINES IN OPERATION
(2) 012224 012420          MOV   (R4)+,(R0)+ ;LOAD DEVICE PARAMETERS
(2) 012226 100006          BPL   4$           ;IF 20MA MODE SELECTED, SET IT UP
(2) 012230 052760 100000 177772   BIS   #100000,-6(R0);SET THE 20MA FLAG IN DZLVN
(2) 012236 042760 100000 177776   BIC   #100000,-2(R0);CLEAR THE FLAG IN DZPARN
(2) 012244 005020          4$:  CLR   (R0)+      ;DEFAULT OPERATION TO INTERNAL MAINTENANCE MODE
(2) 012246 000754          BR    2$           ;GO BUILD THE NEXT ADDRESS
(2) 012250 012710 177777          5$:  MOV   #-1,(R0)   ;TERMINATE THE DEVICE MAP
(2) 012254 012737 001256 001160   MOV   #DZ.SWREG,SWR ;SET TO SOFTWARE APT SWITCH REGISTER
(2) 012262 000207          RTS    PC           ;RETURN TO PRINT STATUS TABLE

```

```

(2)
(2)
(2)                                     ; *ROUTINE USED TO "AUTO SIZE" THE DZ11
(2)                                     ; *CSR AND VECTOR.
(2)                                     ; *NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
(2)                                     ; * ADDRESS RANGE (160000:163700)
(2)                                     ; * AND THE VECTOR MAY BE ANY WHERE IN THE
(2)                                     ; * FLOATING VECTOR RANGE (300:770)
(2)                                     ; *
(2)

```

```

(2) 012264          AUTO.SIZE:
(2) 012264 000005          RESET                   ;INSURE A BUS INIT.
(2) 012266 105337 001415          DECB                   ;SHOW THAT I WAS HERE
(2) 012272 012702 001500          CSRMAP: MOV    #DZ.MAP,R2 ;LOAD MAP POINTER.
(2) 012276 012703 001320          MOV    #DZ.DW0,R3      ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
(2) 012302 005022          1$:  CLR   (R2)+          ;ZERO ENTIRE MAP
(2) 012304 022702 002000          CMP    #DZ.END,R2     ;ALL DONE?

```

(2)	012310	001374			BNE	1\$:BR IF NO
(2)	012312	105037	001410		CLRB	DZNUM			:SET OCTAL NUMBER OF DZ11'S TO 0
(2)	012316	012702	001500		MOV	#DZ,MAP,R2			
(2)	012322	012701	160000		MOV	#160000,R1			:SET FOR FIRST ADDRESS TO BE TESTED
(2)	012326	012737	012646	000004	MOV	#6\$,a#4			:SET FOR NON-EXISTENT DEVICE TIME OUT
(2)	012334	052711	000040		2\$:	BIS	#BIT5,(R1)		:TRY TO SET MASTER SCAN ENABLE
(2)	012340	052761	000200	000004		BIS	#BIT7,4(R1)		:TRY TO TRANSMIT ON LINE 7
(2)	012346	005000				CLR	R0		:USE R0 AS A COUNTER
(2)	012350	005711			7\$:	TST	(R1)		:HAS TRANSMITTER READY COME UP?
(2)	012352	100403				BMI	8\$:IF SO, GO GET A FINAL CHECK
(2)	012354	005300				DEC	R0		:REDUCE COUNT. TIME UP?
(2)	012356	001374				BNE	7\$:IF NOT, KEEP WAITING
(2)	012360	000463				BR	3\$:ASSUME IT'S NOT A DZ11
(2)	012362	032761	000200	000004	8\$:	BIT	#BIT7,4(R1)		:IS LINE 7 ENABLE STILL SET? IT SHOULD BE
(2)	012370	001457				BEQ	3\$:IF IT'S NOT, ASSUME IT'S NOT A DZ11
(2)	012372	032711	000040			BIT	#BIT5,(R1)		:IS MASTER SCAN ENABLE STILL SET?
(2)	012376	001454				BEQ	3\$:IF NOT, ASSUME IT'S NOT A DZ11
(2)	012400	005000				CLR	R0		
(2)	012402	052711	000020			BIS	#20,(R1)		:SET DEVICE CLEAR
(2)	012406	032711	000020			BIT	#20,(R1)		:SHOULD STAY SET FOR A WHILE IF DZ
(2)	012412	001446				BEQ	3\$:BR IF NOT DZ11
(2)	012414	032711	000020			BIT	#20,(R1)		:WAIT FOR BIT TO CLEAR
(2)	012420	001404				BEQ	+.12		:BR WHEN CLEARED
(2)	012422	104414				DELAY			
(2)	012424	005200				INC	R0		
(2)	012426	001372				BNE	.-12		
(2)	012430	000437				BR	3\$:BIT NOT CLEARED! MUST NOT BE DZ11
(2)	012432	005011				CLR	(R1)		:GET RID OF MASTER SCAN ENABLE
(2)	012434	005061	000004			CLR	4(R1)		:GET RID OF LINE 7 ENABLE
(2)									:AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZ11 CSR ADDRESS.
(2)	012440	010122				MOV	R1,(R2)+		:STORE CSR IN CORE TABLE.
(2)	012442	005722				TST	(R2)+		:POP OVER VECTOR STORE AREA
(2)	012444	012722	000005			MOV	#5,(R2)+		:SET THE DEFAULT BUS LEVEL
(2)	012450	052761	177400	000004		BIS	#177400,4(R1)		:TRY TO SET ALL DTR BITS
(2)	012456	032761	177400	000004		BIT	#177400,4(R1)		:IF ANY SET ASSUME EIA BOARD
(2)	012464	001003				BNE	9\$:IF NONE SET ASSUME BOARD IS
(2)	012466	052762	100000	177776		BIS	#BIT15,-2(R2)		:20 MA, SET 20 MA FLAG
(2)	012474	012722	000377		9\$:	MOV	#377,(R2)+		:SET THE DEFAULT LINE SELECTION PARAMETER
(2)	012500	012712	017070			MOV	#17070,(R2)		:SET THE DEFAULT PARAMETERS
(2)	012504	012223				MOV	(R2)+,(R3)+		:COPY PARAMETERS INTO ETABLE DESCRIPTOR
(2)	012506	005022				CLR	(R2)+		:SET THE DEFAULT MODE OF OPERATION
(2)	012510	012712	177777			MOV	#-1,(R2)		:TERMINATE LIST
(2)	012514	105237	001410			INCB	DZNUM		:UPDATE DEVICE COUNTER
(2)	012520	122737	000020	001410		CMPB	#20,DZNUM		:ARE MAX. NO. OF DEV FOUND?
(2)	012526	001405				BEQ	100\$:YES DON'T LOOK FOR ANY MORE.
(2)	012530	062701	000010		3\$:	ADD	#10,R1		:UPDATE CSR POINTER ADDRESS
(2)	012534	022701	163700			CMP	#163700,R1		
(2)	012540	001275				BNE	2\$:BR IF MORE ADDRESS TO CHECK.
(2)	012542				100\$:				
(2)	012542	105737	001410			TSTB	DZNUM		:WERE ANY DZ11'S FOUND AT ALL?
(2)	012546	001432				BEQ	5\$:ERROR AUTO SIZER FOUND NO DZ11'S IN THIS SYS.
(2)	012550	113701	001410			MOVB	DZNUM,R1		
(2)	012554	110137	001411			MOVB	R1,SAVNUM		:SAVE NUMBER OF DEVICES
(2)	012560	012737	000001	001404		MOV	#1,DZACTV		
(2)	012566	005301			4\$:	DEC	R1		
(2)	012570	001404				BEQ	98\$		

```

(2) 012572 000261          SEC
(2) 012574 006137 001404  ROL    DZACTV
(2) 012600 000772          BR     4$
(2) 012602 013737 001500 001310 98$:  MOV   DZCRO,$BASE ;POINT TO THE ADDRESS OF FIRST DEVICE
(2) 012610 013737 001512 001314    MOV   MANTO,$CDW1 ;INDICATE TO ETABLE WHAT MODE IS BEING USED
(2) 012616 012737 000006 000004 99$:  MOV   #6,@#4 ;RESTORE TRAP VECTOR
(2) 012624 013737 001404 001312    MOV   DZACTV,$DEVM ;SAVE ACTIVE REGISTER
(2) 012632 000410          BR     VECMAP ;GO FIND THE VECTOR NOW.
(2) 012634 104402 010574    5$:  TYPE ,MERR2 ;NOTIFY OPR THAT NO DZ11'S FOUND.
(2) 012640 005000          CLR    RO ;MAKE DATA DISPLAY ZERO
(2) 012642 000000          HALT
(2) 012644 000776          BR     -2 ;STOP THE SHOW
(2) 012646 012716 012530 6$:  MOV   #3$, (SP) ;DISABLE CONT. SW.
(2) 012652 000002          RTI ;ENTERED BY NON-EXISTENT TIME-OUT
(2)                                ;RETURN TO MAINSTREAM
(2) 012654 012737 000340 000022 VECMAP: MOV #340,@#22 ;SET IOT TRAP PRIORITY TO 7
(2) 012662 012737 013000 000020    MOV #4$,@#20 ;SET IOT TRAP VECTOR
(2) 012670 012702 001500          MOV #DZ.MAP,R2 ;SET SOFTWARE POINTER
(2) 012674 012700 000300          MOV #300,R0 ;FLOATING VECTORS START HERE.
(2) 012700 012701 000302          MOV #302,R1 ;PC OF IOT INSTR.
(2) 012704 010120          1$:  MOV R1,(R0)+ ;START FILLING VECTOR AREA
(2) 012706 012721 000004          MOV #4,(R1)+ ;WITH +2; IOT
(2) 012712 022021          CMP (R0)+,(R1)+ ;ADD 2 TO R0 +R1
(2) 012714 020127 001000          CMP R1,#1000 ;HAS THE VECTOR AREA BEEN EXCEEDED?
(2) 012720 101771          BLOS 1$ ;BR IF MORE TO FILL
(2) 012722 013704 001404          2$:  MOV   DZACTV,R4 ;STORE TEMPORARILY
(2) 012726 000241          CLC
(2) 012730 006004          ROR R4 ;BRING OUT A BIT
(2) 012732 103036          BCC 5$ ;BR IF ALL DONE
(2) 012734 106427 000000          MTPS #0 ;ZERO CPU PRIO
(2) 012740 012772 040040 000000    MOV #BIT14+BIT5,@(R2)
(2) 012746 011201          MOV (R2),R1 ;GET CSR
(2) 012750 112761 000200 000004    MOVB #BIT7,4(R1) ;SET THE TCR BIT!
(2)                                ;ATTEMPT TO FORCE AN INTERRUPT
(2) 012756 005200          INC R0 ;STALL
(2) 012760 001376          BNE -2 ;
(2) 012762 012762 000300 000002    MOV #300,2(R2) ;NO INTERRUPT ASSUME 300 AND FIX DZ11 LATER
(2) 012770 000005          3$:  RESET ;INIT
(2) 012772 062702 000014          ADD #14,R2 ;POP SOFTWARE POINTER
(2) 012776 000753          BR 2$ ;KEEP GOING
(2) 013000 011662 000002          4$:  MOV (SP),2(R2) ;GET VECTOR ADDRESS
(2) 013004 162762 000010 000002    SUB #10,2(R2) ;POINT BACK TO THE CORRECT VECTOR
(2) 013012 042762 000007 000002    BIC #7,2(R2) ;CLEAR JUNK
(2) 013020 022626          POP2SP ;POP IOT JUNK OFF STACK
(2) 013022 012716 012770          MOV #3$, (SP) ;SET FOR RETURN
(2) 013026 000002          RTI
(2) 013030 013737 001502 001304 5$:  MOV DZVCO,$VECT1 ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
(2) 013036 012737 005122 000020    MOV #.SCOPE,IOTVEC ;RESTORE THE SCOPE TRAP
(2) 013044 000207          RTS PC ;ALL DONE WITH 'AUTO SIZING'
(2)

```

8621

8622

(1)

(1)

(1)

(5)

(6)

(5)

(3)

(3)

(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)

8623

8624

8625

8626

8628

(5)

(4)

(2)

(2)

8629

8630

8631

***** TEST 1 *****

*THIS TEST PROVES THE SLAVE SYNC RESPONSE
*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
* DZCSR, DZRBUF, DZTCR, DZMSR

::* TEST 1

```

TST1:  SCOPE
        MOV     #1,$STSTNM      ;LOAD THE NUMBER OF THIS TEST
        MOV     #TST2,NEXT     ;POINT TO THE START OF THE NEXT TEST
        MOV     #5$,4          ;SET TRAP VECTOR
        MOV     #PR7,6         ;SET PRIORITY TO LEVEL 7
        MOV     #1$,LOCK       ;SET RETURN IF SW09=11
1$:     MOV     DZCSR,R0        ;SET ADDRESS TO TEST
        MOV     (R0),R1        ;READ THE ADDRESS
        NOP                               ;WASTE TIME
        CLR     (R0)           ;WRITE THE ADDRESS
        NOP                               ;WASTE TIME
2$:     MOV     #2$,LOCK       ;SET RETURN ADDRESS FOR SW09
        MOV     DZRBUF,R0      ;SET ADDRESS TO TEST
        MOV     (R0),R1        ;READ THE ADDRESS
        NOP                               ;
        CLR     (R0)           ;WRITE THE ADDRESS
        NOP                               ;WASTE TIME
3$:     MOV     #3$,LOCK       ;SET RETURN ADDRESS FOR SW09
        MOV     DZTCR,R0      ;SET ADDRESS TO TEST
        MOV     (R0),R1        ;READ THE ADDRESS
        NOP                               ;
        CLR     (R0)           ;WRITE THE ADDRESS
        NOP                               ;WASTE TIME
4$:     MOV     #4$,LOCK       ;SET RETURN ADDRESS
        MOV     DZMSR,R0      ;SET ADDRESS TO TEST
        MOV     (R0),R1        ;READ FROM ADDRESS
        NOP                               ;
        CLR     (R0)           ;WRITE THE ADDRESS
        NOP                               ;
        MOV     #6$,4          ;SET TRAP CATCHER BACK TO NORMAL
        CLR     6              ;
5$:     ADVANCE                ;SCOPE THIS TEST
        MOV     (SP),R1        ;SAVE PC OF TRAP
        CMP     (SP)+,(SP)+    ;POP TRAP OFF STACK
        ERROR  1              ;*NO SLAVE SYNC RESPONSE.
        SCOPE1                ;SW09=1?
        JMP     (R1)          ;RTI

```

***** TEST 2 *****

*THIS TEST PROVES THAT BIT 'DCLR'
*CAN BE SET AND THAT IT WILL CLEAR
*BY ITSELF AFTER A PERIOD OF TIME.

::* TEST 2

```

TST2:  SCOPE
        MOV     #2,$STSTNM      ;LOAD THE NUMBER OF THIS TEST
        MOV     #TST3,NEXT     ;POINT TO THE START OF THE NEXT TEST
        MOV     DZCSR,R0        ;SET POINTER
        MOV     #DCLR,R5       ;SET DCLR
        MOV     R5,(R0)        ;WRITE DCLR INTO DZCSR

```

```

8632 013266 011004      MOV      (R0),R4      ;READ BACK DZCSR
8633 013270 020504      CMP      R5,R4      ;DZCSR OK?
8634 013272 001401      BEQ      1$          ;IF IT IS SET SKIP THE ERROR CALL
8635 013274 104002      ERROR   2           ;*DCLR SHOULD BE SET..MOMENTARILY
8636                                     ;NOW LETS WATCH IT DISAPPEAR
8637 013276 005002      1$:  CLR      R2          ;SET COUNTER TO 0
8638 013300 005005      CLR      R5          ;SET EXPECTED TO 0
8639 013302 005003      CLR      R3          ;DUAL LOOP COUNTER
8640 013304 011004      2$:  MOV      (R0),R4      ;IS DCLR CLEAR?
8641 013306 001405      BEQ      3$          ;IF YES , GO TO THE NEXT TEST
8642 013310 005203      INC      R3          ;IF NO,COUNT 1 OF 65535 TICKS
8643                                     ;THE WORD CREATED BY THE IMMEDIATE 0 WILL BE
8644                                     ;THE COUNTER
8645 013312 001374      BNE      2$          ;HAS THE TIME EXPIRED? IF NO, GO TEST BIT AGAIN
8646 013314 005302      DEC      R2          ;HAS THE TOTAL TIME EXPIRED?
8647 013316 001372      BNE      2$          ;IF NO, CHECK THE BIT AGAIN
8648 013320 104002      ERROR   2           ;*DCLR FAILED TO CLEAR
8649 013322

```

```

8652 (2) ;***** TEST 3 *****
      (2) ;*TEST TO VERIFY THAT BIT 'MAINT' CAN
      (2) ;*BE SET. THEN VERIFY THAT BIT 'MAINT' CAN
      (2) ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
      (2) ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
      (2) ;*CLEARED BY A 'DEVICE CLEAR'

```

::* TEST 3

```

(4) 013322 000004      TST3: SCOPE
(4) 013324 012737 000003 001122      MOV      #3,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(4) 013332 012737 013414 001360      MOV      #TST4,NEXT     ;POINT TO THE START OF THE NEXT TEST
(2) 013340 013700 002042      MOV      DZCSR,R0      ;GET BASE ADDRESS
(2) 013344 012705 000010      MOV      #MAINT,R5     ;SET BIT
(2) 013350 010510      MOV      R5,(R0)       ;SET SET IN DEVICE
(2) 013352 011004      MOV      (R0),R4      ;READ THE BIT FROM DEVICE
(2) 013354 020504      CMP      R5,R4        ;WAS BIT SET?
(2) 013356 001401      BEQ      1$          ;BR IF YES
(2) 013360 104002      ERROR   2           ;*BIT R/W FAILURE
(2) 013362 040510      1$:  BIC      R5,(R0)     ;CLEAR THE BIT.
(2) 013364 011004      MOV      (R0),R4      ;READ DEVICE
(2) 013366 001404      BEQ      2$          ;BR IF BITS WERE CLEARED.
(2) 013370 010546      MOV      R5,-(SP)     ;SAVE THE BIT
(2) 013372 005005      CLR      R5          ;SET EXPECTED RESULTS TO 0
(2) 013374 104002      ERROR   2           ;*BIT FAILED TO CLEAR
(2) 013376 012605      MOV      (SP)+,R5     ;RESTORE THE BIT.
(2) 013400 010510      2$:  MOV      R5,(R0)     ;SET THE BIT AGAIN
(2) 013402 104413      DEVICE.CLR          ;ISSUE DEVICE CLEAR
(2) 013404 011004      MOV      (R0),R4      ;READ THE BIT.
(2) 013406 001402      BEQ      3$          ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013410 005005      CLR      R5          ;SET EXPECTED TO ZERO
(2) 013412 104002      ERROR   2           ;*BIT NOT CLEARED BY DEVICE CLEAR
(2) 013414

```

```

(4) ;***** TEST 4 *****
(2) ;*TEST TO VERIFY THAT BIT 'MSENAB' CAN
(2) ;*BE SET. THEN VERIFY THAT BIT 'MSENAB' CAN
(2) ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
(2) ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
(2) ;*CLEARED BY A 'DEVICE CLEAR'

```

```

(4)          ;:* TEST 4
(7)          ;:*****
(6) 013414 000004          TST4: SCOPE
(4) 013416 012737 000004 001122      MOV #4,$STSTNM      ;LOAD THE NUMBER OF THIS TEST
(4) 013424 012737 013506 001360      MOV #TST5,NEXT     ;POINT TO THE START OF THE NEXT TEST
(2) 013432 013700 002042              MOV DZCSR,R0       ;GET BASE ADDRESS
(2) 013436 012705 000040              MOV #MSENAB,R5     ;SET BIT
(2) 013442 010510              MOV R5,(R0)        ;SET SET IN DEVICE
(2) 013444 011004              MOV (R0),R4        ;READ THE BIT FROM DEVICE
(2) 013446 020504              CMP R5,R4          ;WAS BIT SET?
(2) 013450 001401              BEQ 1$             ;BR IF YES
(2) 013452 104002              ERROR 2            ;*BIT R/W FAILURE
(2) 013454 040510 1$: BIC R5,(R0)          ;CLEAR THE BIT.
(2) 013456 011004              MOV (R0),R4        ;READ DEVICE
(2) 013460 001404              BEQ 2$             ;BR IF BITS WERE CLEARED.
(2) 013462 010546              MOV R5,-(SP)       ;SAVE THE BIT
(2) 013464 005005              CLR R5             ;SET EXPECTED RESULTS TO 0
(2) 013466 104002              ERROR 2            ;*BIT FAILED TO CLEAR
(2) 013470 012605              MOV (SP)+,R5       ;RESTORE THE BIT.
(2) 013472 010510 2$: MOV R5,(R0)          ;SET THE BIT AGAIN
(2) 013474 104413              DEVICE.CLR        ;ISSUE DEVICE CLEAR
(2) 013476 011004              MOV (R0),R4        ;READ THE BIT.
(2) 013500 001402              BEQ 3$             ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013502 005005              CLR R5             ;SET EXPECTED TO ZERO
(2) 013504 104002              ERROR 2            ;*BIT NOT CLEARED BY DEVICE CLEAR
(2) 013506          3$:
(4)          ;:***** TEST 5 *****
(2)          ;:*TEST TO VERIFY THAT BIT "SILOEN" CAN
(2)          ;:*BE SET. THEN VERIFY THAT BIT "SILOEN" CAN
(2)          ;:*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
(2)          ;:*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
(2)          ;:*CLEARED BY A "DEVICE CLEAR"
(4)          ;:* TEST 5
(7)          ;:*****
(6) 013506 000004          TST5: SCOPE
(4) 013510 012737 000005 001122      MOV #5,$STSTNM      ;LOAD THE NUMBER OF THIS TEST
(4) 013516 012737 013600 001360      MOV #TST6,NEXT     ;POINT TO THE START OF THE NEXT TEST
(2) 013524 013700 002042              MOV DZCSR,R0       ;GET BASE ADDRESS
(2) 013530 012705 010000              MOV #SILOEN,R5     ;SET BIT
(2) 013534 010510              MOV R5,(R0)        ;SET SET IN DEVICE
(2) 013536 011004              MOV (R0),R4        ;READ THE BIT FROM DEVICE
(2) 013540 020504              CMP R5,R4          ;WAS BIT SET?
(2) 013542 001401              BEQ 1$             ;BR IF YES
(2) 013544 104002              ERROR 2            ;*BIT R/W FAILURE
(2) 013546 040510 1$: BIC R5,(R0)          ;CLEAR THE BIT.
(2) 013550 011004              MOV (R0),R4        ;READ DEVICE
(2) 013552 001404              BEQ 2$             ;BR IF BITS WERE CLEARED.
(2) 013554 010546              MOV R5,-(SP)       ;SAVE THE BIT
(2) 013556 005005              CLR R5             ;SET EXPECTED RESULTS TO 0
(2) 013560 104002              ERROR 2            ;*BIT FAILED TO CLEAR
(2) 013562 012605              MOV (SP)+,R5       ;RESTORE THE BIT.
(2) 013564 010510 2$: MOV R5,(R0)          ;SET THE BIT AGAIN
(2) 013566 104413              DEVICE.CLR        ;ISSUE DEVICE CLEAR
(2) 013570 011004              MOV (R0),R4        ;READ THE BIT.
(2) 013572 001402              BEQ 3$             ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013574 005005              CLR R5             ;SET EXPECTED TO ZERO

```

(2) 013576 104002
(2) 013600

ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR

3\$:

***** TEST 6 *****
;*TEST TO VERIFY THAT BIT 'RIE' CAN
;*BE SET. THEN VERIFY THAT BIT 'RIE' CAN
;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
;*CLEARED BY A 'DEVICE CLEAR'

::* TEST 6

(6) 013600 000004
(4) 013602 012737 000006 001122
(4) 013610 012737 013672 001360
(2) 013616 013700 002042
(2) 013622 012705 000100
(2) 013626 010510
(2) 013630 011004
(2) 013632 020504
(2) 013634 001401
(2) 013636 104002
(2) 013640 040510
(2) 013642 011004
(2) 013644 001404
(2) 013646 010546
(2) 013650 005005
(2) 013652 104002
(2) 013654 012605
(2) 013656 010510
(2) 013660 104413
(2) 013662 011004
(2) 013664 001402
(2) 013666 005005
(2) 013670 104002
(2) 013672

TST6: SCOPE
MOV #6,\$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST7,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;GET BASE ADDRESS
MOV #RIE,R5 ;SET BIT
MOV R5,(R0) ;SET SET IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 1\$;BR IF YES
ERROR 2 ;*BIT R/W FAILURE
1\$: BIC R5,(R0) ;CLEAR THE BIT.
MOV (R0),R4 ;READ DEVICE
BEQ 2\$;BR IF BITS WERE CLEARED.
MOV R5,-(SP) ;SAVE THE BIT
CLR R5 ;SET EXPECTED RESULTS TO 0
ERROR 2 ;*BIT FAILED TO CLEAR
2\$: MOV (SP)+,R5 ;RESTORE THE BIT.
MOV R5,(R0) ;SET THE BIT AGAIN
DEVICE.CLR ;ISSUE DEVICE CLEAR
MOV (R0),R4 ;READ THE BIT.
BEQ 3\$;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
CLR R5 ;SET EXPECTED TO ZERO
3\$: ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR

(4)
(2)
(2)
(2)
(2)
(2)
(4)
(7)

***** TEST 7 *****
;*TEST TO VERIFY THAT BIT 'TIE' CAN
;*BE SET. THEN VERIFY THAT BIT 'TIE' CAN
;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
;*CLEARED BY A 'DEVICE CLEAR'

::* TEST 7

(6) 013672 000004
(4) 013674 012737 000007 001122
(4) 013702 012737 013764 001360
(2) 013710 013700 002042
(2) 013714 012705 040000
(2) 013720 010510
(2) 013722 011004
(2) 013724 020504
(2) 013726 001401
(2) 013730 104002
(2) 013732 040510
(2) 013734 011004
(2) 013736 001404
(2) 013740 010546

TST7: SCOPE
MOV #7,\$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST10,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;GET BASE ADDRESS
MOV #TIE,R5 ;SET BIT
MOV R5,(R0) ;SET SET IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 1\$;BR IF YES
ERROR 2 ;*BIT R/W FAILURE
1\$: BIC R5,(R0) ;CLEAR THE BIT.
MOV (R0),R4 ;READ DEVICE
BEQ 2\$;BR IF BITS WERE CLEARED.
MOV R5,-(SP) ;SAVE THE BIT

```

(2) 013742 005005          CLR      R5          ;SET EXPECTED RESULTS TO 0
(2) 013744 104002          ERROR    2          ;*BIT FAILED TO CLEAR
(2) 013746 012605          MOV      (SP)+,R5   ;RESTORE THE BIT.
(2) 013750 010510          2$:      MOV      R5,(R0) ;SET THE BIT AGAIN
(2) 013752 104413          DEVICE.CLR        ;ISSUE DEVICE CLEAR
(2) 013754 011004          MOV      (R0),R4   ;READ THE BIT.
(2) 013756 001402          BEQ      3$        ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013760 005005          CLR      R5          ;SET EXPECTED TO ZERO
(2) 013762 104002          ERROR    2          ;*BIT NOT CLEARED BY DEVICE CLEAR
(2) 013764

```

```

8653
(1)
(1)
(1)
(1)
(3)
(6)
::* TEST 10
*****

```

```

(5) 013764 000004          TST10: SCOPE
(3) 013766 012737 000010 001122  MOV      #10,$STSTM ;LOAD THE NUMBER OF THIS TEST
(3) 013774 012737 014122 001360  MOV      #TST11,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 014002 013700 002056          MOV      DZTCR,R0   ;SET DEVICE ADDRESS
(1) 014006 012705 000001          MOV      #TCR0,R5   ;SET EXPECTED RESULTS
(1) 014012 012737 014020 001362  MOV      #1$,LOCK   ;SET FOR SW09
(1) 014020 010510          1$:      MOV      R5,(R0)   ;SET THE BIT
(1) 014022 011004          MOV      (R0),R4   ;READ THE BIT FROM THE DEVICE
(1) 014024 042704 177400          BIC      #^C<377>,R4 ;CLEAR HIGH BYTE
(1) 014030 020504          CMP      R5,R4     ;WAS BIT OK?
(1) 014032 001401          BEQ      2$        ;BR IF YES
(1) 014034 104002          ERROR    2          ;*BIT FAILED TO SET.
(1) 014036 040510          2$:      BIC      R5,(R0)   ;CLEAR THE BIT
(1) 014040 011004          MOV      (R0),R4   ;READ THE REGISTER
(1) 014042 042704 177400          BIC      #^C<377>,R4 ;CLEAR HIGH BYTE
(1) 014046 005704          TST      R4        ;BITS CLEAR?
(1) 014050 001404          BEQ      3$        ;BR IF YES
(1) 014052 010546          MOV      R5,-(SP)  ;SAVE GOOD RESULTS
(1) 014054 005005          CLR      R5          ;SET EXPECTED TO 0
(1) 014056 104002          ERROR    2          ;*REPORT BIT NOT CLEAR
(1) 014060 012605          MOV      (SP)+,R5  ;RESTORE R5
(1) 014062 010510          3$:      MOV      R5,(R0)   ;SET THE BIT AGAIN.
(1) 014064 104413          DEVICE.CLR        ;ISSUE DEVICE CLEAR
(1) 014066 011004          MOV      (R0),R4   ;READ THE REGISTER
(1) 014070 042704 177400          BIC      #^C<377>,R4 ;CLEAR HIGH BYTE
(1) 014074 005704          TST      R4        ;BITS CLEAR?
(1) 014076 001404          BEQ      4$        ;BR IF YES
(1) 014100 010546          MOV      R5,-(SP)  ;SAVE GOOD RESULTS
(1) 014102 005005          CLR      R5          ;SET EXPECTED TO 0
(1) 014104 104002          ERROR    2          ;*REPORT BIT NOT CLEAR
(1) 014106 012605          MOV      (SP)+,R5  ;RESTORE R5
(1) 014110 104401          4$:      SCOP1          ;LOCK ON BIT? SET SW09=1
(1) 014112 106305          ASLB     R5          ;CHANGE TO NEXT BIT
(1) 014114 001341          BNE     1$          ;CONTINUE TESTING
(1) 014116 005037 001362          CLR      LOCK      ;MAKE SURE TIGHT LOOP IS CLEANED UP

```

```

8654
(1)
(1)
(1)
***** TEST 11 *****
*THIS TESTS THAT ALL OF THE FOLLOWING
*BITS CAN BE: SET, CLEARED, CLEARED BY 'RESET INSTR *NOT* DEVICE CLEAR ''
*BITS TESTED ARE:

```

```

(1)                                     ;* DTR0, DTR1, DTR2, DTR3, DTR4, DTR5, DTR6, DTR7
(1)                                     ;*THIS TEST IS NOT DONE IF MODULE IS 20MA VERSION
(3)                                     ;:* TEST 11
(6)                                     ;*****
(5) 014122 000004                       TST11: SCOPE
(3) 014124 012737 000011 001122         MOV #11,$STSTNM ;LOAD THE NUMBER OF THIS TEST
(3) 014132 012737 014304 001360         MOV #TST12,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 014140 013700 002056                 MOV DZTCR,R0 ;SET DEVICE ADDRESS
(1) 014144 012705 000400                 MOV #DTR0,R5 ;SET EXPECTED RESULTS
(1) 014150 012737 014166 001362         MOV #1$,LOCK ;SET FOR SW09
(1) 014156 105737 001414                 TSTB EIAFLG ;20MA OR EIA
(1) 014162 100001                         BPL 1$ ;BR IF EIA
(1) 014164 104400                         ADVANCE ;EXIT TEST
(1) 014166 010510 1$: MOV R5,(R0) ;SET THE BIT
(1) 014170 011004                         MOV (R0),R4 ;READ THE BIT FROM THE DEVICE
(1) 014172 105004                         CLRB R4 ;CLEAR LOW BYTE
(1) 014174 020504                         CMP R5,R4 ;WAS BIT OK?
(1) 014176 001401                         BEQ 2$ ;BR IF YES
(1) 014200 104002                         ERROR 2 ;*BIT FAILED TO SET.
(1) 014202 040510 2$: BIC R5,(R0) ;CLEAR THE BIT
(1) 014204 011004                         MOV (R0),R4 ;READ THE REGISTER
(1) 014206 105004                         CLRB R4 ;CLEAR LOW BYTE
(1) 014210 005704                         TST R4 ;BITS CLEAR?
(1) 014212 001404                         BEQ 3$ ;BR IF YES
(1) 014214 010546                         MOV R5,-(SP) ;SAVE GOOD RESULTS
(1) 014216 005005                         CLR R5 ;SET EXPECTED TO 0
(1) 014220 104002                         ERROR 2 ;*REPORT BIT NOT CLEAR
(1) 014222 012605                         MOV (SP)+,R5 ;RESTORE R5
(1) 014224 010510 3$: MOV R5,(R0) ;SET THE BIT AGAIN.
(1) 014226 104413                         DEVICE.CLR ;ISSUE DEVICE CLEAR
(1) 014230 011004                         MOV (R0),R4 ;READ THE REGISTER
(1) 014232 105004                         CLRB R4 ;CLEAR LOW BYTE
(1) 014234 030510                         BIT R5,(R0) ;WAS BIT CLEARED BY DEVICE.CLR?
(1) 014236 001001                         BNE 4$ ;BR IF NO (IT SHOULDN'T BE CLEAR)
(1) 014240 104002                         ERROR 2 ;*BIT CLEARED BY DEVICE.CLR
(1) 014242 104401 4$: SCOP1 ;LOCK ON BIT? SW09=1
(1) 014244 006305                         ASL R5 ;CHANGE TO NEXT BIT
(1) 014246 001347                         BNE 1$ ;IF NOT DONE LOOP
(1) 014250 012710 177400                 MOV #177400,(R0) ;SET ALL DTR BITS
(1) 014254 005005                         CLR R5 ;CLEAR LOCATION FOR ERROR PRINTOUT
(1) 014256 005227 000000 5$: INC #0 ;ACT DELAY LOOP FOR
(1) 014262 001375                         BNE 5$ ;RESET INSTRUCTION
(1) 014264 000005                         RESET ;ISSUE A BUS INIT
(1) 014266 011004                         MOV (R0),R4 ;READ REGISTER
(1) 014270 105004                         CLRB R4 ;CLEAR LOW BYTE
(1) 014272 005704                         TST R4 ;DTR BITS CLEAR?
(1) 014274 001401                         BEQ +4 ;IF YES CONTINUE
(1) 014276 104002                         ERROR 2 ;IF NO PRINT ERROR
(1) 014300 005037 001362                 CLR LOCK ;MAKE SURE TIGHT LOOP IS CLEANED UP
8655 ;***** TEST 12 *****
(1) ;*THIS TEST PERFORMS RESET TESTING &
(1) ;*TESTING OF WRITE ONLY OR READ ONLY BIT
(1) ;* TEST BITS 'RDONE, BIT11, BIT10, BIT9, BIT8, BIT2, BIT1
(1) ;* ;* BIT0, SILOAL' ARE READ ONLY AND THAT TRDY IS
(1) ;* ;* ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.
(1) ;*

```

```

(3)
(6)
(5) 014304 000004
(3) 014306 012737 000012 001122
(3) 014314 012737 014422 001360
(1) 014322 013700 002042
(1) 014326 005005
(1) 014330 012710 027607
(1)
(1) 014334 011004
(1) 014336 001401
(1) 014340 104002
(1) 014342 012710 100000
(1) 014346 011004
(1) 014350 001401
(1) 014352 104002
(1) 014354 012705 100000
(1) 014360 005077 165466
(1) 014364 052777 000001 165464
(1) 014372 052710 000040
(1) 014376 052705 000040
(1) 014402 005002
(1) 014404 011004
(1) 014406 020504
(1) 014410 001404
(1) 014412 104414
(1) 014414 005202
(1) 014416 001372
(1) 014420 104002
(1) 014422

```

```

::* TEST 12
:*****
TST12: SCOPE
MOV #12,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST13,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;SET ADDRESS TO R0
CLR R5 ;SET EXPECTED TO 0
MOV #RDONE+BIT11+BIT10+BIT9+BIT8+BIT2+BIT1+BIT0+SILOAL,(R0)
;WRITE THE BITS
MOV (R0),R4 ;READ BACK THE BITS
BEQ 1$ ;BR IF NONE ARE SET.
ERROR 2 ;*BITS WERE SET.
1$: MOV #TRDY,(R0) ;ATTEMPT TO WRITE TRDY
MOV (R0),R4 ;READ TRDY
BEQ 2$ ;BR IF NOT SET
ERROR 2 ;*
2$: MOV #TRDY,R5 ;SET EXPECTED BIT
CLR @DZLPR ;LOAD LINE 0
BIS #TCR0,@DZTCR ;SET TCR BIT
BIS #MSENAB,(R0)
BIS #MSENAB,R5 ;SET SCAN ENABLE
CLR R2 ;SET COUNTER TO ZERO
3$: MOV (R0),R4 ;READ THE REGISTER
CMP R5,R4 ;BIT SET?
BEQ 4$ ;BR IF YES
DELAY ;STALL TIME
INC R2 ;UPDATE COUNTER
BNE 3$ ;BR IF COUNTER NOT DONE.
4$: ERROR 2 ;*TRDY NOT SET!

```

```

8656
8657
8658
8659
8660
8661
8662
8663
8665

```

```

:***** TEST 13 *****
:THIS TEST PERFORMS RESET TESTING AND
:TESTING OF READ ONLY AND WRITE ONLY BITS
: IN REGISTER DZCSR
:VERIFY THAT 'TIE', 'SILOEN', 'RIE', 'MSENAB', 'MAINT'
:ARE THE ONLY R/W BITS IN THE DZCSR.
:THEN VERIFY THAT A RESET WILL CLEAR THESE BITS
:THIS TEST ALSO CHECKS BYTE OPERATIONS ON THE CSR

```

```

(5)
(4) 014422 000004
(2) 014424 012737 000013 001122
(2) 014432 012737 014552 001360
8666 014440 104413
8667 014442 013700 002042
8668 014446 012710 177757
8669 014452 012705 050150
8670 014456 011004
8671 014460 020405
8672 014462 001401
8673 014464 104002
8674 014466 105010
8675 014470 105005
8676 014472 011004
8677 014474 020405
8678 014476 001401

```

```

::* TEST 13
:*****
TST13: SCOPE
MOV #13,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST14,NEXT ;POINT TO THE START OF THE NEXT TEST
DEVICE.CLR
MOV DZCSR,R0 ;SET UP FOR ERROR MESSAGE
MOV #^C<DCLR>,(R0) ;TRY TO WRITE
MOV #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP EXPECTED VS ACTUAL
BEQ 1$ ;YES
ERROR 2 ;*NO
1$: CLRB (R0) ;CLEAR LOWER BYTE OF CSR
CLRB R5 ;SET EXPECTED
MOV (R0),R4 ;READ CSR BITS
CMP R4,R5 ;COMPARE ACTUAL TO EXPECTED
BEQ 3$ ;BRANCH IF SAME

```

8679 014500 104002
 8680 014502 012710 177757
 8681 014506 105077 165332
 8682 014512 012705 000150
 8683
 8684 014516 011004
 8685 014520 020405
 8686 014522 001401
 8687 014524 104002
 8688 014526 012710 177757
 8689 014532 005005
 8690 014534 005227 000000
 8691 014540 001375
 8692 014542 000005
 8693 014544 011004
 8694 014546 001401
 8695 014550 104002
 8696 014552

```

3$: ERROR 2 ; OTHERWISE PRINT ERROR
MOV #^C<DCLR>,(R0) ; RESET CSR BITS
CLRB @HDZCSR ; CLEAR HIGH BYTE OF CSR
MOV #RIE!MSENAB!MAINT,R5 ; SET R5 TO EXPECTED RESULTS
MOV (R0),R4 ; READ CSR
CMP R4,R5 ; ACTUAL = EXPECTED?
BEQ 4$ ; BRANCH IF SAME
ERROR 2 ; OTHERWISE PRINTOUT ERROR
4$: MOV #^C<DCLR>,(R0) ; RESET CSR BITS
CLR R5 ; SET R5 TO EXPECTED RESULTS
5$: INC #0 ; DELAY TIMER FOR
BNE 5$ ; ACT-11 COMPATIBILITY
RESET ; ISSUE BUS INIT
MOV (R0),R4 ; READ CSR REGISTER
BEQ 2$ ; BRANCH IF CSR IS CLEAR
ERROR 2 ; IF NOT PRINT ERROR

```

8697
 (1)
 (1)
 (1)
 (3)
 (6)
 (5) 014552 000004
 (3) 014554 012737 000014 001122
 (3) 014562 012737 014642 001360
 (1) 014570 104413
 (1) 014572 013700 002046
 (1) 014576 011005
 (1) 014600 012777 177777 165244
 (1) 014606 011004
 (1) 014610 042705 104000
 (1) 014614 020405
 (1) 014616 001401
 (1) 014620 104002
 (1) 014622 010403
 (1) 014624 005103
 (1) 014626 010377 165220
 (1) 014632 011004
 (1) 014634 020405
 (1) 014636 001401
 (1) 014640 104002
 (1) 014642

```

:***** TEST 14 *****
:*THIS TEST PERFORMS RESET TESTING AND
:*TESTING OF READ ONLY REGISTER DZRBUF
:*AND TESTING OF WRITE ONLY REGISTER DZLPR
::* TEST 14
:*****
TST14: SCOPE
MOV #14,$STSTNM ; LOAD THE NUMBER OF THIS TEST
MOV #TST15,NEXT ; POINT TO THE START OF THE NEXT TEST
DEVICE.CLR ; CLEAR DZ11
MOV DZRBUF,R0 ; SET UP FOR ERROR MESSAGE
MOV (R0),R5 ; SET EXPECTED
MOV #-1,@DZLPR ; TRY TO WRITE ALL 1'S
MOV (R0),R4 ; ACTUAL
BIC #DVALID!BIT11,R5 ; DITTO
CMP R4,R5 ; CMP ACTUAL VS EXPECTED
BEQ 1$ ; IF YES,GO CONTINUE PROCESSING
ERROR 2 ; *ERROR- BIT PATTERN NOT CORRECT
1$: MOV R4,R3 ; GET A COPY OF THE ACTUAL BIT PATTERN
COM R3 ; GET THE LOGICAL INVERSE OF THE BIT PATTERN
MOV R3,@DZLPR ; TRY TO WRITE
MOV (R0),R4 ; ACTUAL
CMP R4,R5 ; CMP ACTUAL VS EXPECTED
BEQ 2$ ; IF YES, GET OUT OF THIS TEST
ERROR 2 ; *NO
2$:

```

8698
 (1)
 (1)
 (1)
 (3)
 (6)
 (5) 014642 000004
 (3) 014644 012737 000015 001122
 (3) 014652 012737 014726 001360
 (1) 014660 104413
 (1) 014662 013700 002062
 (1) 014666 011005

```

:***** TEST 15 *****
:*THIS TEST PERFORMS RESET TESTING AND
:*TESTING OF READ ONLY REGISTER DZMSR
:*AND TESTING OF WRITE ONLY REGISTER DZTDR
::* TEST 15
:*****
TST15: SCOPE
MOV #15,$STSTNM ; LOAD THE NUMBER OF THIS TEST
MOV #TST16,NEXT ; POINT TO THE START OF THE NEXT TEST
DEVICE.CLR ; CLEAR DZ11
MOV DZMSR,R0 ; SET UP FOR ERROR MESSAGE
MOV (R0),R5 ; SET EXPECTED

```

```

(1) 014670 012777 177777 165170      MOV    #-1,@DZTDR      ;TRY TO WRITE ALL 1'S
(1) 014676 011004                      MOV    (R0),R4         ;ACTUAL
(1) 014700 020405                      CMP    R4,R5           ;CMP ACTUAL VS EXPECTED
(1) 014702 001401                      BEQ    1$              ;IF YES,GO CONTINUE PROCESSING
(1) 014704 104002                      ERROR  2              ;*ERROR- BIT PATTERN NOT CORRECT
(1) 014706 010403                      1$:  MOV    R4,R3       ;GET A COPY OF THE ACTUAL BIT PATTERN
(1) 014710 005103                      COM    R3              ;GET THE LOGICAL INVERSE OF THE BIT PATTERN
(1) 014712 010377 165150              MOV    R3,@DZTDR      ;TRY TO WRITE
(1) 014716 011004                      MOV    (R0),R4         ;ACTUAL
(1) 014720 020405                      CMP    R4,R5           ;CMP ACTUAL VS EXPECTED
(1) 014722 001401                      BEQ    2$              ;IF YES, GET OUT OF THIS TEST
(1) 014724 104002                      ERROR  2              ;*NO
(1) 014726

```

8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8712
8713

```

:***** TEST 16 *****
:*VERIFY THAT IF WE ARE IN 'STAGGERED' MODE
:*THAT SETTING 'DTR' FOR A LINE WILL
:*BRING UP 'RING' AND 'CARRIER' FOR THE
:*ASSOCIATED LINE IN WHICH WE ARE STAGGERED!
:* LINE0 DTR= LINE1 RING AND CARRIER
:* LINE1 DTR= LINE0 RING AND CARRIER
:* LINE2 DTR= LINE3 RING AND CARRIER
:* LINE3 DTR= LINE 4 RING AND CARRIER
:*
:* ETC...

```

::* TEST 16

```

(5)
(4) 014726 000004
(2) 014730 012737 000016 001122      TST16: SCOPE
(2) 014736 012737 015122 001360      MOV    #16,$STSTNM    ;LOAD THE NUMBER OF THIS TEST
(1) 014744 012737 015016 001362      MOV    #TST17,NEXT    ;POINT TO THE START OF THE NEXT TEST
8714 014752 105737 001414              MOV    #1$,LOCK       ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8715 014756 100001                      TSTB   EIAFLG         ;EIA OR 20MA?
8716 014760 104400                      BPL    10$            ;BR IF EIA
8717 014762 013700 002062              ADVANCE ;EXIT TEST
8718 014766 104413                      10$:  MOV    DZMSR,R0    ;SET REGISTER
8719 014770 005003                      DEVICE.CLR           ;INIT DZ11
8720 014772 012702 000001              CLR    R3             ;ZERO LINE NUMBER
8721 014776 005737 001370              MOV    #1,R2         ;SET POINTER
8722 015002 100405                      TST    MODE          ;ARE WE IN STAGGERED MODE?
8723 015004 013737 001360 001126      BMI    1$            ;YES WE ARE!
8724 015012 000177 164110              MOV    NEXT,$LPADR   ;LEAVE THIS TEST! NOT STAGGERED
8725 015016 130237 001364              JMP    2$LPADR       ;EXIT
8726 015022 001004                      1$:  BITB   R2,LINE    ;TEST THIS LINE?
8727 015024 005203                      BNE    3$            ;YES
8728 015026 106302                      2$:  INC    R3         ;LINE #
8729 015030 103372                      ASLB   R2             ;GET NEXT LINE
8730 015032 104400                      BCC    1$            ;KEEP TESTING
8731 015034 010204                      ADVANCE ;ADVANCE THIS TEST
8732 015036 032703 000001              3$:  MOV    R2,R4     ;SAVE BINARY BIT FOR LINE #
8733 015042 001402                      BIT    #BIT0,R3      ;GET STAGGERED COMPANION LINE
8734 015044 006204                      BEQ    4$            ;BR IF LINE EVEN
8735 015046 000401                      ASR    R4             ;ADJUST LINE
8736 015050 006304                      BR     5$            ;
8737 015052 005005                      4$:  ASL    R4         ;ADJUST LINE
5$:  CLR    R5         ;SET EXPECTED

```

8738	015054	150405			BICB	R4,R5	:
8739	015056	000305			SWAB	R5	:
8740	015060	150405			BISB	R4,R5	:
8741	015062	150277	164772		BISB	R2,@HDZTCR	:SET DTR
8742	015066	104414			DELAY		:CABLE DELAY
8743	015070	011004			MOV	(R0),R4	:READ MSR REGISTER
8744	015072	020504			CMP	R5,R4	:OK?
8745	015074	001401			BEQ	6\$:YES
8746	015076	104002			ERROR	2	:*ERROR IN RING OR CARRIER
8747	015100	140277	164754	6\$:	BICB	R2,@HDZTCR	:CLEAR DTR
8748	015104	104414			DELAY		:CABLE DELAY
8749	015106	011004			MOV	(R0),R4	:READ MSR
8750	015110	001402			BEQ	7\$:BR IF THEY CLEARED
8751	015112	005005			CLR	R5	:SET EXPECTED TO 0
8752	015114	104002			ERROR	2	:*BITS NOT CLEARED
8753	015116	104401		7\$:	SCOPE		:LOCK ON SIGNAL?
8754	015120	000741			BR	2\$:CONTINUE TEST

```

8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8767

```

***** TEST 17 *****
: *TEST TO VERIFY THAT IF IN 'EXTERNAL'
: *MODE; SETTING DTR FOR SELECTED LINES
: *WILL BRING UP 'CARRIER' AND 'RING'
: *FOR THAT SAME LINE. NOTE: IF YOU HAVE
: *SELECTED MODE AS 'EXTERNAL'; THE H325 TEST CONNECTER
: *MUST BE USED ON ALL SPECIFIED LINES.
: *LINES MAY BE SPECIFIED BY SWR03=1
: *AND SWR00=1 AT START TIME OR ALTERING
: *STATUS MAP.

```

8767
(5)
(4) 015122 000004
(2) 015124 012737 000017 001122
(2) 015132 012737 015260 001360
(1) 015140 012737 015174 001362
8768 015146 105737 001370
8769 015152 100401
8770 015154 104400
8771 015156 105737 001414
8772 015162 100774
8773 015164 013700 002062
8774 015170 012702 000001
8775 015174 130237 001364
8776 015200 001003
8777 015202 106302
8778 015204 103373
8779 015206 104400
8780 015210 005005
8781 015212 150205
8782 015214 000305
8783 015216 150205
8784 015220 150277 164634
8785 015224 104414
8786 015226 011004
8787 015230 020504
8788 015232 001401
8789 015234 104002

```

::: TEST 17
:*****
TST17: SCOPE
MOV #17,\$TSTNM :LOAD THE NUMBER OF THIS TEST
MOV #TST20,NEXT :POINT TO THE START OF THE NEXT TEST
MOV #3\$,LOCK :USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
TSTB MODE :EXTERNAL?
BMI 2\$:BR IF YES
ADVANCE :EXIT TEST
TSTB EIAFLG :YOU BETTER BE IN
BMI 1\$:EIA MODE FOR THIS TEST.
MOV DZMSR,R0 :SET REGISTER
MOV #1,R2 :SET LINE POINTER
BITB R2,LINE :LINE SELECTED?
BNE 5\$:BR IF YES
ASLB R2 :NEXT LINE
BCC 3\$:CONTINUE TEST
ADVANCE :ADVANCE THIS TEST
CLR R5 :SET EXPECTED
BISB R2,R5
SWAB R5
BISB R2,R5
BISB R2,@HDZTCR :SET DTR
DELAY :CABLE DELAY
MOV (R0),R4 :READ MSR
CMP R5,R4 :BITS OK?
BEQ 6\$:BR IF YES
ERROR 2 :CARRIER OR RING ERROR

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-54
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0074

8790	015236	140277	164616
8791	015242	104414	
8792	015244	011004	
8793	015246	001402	
8794	015250	005005	
8795	015252	104002	
8796	015254	104401	
8797	015256	000751	
8798			
8799			

```

6$: BICB R2,@HDZTCR :CLEAR DTR
    DELAY :CABLE DELAY
    MOV (R0),R4 :READ MSR
    BEQ 7$ :BR IF BITS CLEARED
    CLR R5 :CLEAR EXPECTED LOC.
    ERROR 2 :BITS NOT CLEARED.
7$: SCOP1 :LOCK ON LINE?
    BR 4$ :CONTINUE TEST

```

```

***** TEST 20 *****
* THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
* IS READY TO BE LOADED, AND THAT THE LINE SPECI-
* FIED IN BITS 8-10 OF DZCSR CORRESPOND
* TO THE LINE SELECTED IN DZTCR

```

::* TEST 20

(1)				
(1)				
(1)				
(1)				
(3)				
(6)				
(5)	015260	000004		
(3)	015262	012737	000020	001122
(3)	015270	012737	015404	001360
(1)	015276	104413		
(1)	015300	013700	002042	
(1)	015304	012705	100040	
(1)	015310	005037	001372	
(1)	015314	012702	000001	
(1)	015320	130237	001364	
(1)	015324	001420		
(1)	015326	050277	164524	
(1)	015332	052710	000040	
(1)	015336	005004		
(1)	015340	032710	100000	
(1)	015344	001004		
(1)	015346	104414		
(1)	015350	005204		
(1)	015352	001372		
(1)	015354	104003		
(1)	015356	011004		
(1)	015360	020405		
(1)	015362	001401		
(1)	015364	104002		
(1)	015366	062705	000400	
(1)	015372	104413		
(1)	015374	005237	001372	
(1)	015400	106302		
(1)	015402	103346		
(1)	015404			

```

TST20: SCPE
    MOV #20,$TSTNM :LOAD THE NUMBER OF THIS TEST
    MOV #TST21,NEXT :POINT TO THE START OF THE NEXT TEST
    DEVICE.CLR :ISSUE A 'DEVICE CLEAR' (RESET)
    MOV DZCSR,R0 :SET POINTER
    MOV #MSENAB!TRDY,R5 :START THE EXPECTED LINE NUMBER AT 0
    CLR SAVLIN :SET UP FOR ERROR PRINTOUTS
    MOV #1,R2 :USING R2 AS A BIT POINTER, POINT TO LINE 0
1$: BITB R2,LINE :IS THIS LINE SELECTED?
    BEQ 5$ :IF NO, SKIP THE STARTUP
2$: BIS R2,@DZTCR :SET THE GO BIT FOR THIS LINE
    BIS #MSENAB,(R0) :START THE SCANNER
    CLR R4 :SET FOR DELAY
3$: BIT #TRDY,(R0) :TX READY?
    BNE 4$ :BR IF YES
    DELAY :DELAY
    INC R4 :COUNTER
    BNE 3$ :BR IF <>0!
    ERROR 3 :*TX NOT READY!
4$: MOV (R0),R4 :GET THE LINE POINTED TO BY THE SCANNER
    CMP R4,R5 :IS THE LINE NUMBER WHAT IT SHOULD BE?
    BEQ 5$ :IF YES,GO WORK ON THE NEXT LINE
    ERROR 2 :*LINE NUMBER DID NOT MATCH TCR BIT
5$: ADD #400,R5 :POINT TO THE NEXT EXPECTED LINE
    DEVICE.CLR :ISSUE A 'DEVICE CLEAR' (RESET)
    INC SAVLIN :ADJUST FOR NEXT LINE
    ASLB R2 :POINT TO THE NEXT LINE.ARE ALL LINES TESTED?
    BCC 1$ :IF NOT, GO DO THE NEXT LINE
6$:

```

```

***** TEST 21 *****
*TEST TO TRANSMIT ONE CHAR AND
*RECEIVE ONE CHAR ON ONE LINE
*AT A TIME. THE CHAR IS '252' AND
*ALL SELECTED LINES WILL BE TURNED ON
*ONE AT A TIME. THIS IS THE FIRST TIME ANY
*DATA IS CHECKED IN THE RECEIVER.
*USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
*WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.

```

::* TEST 21

8800
8801
8802
8803
8804
8805
8806
8807
8808
8810
(5)

```

(4) 015404 000004          TST21: SCOPE
(2) 015406 012737 000021 001122  MOV      #21,$STSTM      ;LOAD THE NUMBER OF THIS TEST
(2) 015414 012737 015742 001360  MOV      #TST22,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 015422 012737 015720 001362  MOV      #16$,LOCK      ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8811 015430 104417          DCLASM
(1) 015432 013701 001366          MOV      PAR,R1         ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 015436 012702 000001          MOV      #1,R2         ;PICK UP PARAMETERS
(1) 015442 030237 001364          1$:  BIT      R2,LINE      ;PICK UP INIT POINTER
(1) 015446 001402          BEQ      2$            ;SHOULD THIS LINE BE SET UP ?
(1) 015450 010177 164376          MOV      R1,@DZLPR     ;NO
(1) 015454 005201          2$:  INC      R1         ;SET UP LINE PARAMETERS
(1) 015456 106302          ASLB     R2            ;POSITION POINTER TO THE NEXT LINE
(1) 015460 103370          BCC      1$           ;GOT 'EM ALL ?
(1) 015462 005037 001372          CLR      SAVLIN        ;IF NO, GO SET UP THE NEXT LINE
8812 015466 012702 000001          MOV      #1,R2         ;CLEAR LINE # INDICATOR
8813 015472 052777 000040 164342  BIS      #MSENAB,@DZCSR ;LINE POINTER
8814 015500 030237 001364          3$:  BIT      R2,LINE      ;START SCANNER
8815 015504 001462          BEQ      14$          ;VALID LINE ?
8816 015506 010277 164344          MOV      R2,@DZTCR    ;NO SET UP NEXT LINE
8817 015512 032777 000200 164322  4$:  BIT      #RDONE,@DZCSR ;SET TCR BIT
8818 015520 001401          BEQ      5$           ;IS REC DONE = 0 ?
8819 015522 104020          ERROR   20           ;IF YES, ALLOW TIME FOR TRDY TO SET
8820 015524 005005          CLR      R5           ;*REC DONE SHOULD = 0
8821 015526 032777 100000 164306  5$:  BIT      #TRDY,@DZCSR
8822 015534 001004          BNE      7$           ;*TRDY FAILED TO SET!
8823 015536 104414          DELAY
8824 015540 105205          INCB     R5           ;LOAD CHARACTER
8825 015542 001371          BNE      6$           ;MAKE EXPECTED LINE #
8826 015544 104003          ERROR   3            ;IS THIS TEST IN STAGGERED MODE?
8827 015546 112777 000252 164312  7$:  MOVB     #252,@DZTDR   ;IF NOT, SKIP STAGGERED SETUP
8828 015554 013705 001372          MOV      SAVLIN,R5
8829 015560 105737 001371          TSTB     MODE+1
(1) 015564 001406          BEQ      10$
(1)
(1)
(1)
(1) 015566 006205          ASR      R5           ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1) 015570 103402          BCS      8$           ;GET THE LAST BIT INTO THE CARRY BIT
(1) 015572 000261          SEC
(1) 015574 000401          BR       9$           ;IF IT IS SET, GO CLEAR IT
(1) 015576 000241          8$:  CLC
(1) 015600 006105          9$:  ROL      R5         ;IF IT IS CLEAR SET IT HERE
8830 015602 000305          10$: SWAB     R5         ;SKIP THE CLEARING
8831 015604 152705 000252          BISB     #252,R5      ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
8832 015610 052705 100000          BIS      #DVALID,R5  ;GET THE NEW BIT BACK INTO R5
8833 015614 005003          CLR      R3           ;MOVE THE LINE NUMBER TO THE UPPER BYTE
8834 015616 032777 000200 164216  11$: BIT      #RDONE,@DZCSR ;ADD CHARACTER
8835 015624 001004          BNE      12$          ;ADD DATA VALID
8836 015626 104414          DELAY
8837 015630 005203          INC      R3           ;*RDONE FAILED TO SET!
8838 015632 001371          BNE      11$
8839 015634 104004          ERROR   4            ;LOAD THE VALUE ACTUALLY RECEIVED
8840 015636 017704 164204          12$: MOV      @DZRBUF,R4  ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
8841 015642 020405          CMP      R4,R5
8842 015644 001401          BEQ      13$
8843 015646 104006          ERROR   6            ;IF YES, GO DO THE NEXT LINE
; *NO DATA/CONTENTS DID NOT COMPARE

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-56
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0076

```

8844 015650 104401          13$: SCOP1          ;CHECK TO SEE IF SWITCH NINE IS SET
8845 015652 040277 164200 14$: BIC R2,@DZTCR ;CLEAR TCR BIT FOR THAT LINE.
8846 015656 005237 001372 15$: INC SAVLIN ;INC EXPECTED LINE
8847 015662 013700 001372 MOV SAVLIN,R0 ;SET UP CHARACTER OFFSET
8848 015666 006300 ASL R0 ;MAKE THE OFFSET A POWER OF TWO
8849 015670 106302 ASLB R2 ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
8850 015672 103302 BCC 3$ ;IF NO, GO AROUND AGAIN FOR NEXT LINE
8851 015674 005003 CLR R3 ;THIS CODE HAS BEEN INSERTED
8852 015676 104414 17$: DELAY ;TO DETECT A PROBLEM FOUND IN FAULT
8853 015700 105203 INCB R3 ;INSERTION. IF AN ERROR OCCURS MORE
8854 015702 001375 BNE 17$ ;THAN ONE WORD WAS RECIEVED ON
8855 015704 032777 000200 164130 BIT #RDONE,@DZCSR ;LINE 7.
8856 015712 001401 BEQ 18$
8857 015714 104020 ERROR 20
8858 015716 104400 18$: ADVANCE ;GO TO NEXT TEST
8859
8860 ;TIGHT SCOPE LOOP FOR THIS TEST. LOOP TRANSMITS CHARACTERS ONLY
8861
8862 015720 032777 100000 164114 16$: BIT #TRDY,@DZCSR ;IS TRANSMITTER READY?
8863 015726 001774 BEQ 16$ ;IF NOT, WAIT FOR IT
8864 015730 112777 000252 164130 MOVB #252,@DZTDR ;LOAD THE CHARACTER
8865 015736 104401 SCOP1 ;LOOP AGIN IF SW09=1
8866 015740 000744 BR 14$ ;OTHERWISE, GO PICK UP THE TEST NORMALLY
8867
8868 ;***** TEST 22 *****
8869 ;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
8870 ;*CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
8871 ;*(ONE LINE AT A TIME BASED UPON VALID LINES)
8872 ;*THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
8874 ;::* TEST 22
(5) ;*****
(4) 015742 000004 TST22: SCOPE
(2) 015744 012737 000022 001122 MOV #22,$STNM ;LOAD THE NUMBER OF THIS TEST
(2) 015752 012737 016270 001360 MOV #TST23,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 015760 012737 016074 001362 MOV #4$,LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8875 01576 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 0157 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
(1) 0157 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
(1) 016000 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
(1) 016004 001402 BEQ 2$ ;NO
(1) 016006 010177 164040 MOV R1,@DZLPR ;SET UP LINE PARAMETERS
(1) 016012 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
(1) 016014 106302 ASLB R2 ;GOT 'EM ALL ?
(1) 016016 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
(1) 016020 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
8876 016024 012700 001422 MOV #TDO,R0 ;POINT TO THE DATA AREA
8877 016030 005020 CLR (R0)+ ;CLEAR A DATA WORD
8878 016032 022700 001462 CMP #STOP,R0 ;FINISHED ?
8879 016036 001374 BNE .-6 ;NO
8880 016040 005000 CLR R0 ;CLEAR OFFSET
8881 016042 013737 002046 001400 MOV DZRBUF,REGIST ;SAVE FOR ERROR MSG
8882 016050 012702 000001 MOV #1,R2 ;LINE POINTER
8883 016054 052777 000040 163760 BIS #MSENAB,@DZCSR ;START SCANNER
8884 016062 030237 001364 3$: BIT R2,LINE ;VALID LINE ?
8885 016066 001465 BEQ 14$ ;NO SET UP NEXT LINE
8886 016070 010277 163762 MOV R2,@DZTCR ;SET TCR BIT

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-57
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0077

```

8887 016074 032777 000200 163740 4$: BIT #RDONE,@DZCSR ;IS REC DONE = 0 ?
8888 016102 001401 BEQ 5$ ;IF YES, ALLOW TIME FOR TRDY TO SET
8889 016104 104020 ERROR 20 ;*REC DONE SHOULD = 0
8890 016106 005005 5$: CLR R5
8891 016110 032777 100000 163724 6$: BIT #TRDY,@DZCSR
8892 016116 001004 BNE 7$
8893 016120 104414 DELAY
8894 016122 105205 INCB R5
8895 016124 001371 BNE 6$
8896 016126 104003 ERROR 3 ;*TRDY FAILED TO SET!
8897 016130 116077 001422 163730 7$: MOVB TDO(R0),@DZTDR ;LOAD CHARACTER
8898 016136 013705 001372 MOV SAVLIN,R5 ;MAKE EXPECTED LINE #
8899 016142 105737 001371 TSTB MODE+1 ;IS THIS TEST IN STAGGERED MODE?
(1) 016146 001406 BEQ 10$ ;IF NOT, SKIP STAGGERED SETUP
(1)
(1) ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1)
(1) 016150 006205 ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
(1) 016152 103402 BCS 8$ ;IF IT IS SET, GO CLEAR IT
(1) 016154 000261 SEC ;IF IT IS CLEAR SET IT HERE
(1) 016156 000401 BR 9$ ;SKIP THE CLEARING
(1) 016160 000241 8$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 016162 006105 9$: ROL R5 ;GET THE NEW BIT BACK INTO R5
8900 016164 000305 10$: SWAB R5 ;MOVE THE LINE NUMBER TO THE UPPER BYTE
8901 016166 156005 001422 BISB TDO(R0),R5 ;ADD CHARACTER
8902 016172 052705 100000 BIS #DVALID,R5 ;ADD DATA VALID
8903 016176 005003 CLR R3
8904 016200 032777 000200 163634 11$: BIT #RDONE,@DZCSR
8905 016206 001004 BNE 12$
8906 016210 104414 DELAY
8907 016212 005203 INC R3
8908 016214 001371 BNE 11$
8909 016216 104004 ERROR 4 ;*RDONE FAILED TO SET!
8910 016220 017704 163622 12$: MOV @DZRBUF,R4 ;LOAD THE VALUE ACTUALLY RECEIVED
8911 016224 020405 CMP R4,R5 ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
8912 016226 001401 BEQ 13$ ;IF YES, GO DO THE NEXT LINE
8913 016230 104006 ERROR 6 ;*NO DATA/CONTENTS DID NOT COMPARE
8914 016232 104401 13$: SCOP1 ;CHECK TO SEE IF SWITCH NINE IS SET
8915 016234 105260 001422 INCB TDO(R0) ;INCREMENT BINARY PATTERN FOR THIS LINE
8916 016240 001315 BNE 4$ ;GO 'ROUND AGAIN FOR NEXT CHARACTER
8917 016242 040277 163610 14$: BIC R2,@DZTCR ;CLEAR TCR BIT FOR THAT LINE.
8918 016246 005237 001372 15$: INC SAVLIN ;INC EXPECTED LINE
8919 016252 013700 001372 MOV SAVLIN,R0 ;SET UP CHARACTER OFFSET
8920 016256 006300 ASL R0 ;MAKE THE OFFSET A POWER OF TWO
8921 016260 106302 ASLB R2 ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
8922 016262 103277 BCC 3$ ;IF NO, GO AROUND AGAIN FOR NEXT LINE
8923 016264 005037 001362 CLR LOCK ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST

```

```

8924
8925
8926 ;***** TEST 23 *****
8927 ;*THIS TEST WILL PROVE THAT EACH RECEIVING LINE CAN
8928 ;*BE DISABLED BY SETTING THE RCVON BIT TO ZERO
8929 ;*FOR EACH LINE IN THE LPR REGISTER. IT ALSO
8930 ;*VERIFIES THAT MASTER CLEAR WILL ZERO DVALID FOR
8931 ;*CHARACTERS STORED IN THE SILO.
8933 ;* TEST 23

```

```

(5)
(4) 016270 000004
(2) 016272 012737 000023 001122
(2) 016300 012737 016622 001360
8934 016306 105037 001420
8935 016312 005037 001372
8936 016316 104417
8937 016320 013701 001366
8938 016324 042701 010000
8939 016330 012702 000001
8940 016334 010177 163512
8941 016340 005201
8942 016342 106302
8943 016344 103373
8944 016346 012701 000252
8945 016352 013702 001364
8946 016356 010277 163474
8947 016362 052777 000040 163452
8948 016370 005005
8949 016372 005777 163444
8950 016376 100404
8951 016400 104414
8952 016402 005205
8953 016404 001372
8954 016406 104003
8955 016410 117705 163430
8956 016414 012703 000001
8957 016420 042705 177770
8958 016424 001403
8959 016426 106303
8960 016430 005305
8961 016432 001375
8962 016434 030302
8963 016436 001007
8964 016440 140377 163412
8965 016444 001351
8966 016446 105737 001420
8967 016452 001040
8968 016454 000404
8969 016456 110177 163404
8970 016462 040302
8971 016464 000741
8972 016466 005077 163364
8973 016472 005005
8974 016474 104414
8975 016476 005205
8976 016500 001375
8977 016502 105777 163334
8978 016506 100003
8979 016510 005037 001372
8980 016514 104020
8981 016516 017704 163324
8982 016522 100007
8983 016524 000304
8984 016526 042704 177770
8985 016532 010437 001372

```

```

TST23: SCOPE
MOV #23,$STSTM ;LOAD THE NUMBER OF THIS TEST
MOV #TST24,NEXT ;POINT TO THE START OF THE NEXT TEST
CLRB DONFLG ;INITIALIZE FOR FIRST TEST LOOP
CLR SAVLIN ;ZERO LINE NO. FOR ERROR REPORT
DCLASM ;EXECUTE MASTER CLEAR
MOV PAR,R1 ;STORE DEFAULT PARAMETERS
BIC #RCVON,R1 ;CLEAR RCVON BIT
1$: MOV #1,R2 ;INIT LINE POINTER
2$: MOV R1,@DZLPR ;LOAD LINE PARAMETER REGISTER
INC R1 ;SET R1 FOR NEXT LINE
ASLB R2 ;SHIFT R2 TO NEXT LINE
BCC 2$ ;ALL LINES LOADED?
MOV #252,R1 ;LOAD TRANSMITTING CHARACTER
MOV LINE,R2 ;COPY ACTIVE LINE BITS
MOV R2,@DZTCR ;LOAD TCR BITS
BIS #MSENAB,@DZCSR ;SET SCANNER
3$: CLR R5 ;INIT DELAY COUNTER
4$: TST @DZCSR ;TRDY SET?
BMI 5$ ;IF YES BRANCH
DELAY ;IF NOT THEN WAIT
INC R5 ;INCREMENT DELAY COUNTER
BNE 4$ ;DELAY DONE?
5$: MOV @HDZCSR,R5 ;MOVE LINE NO. INTO R5
MOV #1,R3 ;INIT TCR POINTER
BIC #^C<7>,R5 ;ISOLATE LINE NO.
BEQ 21$ ;IF LINE 0 GO TEST TRANSM. FLAG
20$: ASLB R3 ;POINT R3 TO NEXT TCR BIT
DEC R5 ;DECREMENT R5 UNTIL R3 POINTS
BNE 20$ ;TO CORRECT TCR BIT
21$: BIT R3,R2 ;HAS THIS LINE BEEN SERVICED?
BNE 6$ ;IF NOT GO SEND CHARACTER
BICB R3,@DZTCR ;IF YES CLEAR TCR BIT
BNE 3$ ;IF MORE LINES SET BRANCH
TSTB DONFLG ;IF ALL LOADED IS THIS SECOND PASS
BNE 12$ ;IF YES BRANCH TO SECOND PART OF TEST
BR 7$ ;OTHERWISE CONTINUE WITH FIRST PART
6$: MOV R1,@DZTDR ;TRANSMIT CHARACTER
BIC R3,R2 ;CLEAR FLAG FOR THIS LINE
BR 3$ ;GO WAIT FOR NEXT LINE
7$: CLR @DZTCR ;CLEAR TCR BITS
CLR R5 ;CLEAR DELAY COUNTER
8$: DELAY ;WAIT FOR LAST CHARACTER
INC R5 ;INCREMENT DELAY COUNTER
BNE 8$ ;IF NOT FINISHED CONTINUE WAITING
TSTB @DZCSR ;RDONE BIT SET?
BPL 10$ ;IF NO CONTINUE
CLR SAVLIN ;IF YES SET LINE NO. TO ZERO
ERROR 20 ;AND PRINT ERROR
10$: MOV @DZRBUF,R4 ;READ SILO
BPL 11$ ;IF DVALID IS ZERO BRANCH
SWAB R4 ;IF SET THEN
BIC #^C<7>,R4 ;ISOLATE LINE NO. IN R4
MOV R4,SAVLIN ;SET SAVLIN FOR ERROR REPORT

```

```

8986 016536 104017          ERROR 17          ;DATA VALID SHOULD NOT BE SET
8987 016540 000766          BR 10$           ;GO READ SILO AGAIN
8988 016542 105237 001420 11$: INCB DONFLG    ;PREPARE FOR SECOND PART OF TEST
8989 016546 013701 001366    MOV PAR,R1       ;MOVE DEFAULT PARAMETERS TO R1
8990 016552 000666          BR 1$            ;GO LOAD LPR REGISTER
8991 016554 005005          CLR R5           ;INIT DELAY COUNTER
8992 016556 104414          13$: DELAY        ;WAIT FOR LAST CHARACTER
8993 016560 005205          INC R5           ;TO BE RECEIVED
8994 016562 001375          BNE 13$         ;DELAY FINISHED?
8995 016564 104413          DEVICE.CLR     ;IF YES EXECUTE MASTER CLEAR
8996 016566 000240          NOP
8997 016570 000240          NOP
8998 016572 105777 163244    TSTB @DZCSR     ;RDONE SET?
8999 016576 100003          BPL 14$         ;IF NOT BRANCH
9000 016600 005037 001372    CLR SAVLIN      ;IF YES THEN PRINT OUT
9001 016604 104020          ERROR 20        ;REPORT
9002 016606 017704 163234 14$: MOV @DZRBUF,R4 ;READ SILO
9003 016612 100003          BPL 15$         ;DATA VALID SET?
9004 016614 005037 001372    CLR SAVLIN      ;IF YES THEN PRINT OUT
9005 016620 104017          ERROR 17        ;ERROR REPORT
9006 016622
9007
9008
9009
9010
9011
9012
9013
9014
9015
9017
(5)
(4) 016622 000004
(2) 016624 012737 000024 0C1122
(2) 016632 012737 017100 001360
9018 016640 012737 016736 001362
9019 016646 005737 001370
9020 016652 001510
9021 016654 104417
9022 016656 013701 001366
9023 016662 052701 000300
9024 016666 012700 000001
9025 016672 030037 001364 1$: BIT R0,LINE ;SHOULD THIS LINE BE SET UP ?
9026 016676 001402          BEQ 2$          ;IF NOT,DON'T SET IT UP
9027 016700 010177 163146    MOV R1,@DZLPR   ;OTHERWISE, SET UP LINE PARAMETERS
9028 016704 005201          2$: INC R1
9029 016706 106300          ASLB R0         ;GOT 'EM ALL ?
9030 016710 103370          BCC 1$         ;NO
9031 016712 005037 001372    CLR SAVLIN      ;CLEAR LINE #
9032 016716 012702 000001          MOV #1,R2      ;LINE POINTER
9033 016722 052777 000040 163112          BIS #MSENAB,@DZCSR ;SET MASTER SCAN ENABLE
9034 016730 013737 002046 001400          MOV DZRBUF,REGIST ;SAVE FOR ERRR MESSAGE
9035 016736 030237 001364 3$: BIT R2,LINE
9036 016742 001446          BEQ 11$
9037 016744 010277 163106    MOV R2,@DZTCR  ;SET TCR BIT
9038 016750 110277 163114    MOV R2,@HDZTDR ;SET BREAK BIT
;***** TEST 24 *****
;THIS TEST WILL PROVE THAT:
;* 1) THE TRANSMITTER 'BREAK BIT' WORKS
;* 2) THE RECEIVER CAN FLAG 'FRAMING ERRORS'
;* 3) THE RECEIVER CAN FLAG 'PARITY ERRORS'
;*ONLY ONE LINE AT A TIME WILL BE EXERCISED.
;*THIS TEST WILL NOT BE EXERCISED UNLESS
;*CONNECTED BY AN H325, H3271, OR H3190 CONNECTOR
;:* TEST 24
;*****
TST24: SCOPE
MOV #24,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST25,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #3$,LOCK    ;SET FOR LOOP
TST MODE        ;ARE WE RUNNING IN INTERNAL MODE?
BEQ 12$         ;IF SO, SKIP THIS TEST
DCLASM         ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV PAR,R1     ;PICK UP PARAMETERS
BIS #ODDPAR!PARITY,R1 ;FORCE ODD PARITY
MOV #1,R0      ;PICK UP INIT POINTER
BIT R0,LINE    ;SHOULD THIS LINE BE SET UP ?
BEQ 2$         ;IF NOT,DON'T SET IT UP
MOV R1,@DZLPR ;OTHERWISE, SET UP LINE PARAMETERS
2$: INC R1
ASLB R0        ;GOT 'EM ALL ?
BCC 1$         ;NO
CLR SAVLIN     ;CLEAR LINE #
MOV #1,R2     ;LINE POINTER
BIS #MSENAB,@DZCSR ;SET MASTER SCAN ENABLE
MOV DZRBUF,REGIST ;SAVE FOR ERRR MESSAGE
3$: BIT R2,LINE
BEQ 11$
MOV R2,@DZTCR ;SET TCR BIT
MOVB R2,@HDZTDR ;SET BREAK BIT

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-60
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0080

```

9039 016754 112777 000377 163104 4$:  MOVB  #377,@DZTDR ;LOAD CHARACTER
9040 016762 013705 001372          MOV   SAVLIN,R5 ;MAKE EXPECTED DATA
9041 016766 105737 001371          TSTB  MODE+1 ;IS THIS TEST IN STAGGERED MODE?
(1) 016772 001406          BEQ   7$ ;IF NOT, SKIP STAGGERED SETUP
(1)
(1)
(1) ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1) 016774 006205          ASR   R5 ;GET THE LAST BIT INTO THE CARRY BIT
(1) 016776 103402          BCS  5$ ;IF IT IS SET, GO CLEAR IT
(1) 017000 000261          SEC ;IF IT IS CLEAR SET IT HERE
(1) 017002 000401          BR   6$ ;SKIP THE CLEARING
(1) 017004 000241          5$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 017006 006105          6$: ROL R5 ;GET THE NEW BIT BACK INTO R5
9042 017010 000305          7$: SWAB R5 ;PUT LINE NUMBER IN UPPER BYTE
9043 017012 052705 130000          BIS  #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
9044 017016 005004          CLR  R4
9045 017020 032777 000200 163014 8$:  BIT  #RDONE,@DZCSR
9046 017026 001004          BNE  9$
9047 017030 104414          DELAY
9048 017032 005204          INC  R4
9049 017034 001371          BNE  8$
9050 017036 104004          ERROR 4 ;*RDONE FAILED TO SET!
9051 017040 017704 163002          9$: MOV  @DZRBUF,R4 ;ACTUAL
9052 017044 020405          CMP  R4,R5 ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
9053 017046 001401          BEQ  10$ ;IF YES, GO CLEAN UP
9054 017050 104006          ERROR 6 ;*DATA/CONTENTS FAILED TO COMPARE
9055 017052 105077 163012          10$: CLRB @HDZTDR ;CLEAR BREAK BITS
9056 017056 104401          SCOP1 ;LOOP?
9057 017060 005237 001372          11$: INC  SAVLIN ;INC LINE #
9058 017064 040277 162766          BIC  R2,@DZTCR ;CLEAR TCR BIT
9059 017070 106302          ASLB R2
9060 017072 103321          BCC  3$
9061 017074 005037 001362          12$: CLR  LOCK ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST
9062
(1) ;***** TEST 25 *****
(1) ;* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
(1) ;*WHILE THE PROCESSOR STATUS IS SET EXACTLY
(1) ;*TO WHAT THE DZ11 PRIORITY IS SET TO.
(3) ;*DEFAULT PRIORITY IS AT 5 (240).
(6) ;** TEST 25
(5) ;*****
(3) 017100 000004          TST25: SCOPE
(3) 017102 012737 000025 001122          MOV  #25,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3) 017110 012737 017410 001360          MOV  #TST26,NEXT ;POINT TO THE START OF THE NEXT TEST
(3) 017116 104417          DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2) 017120 013701 001366          MOV  PAR,R1 ;PICK UP PARAMETERS
(2) 017124 012702 000001          MOV  #1,R2 ;PICK UP INIT POINTER
(2) 017130 030237 001364          1$: BIT  R2,LINE ;SHOULD THIS LINE BE SET UP ?
(2) 017134 001402          BEQ  2$ ;NO
(2) 017136 010177 162710          MOV  R1,@DZLPR ;SET UP LINE PARAMETERS
(2) 017142 005201          2$: INC  R1 ;POSITION POINTER TO THE NEXT LINE
(2) 017144 106302          ASLB R2 ;GOT 'EM ALL ?
(2) 017146 103370          BCC  1$ ;IF NO, GO SET UP THE NEXT LINE
(2) 017150 005037 001372          CLR  SAVLIN ;CLEAR LINE # INDICATOR
(1) 017154 106437 027532          MTPS @#DZPRT ;SET CPU STATUS TO DZ11 PRIO,
(1) 017160 113777 001364 162670          MOV  LINE,@DZTCR ;ENABLE THE VALID LINES
(1) 017166

```

```

(2) 017166 012777 017256 162702      MOV      #6$,@DZTIV      ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017174 012777 017264 162670      MOV      #7$,@DZRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017202 013777 027532 162664      MOV      DZPRT,@DZ RIS   ;SET THE INTERRUPT VECTOR STATUS
(2) 017210 013777 027532 162662      MOV      DZPRT,@DZTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017216 052777 040040 162616      BIS      #TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(1) 017224 005005                CLR      R5
(1) 017226 032777 100000 162606 4$:  BIT      #TRDY,@DZCSR
(1) 017234 001403                BEQ      5$
(1) 017236 000240                NOP
(1) 017240 000240                NOP
(1) 017242 000412                BR       8$
(1) 017244 104414                5$:  DELAY
(1) 017246 005205                INC      R5
(1) 017250 001366                BNE     4$
(1) 017252 104003                ERROR   3                ;*TRDY NOT SET!
(1) 017254 000405                BR       8$
(1) 017256 104010                6$:  ERROR   10                ;*TRANSMITTER SHOULD NOT INTERRUPT
(1) 017260 022626                CMP     (SP)+,(SP)+      ;POP FOR FAKE RTI
(1) 017262 000402                BR       8$                ;CONTINUE TEST
(1) 017264 104012                7$:  ERROR   12                ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017266 022626                CMP     (SP)+,(SP)+      ;POP FOR FAKE RTI
(1) 017270 042777 040000 162544 8$:  BIC      #TIE,@DZCSR      ;RESET TRANSMITTER INTERRUPT ENABLE
(2) 017276 012777 017374 162572      MOV      #11$,@DZTIV     ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017304 012777 017402 162560      MOV      #12$,@DZRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017312 013777 027532 162554      MOV      DZPRT,@DZ RIS   ;SET THE INTERRUPT VECTOR STATUS
(2) 017320 013777 027532 162552      MOV      DZPRT,@DZTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017326 052777 000140 162506      BIS      #RIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(1) 017334 113777 001422 162524      MOVB    TD0,@DZTDR       ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
(1) 017342 005005                CLR      R5
(1) 017344 032777 000200 162470 9$:  BIT      #RDONE,@DZCSR
(1) 017352 001403                BEQ     10$
(1) 017354 000240                NOP
(1) 017356 000240                NOP
(1) 017360 000412                BR      13$
(1) 017362 104414                10$: DELAY
(1) 017364 005205                INC      R5
(1) 017366 001366                BNE     9$
(1) 017370 104004                ERROR   4                ;*NO RX DONE! (NOT SET)
(1) 017372 000405                BR      13$                ;CONTINUE TEST
(1) 017374 104010                11$: ERROR   10                ;*TRANSMITTER SHOULD NOT INTERRUPT
(1) 017376 022626                CMP     (SP)+,(SP)+      ;POP FOR FAKE RTI
(1) 017400 000402                BR      13$                ;CONT TEST
(1) 017402 104012                12$: ERROR   12                ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017404 022626                CMP     (SP)+,(SP)+      ;POP FOR FAKE RTI
(1) 017406                13$:
(2) 017406 104413                DEVICE.CLR                ;ISSUE DEVICE CLEAR (RESET)
9063
(1)                ;***** TEST 26 *****
(1)                ;* THIS TEST VERIFIES THAT THE DEVICE DOES INTERRUPT
(1)                ;*WHILE THE PROCESSOR STATUS IS SET TO EXACTLY
(1)                ;*ONE LEVEL LOWER THAN THE DZ11. DZ11 PRIORITY
(1)                ;*DEFAULT TO LEVEL 5 MINUS ONE LEVEL IS LEVEL 4.
(3)                ;:* TEST 26
(6)                ;*****
(5) 017410 000004                TST26: SCOPE
(3) 017412 012737 000026 001122      MOV      #26,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3) 017420 012737 017736 001360      MOV      #TST27,NEXT     ;POINT TO THE START OF THE NEXT TEST

```

```

(3) 017426 104417          DCLASM          ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2) 017430 013701 001366  MOV PAR,R1      ;PICK UP PARAMETERS
(2) 017434 012702 000001  MOV #1,R2      ;PICK UP INIT POINTER
(2) 017440 030237 001364  1$: BIT R2,LINE  ;SHOULD THIS LINE BE SET UP ?
(2) 017444 001402          BEQ 2$         ;NO
(2) 017446 010177 162400  MOV R1,@DZLPR ;SET UP LINE PARAMETERS
(2) 017452 005201          2$: INC R1         ;POSITION POINTER TO THE NEXT LINE
(2) 017454 106302          ASLB R2        ;GOT 'EM ALL ?
(2) 017456 103370          BCC 1$        ;IF NO, GO SET UP THE NEXT LINE
(2) 017460 005037 001372  CLR SAVLIN    ;CLEAR LINE # INDICATOR
(1) 017464 106437 027534  MTPS @#LESS1  ;MAKE CPU ONE LEVEL LOWER THAN DZ11
(1) 017470 113777 001364 162360 MOVB LINE,@DZTCR ;ENABLE THE VALID LINES
(1) 017476          3$:
(2) 017476 012777 017570 162372 MOV #6$,@DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017504 012777 017606 162360 MOV #7$,@DZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017512 013777 027532 162354 MOV DZPRT,@DZ RIS ;SET THE INTERRUPT VECTOR STATUS
(2) 017520 013777 027532 162352 MOV DZPRT,@DZTIS ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017526 052777 040040 162306 BIS #TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(1) 017534 005005          CLR R5
(1) 017536 032777 100000 162276 4$: BIT #TRDY,@DZCSR
(1) 017544 001404          BEQ 5$
(1) 017546 000240          NOP
(1) 017550 000240          NOP
(1) 017552 104007          ERROR 7       ;*TRANSMITTER FAILED TO INTERRUPT
(1) 017554 000416          BR 8$
(1) 017556 104414          5$: DELAY
(1) 017560 005205          INC R5
(1) 017562 001365          BNE 4$
(1) 017564 104003          ERROR 3       ;*TRDY NOT SET!
(1) 017566 000411          BR 8$
(1) 017570 022626          6$: POP2SP
(1) 017572 042777 040000 162242 BIC #TIE,@DZCSR ;REMOVE THE INTERRUPT FROM THE STACK
(1) 017600 106437 027534          MTPS @#LESS1  ;DON'T LET ANY MORE INTERRUPTS OCCUR
(1) 017604 000402          BR 8$        ;MAKE CPU ONE LEVEL LOWER THAN DZ11
(1) 017606 104012          7$: ERROR 12   ;RETURN TO THE NORMAL FLOW
(1) 017610 022626          CMP (SP)+,(SP)+ ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017612 042777 040000 162222 8$: BIC #TIE,@DZCSR ;POP FOR FAKE RTI
(2) 017620 012777 017720 162250 MOV #11$,@DZTIV ;RESET TRANSMITTER INTERRUPT ENABLE
(2) 017626 012777 017726 162236 MOV #12$,@DZRIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017634 013777 027532 162232 MOV DZPRT,@DZ RIS ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017642 013777 027532 162230 MOV DZPRT,@DZTIS ;SET THE INTERRUPT VECTOR STATUS
(2) 017650 052777 000140 162164 BIS #RIE!MSENAB,@DZCSR ;SET TRANSMITTER INTERRUPT PRIORITY
(1) 017656 113777 001422 162202 MOVB TD0,@DZTDR ;ENABLE THE DEVICE
(1) 017664 005005          CLR R5        ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
(1) 017666 032777 000200 162146 9$: BIT #RDONE,@DZCSR
(1) 017674 001404          BEQ 10$
(1) 017676 000240          NOP
(1) 017700 000240          NOP
(1) 017702 104011          ERROR 11     ;*RECEIVER FAILED TO INTERRUPT
(1) 017704 000413          BR 13$
(1) 017706 104414          10$: DELAY
(1) 017710 005205          INC R5
(1) 017712 001365          BNE 9$
(1) 017714 104004          ERROR 4       ;*NO RX DONE! (NOT SET)
(1) 017716 000406          BR 13$      ;CONTINUE TEST
(1) 017720 104010          11$: ERROR 10  ;*TRANSMITTER SHOULD NOT INTERRUPT

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-63
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0083

```

(1) 017722 022626          CMP      (SP)+,(SP)+      ;POP FOR FAKE RTI
(1) 017724 000403          BR       13$              ;CONT TEST
(1) 017726 022626          12$:    POP2SP            ;REMOVE THE INTERRUPT FROM THE STACK
(1) 017730 005077 162106   CLR      @DZCSR           ;DON'T ALLOW ANY MORE INTERRUPTS
(1) 017734          13$:    DEVICE.CLR             ;ISSUE DEVICE CLEAR (RESET)
(2) 017734 104413
9064
9065          ;***** TEST 27 *****
9066          ;*THIS TEST VERIFIES THAT THE RECEIVER WILL
9067          ;*INTERRUPT BEFORE THE TRANSMITTER EVEN
9068          ;*THOUGH THE TRANSMITTER WAS ENABLED
9069          ;*FIRST. SET PS TO LEVEL 7;
9070          ;*GET RDONE AND TRDY TO SET;
9071          ;*SET TX IE AND RX IE;
9072          ;*CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
9074          ;:* TEST 27
(5)          ;*****
(4) 017736 000004          TST27:  SCOPE
(2) 017740 012737 000027 001122  MOV      #27,$STNM        ;LOAD THE NUMBER OF THIS TEST
(2) 017746 012737 020370 001360  MOV      #TST30,NEXT     ;POINT TO THE START OF THE NEXT TEST
9075 017754 104417          DCLASM                    ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 017756 013701 001366          MOV      PAR,R1          ;PICK UP PARAMETERS
(1) 017762 012702 000001          MOV      #1,R2           ;PICK UP INIT POINTER
(1) 017766 030237 001364          1$:    BIT      R2,LINE     ;SHOULD THIS LINE BE SET UP ?
(1) 017772 001402          BEQ      2$              ;NO
(1) 017774 010177 162052          MOV      R1,@DZLPR      ;SET UP LINE PARAMETERS
(1) 020000 005201          2$:    INC      R1          ;POSITION POINTER TO THE NEXT LINE
(1) 020002 106302          ASLB    R2              ;GOT 'EM ALL ?
(1) 020004 103370          BCC     1$              ;IF NO, GO SET UP THE NEXT LINE
(1) 020006 005037 001372          CLR     SAVLIN          ;CLEAR LINE # INDICATOR
9076 020012 012777 020242 162052  MOV      #8$,@DZRIV      ;SETUP INTERRUPT STUFF
9077 020020 013777 027532 162046  MOV      DZPRT,@DZ RIS
9078 020026 012777 020332 162042  MOV      #12$,@DZTIV
9079 020034 013777 027532 162036  MOV      DZPRT,@DZTIS
9080 020042 052777 000040 161772  BIS     #MSENAB,@DZCSR
9081 020050 012702 000001          MOV      #1,R2          ;LINE POINTER
9082 020054 030237 001364          3$:    BIT      R2,LINE     ;VALID LINE ?
9083 020060 001004          BNE     4$              ;
9084 020062 005237 001372          INC     SAVLIN
9085 020066 106302          ASLB    R2
9086 020070 000771          BR      3$
9087 020072 106427 000340          4$:    MTPS    #PR7
9088 020076 000240          NOP
9089 020100 000240          NOP
9090 020102 110277 161750          MOVB    R2,@DZTCR       ;SET TCR BIT
9091 020106 005777 161734          TST     @DZRBUF         ;VALID DATA?
9092 020112 100001          BPL     .+4             ;IT BETTER NOT BE SET
9093 020114 104017          ERROR  17              ;DATA VALID SHOULD NOT BE SET
9094 020116 105777 161720          5$:    TSTB    @DZCSR     ;RECEIVER DONE ?
9095 020122 100001          BPL     .+4
9096 020124 104020          ERROR  20              ;RECEIVER DONE BIT SHOULD NOT BE SET
9097 020126 005005          CLR     R5
9098 020130 005004          CLR     R4
9099 020132 005777 161704          99$:   TST     @DZCSR       ;WAIT FOR TRDY
9100 020136 100404          BMI     100$           ;BR IF READY
9101 020140 104414          DELAY

```

```

9102 020142 005204          INC      R4          ;
9103 020144 001372          BNE     99$          ;
9104 020146 104003          ERROR   3           ;TRDY FAILED TO SET
9105 020150 105077 161712   100$:  CLR    @DZTDR
9106 020154 005004          CLR     R4
9107 020156 032777 000200 161656 6$:  BIT    #RDONE,@DZCSR
9108 020164 001004          BNE     7$
9109 020166 104414          DELAY
9110 020170 005204          INC     R4
9111 020172 001371          BNE     6$
9112 020174 104004          ERROR   4           ;*RDONE FAILED TO SET!
9113 020176 005777 161640   7$:  TST    @DZCSR       ;TRANS DONE BIT = 1 ?
9114 020202 100401          BMI    +4          ;YES
9115 020204 104003          ERROR   3           ;*NO TRANS DONE FAILED TO SET
9116          ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
9117          ;SET INTERRUPT ENABLES AND WATCH THE FUR FLY
9118 020206 052777 040000 161626  BIS    #TIE,@DZCSR
9119 020214 052777 000100 161620  BIS    #RIE,@DZCSR
9120 020222 106427 000000  MTPS   #0
9121 020226 000240          NOP
9122 020230 000240          NOP
9123 020232 104007          ERROR   7           ;*TRANSMITTER FAILED TO INTERRUPT
9124 020234 104011          ERROR   11          ;*RECEIVER FAILED TO INTERRUPT
9125          ;CHECK BR LEVEL
9126 020236 000137 020336   JMP    13$          ;GET OUT
9127
9128          ;RECEIVER INTERRUPT ROUTINE
9129 020242 017704 161600   8$:  MOV    @DZRBUF,R4      ;ACTUAL
9130 020246 010403          MOV    R4,R3
9131 020250 000303          SWAB   R3
9132 020252 042703 177770   BIC    #^C<7>,R3     ;STRIP JUNK
9133 020256 105737 001371   TSTB   MODE+1       ;IS THIS TEST IN STAGGERED MODE?
(1) 020262 001406          BEQ    11$          ;IF NOT, SKIP STAGGERED SETUP
(1)
(1)
(1)          ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1) 020264 006203          ASR    R3           ;GET THE LAST BIT INTO THE CARRY BIT
(1) 020266 103402          BCS    9$           ;IF IT IS SET, GO CLEAR IT
(1) 020270 000261          SEC
(1) 020272 000401          BR     10$          ;IF IT IS CLEAR SET IT HERE
(1) 020274 000241          CLC     10$        ;SKIP THE CLEARING
(1) 020276 006103          ROL    R3           ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
9134 020300 020337 001372   10$:  ROL    R3           ;GET THE NEW BIT BACK INTO R3
9135 020304 001401          CMP    R3,SAVLIN    ;IS THIS A VALID LINE
9136 020306 104015          BEQ    +4          ;YES
9137 020310 042704 177400   ERROR   15          ;*INVALID LINE
9138 020314 120504          BIC    #^C<377>,R4  ;STRIP JUNK
9139 020316 001401          CMPB   R5,R4        ;DATA COMPARE ?
9140 020320 104005          BEQ    +4          ;YES
9141 020322 040277 161530   ERROR   5           ;*DATA DOES NOT COMPARE
9142 020326 022626          BIC    R2,@DZTCR   ;CLEAR TCR BIT
9143 020330 000402          POP2SP ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
9144          BR     13$          ;GO GET OUT OF INTERRUPT MODE
9145 020332 104011          ;TRANSMITTER INTERRUPT SVC ROUTINE
9146          ERROR   11          ;THE RECEIVER INTERRUPT FAILED
9147 020334 022626          ;TO OVERRIDE THE TRANSMITTER
          POP2SP        ;REMOVE THE INTERRUPT VECTOR FROM THE STACK

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 82-65
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0085

```

9148 020336 042777 040100 161476 13$: BIC #TIE!RIE,@DZCSR ;CLEAR INTERRUPT ENABLES
9149 020344 013777 002074 161520 MOV DZRIS,@DZRIV ;RESTORE TRAPCATCHER
9150 020352 005077 161516 CLR @DZRIS
9151 020356 013777 002100 161512 MOV DZTIS,@DZTIV
9152 020364 005077 161510 CLR @DZTIS
9153 :***** TEST 30 *****
9154 :*TEST TO VERIFY THAT 'RDONE DOES NOT SET
9155 :*IF THE SCANNER IS DISABLED.
9156 :*TURN ON SCANNER, WAIT FOR TRDY,
9157 :*TURN OFF SCANNER, TRANSMIT A CHARACTER
9158 :*'RDONE SHOULD NOT SET.
9160 ::* TEST 30
(5) :*****
(4) 020370 000004 TST30: SCOPE
(2) 020372 012737 000030 001122 MOV #30,$STSTNM ;LOAD THE NUMBER OF THIS TEST
(2) 020400 012737 020556 001360 MOV #TST31,NEXT ;POINT TO THE START OF THE NEXT TEST
9161 020406 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 020410 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
(1) 020414 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
(1) 020420 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
(1) 020424 001402 BEQ 2$ ;NO
(1) 020426 010177 161420 MOV R1,@DZLPR ;SET UP LINE PARAMETERS
(1) 020432 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
(1) 020434 106302 ASLB R2 ;GOT 'EM ALL ?
(1) 020436 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
(1) 020440 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
9162 020444 052777 000040 161370 BIS #MSENAB,@DZCSR ;TURN ON SCANNER
9163 020452 012702 000001 MOV #1,R2 ;INIT LINE COUNTER
9164 020456 030237 001364 3$: BIT R2,LINE ;FIND A VALID LINE
9165 020462 001004 BNE 4$ ;IF WE FOUND ONE GO TO TEST
9166 020464 005237 001372 INC SAVLIN ;IF NOT
9167 020470 106302 ASLB R2 ;KEEP LOOKING
9168 020472 000771 BR 3$
9169 020474 110277 161356 4$: MOV B R2,@DZTCR ;SET TCR BIT
9170 020500 005005 CLR R5
9171 020502 005777 161334 5$: TST @DZCSR ;IS TRDY SET
9172 020506 100404 BMI 6$ ;CON'T TESTING IF IT IS
9173 020510 104414 DELAY ;IF IT NOT WAIT A WHILE
9174 020512 005205 INC R5
9175 020514 001372 BNE 5$
9176 020516 104003 ERROR 3 ;WE WAITED LONG ENOUGH-ERROR
9177 020520 042777 000040 161314 6$: BIC #MSENAB,@DZCSR ;TURN OFF SCANNER
9178 020526 105077 161334 CLR @DZTDR ;TRANSMIT A CHARACTER
9179 020532 005005 CLR R5 ;CLEAR COUNTER
9180 020534 104414 7$: DELAY ;WAIT SUFFICIENT TIME FOR
9181 020536 005205 INC R5 ;RDONE TO SET
9182 020540 001375 BNE 7$
9183 020542 032777 000200 161272 BIT #RDONE,@DZCSR ;RDONE SET
9184 020550 001401 BEQ 8$ ;IT SHOULDN'T BE-CONTINUE
9185 020552 104020 ERROR 20 ;IF IT IS THERE'S AN ERROR
9186 020554 104400 8$: ADVANCE
9187 :***** TEST 31 *****
9188 :*THIS TEST VERIFIES OVERRUN AND SILO ALARM
9189 :*ONE LINE AT A TIME - BASED UPON VALID LINES
9190 :*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
9191 :*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN

```

```

9192
9193
9194
9195
9196
9197
9199
(5)
(4) 020556 000004
(2) 020560 012737 000031 001122
(2) 020566 012737 021304 001360
9200 020574 012737 021210 001362
9201 020602 104417
(1) 020604 013701 001366
(1) 020610 012702 000001
(1) 020614 030237 001364
(1) 020620 001402
(1) 020622 010177 161224
(1) 020626 005201
(1) 020630 106302
(1) 020632 103370
(1) 020634 005037 001372
9202 020640 012700 001422
9203 020644 005020
9204 020646 022700 001462
9205 020652 001374
9206 020654 005000
9207 020656 012702 000001
9208 020662 052777 010040 161152
9209 020670 030237 001364
9210 020674 001002
9211 020676 000137 021172
9212 020702 013700 001372
9213 020706 006300
9214 020710 010277 161142
9215 020714 105777 161122
9216 020720 100001
9217 020722 104020
9218 020724 005003
9219 020726 005004
9220 020730 032777 100000 161104
9221 020736 001004
9222 020740 104414
9223 020742 105204
9224 020744 001371
9225 020746 104003
9226 020750 116077 001422 161110
9227 020756 005260 001422
9228 020762 020327 000017
9229 020766 103006
9230 020770 032777 020000 161044
9231 020776 001401
9232 021000 104013
9233
9234 021002 000411
9235 021004 005004

```

```

: * EXPECTS SILO ALARM TO SET, THEN THE ENTIRE
: * SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
: * CHAR PULLED OUT OUT THE SILO.
: * USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
: * ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
: * USED TO SCOPE SILO ALARM PULSES, ETC.
: * TEST 31
: *****
TST31: SCOPE
MOV #31,$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST32,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #18$,LOCK ;SET FOR LOOP
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV PAR,R1 ;PICK UP PARAMETERS
MOV #1,R2 ;PICK UP INIT POINTER
1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
BEQ 2$ ;NO
MOV R1,@DZLPR ;SET UP LINE PARAMETERS
2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
ASLB R2 ;GOT 'EM ALL ?
BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
CLR SAVLIN ;CLEAR LINE # INDICATOR
MOV #TDO,R0 ;POINT TO THE DATA AREA
CLR (R0)+ ;CLEAR A DATA WORD
CMP #STOP,R0 ;FINISHED ?
BNE -6 ;NO
CLR R0 ;CLEAR OFFSET
MOV #1,R2 ;LINE POINTER
BIS #MSENAB!SILOEN,@DZCSR ;START SCANNER & SET SILO ENABLE
3$: BIT R2,LINE ;VALID LINE?
BNE +6 ;YES
JMP 22$ ;TRY NEXT LINE
MOV SAVLIN,R0 ;MAKE OFFSET
ASL R0 ;MAKE POWER OF TWO
MOV R2,@DZTCR ;SET TCR BIT
4$: TSTB @DZCSR ;REC DONE = 1 ?
BPL +4
ERROR 20 ;REC DONE SHOULD NOT = 1
CLR R3 ;SET CHARACTER COUNT
CLR R4
5$: BIT #TRDY,@DZCSR
BNE 7$
DELAY
INCB R4
BNE 6$
ERROR 3
7$: MOVB TDO(R0),@DZTDR ;*TRDY FAILED TO SET
;LOAD A CHARACTER
INC TDO(R0) ;SET UP NEXT CHARACTER
CMP R3,#15. ;16 CHARACTERS ?
BHS 8$
BIT #SILOAL,@DZCSR ;SILO ALARM = 0 ?
BEQ +4 ;YES
ERROR 13 ;*SILO ALARM SHOULD NOT = 1
;UNTIL 16. DATA CHARACTERS
8$: BR 10$
CLR R4

```

```

9236 021006 032777 020000 161026 9$: BIT #SILOAL,@DZCSR
9237 021014 001004 BNE 10$
9238 021016 104414 DELAY
9239 021020 005204 INC R4
9240 021022 001371 BNE 9$
9241 021024 104014 ERROR 14 ;*SILO ALARM FAILED TO SET!
9242 ;SILO ALARM SHOULD =1 AFTER 16.
9243 ;DATA CHARACTERS
9244 021026 005203 10$: INC R3 ;INC CHAR COUNT
9245 021030 022703 000102 CMP #66.,R3 ;FINISHED SENDING CHARACTERS ?
9246 021034 001334 BNE 5$ ;NO
9247 021036 005004 CLR R4
9248 021040 104414 DELAY
9249 021042 105204 INCB R4
9250 021044 001375 BNE -4
9251 ;NOW LETS READ THE SILO
9252 021046 013705 001372 MOV SAVLIN,R5 ;MAKE EXPECTED LINE #
9253 021052 105737 001371 TSTB MODE+1 ;IS THIS TEST IN STAGGERED MODE?
(1) 021056 001406 BEQ 13$ ;IF NOT, SKIP STAGGERED SETUP
(1)
(1) ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1)
(1) 021060 006205 ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
(1) 021062 103402 BCS 11$ ;IF IT IS SET, GO CLEAR IT
(1) 021064 000261 SEC ;IF IT IS CLEAR SET IT HERE
(1) 021066 000401 BR 12$ ;SKIP THE CLEARING
(1) 021070 000241 11$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 021072 006105 12$: ROL R5 ;GET THE NEW BIT BACK INTO R5
9254 021074 000305 13$: SWAB R5 ;PUT IN UPPER BYTE
9255 021076 052705 100000 BIS #DVALID,R5 ;ADD DATA VALID
9256 021102 017704 160740 14$: MOV @DZRBUF,R4 ;ACTUAL
9257 021106 020405 CMP R4,R5 ;ACTUAL VS. EXPECTED
9258 021110 001401 BEQ 15$ ;YES
9259 021112 104006 ERROR 6 ;*DATA/CONTENTS DID NOT COMPARE
9260 021114 032777 020000 160720 15$: BIT #SILOAL,@DZCSR ;SILO ALARM= 0 ?
9261 021122 001401 BEQ 16$ ;YES
9262 021124 104016 ERROR 16 ;READING DZRBUF DID NOT CLEAR SILO ALARM
9263 021126 005205 16$: INC R5 ;UP CHARACTER
9264 021130 120527 000077 CMPB R5,#63. ;LAST SILO CHAR ?....64TH CHAR
9265 021134 101762 BLOS 14$
9266 021136 005205 INC R5 ;ADD 1 MORE FOR THE CLOBBERED CHAR
9267 021140 052705 040000 BIS #OVERRUN,R5 ;ADD OVERRUN TO EXPECTED
9268 021144 120527 000101 CMPB R5,#65. ;LAST CHARACTER ?
9269 021150 001754 BEQ 14$
9270 021152 017704 160670 MOV @DZRBUF,R4 ;FOR GOOD MEASURE
9271 021156 005704 TST R4 ;DATA VALID SHOULD = 0
9272 021160 100001 BPL 17$ ;YES
9273 021162 104017 ERROR 17 ;DATA VALID SHOULD = 0
9274 021164 040277 160666 17$: BIC R2,@DZTCR ;CLR TCR BIT
9275 021170 104401 SCOP1 ;LOOP?
9276 021172 005237 001372 22$: INC SAVLIN ;INC EXPECTED LINE
9277 021176 106302 ASLB R2 ;NEXT LINE
9278 021200 103402 BCS +6 ;NO
9279 021202 000137 020670 JMP 3$ ;YES
9280 021206 104400 ADVANCE ;GO TO NEXT TEST
9281

```

```

9282 ;TIGHT SCOPE LOOP FOR THIS TEST. SENDS 20. CHARACTERS
9283 ;ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
9284 ;USED TO SCOPE SILO ALARM PULSES, ETC.
9285
9286 021210 052777 010040 160624 18$: BIS #MSENAB!SILOEN,@DZCSR ;SETUP DEVICE
9287 021216 012777 021274 160652 MOV #20$,@DZTIV ;SETUP TRANSMITTER VECTOR
9288 021224 012737 000024 001216 MOV #20,$TMP0 ;TEMPORARY COUNT OF CHARACTER BURST
9289 021232 050277 160620 BIS R2,@DZTCR ;ENABLE LINE
9290 021236 052777 040000 160576 BIS #TIE,@DZCSR ;ENABLE INTERRUPTS
9291 021244 106427 000000 MTPS #0 ;LOWER PRIORITY
9292 021250 000001 19$: WAIT ;ALLOW INTERRUPTS
9293 021252 005337 001216 DEC $TMP0 ;REDUCE COUNT. ALL CHARACTERS SENT?
9294 021256 001374 BNE 19$ ;IF NO, WAIT FOR MORE
9295 021260 042777 050040 160554 BIC #SILOEN!MSENAB!TIE,@DZCSR ;RESET SILO COUNTER, CLEAR STROBE
9296 021266 104401 SCOP1 ;LOOP AGAIN?
9297 021270 000137 021164 JMP 17$ ;IF NOT, RETURN TO WHERE YOU LEFT OFF
9298 021274 112777 000252 160564 20$: MOVB #252,@DZTDR ;SEND A CHARACTER
9299 021302 000002 RTI ;ALLOW MORE CHARACTERS TO COME
9300 ;***** TEST 32 *****
9301 ;*THIS TEST THAT "SILO ENABLE" WILL INHIBIT
9302 ;*RECEIVER INTERRUPTS AND THAT ON THE
9303 ;*16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
9304 ;*INTERRUPT WITH "RIE" SET.
9305 ;*THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
9307 ;:* TEST 32
(5) ;*****
(4) 021304 000004 TST32: SCOPE
(2) 021306 012737 000032 001122 MOV #32,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(2) 021314 012737 021666 001360 MOV #TST33,NEXT ;POINT TO THE START OF THE NEXT TEST
9308 021322 012737 021410 001362 MOV #3$,LOCK ;SET FOR LOOP
9309 021330 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 021332 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
(1) 021336 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
(1) 021342 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
(1) 021346 001402 BEQ 2$ ;NO
(1) 021350 010177 160476 MOV R1,@DZLPR ;SET UP LINE PARAMETERS
(1) 021354 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
(1) 021356 106302 ASLB R2 ;GOT 'EM ALL ?
(1) 021360 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
(1) 021362 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
9310 021366 012700 001422 MOV #TD0,R0 ;POINT TO THE DATA AREA
9311 021372 005020 CLR (R0)+ ;CLEAR A DATA WORD
9312 021374 022700 001462 CMP #STOP,R0 ;FINISHED ?
9313 021400 001374 BNE .-6 ;NO
9314 021402 005000 CLR R0 ;CLEAR OFFSET
9315 021404 012702 000001 MOV #1,R2 ;LINE POINTER
9316 021410 012777 021630 160454 3$: MOV #11$,@DZRIV ;SET FOR UNEXPECTED INTER.
9317 021416 012777 000340 160450 MOV #PR7,@DZ RIS ;SET PRIO.
9318 021424 052777 010140 160410 BIS #MSENAB!SILOEN!RIE,@DZCSR ;START SCANNER & SET SILO ENABLE
9319 ;VALID LINE?
9320 021432 030237 001364 BIT R2,LINE ;YES
9321 021436 001002 BNE .+6 ;TRY NEXT LINE
9322 021440 000137 021646 JMP 22$ ;EMPTY THE SILO
9323 021444 005777 160376 TST @DZRBUF ;BR IF DATA VALID IS SET!
9324 021450 100775 BMI .-4 ;SET PROCESSOR PRIORITY TO 0
9325 021452 106427 000000 MTPS #0

```

9326	021456	013700	001372			MOV	SAVLIN,R0		:MAKE OFFSET
9327	021462	006300				ASL	R0		:MAKE POWER OF TWO
9328	021464	010277	160366			MOV	R2,@DZTCR		:SET TCR BIT
9329	021470	005004			5\$:	CLR	R4		
9330	021472	032777	100000	160342	6\$:	BIT	#TRDY,@DZCSR		
9331	021500	001004				BNE	7\$		
9332	021502	104414				DELAY			
9333	021504	005204				INC	R4		
9334	021506	001371				BNE	6\$		
9335	021510	104003				ERROR	3		:*TRDY FAILED TO SET
9336	021512	116077	001422	160346	7\$:	MOVB	TDO(R0),@DZTDR		:LOAD A CHARACTER
9337	021520	005260	001422			INC	TDO(R0)		:SET UP NEXT CHARACTER
9338	021524	022760	000017	001422		CMP	#15.,TDO(R0)		:15 CHARS YET?
9339	021532	001406				BEQ	8\$		
9340	021534	032777	020000	160300		BIT	#SILOAL,@DZCSR		:SILO ALARM = 0 ?
9341	021542	001401				BEQ	.+4		:YES
9342	021544	104013				ERROR	13		:*SILO ALARM SHOULD NOT = 1
9343									:UNTIL 16. DATA CHARACTERS
9344	021546	000750				BR	5\$		
9345	021550	012777	021636	160314	8\$:	MOV	#12\$,@DZRIV		:SET NEW VECTOR
9346	021556	032777	100000	160256		BIT	#TRDY,@DZCSR		:READY FOR 16TH CHAR
9347	021564	001774				BEQ	.-6		
9348	021566	016077	001422	160272		MOV	TDO(R0),@DZTDR		:LOAD THE 16TH CHAR.
9349	021574	005004				CLR	R4		
9350	021576	032777	020000	160236	9\$:	BIT	#SILOAL,@DZCSR		
9351	021604	001005				BNE	10\$		
9352	021606	104414				DELAY			
9353	021610	005204				INC	R4		
9354	021612	001371				BNE	9\$		
9355	021614	104014				ERROR	14		:*SILO ALARM FAILED TO SET!
9356	021616	000410				BR	17\$:SILO ALARM SHOULD =1 AFTER 16.
9357									:DATA CHARACTERS
9358	021620	000240			10\$:	NOP			:STALL
9359	021622	000240				NOP			
9360	021624	104000				ERROR			:SILO ALARM NOT INTERRUPTING.
9361	021626	000404				BR	17\$:CONTINUE TEST.
9362	021630	022626			11\$:	CMP	(SP)+,(SP)+		:FAKE RTI
9363	021632	104012				ERROR	12		:RX SHOULD NOT INTERRUPT
9364	021634	000401				BR	17\$:CONTINUE
9365	021636	022626			12\$:	CMP	(SP)+,(SP)+		:GOOD INTERRUPT TO HERE.
9366	021640	040277	160212		17\$:	BIC	R2,@DZTCR		:CLR TCR BIT
9367	021644	104401				SCOP1			:LOOP?
9368	021646	005237	001372		22\$:	INC	SAVLIN		:INC EXPECTED LINE
9369	021652	106302				ASLB	R2		:NEXT LINE
9370	021654	103402				BCS	.+6		:NO
9371	021656	000137	021410			JMP	3\$:YES
9372	021662	005037	001362			CLR	LOCK		:CLEAR TIGHT LOOP FOR NEXT TEST

```

9374
9375
9376
9377
9378
9380
(5)
(4) 021666 000004
(2) 021670 012737 000033 001122
(2) 021676 012737 022474 001360
9381 021704 104417
9382 021706 013737 001364 022472
9383 021714 013701 001366
9384 021720 012700 000001
9385 021724 030037 001364
9386 021730 001402
9387 021732 010177 160114
9388 021736 005201
9389 021740 106300
9390 021742 103370
9391 021744 012700 001422
9392 021750 005020
9393 021752 022700 001462
9394 021756 001374
9395 021760 012777 022214 160104
9396 021766 012777 000340 160100
9397 021774 012777 022116 160074
9398 022002 012777 000340 160070
9399 022010 052777 000100 160024
9400 022016 052777 040000 160016
9401 022024 052777 000040 160010
9402 022032 113777 001364 160016
9403 022040 106437 027534
9404
9405
9406 022044 005037 022114
9407 022050 013727 007214
9408 022054 000000
9409 022056 005337 022054
9410 022062 001375
9411 022064 105737 022472
9412 022070 001002
9413 022072 000137 022372
9414 022076 005237 022114
9415 022102 001362
9416 022104 104007
9417 022106 104011
9418 022110 000137 022444
9419 022114 000000
9420
9421
9422 022116 005777 157720
9423 022122 100401
9424 022124 104003
9425 022126 117703 157712
9426

```

```

:***** TEST 33 *****
:*THIS TEST RUNS ALL LINES FULL BORE
:*BASED UPON QUALIFIED LINES
:*..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
:*TRANSMITTER
:.* TEST 33
:*****
TST33: SCOPE
MOV #33,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST34,NEXT ;POINT TO THE START OF THE NEXT TEST
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV LINE,RXTCR ;SET IMAGE OF TCR BITS
RSTART: MOV PAR,R1 ;PICK UP PARAMETER
MOV #1,R0 ;PICK UP INIT POINTER
INIT: BIT R0,LINE ;SHOULD THIS LINE BE SET UP
BEQ 1$ ;NO
MOV R1,@DZLPR ;SET UP LINE PARAM REGISTER
1$: INC R1
ASLB R0 ;GOT 'EM ALL ?
BCC INIT ;NO
MOV #TD0,R0 ;CLEAR TRANS DATA POINTER & REC POINTERS
INIT1: CLR (R0)+
CMP #STOP,R0 ;FINISHED ?
BNE INIT1 ;NO, CONTINUE CLEARING
MOV #RXSVC,@DZRIV ;SET UP REC INTR VECTOR
MOV #PR7,@DZRI5 ;STATUS
MOV #TXSVC,@DZTIV ;SET UP TRANS INTR VECTOR
MOV #PR7,@DZTIS ;STATUS
BIS #RIE,@DZCSR ;SET REC INTR ENABLE
BIS #TIE,@DZCSR ;SET TRANS INTR ENABLE
BIS #MSENAB,@DZCSR ;SET MASTER SCAN ENABLE
MOVB LINE,@DZTCR ;SET TCR BITS...UP UP AND AWAY !
MTPS @#LESS1 ;ALLOW INTERRUPTS

SNAP: CLR 66$
67$: MOV DLYCNT,(PC)+ ;SET FOR DELAY
68$: 0
DEC 68$
BNE -4
TSTB RXTCR ;WAIT FOR ALL RECIEVERS TO FINISH
BNE 3$
JMP OUT
3$: INC 66$
BNE 67$
ERROR 7 ;*TRANSMITTER FAILED TO INTERRUPT
ERROR 11 ;*RECEIVER FAILED TO INTERRUPT
66$: 0

;TRANS INTR SVC ROUTINE
TXSVC: TST @DZCSR ;TRANS INTR ?
BMI +4
ERROR 3 ;*TRANSMITTER FAILED
MOVB @HDZCSR,R3 ;SAVE IT
;NOW TEST FOR LINE # ETC

```

9427	022132	042703	177770		BIC	#^C<7>,R3	:STRIP JUNK
9428	022136	010304			MOV	R3,R4	:SAVE
9429	022140	010337	001372		MOV	R3,SAVLIN	:ADJUST LOCATION FOR ERROR PRINTOUT
9430	022144	012702	000001		MOV	#1,R2	:SET UP POSITION POINTER
9431	022150	105303		3\$:	DECB	R3	:IS IT THIS LINE ?
9432	022152	100402			BMI	4\$:YES
9433	022154	006302			ASL	R2	:UP THE LINE #
9434	022156	000774			BR	3\$:GO 'ROUND AGAIN
9435	022160	030237	001364	4\$:	BIT	R2,LINE	:VALID LINE?
9436	022164	001001			BNE	+.4	:YES
9437	022166	104010			ERROR	10	:NO,INVALID LINE!!!!
9438	022170	006304			ASL	R4	:MAKE POWER OF 2
9439	022172	116477	001422	157666	MOVB	TD0(R4),@DZTDR	:LOAD CHARACTER
9440	022200	105264	001422		INCB	TD0(R4)	:SET UP NEXT CHARACTER
9441	022204	001002			BNE	5\$:LAST CHARACTER ?
9442	022206	040277	157644		BIC	R2,@DZTCR	:YES ,CLEAR TCR BIT
9443	022212	000002		5\$:	RTI		
9444							
9445							
9446							
9447	022214	105777	157622		:REC INTR SVC ROUTINE		
9448	022220	100401			RXSVC: TSTB	@DZCSR	:REC DONE ?
9449	022222	104004			BMI	+.4	:YES
9450	022224	017704	157616		ERROR	4	:FALSE INTERRUPT
9451	022230	010403			MOV	@DZRBUF,R4	:SAVE IT
9452	022232	000303			MOV	R4,R3	
9453	022234	042703	177770		SWAB	R3	
9454	022240	010337	001372		BIC	#^C<7>,R3	:STRIP JUNK
9455	022244	032777	020000	157570	MOV	R3,SAVLIN	:SAVE LINE NUMBER
9456	022252	001401			BIT	#SILOAL,@DZCSR	:SILO ALARM?
9457	022254	104000			BEQ	+.4	:NO
9458	022256	005704			ERROR		:SILO ALARM SHOULD NOT =1
9459	022260	100401			TST	R4	:DATA VALID SET?
9460	022262	104023			BMI	+.4	:YES
9461	022264	032704	070000		ERROR	23	:YOU LOSE ...DATA VALID WAS'NT SET
9462	022270	001401			BIT	#OVRRUN!FRMERR!PARER,R4	
9463	022272	104000			BEQ	+.4	
9464	022274	012702	000001		ERROR		:RECEIVER ERROR FLAG/S WERE SET
9465	022300	105303		5\$:	MOV	#1,R2	:SET UP POSITION POINTER
9466	022302	100402			DECB	R3	
9467	022304	006302			BMI	6\$	
9468	022306	000774			ASL	R2	:RE POSITION POINTER
9469	022310	030237	001364	6\$:	BR	5\$:GO 'ROUND AGAIN
9470	022314	001001			BIT	R2,LINE	:LINE VALID ?
9471	022316	104011			BNE	+.4	:YES
9472	022320	013703	001372		ERROR	11	:INVALID LINE #
9473	022324	006303			MOV	SAVLIN,R3	:GET THE LINE NUMBER AGAIN
9474	022326	126304	001442		ASL	R3	:USE R3 AS A POINTER IN THE DATA TABLE
9475	022332	001405			CMPB	TRO(R3),R4	:DOES THE DATA CHARACTER COMPARE ?
9476	022334	016305	001442		BEQ	2\$:YES
9477	022340	042704	177400		MOV	TRO(R3),R5	:SAVE EXPECTED
9478					BIC	#^C<377>,R4	:CLEAR JUNK
9479							:R2 = LINE # BY BIT POSITION
9480							:R4 = ACTUAL DATA
9481	022344	104005			ERROR	5	:R5 = EXPECTED DATA
9482	022346	005263	001442	2\$:	INC	TRO(R3)	:*NO, DATA DOES NOT COMPARE :SET UP FOR NEXT CHARACTER

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 83-2
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0092

```

9483 022352 105763 001442      TSTB   TR0(R3) ;ALL CHARS DONE?
9484 022356 001002              BNE     .+6
9485 022360 040237 022472      BIC     R2,RXTCR      ;ZERO LINE DONE INDICATOR.
9486 022364 012716 022044      MOV     #SNAP,(SP)   ;RESET THE BACKGROUND TIMING LOOP
9487 022370 000002              RTI
9488
9489
9490                                ;FINISH UP ROUTINE
9491 022372 106427 000340      OUT:   MTPS   #PR7              ;STOP ALL INTERRUPTS
9492 022376 104413              DEVICE.CLR           ;CLEAR ALL INTERRUPTS AWAY
9493 022400 005003              CLR     R3
9494 022402 005037 001372      CLR     SAVLIN
9495 022406 012702 000001      MOV     #1,R2
9496 022412 030237 001364      1$:   BIT     R2,LINE      ;VALID LINE ?
9497 022416 001405              BEQ     2$            ;NO
9498 022420 022763 000400 001442  CMP     #400,TR0(R3)   ;RECEIVED A BINARY COUNT PATTERN ?
9499 022426 001401              BEQ     .+4           ;YES
9500 022430 104027              ERROR   27            ;THE LINE FAILED TO RECEIVE A FULL
9501                                ;BINARY COUNT PATTERN
9502 022432 005237 001372      2$:   INC     SAVLIN      ;SET UP FOR NEXT LINE
9503 022436 005723              TST     (R3)+         ;ADD 2
9504 022440 106302              ASLB   R2              ;SET UP NEXT LINE POINTER
9505 022442 103363              BCC    1$             ;FINISHED ?
9506 022444
9507 022444 013777 002074 157420  FINI:  MOV     DZRRIS,@DZRRIV  ;RESTORE TRAPCATCHER
9508 022452 005077 157416      CLR     @DZRRIS
9509 022456 013777 002100 157412  MOV     DZTIS,@DZTIV
9510 022464 005077 .57410      CLR     @DZTIS
9511 022470 104400              ADVANCE              ;GO TO THE NEXT TEST
9512 022472 000000      RXTCR: 0              ;RX IMAGE OF TCR BITS
9513
9514
9515
9516
9517
9518
9519
9520
9521
9522
9523
9524
9525
9526
9527
9528
9529
9530
9532

```

```

***** TEST 34 *****
;*DZ11 RELATIVE TIMING TEST.
;*EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
;*AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
;*WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
;*DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
;*THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
;* AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
;*PARAMETERS ARE:
;* EIGHT BITS/PER/CHAR - TWO STOP BITS AT
;* 50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
;* 2400, 3600, 4800, 7200, 9600 BAUD.
;* THEN, 9600 BAUD - TWO STOP BITS AT
;* SEVEN, SIX, FIVE BITS/PER/CHAR.
;*AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
;*THE NEXT SELECTED LINE IS THE TESTED.

```

::* TEST 34

```

(5)
(4) 022474 000004
(2) 022476 012737 000034 001122  TST34: SCOPE
(1) 022504 012737 000002 001226  MOV     #34,$STNM      ;LOAD THE NUMBER OF THIS TEST
(2) 022512 012737 023204 001360  MOV     #2,$TIMES
9533 022520 012737 022644 001362  MOV     #TST35,NEXT   ;POINT TO THE START OF THE NEXT TEST
9534 022526 005037 024440      MOV     #3$,LOCK      ;SET FOR LOOP
                                CLR     OFFSET           ;RESET THIS VARIABLE

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 83-3
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0093

```

9535 022532 005037 001372 CLR SAVLIN ;RESET LINE NUMBER INDICATOR
9536 022536 005037 001374 CLR XMTLIN ;USE THIS WORD TO TELL WHAT LINE TRANSMITTED
9537 022542 012737 000001 001216 MOV #1,$TMP0 ;USE $TMP0 AS A BIT POINTER
9538 022550 012737 010070 023202 MOV #RCVON!S50!EIGHT!TWOSTOP,7$ ;BUILD TEMPORARY PARAMETERS
9539 022556 033737 001216 001364 1$: BIT $TMP0,LINE ;IS THIS LINE ACTIVE?
9540 022564 001027 BNE 3$ ;IF SO, GO GET STARTED
9541 022566 012737 010070 023202 2$: MOV #RCVON!S50!EIGHT!TWOSTOP,7$ ;LOAD PARAMETERS TEMPORARILY
9542 022574 012700 001422 MOV #TDO,R0 ;POINT TO THE DATA AREA
9543 022600 005020 CLR (R0)+ ;CLEAR A DATA WORD
9544 022602 022700 001462 CMP #STOP,R0 ;FINISHED ?
9545 022606 001374 BNE .-6 ;NO
9546 022610 005237 001374 INC XMTLIN ;POINT TO THE NEXT LINE TO TRANSMIT
9547 022614 042737 000007 023202 BIC #7,7$ ;MAKE SURE TEMPORARY PARAMETERS POINT TO 0
9548 022622 053737 001374 023202 BIS XMTLIN,7$ ;ADD DESIRED LINE NUMBER
9549 022630 005037 024440 CLR OFFSET
9550 022634 106337 001216 ASLB $TMP0 ;POINT TO THE NEXT LINE
9551 022640 103346 BCC 1$ ;PROCESS THE NEXT LINE
9552 022642 104400 ADVANCE ;TEST TO SEE IF THIS TEST GETS REPEATED
9553 022644 3$:
(1) 022644 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
9554 022646 042737 010000 023202 BIC #RCVON,7$ ;ZERO PARAMTERS FOR TX LINE
9555 022654 013777 023202 157170 MOV 7$,@DZLPR ;LOAD PARAMTERS FOR TX
9556 022662 005737 001370 TST MODE ;STAGGERED?
9557 022666 100011 BPL 100$ ;BR IF NO
9558 022670 000241 CLC ;SET UP LINE
9559 022672 006037 023202 ROR 7$ ;
9560 022676 103002 BCC 98$ ;BR IF LINE WAS EVEN
9561 022700 000241 CLC ;PREPARE TO MKE LINE EVEN
9562 022702 000401 BR 99$ ;CONTINUE
9563 022704 C00261 98$: SEC ;PREPARE TO MAKE LINE ODD
9564 022706 006137 023202 99$: ROL 7$ ;SET ALTERED LINE
9565 022712 052737 010000 023202 100$: BIS #RCVON,7$ ;SET RX ON
9566 022720 013777 023202 157124 MOV 7$,@DZLPR ;LOAD RX PARAMETERS
9567 022726 013737 023202 001372 MOV 7$,SAVLIN ;ADJUST LOCATION FOR ERROR PRINTOUT
9568 022734 042737 177770 001372 BIC #^C<7>,SAVLIN ;STRIP JUNK
9569 022742 042737 000007 023202 BIC #7,7$ ;CLEAR OLD LINE #
9570 022750 053737 001374 023202 BIS XMTLIN,7$ ;SET LINE UP AGAIN
9571 022756 013737 023202 001400 MOV 7$,REGIST ;SAVE PARAMETERS FOR PRINTOUT
9572 022764 012700 001422 MOV #TDO,R0 ;POINT TO THE DATA AREA
9573 022770 005020 CLR (R0)+ ;CLEAR A DATA WORD
9574 022772 022700 001462 CMP #STOP,R0 ;FINISHED ?
9575 022776 001374 BNE .-6 ;NO
9576 023000 005002 CLR R2 ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
9577 023002 005003 CLR R3 ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
9578 023004 005037 001220 CLR $TMP1 ;INITIALIZE THE TIMER
9579 023010 005037 001224 CLR $TMP3 ;INITIALIZE THESE BITS ALSO
9580 023014 012737 000020 001376 MOV #20,XMTCNT ;SET HOW MANY CHARACTERS TO TRANSMIT
9581 023022 012777 024062 157046 MOV #XMTSRV,@DZTIV
9582 023030 012777 024226 157034 MOV #RXISR1,@DZRIV
9583 023036 013777 027532 157030 MOV DZPRT,@DZRI5
9584 023044 013777 027532 157026 MOV DZPRT,@DZTI5
9585 023052 113777 001216 156776 MOVB $TMP0,@DZTCR ;START THE VALID LINE
9586 023060 052777 040140 156754 BIS #TIE!RIE!MSENAB,@DZCSR
9587 023066 106427 000000 MTPS #0 ;LOWER THE PRIORITY TO ALLOW INTERRUPTS
9588 023072 032777 000100 156742 4$: BIT #RIE,@DZCSR ;IS ROUTINE DONE?
9589 023100 001407 BEQ 5$ ;WHEN ALL IS DONE RX IE IS CLEARED IN ISR.

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 83-4
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0094

```

9590 023102 005237 001220      INC      $TMP1      ;COUNT TIME
9591 023106 001371              BNE      4$        ;CONTINUE TEST
9592 023110 105237 001224      INCB     $TMP3     ;DOUBLE COUNT
9593 023114 001366              BNE      4$        ;CONTINUE TEST
9594 023116 104011              ERROR    11        ;INTERRUPTS NOT FINISHED
9595 023120 004737 007652      JSR     PC,SERV.G  ;<^G>?
9596 023124 104401              SCOPI    ;LOOP?
9597 023126 062737 000002 024440      ADD     #2,OFFSET
9598 023134 013700 023202      MOV     7$,R0
9599 023140 042700 170377      BIC     #^C<17*400>,R0
9600 023144 022700 007000      CMP     #<16*400>,R0
9601 023150 001010              BNE     6$
9602 023152 032737 000030 023202      BIT     #BIT4+BIT3,7$
9603 023160 001602              BEQ     2$
9604 023162 162737 000010 023202      SUB     #BIT3,7$
9605 023170 000625              BR      3$
9606 023172 062737 000400 023202 6$:      ADD     #400,7$
9607 023200 000621              BR      3$
9608 023202 000000      7$:      0
9609 (1) :***** TEST 35 *****
(1) :* THIS TEST VERIFIES THAT EVEN PARITY WORKS
(1) :* FOR ALL ODD LINES SELECTED AND THAT ODD PARITY WORKS FOR ALL
(1) :* EVEN LINES SELECTED.
(1) :*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
(1) :*THAT 'PE' (PARITY ERROR) CAN BE FLAGGED BY
(1) :*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
(1) :*YOU ARE IN 'STAGGERED' MODE.
(1) :*40(8) CHARS ARE USED FOR THIS TEST.
(1) :*ALL SELECTED LINES WILL BE ENABLED
(1) :*AT THE SAME TIME!
(3) :::* TEST 35
(6) :*****
(5) 023204 000004      TST35: SCOPE
(3) 023206 012737 000035 0C1122      MOV     #35,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3) 023214 012737 023444 001360      MOV     #TST36,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 023222 005737 001370      TST     MODE       ;IS THIS STAGGERED MODE?
(1) 023226 100105      BPL     6$         ;IF NOT, DON'T DO THIS TEST
(2) 023230 104417      DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 023232 013701 001366      MOV     PAR,R1     ;USE R1 TO BUILD PARAMETERS TO BE LOADED
(1) 023236 042701 000200      BIC     #ODDPAR,R1 ;MAKE SURE ODD PARITY ISN'T SET
(1) 023242 052701 000100      BIS     #PARITY,R1 ;MAKE SURE PARITY IS TURNED ON
(1) 023246 012702 000001      MOV     #1,R2      ;USE R2 AS A LINE POINTER
(1) 023252 030237 001364      1$:     BIT     R2,LINE  ;IS THIS A VALID LINE?
(1) 023256 001411      BEQ     3$         ;IF NOT, SKIP TO THE NEXT LINE
(1) 023260 032701 000001      BIT     #BIT0,R1   ;IS THIS LINE AN ODD LINE?
(1) 023264 001002      BNE     2$         ;IF IT'S ODD, USE EVEN PARITY
(1) 023266 052701 000200      BIS     #ODDPAR,R1 ;IF IT'S EVEN, USE ODD PARITY
(1) 023272 010177 156554      2$:     MOV     R1,@DZLPR ;LOAD THE LINE PARAMETER REGISTER
(1) 023276 042701 000200      BIC     #ODDPAR,R1 ;SET UP THE NEXT PARITY TO EVEN
(1) 023302 005201      3$:     INC     R1        ;POINT TO THE NEXT LINE
(1) 023304 106302      ASLB    R2         ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
(1) 023306 103361      BCC     1$         ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
(1) 023310 005037 001372      CLR     SAVLIN     ;CLEAR THE LINE NUMBER INDICATOR
(1) 023314 005002      CLR     R2         ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
(1) 023316 005003      CLR     R3         ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
(1) 023320 012737 000040 001376      MOV     #40,XMTCNT ;TRANSMIT A BINARY COUNT PATTERN(00-40)

```

```

(1) 023326 01270C 001422      MOV      #TDO,R0          ;POINT TO THE DATA AREA
(1) 023332 005020              CLR      (R0)+            ;CLEAR A DATA WORD
(1) 023334 022700 001462      CMP      #STOP,R0        ;FINISHED ?
(1) 023340 001374              BNE     .-6              ;NO
(1) 023342 005000              CLR      R0              ;CLEAR OFFSET
(2) 023344 012777 024062 156524  MOV      #XMTSRV,@DZTIV  ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 023352 012777 023704 156512  MOV      #PARESE,@DZRIV  ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 023360 013777 027532 156506  MOV      DZPRT,@DZRRIS  ;SET THE INTERRUPT VECTOR STATUS
(2) 023366 013777 027532 156504  MOV      DZPRT,@DZTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 023374 052777 040140 156440  BIS      #RIE!TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(1) 023402 113777 001364 156446  MOV      LINE,@DZTCR    ;ENABLE ALL SELECTED LINES
(1) 023410 106427 000000              MTPS     #0              ;ALLOW INTERRUPTS
(1) 023414 032777 000100 156420 5$:  BIT      #RIE,@DZCSR    ;WHEN RX DONE; RIE WILL =0
(1) 023422 001407              BEQ     6$              ;BR IF ALL DONE
(1) 023424 005237 024056              INC      COUNT0
(1) 023430 102771              BVS     5$
(1) 023432 105237 024060              INCB     COUNT1
(1) 023436 100366              BPL     5$
(1) 023440 104011              ERROR   11              ;*RX FAILED TO FINISH (INTERRUPT)
(1) 023442 104400 6$:  ADVANCE ;ADVANCE LOOP
9610
(1) :***** TEST 36 *****
(1) :*THIS TEST VERIFIES THAT ODD PARITY WORKS FOR ALL ODD LINES
(1) :* SELECTED AND THAT EVEN PARITY WORKS FOR ALL EVEN LINES SELECTED
(1) :*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
(1) :*THAT 'PE' (PARITY ERROR) CAN BE FLAGGED BY
(1) :*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
(1) :*YOU ARE IN "STAGGERED" MODE.
(1) :*40(8) CHARS ARE USED FOR THIS TEST.
(1) :*ALL SELECTED LINES WILL BE ENABLED
(1) :*AT THE SAME TIME!
(3) :::* TEST 36
(6) :*****
(5) 023444 000004 TST36: SCOPE
(3) 023446 012737 000036 0C1122  MOV      #36,$TSTNM     ;LOAD THE NUMBER OF THIS TEST
(2) 023454 012737 004712 001360  MOV      #$EOP,NEXT    ;POINT TO THE END-OF-PASS HANDLER
(1) 023462 005737 001370              TST      MODE          ;IS THIS STAGGERED MODE?
(1) 023466 100105              BPL     6$              ;IF NOT, DON'T DO THIS TEST
(2) 023470 104417              DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(1) 023472 013701 001366              MOV      PAR,R1        ;USE R1 TO BUILD PARAMETERS TO BE LOADED
(1) 023476 042701 000200              BIC     #ODDPAR,R1     ;MAKE SURE ODD PARITY ISN'T SET
(1) 023502 052701 000100              BIS     #PARITY,R1     ;MAKE SURE PARITY IS TURNED ON
(1) 023506 012702 000001              MOV     #1,R2          ;USE R2 AS A LINE POINTER
(1) 023512 030237 001364 1$:  BIT      R2,LINE       ;IS THIS A VALID LINE?
(1) 023516 001411              BEQ     3$              ;IF NOT, SKIP TO THE NEXT LINE
(1) 023520 032701 000001              BIT     #BIT0,R1       ;IS THIS LINE AN ODD LINE?
(1) 023524 001402              BEQ     2$              ;IF IT'S EVEN, USE EVEN PARITY
(1) 023526 052701 000200              BIS     #ODDPAR,R1     ;IF IT'S ODD, USE ODD PARITY
(1) 023532 010177 156314 2$:  MOV      R1,@DZLPR     ;LOAD THE LINE PARAMETER REGISTER
(1) 023536 042701 000200              BIC     #ODDPAR,R1     ;SET UP THE NEXT PARITY TO EVEN
(1) 023542 005201 3$:  INC      R1            ;POINT TO THE NEXT LINE
(1) 023544 106302              ASLB    R2              ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
(1) 023546 103361              BCC     1$              ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
(1) 023550 005037 001372              CLR     SAVLIN         ;CLEAR THE LINE NUMBER INDICATOR
(1) 023554 005002              CLR     R2             ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
(1) 023556 005003              CLR     R3             ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
(1) 023560 012737 000040 001376  MOV      #40,XMTCNT    ;TRANSMIT A BINARY COUNT PATTERN(00-40)

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 83-6
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0096

```

(1) 023566 012700 001422      MOV    #TDO,R0      ;POINT TO THE DATA AREA
(1) 023572 005020              CLR    (R0)+        ;CLEAR A DATA WORD
(1) 023574 022700 001462      CMP    #STOP,R0    ;FINISHED ?
(1) 023600 001374              BNE    .-6         ;NO
(1) 023602 005000              CLR    R0          ;CLEAR OFFSET
(2) 023604 012777 024062 156264  MOV    #XMTSRV,@DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 023612 012777 023704 156252  MOV    #PARESE,@DZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 023620 013777 027532 156246  MOV    DZPRT,@DZ RIS ;SET THE INTERRUPT VECTOR STATUS
(2) 023626 013777 027532 156244  MOV    DZPRT,@DZTIS ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 023634 052777 040140 156200  BIS    #RIE!TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(1) 023642 113777 001364 156206  MOVB  LINE,@DZTCR  ;ENABLE ALL SELECTED LINES
(1) 023650 106427 000000              MTPS  #0          ;ALLOW INTERRUPTS
(1) 023654 032777 000100 156160 5$: BIT    #RIE,@DZCSR ;WHEN RX DONE; RIE WILL =0
(1) 023662 001407              BEQ    6$         ;BR IF ALL DONE
(1) 023664 005237 024056              INC   COUNT0
(1) 023670 102771              BVS   5$
(1) 023672 105237 024060              INCB  COUNT1
(1) 023676 100366              BPL   5$
(1) 023700 104011              ERROR 11         ;*RX FAILED TO FINISH (INTERRUPT)
(1) 023702 104400              6$:  ADVANCE     ;ADVANCE LOOP

```

```

9612
9613
9614
9615 023704 017704 156136
9616 023710 010401
9617 023712 000301
9618 023714 042701 177770
9619 023720 010137 001372
9620 023724 005704
9621 023726 100401
9622 023730 104023
9623 023732 006301
9624 023734 032704 010000
9625 023740 001013
9626 023742 013737 002046 001400
9627 023750 010405
9628 023752 042705 000377
9629 023756 156105 001442
9630 023762 052705 110000
9631 023766 104006
9632 023770 126104 001442
9633 023774 001407
9634 023776 116105 001442
9635 024002 042705 177400
9636 024006 042704 177400
9637 024012 104005
9638 024014 005261 001442
9639 024020 005203
9640 024022 005037 024056
9641 024026 005037 024060
9642 024032 032777 040000 156002
9643 024040 001005
9644 024042 020203
9645 024044 001003
9646 024046 042777 000100 155766
9647 024054 000002
9648 024056 000000
9649 024060 000000
9650
9651
9652
9653
9654
9655 024062 117701 155756
9656 024066 100411
9657 024070 013700 001372
9658 024074 042701 177770
9659 024100 010137 001372
9660 024104 104003
9661 024106 010037 001372
9662 024112 042701 177770
9663 024116 006301
9664 024120 116177 001422 155740
9665 024126 005261 001422
9666
9667

```

:RECEIVER SERVICE ROUTINE(PARITY TEST ONLY)

```

PARESE: MOV @DZRBUF,R4 ;GET THE CHARACTER
MOV R4,R1 ;COPY THE RECEIVED INFORMATION
SWAB R1 ;GET THE LINE NUMBER IN THE LOWER BYTE
BIC #^C<7>,R1 ;ISOLATE THE LINE NUMBER
MOV R1,SAVLIN ;FILL LOC. FOR ERROR PRINTOUT
TST R4 ;WAS DATA VALID?
BMI 10$ ;BRANCH IF YES
ERROR 23 ;ERROR - DATA VALID NOT SET!
10$: ASL R1 ;ALIGN IT ON A WORD BOUNDARY
BIT #PAR_R,R4 ;PARITY ERROR SHOULD BE SET. IS IT?
BNE 11$ ;IF SO, GO CHECK CHARACTER
MOV DZRBUF,REGIST ;SET UP FOR THE ERROR MESSAGE
MOV R4,R5
BIC #377,R5
BISB TR0(R1),R5 ;GET THE CORRECT CHARACTER
BIS #DVALID!PARER,R5 ;BUILD WHAT WAS EXPECTED
ERROR 6 ;*ERROR- DID NOT GET CORRECT INFORMATION
11$: CMPB TR0(R1),R4 ;CHECK THE CHARACTER. IS IT CORRECT?
BEQ 12$ ;IF SO, GO SET UP NEXT CHARACTER
MOVB TR0(R1),R5 ;LOAD THE CHARACTER FOR ERROR REPORTING
BIC #^C<377>,R5 ;CLEAR SIGN EXTEND
BIC #^C<377>,R4 ;REMOVE THE JUNK FROM R4, THE ACTUAL CHARACTER
ERROR 5 ;DATA ERROR
12$: INC TR0(R1) ;SET UP THE NEXT CHARACTER
INC R3 ;ADD TO THE TOTAL RECEIVED COUNT
CLR COUNT0 ;RESET COUNTERS TO NEXT
CLR COUNT1 ;RECEIVER INTERRUPT
BIT #TIE,@DZCSR ;ARE TRANSMISSIONS DONE?
BNE 13$ ;IF NO, GO RECEIVE SOME MORE
CMP R2,R3 ;ARE ALL CHARACTERS RECEIVED?
BNE 13$ ;IF NO, GO RECEIVE SOME MORE
BIC #RIE,@DZCSR ;DISABLE RECEIVER INTERRUPTS
13$: RTI ;GO BACK TO RECEIVER WAIT LOOP
COUNT0: 0
COUNT1: 0

```

:TRANSMITTER INTERRUPT SERVICE

```

-----
XMTSRV: MOVB @HDZCSR,R1 ;GET THE LINE NUMBER. IS THE TRANSMITTER
BMI 1$ ;REALLY READY? IF SO, GO LOAD THE CHARACTER
MOV SAVLIN,R0 ;ADJUST LOCATION SAVLIN
BIC #^C<7>,R1 ;ISOLATE THE LINE NUMBER
MOV R1,SAVLIN ;FOR ERROR PRINTOUT
ERROR 3 ;*TRANSMITTER NOT READY- FALSE INTERRUPT
MOV R0,SAVLIN ;RESET SAVLIN TO PREVIOUS VALUE
1$: BIC #^C<7>,R1 ;ISOLATE THE LINE NUMBER
ASL R1 ;MAKE SURE IT REFERENCES A WORD BOUNDARY
MOVB TD0(R1),@DZTDR ;LOAD THE CURRENT CHARACTER FOR THIS LINE
INC TD0(R1) ;SET UP NEXT CHARACTER FOR THIS LINE

```

:*****

```
9668 ;*****
9669 024132 023761 001376 001422 ;      CMP      XMTCNT,TD0(R1) ;HAVE WE DONE ALL PATTERNS ON THIS LINE?
9670 024140 001015 ;      BNE      4$ ;IF NOT, KEEP ON TRANSMITTING
9671 024142 012700 000001 ;      MOV      #1,R0 ;SET UP A DESELECTION POINTER
9672 024146 006201 ;      ASR      R1 ;GET THE LINE NUMBER AGAIN
9673 024150 005301 2$: ;      DEC      R1 ;REDUCE THE COUNT. WAS THIS THE LINE?
9674 024152 100402 ;      BMI      3$ ;IF SO, GO DISABLE THE ENABLE BIT FOR IT
9675 024154 006300 ;      ASL      R0 ;MOVE THE POINTER TO THE NEXT LINE
9676 024156 000774 ;      BR       2$ ;GO CHECK THE NEXT LINE
9677 024160 140077 155672 3$: ;      BICB     R0,@DZTCR ;DISABLE THE LINE POINTED TO BY R0
9678 024164 001003 ;      BNE      4$ ;IF MORE LINES ARE ACTIVE, GO CONTINUE TRANSMIT
9679 024166 042777 040000 155646 ;      BIC      #TIE,@DZCSR ;IF NOT, DISABLE TRANSMITTER INTERRUPTS
9680 ;*****
9681 024174 005202 4$: ;      INC      R2 ;UP THE NUMBER OF TRANSMISSIONS (REV. F0)
9682 ;*****
9683 024176 000002 ;      RTI ;RETURN TO THE TIMING LOOP
9684
9685 ; RELATIVE TIME BUILDING ROUTINE
9686 ; -----
9687
9688 024200 012737 000004 001222 BUILD: MOV      #4,$TMP2 ;ROTATE 4 BITS BACK INTO $TMP1
9689 024206 006037 001224 1$: ROR      $TMP3 ;GET THE BITS FROM $TMP3, THE HIGH BYTE
9690 024212 006037 001220 ROR      $TMP1 ;OF THE RELATIVE TIME COUNTER. PUT THEM BACK
9691 024216 005337 001222 DEC      $TMP2 ;INTO $TMP1 USING THE CARRY BIT WITH
9692 ;ROTATE INSTRUCTIONS
9693 024222 001371 BNE      1$ ;REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
9694 024224 000207 RTS      PC ;RETURN TO CALLING TEST
9695
```

```
9697                                     ;RECEIVER SERVICE ROUTINE
9698
9699 024226 105777 155610          RXISR1: TSTB   @DZCSR      ;IS THE RECEIVER REALLY READY?
9700 024232 100401                BMI     1$          ;IF SO, GO SERVICE IT
9701 024234 104004                ERROR   4           ;*ERROR- RECEIVER DONE FLAG ISN'T SET
9702 024236 017704 155604          1$:   MOV     @DZRBUF,R4    ;SAVE THE RECEIVER INFORMATION
9703 024242 100401                BMI     2$          ;IF IT WAS VALID, GO PROCESS IT
9704 024244 104023                ERROR   23          ;ERROR- DATA VALID WASN'T SET
9705 024246 032704 070000          2$:   BIT     #OVRRUN!FRMERR!PARER,R4 ;ARE ANY ERROR FLAGS SET?
9706 024252 001403                BEQ     3$          ;IF NOT, GO CONTINUE PROCESSING
9707 024254 013700 002046          MOV     DZRBUF,R0    ;SET UP FOR ERROR REPORTING
9708 024260 104002                ERROR   2           ;ERROR- RECEIVER ERROR FLAG SET
9709 024262 010401                3$:   MOV     R4,R1      ;COPY THE RECEIVER INFORMATION
9710 024264 000301                SWAB    R1           ;GET THE LINE NUMBER IN THE LOWER BYTE
9711 024266 042701 177770          BIC     #^C<7>,R1    ;ISOLATE THE LINE NUMBER
9712 024272 006301                ASL     R1           ;ALIGN IT ON A WORD BOUNDARY
9713 024274 120461 001442          CMPB   R4,TR0(R1)   ;IS THE CHARACTER WHAT IT SHOULD BE?
9714 024300 001413                BEQ     4$          ;IF SO,GO CONTINUE PROCESSING
9715 024302 116105 001442          MOVB   TR0(R1),R5   ;GET WHAT WAS EXPECTED FOR ERROR REPORTING
9716 024306 042705 177400          BIC     #^C<377>,R5  ;ELIMINATE PROPAGATED SIGN
9717 024312 042704 177400          BIC     #^C<377>,R4  ;ISOLATE THE ACTUAL CHARACTER
9718 024316 010137 001372          MOV     R1,SAVLIN   ;GET THE LINE NUMBER OF THE RECEIVER ERROR
9719 024322 006237 001372          ASR    SAVLIN       ;ALIGN IT CORRECTLY FOR REPORTING
9720 024326 104005                ERROR   5           ;*DATA ERROR
9721 024330 005261 001442          4$:   INC     TR0(R1)   ;SET UP THE NEXT EXPECTED CHARACTER
9722 024334 005203                INC     R3           ;INCREMENT THE COUNT OF RECEIVED CHARACTERS
9723 024336 032761 000020 001442  BIT     #20,TR0(R1) ;HAVE ALL CHARACTERS BEEN RECEIVED?
9724 024344 001402                BEQ     5$          ;IF NOT, GO RECEIVE SOME MORE
9725 024346 020203                CMP     R2,R3       ;HAVE WE RECEIVED ALL CHARACTERS?
9726 024350 001401                BEQ     6$          ;IF SO,GO DETERMINE THE TIMING
9727 024352 000002                5$:   RTI           ;GO CONTINUE TIMING AND ALLOW INTERRUPTS
9728 024354 004737 024200          6$:   JSR     PC,BUILD ;GET THE RELATIVE TIME (SIGNIFICANT BITS)
9729
9730 024360 013700 024440          MOV     OFFSET,R0   ;GET POINTER
9731 024364 013760 001220 002102  MOV     $TMP1,TMTBL(R0) ;SAVE THIS TEST'S TIME
9732 024372 005737 024440          TST    OFFSET      ;FIRST TEST?
9733 024376 001414                BEQ     7$          ;IF NOT, GO CHECK THE TIME
9734 024400 005740                TST    -(R0)       ;POINT TO THE PREVIOUS TIME TAKEN
9735 024402 026037 002102 001220  CMP     TMTBL(R0),$TMP1 ;IS THIS TIME WHAT IT SHOULD BE?
9736 024410 101007                BHI     7$          ;IF SO, GO TO THE NEXT TEST
9737 024412 016005 002102          MOV     TMTBL(R0),R5 ;PLACE WHAT WAS EXPECTED IN R5
9738 024416 010137 001372          MOV     R1,SAVLIN   ;GET THE LINE NUMBER OF THE RECEIVER
9739 024422 006237 001372          ASR    SAVLIN       ;MAKE SURE IT'S THE LINE NUMBER
9740 024426 104021                ERROR   21          ;TIMING ERROR
9741 024430 042777 000140 155404  7$:   BIC     #RIE!MSENAB,@DZCSR ;DISABLE THE DEVICE
9742 024436 000002                RTI           ;RETURN TO THE PROGRAM
9743 024440 000000                OFFSET: 0
```

```

9745          ;DZ11 ECHO/CABLE TEST
9746
9747          ;*STARTING PROCEDURE
9748          ;*LOAD PROGRAM
9749          ;*LOAD ADDRESS 000210
9750          ;*PRESS START
9751          ;*PROGRAM WILL TYPE DZ11 ECHO/CABLE TEST
9752          ;*PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
9753          ;*TYPE IN E OR C RESPECTIVELY
9754          ;*PROGRAM WILL TYPE 'VECTOR ADDRESS-'
9755          ;*TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
9756          ;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
9757          ;*PROGRAM WILL TYPE 'CONTROL REGISTER ADDRESS-'
9758          ;*TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
9759          ;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
9760          ;*PROGRAM WILL TYPE 'LINE NUMBER-'
9761          ;*TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
9762          ;*, FOLLOWED BY <CARRIAGE RETURN>
9763          ;*PROGRAM WILL TYPE 'BAUD RATE-'
9764          ;*TYPE IN THE BAUD RATE OF THE DZ11 TERMINAL
9765          ;*, FOLLOWED BY <CARRIAGE RETURN>
9766          ;*THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL
9767          ;*
9768          ;*      50
9769          ;*      75
9770          ;*      110
9771          ;*      135      (ROUNDED OFF 134.5)
9772          ;*      150
9773          ;*      300
9774          ;*      600
9775          ;*      1200
9776          ;*      1800
9777          ;*      2000
9778          ;*      2400
9779          ;*      3600
9780          ;*      4800
9781          ;*      7200
9782          ;*      9600
9783          ;*ALL OTHERS ARE REJECTED
9784          ;*PROGRAM WILL TYPE 'ECHO' OR 'CABLE TEST' TO INDICATE THAT TESTING HAS STARTE
9788
9789
9790          ;PROGRAM INITIALIZATION
9791          ;LOCK OUT INTERRUPTS
9792          ;SET UP PROCESSOR STACK
9793          ;SET UP POWER FAIL VECTOR
9794          ;CLEAR PROGRAM FLAGS AND COUNTS
9795
9796 024442 012706 001120          XSTART: MOV      #STACK,SP      ;SET UP PROCESSOR STACK
9797 024446 106427 000340          MTPS     #PR7        ;LOCK OUT INTERRUPTS
9798 024452 012737 024442 001126  MOV      #XSTART,$LPADR ;SET UP IN CASE OF POWER FAIL
9799 024460 005037 026636          CLR      STFLG       ;CLEAR TEST START FLAG
9800 024464 005037 001242          CLR      $PASS       ;CLEAR PASS COUNT
9801 024470 005037 001132          CLR      $ERTTL      ;CLEAR ERROR COUNT
9802 024474 105037 001123          CLR     $ERFLG       ;CLEAR ERROR FLAG
9803 024500 005037 026642          CLR      LAST        ;CLEAR LAST ERROR PC

```

9804	024504	032777	000001	154446	VEC1:	BIT	#SW00,@SWR		:IF SW00=1, GET NEW VECTOR
9805	024512	001465				BEQ	OTHER		:AND CSR
9806	024514	012701	000300		VEC2:	MOV	#300,R1		
9807	024520	012702	000302			MOV	#302,R2		
9808	024524	010221			1\$:	MOV	R2,(R1)+		:RESTORE TRAPCATCHER
9809	024526	005022				CLR	(R2)+		:IN FLOATING VECTOR AREA
9810	024530	022122				CMP	(R1)+,(R2)+		:UPDATE THE POINTERS
9811	024532	020127	001000			CMP	R1,#1000		
9812	024536	001372				BNE	1\$		
9813	024540	104403				INSTR			:INPUT ADDRESS OF DEVICE VECTOR
9814	024542	026670				MVECTOR			:MESSAGE 'VECTOR ADDRESS-''
9815	024544	104405				PARAM			:CONVERT STRING TO OCTAL
9816	024546	000300				300			:LOW LIMIT
9817	024550	000770				770			:HIGH LIMIT
9818	024552	002072				DZRIV			:LOCATIONS TO BE FILLED
9819	024554	003			.BYTE	3			:LSB MASK
9820	024555	004			.BYTE	4			:NUMBER OF LOCATIONS
9821	024556	104403				INSTR			:INPUT ADDRESS OF DEVICE CSR
9822	024560	026712				MREGAD			:MESSAGE 'CONTROL REGISTER ADDRESS-''
9823	024562	104405				PARAM			:CONVERT STRING TO OCTAL
9824	024564	160000				160000			:LOW LIMIT
9825	024566	163700				163700			:HIGH LIMIT
9826	024570	002042				DZCSR			:LOCATIONS TO BE FILLED
9827	024572	007			.BYTE	7			:LSB MASK
9828	024573	001			.BYTE	1			:NUMBER OF LOCATIONS
9829	024574	013737	002042	002046		MOV	DZCSR,DZRBUF		:BEGIN BUILDING DEVICE ADDRESSES
9830	024602	062737	000002	002046		ADD	#2,DZRBUF		:FORM THE READ BUFFER ADDRESS
9831	024610	013737	002046	002052		MOV	DZRBUF,DZLPR		:REMEMBER THAT THIS IS ALSO LINE PARAMETER REG.
9832	024616	013737	002046	002056		MOV	DZRBUF,DZTCR		:BEGIN BUILDING TRANSMITTER CONTROL REGISTER
9833	024624	062737	000002	002056		ADD	#2,DZTCR		:FORM THE TRANSMITTER CONTROL REGISTER POINTER
9834	024632	013737	002056	002060		MOV	DZTCR,HDZTCR		
9835	024640	005237	002060			INC	HDZTCR		
9836	024644	013737	002056	002066		MOV	DZTCR,DZTDR		:BEGIN FORMING TRANSMITTER DATA REGISTER
9837	024652	062737	000002	002066		ADD	#2,DZTDR		:FORM THE TRANSMITTER DATA REGISTER
9838	024660	013737	002066	002062		MOV	DZTDR,DZMSR		
9839	024666	032777	000002	154264	OTHER:	BIT	#SW01,@SWR		:RESELECT OF TEST?
9840	024674	001427				BEQ	XBEGIN		:IF NOT, SKIP ASKING WHICH ONE
9841	024676	104403				INSTR			:INPUT WHICH TEST YOU ARE RUNNING
9842	024700	027076				MWHICH			:ECHO OR CABLE
9843	024702	104416				PAWCH			:SET FLAG
9844	024704	026634				WCHFLG			:THIS FLAG
9845	024706	104403			BAUD:	INSTR			:INPUT BAUD RATE
9846	024710	027020				MSPEED			:MESSAGE 'BAUD RATE-''
9847	024712	104415				PARMD			:CONVERT DECIMAL STRING TO OCTAL
9848	024714	000062				50			:LOW LIMIT
9849	024716	022600				9600			:HIGH LIMIT
9850	024720	026652				LINESP			:LOCATION TO BE FILLED
9851	024722	000			.BYTE	0			:LSB MASK
9852	024723	001			.BYTE	1			:NUMBER OF LOCATIONS
9853	024724	104413			LINEX:	DEVICE.CLR			:CLEAR DEVICE
9854	024726	005037	026636			CLR	STFLG		:CLEAR PROGRAM START FLAG
9855	024732	104403				INSTR			:INPUT LINE NUMBER
9856	024734	027010				MLINE			:MESSAGE 'LINE NUMBER-''
9857	024736	104405				PARAM			:CONVERT STRING TO OCTAL
9858	024740	000000				0			:LOW LIMIT
9859	024742	000007				7			:HIGH LIMIT

```

9860 024744 001372          SAVLIN          ;LOCATION TO BE FILLED
9861 024746      000      .BYTE 0          ;LSB MASK
9862 024747      001      .BYTE 1          ;NUMBER OF LOCATIONS
9863 024750 004537 026440      JSR R5,SET
9864
9865 024754 106427 000340      XBEGIN: MTPS #PR7          ;LOCK OUT INTERRUPTS
9866 024760 012706 001120      MOV #STACK,SP          ;SET UP PROCESSOR STACK
9867 024764 005037 026640      CLR LOCKUP          ;CLEAR TIMEOUT
9868 024770 005737 026634      TST WCHFLG          ;ECHO OR CABLE TEST ?
9869 024774 001413          BEQ 2$          ;ECHO
9870 024776 012737 025512 001126      MOV #TEST2,$LPADR          ;CABLE TEST
9871 025004 005737 026636      TST STFLG          ;ARE YOU LOOPING ?
9872 025010 001017          BNE 1$          ;YES
9873 025012 005137 026636      COM STFLG          ;NO
9874 025016 104402 027171      TYPE ,MCABLE          ;TYPE CABLE TEST
9875 025022 000412          BR 1$
9876 025024 012737 025054 001126 2$:      MOV #TEST1,$LPADR          ;SET UP ECHO TEST
9877 025032 005737 026636      TST STFLG          ;ARE YOU LOOPING ?
9878 025036 001004          BNE 1$          ;YES
9879 025040 005137 026636      COM STFLG          ;NO
9880 025044 104402 027144      TYPE ,MTERM          ;TYPE ECHO TEST
9881 025050 000177 154052      1$:      JMP @$LPADR          ;START TESTING
9882          ;THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
9883          ; (IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
9884          ; ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
9885
9886 025054 104413          TEST1: DEVICE.CLR          ;CLEAR DZ11
9887 025056 012737 000001 001122      MOV #1,$STSTNM
9888 025064 013777 026660 154764      MOV NUMTCR,@DZTCR          ;SET TCR BIT
9889 025072 013737 026656 001366      MOV NUMLIN,PAR          ;SET PARAMETERS
9890 025100 053737 026654 001366      BIS SPEED,PAR          ;SET BAUD RATE
9891 025106 013777 001366 154736      MOV PAR,@DZLPR          ;LOAD PARAM.
9892 025114 012777 000040 154720      MOV #MSENAB,@DZCSR          ;SET SCAN# ENABLE
9893 025122 005004          CLR R4
9894 025124 012705 027206          MOV #MQUICK,R5          ;SET MESSAGE BUFFER
9895 025130 005777 154706      3$:      TST @DZCSR          ;TRDY?
9896 025134 100404          BMI 2$          ;BR IF YES
9897 025136 104414          DELAY          ;WAIT
9898 025140 005304          DEC R4
9899 025142 001372          BNE 3$
9900 025144 104003          ERROR 3          ;NO TRDY SET! WHY?
9901 025146 005004          CLR R4          ;RESET COUNTER TO 0
9902 025150 112577 154712      2$:      MOV#B (R5)+,@DZTDR          ;LOAD CHAR
9903 025154 001365          BNE 3$
9904 025156 004737 007652          JSR PC,SERV.G          ;<^G>?
9905 025162 122777 000377 153770      CMP#B #377,@SWR          ;RE-DO QUICK BROWN?
9906 025170 001731          BEQ TEST1          ;BR IF REPEAT PATTERN
9907 025172 104413          DEVICE.CLR
9908 025174 106427 000340      MTPS #PR7          ;LOCK OUT INTERRUPTS
9909 025200 012737 026150 001360      MOV #XEOP,NEXT
9910 025206 104413          DEVICE.CLR
9911 025210 013737 026656 001366      MOV NUMLIN,PAR          ;SELECT LINE # & SET INTERRUPT ENABLE
9912 025216 053737 026654 001366      BIS SPEED,PAR          ;SET LINE SPEED AND
9913          ;CHARACTER LENGTH (TRANS. & REC.)
9914 025224 052737 010000 001366      BIS #RCVON,PAR          ;MAKE SURE RECEIVER IS TURNED ON
9915 025232 013777 001366 154612      MOV PAR,@DZLPR          ;LOAD THE LINE PARAMETER REGISTER

```

```

9916 025240 012777 025314 154624      MOV    #INTSVC,@DZRIV ;SET UP INTERRUPT SERVICE
9917 025246 013777 026662 154620      MOV    PRIO,@DZRIS   ;AND LEVEL
9918 025254 106437 027534                MTPS   @#LESS1      ;ALLOW INTERRUPTS
9919 025260 012777 000140 154554      MOV    #RIE!MSENAB,@DZCSR ;SET RECEIVER INTERRUPT ENABLE
9920 025266 104402 027036                TYPE  ,MCHAR        ;TYPE 'ANY CHARACTER'
9921 025272 105777 153666      1$:   TSTB   @$TKS      ;IF SOMBODY HITS A KEY- GET NEW LINE #
9922 025276 100375                BPL    1$           ;LOOP HERE
9923 025300 005777 153662      TST    @$TKB        ;CLEAR CHAR
9924 025304 004737 007652      JSR    PC,SERV.G    ;MAKE SURE IT WASN'T <^G>
9925 025310 000137 024724      JMP    LINEX        ;

```

```

9926
9927
9928
9929 025314 105777 154522      INTSVC: TSTB   @DZCSR    ;THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
9930 025320 100401                BMI    .+4          ;TEST REC. FLAG
9931 025322 104004                ERROR  4            ;ERROR - INTERRUPT NOT CAUSED BY FLAG
9932 025324 017737 154516 026664      MOV    @DZRBUF,RECDAT
9933 025332 100401                BMI    .+4
9934 025334 104023                ERROR  23          ;NON- VALID CHARACTER
9935 025336 032737 020000 026664      BIT    #BIT13,RECDAT ;CHECK FOR FRAMING ERROR
9936 025344 001401                BEQ    .+4          ;BR IF NO ERROR
9937 025346 104025                ERROR  25          ;EITHER SOMBODY HIT THE
9938                                ;'BREAK KEY' OR YOU HAVE AN ERROR!
9939 025350 113737 026664 026666      MOVB   RECDAT,TBUF  ;MOVE CHARACTER TO OUTPUT AREA
9940 025356 113737 026664 011272      MOVB   RECDAT,INBUF ;MOVE CHARACTER TO CHECK FOR ^C
9941 025364 042737 177600 011272      BIC    #^C<177>,INBUF ;STRIP JUNK PLUS PARITY
9942 025372 042737 174377 026664      BIC    #174377,RECDAT ;SAVE ONLY LINE NUMBER
9943 025400 000337 026664                SWAB   RECDAT
9944 025404 023737 001372 026664      CMP    SAVLIN,RECDAT ;DOES THE LINE # COMPARE?
9945 025412 001401                BEQ    .+4
9946 025414 104015                ERROR  15          ;*WRONG LINE NUMBER
9947 025416 012777 000040 154416      MOV    #MSENAB,@DZCSR ;START THE TRANSMITTERS SCANNER
9948 025424 123727 011272 000003      CMPB   INBUF,#3     ;IS IT A ^C ?
9949 025432 001004                BNE    1$          ;NO
9950 025434 104413                DEVICE.CLR
9951 025436 012716 026150      MOV    #XEOP,(SP)   ;CRUNCH STACK
9952 025442 000002                RTI
9953 025444 005003      1$:   CLR    R3         ;INITIALIZE DELAY
9954 025446 013777 026660 154402      MOV    NUMTCR,@DZTCR ;ENABLE THE LINE
9955 025454 005777 154362      10$:  TST    @DZCSR     ;TRANSMITTER READY?
9956 025460 100403                BMI    2$          ;IF YES BRANCH
9957 025462 005203                INC    R3          ;INCREMENT DELAY
9958 025464 001373                BNE    10$         ;DELAY DONE?
9959 025466 104003                ERROR  3           ;TRANSMIT READY NOT SET!
9960 025470 113777 026666 154370 2$:   MOVB   TBUF,@DZTDR  ;TRANSMIT THE CHARACTER
9961 025476 012777 000140 154336      MOV    #RIE!MSENAB,@DZCSR ;RESTART THE RECEIVER
9962 025504 005077 154346      CLR    @DZTCR      ;CLEAR TCR BIT
9963 025510 000002                RTI

```

```

9964
9965
9966
9967
9968
9969 025512 106427 000340                ;THIS TEST TRANSMITS A BINARY COUNT PATTERN
9970 025516 012737 000002 001122      ;VIA INTERRUPT MODE TO THE RECEIVER
9971 025524 012737 026150 001360      ;...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
TEST2: MTPS   #PR7      ;DISABLE INTERRUPTS
        MOV    #2,$STNM
        MOV    #XEOP,NEXT

```

9972	025532	104413				DEVICE.CLR	:
9973						:*TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE	
9974						:*WILL BRING UP 'CO' AND 'RING' FOR THE SAME LINE	
9975						:*THE DIST PNL MUST HAVE JUMPER FROM DTR TO RQST TO SEND	
9976						:*IN ORDER FOR THIS TEST TO WORK!	
9977	025534	012737	025542	001362		MOV #1\$,LOCK	:LOOP
9978	025542	113777	026660	154310	1\$:	MOV NUMTCR,@HDZTCR	:SET DTR
9979	025550	005005				CLR R5	:
9980	025552	153705	026660			BISB NUMTCR,R5	:BUILD EXPECTED
9981	025556	000305				SWAB R5	:PUT IN HIGH BYTE
9982	025560	153705	026660			BISB NUMTCR,R5	:
9983	025564	104414				DELAY	:WAIT FOR CABLE DELAY
9984	025566	017704	154270			MOV @DZMSR,R4	:READY MODEM BITS
9985	025572	020504				CMP R5,R4	:ARE THEY OK?
9986	025574	001401				BEQ 2\$:BR IF YES
9987	025576	104022				ERROR 22	:IS THE TEST CONNECTOR ON?
9988							:HAS RIGHT LINE BEEN SELECTED?
9989							:IF SO- YOU HAVE A PROBLEM!
9990							:MODEM BITS NOT RIGHT
9991	025600	104401			2\$:	SCOPI	:LOOP
9992	025602	104413			3\$:	DEVICE.CLR	:INIT DZ11
9993	025604	013737	026654	001366		MOV SPEED,PAR	:SET LINE SPEED
9994	025612	053737	026656	001366		BIS NUMLIN,PAR	:SELECT LINE # & REC. INTERRUPT ENABLE
9995	025620	052737	010000	001366		BIS #RCVON,PAR	:ENABLE THE RECEIVER FOR THIS LINE
9996	025626	052777	040140	154206		BIS #TIE!RIE!MSENAB,@DZCSR	:SET TRANSMITTER INTERRUPT ENABLE
9997	025634	012777	025750	154230		MOV #INTREC,@DZRIV	:SET UP INTR SERVICE
9998	025642	013777	026662	154224		MOV PRIO,@DZRI5	:SET UP LEVEL
9999	025650	012777	026130	154220		MOV #INTRAN,@DZTIV	:SET UP INTR SERVICE
10000	025656	013777	026662	154214		MOV PRIO,@DZTIS	:SET UP LEVEL
10001	025664	005001				CLR R1	:RX DATA POINTER- SET TO 0
10002	025666	005002				CLR R2	:TX DATA POINTER- SET TO 0
10003	025670	013777	001366	154154		MOV PAR,@DZLPR	:SET THE PARAMETERS AND TURN ON RECEIVER
10004	025676	106437	027534			MTPS @#LESS1	:ALLOW INTERRUPTS
10005	025702	013777	026660	154146		MOV NUMTCR,@DZTCR	:SET UP TCR BIT
10006							
10007							:YOU RETURN HERE AFTER EVERY RECEIVER INTERRUPT
10008	025710	105777	153250		SPIN:	TSTB @STKS	:IF SOMEBODY HITS A KEY- GET A NEW LINE #
10009	025714	100006				BPL 1\$:BR IF NO KEY HIT
10010	025716	005777	153244			TST @STKB	:CLEAR CHAR
10011	025722	004737	007652			JSR PC,SERV.G	:MAKE SURE IT WASN'T <^G>
10012	025726	000137	024724			JMP LINEX	:SW02=1
10013	025732	005237	026640		1\$:	INC LOCKUP	:INC TIMEOUT FLAG
10014	025736	001364				BNE SPIN	:IF NOT 0 RETURN SPINNING
10015	025740	104011				ERROR 1\$:*RECEIVER FAILED TO INTERRUPT CHECK CABLE/TERMINATOR
10016	025742	104413			QUITS:	DEVICE.CLR	
10017	025744	000137	026150			JMP XEOP	:CALL FOR END OF PASS
10018	025750	005037	026640		INTREC:	CLR LOCKUP	:CLEAR TIMEOUT FLAG
10019	025754	105777	154062			TSTB @DZCSR	:TEST REC DONE
10020	025760	100401				BMI +4	:YES
10021	025762	104004				ERROR 4	:*FALSE INTERRUPT
10022	025764	017737	154056	026664		MOV @DZRBUF,RECDAT	:SAVE WORD
10023	025772	100401				BMI +4	
10024	025774	104023				ERROR 23	:*NON VALID CHARACTER
10025	025776	032737	040000	026664		BIT #BIT14,RECDAT	:DATA OVERRUN ?
10026	026004	001401				BEQ +4	:NO
10027	026006	104024				ERROR 24	:*YES

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052) 24-JUN-81 09:45

24-JUN-81 09:45
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0105

10028	026010	032737	020000	026664	BIT	#BIT13,RECDAT	:FRAMING ERROR ?
10029	026016	001401			BEQ	.+4	:NO
10030	026020	104025			ERROR	25	:*YES
10031	026022	032737	010000	026664	BIT	#BIT12,RECDAT	:PARITY ERROR ?
10032	026030	001401			BEQ	.+4	:NO
10033	026032	104026			ERROR	26	:*YES
10034	026034	110105			MOVB	R1,R5	:SET EXPECTED
10035	026036	042705	177400		BIC	#^C<377>,R5	:CLEAR HIGH BYTE
10036	026042	113704	026664		MOVB	RECDAT,R4	:GET FOUND
10037	026046	042704	177400		BIC	#^C<377>,R4	:CLEAR HIGH BYTE
10038	026052	020504			CMP	R5,R4 ;OK?	
10039	026054	001401			BEQ	.+4	
10040	026056	104005			ERROR	5	:DATA ERROR
10041	026060	042737	174377	026664	BIC	#174377,RECDAT	:SAVE ONLY LINE NUMBER
10042	026066	000337	026664		SWAB	RECDAT	
10043	026072	023737	001372	026664	CMP	SAVLIN,RECDAT	:DOES THE LINE # COMPARE ?
10044	026100	001401			BEQ	.+4	:YES
10045	026102	104015			ERROR	15	:*WRONG LINE #
10046	026104	120127	000377		CMPB	R1,#377	:LAST CHARACTER ?
10047	026110	001003			BNE	1\$:NO
10048	026112	012716	025742		MOV	#QUITS,(SP)	:CRUNCH STACK
10049	026116	000403			BR	2\$	
10050	026120	105201			1\$: INCB	R1	:UPDATE EXPECTED DATA
10051	026122	012716	025710		MOV	#SPIN,(SP)	:CRUNCH STACK
10052	026126	000002			2\$: RTI		
10053							
10054	026130	005777	153706		INTRAN: TST	@DZCSR ;TEST TRANSMIT	
10055	026134	100401			BMI	.+4	
10056	026136	104003			ERROR	3	:*FALSE INTERRUPT
10057	026140	110277	153722		MOVB	R2,@DZTDR	:TRANSMIT A CHARACTER
10058	026144	105202			INCB	R2	:UPDATE TX DATA
10059	026146	000002			RTI	;RETURN	

```

10061
10062
10063
10064
10065 026150 104402
10066 026152 026746
10067 026154 005037 026642
10068 026160 105037 001123
10069 026164 000137 024754
10070
10071
10072 026170 011605
10073 026172 012537 026354
10074 026176 012537 026356
10075 026202 012537 026360
10076 026206 112537 026362
10077 026212 112537 026363
10078 026216 010516
10079 026220 005005
10080 026222 012704 011272
10081 026226 122714 000015
10082 026232 001424
10083 026234 121427 000060
10084 026240 002421
10085 026242 121427 000071
10086 026246 003016
10087 026250 142714 000060
10088 026254 005002
10089 026256 152402
10090 026260 060205
10091 026262 122714 000015
10092 026266 001410
10093 026270 006305
10094 026272 010502
10095 026274 006305
10096 026276 006305
10097 026300 060205
10098 026302 000754
10099 026304 104404
10100 026306 000744
10101
10102
10103
10104 026310 020537 026356
10105 026314 101373
10106 026316 020537 026354
10107 026322 103770
10108 026324 133705 026362
10109 026330 001365
10110
10111
10112
10113 026332 013704 026360
10114 026336 010524
10115 026340 062705 000002
10116 026344 105337 026363

```

```

;END OF PASS
;RESTART TEST
XEOP: TYPE ;TYPE NAME OF TEST
MPASS
CLR LAST ;CLEAR LAST ERROR PC
CLRB $ERFLG ;CLEAR ERROR FLAG
RSTRT: JMP XBEGIN

;CONVERT DECIMAL ASCII STRING TO OCTAL
.PARMD: MOV (SP),R5
MOV (R5)+,6$
MOV (R5)+,7$
MOV (R5)+,8$
MOVB (R5)+,9$
MOVB (R5)+,10$
MOV R5,(SP)
2$: CLR R5
MOV #INBUF,R4
CMPB #15,(R4)
BEQ 3$
1$: CMPB (R4),#'0
BLT 3$
CMPB (R4),#'9
BGT 3$
BICB #'0,(R4)
CLR R2
BISB (R4)+,R2
ADD R2,R5
CMPB #15,(R4)
BEQ 4$
ASL R5 ;X2
MOV R5,R2 ;SAVE X2
ASL R5 ;X4
ASL R5 ;X8
ADD R2,R5 ;TIMES 10
BR 1$
3$: INSTER
BR 2$

;TEST TO SEE IF NUMBER IS WITHIN LIMITS
4$: CMP R5,7$
BHI 3$
CMP R5,6$
BLO 3$
BITB 9$,R5
BNE 3$

;STORE NUMBER AT SPECIFIED ADDRESS
5$: MOV 8$,R4
MOV R5,(R4)+
ADD #2,R5
DECB 10$

```

```

10117 026350 001372          BNE      5$
10118 026352 000002          RTI
10119 026354 000000          6$:      0
10120 026356 000000          7$:      0
10121 026360 000000          8$:      0
10122 026362      000          9$:      .BYTE 0
10123 026363      000          10$:     .BYTE 0
10124
10125
10126          ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
10127          ;BUFFER TO THE CHARACTERS 'E' AND 'C'.
10128          ;IF THE CHARACTER IS 'E' CLEAR THE FLAG
10129          ;IF THE CHARACTER IS 'C' SET THE FLAG
10130
10131 026364 017605 000000      .PAWCH:MOV @ (SP),R5
10132 026370 142737 000040 011272      BICB    #40,INBUF      ;SET FOR LOWER CASE INPUT
10133 026376 122737 000105 011272      CMPB    #'E',INBUF    ;IS IT 'E' ?
10134 026404 001002          BNE      1$
10135 026406 105015          CLR      (R5)          ;000
10136 026410 000406          BR       2$
10137 026412 122737 000103 011272  1$:     CMPB    #'C',INBUF    ;IS IT 'C' ?
10138 026420 001005          BNE      3$
10139 026422 112715 177777      MOV      #-1,(R5)     ;3177
10140 026426 062716 000002      2$:     ADD      #2,(SP)
10141 026432 000002          RTI
10142 026434 104404          3$:     INSTER          ;RETRY
10143 026436 000752          BR       .PAWCH
10144
10145
10146
10147          ;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
10148          ;LINE NUMBER (SAVLIN) FOR DZLPR, DZTCR AND DZCSR
10149          ;REGISTER USAGE.
10150
10151 026440 013737 001372 026656  SET:    MOV      SAVLIN,NUMLIN ;SAVE SAVLIN
10152 026446 013700 001372      XTCRO:  MOV      SAVLIN,R0   ;COPY THE LINE NUMBER FOR LOOP CONTROL
10153 026452 005037 026660          CLR      NUMTCR          ;SET A DEFAULT OF LINE 0 OR NO LINES
10154 026456 012702 000001          MOV      #1,R2          ;SET A BIT POINTER TO THE FIRST LINE
10155 026462 005300          XTCR1:  DEC      R0          ;REDUCE THE INDICATOR.IS IT MINUS YET?
10156 026464 100402          BMI      SET1           ;IF SO, R2 POINTS TO THE RIGHT LINE
10157 026466 006302          ASL      R2            ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
10158 026470 000774          BR       XTCR1         ;GO SEE IF THIS LINE IS THE ONE
10159 026472 012701 026534      SET1:   MOV      #TABLE2,R1
10160 026476 010237 026660          MOV      R2,NUMTCR     ;COPY THE CORRECT BIT POINTER
10161 026502 022137 026652      1$:     CMP      (R1)+,LINESP
10162 026506 001407          BEQ      2$
10163 026510 005721          TST      (R1)+        ;IS IT THE END OF TABLE?
10164 026512 001373          BNE      1$           ;NO
10165 026514 104402 026762          TYPE    ,MINVAL       ;INVALID BAUD RATE,BEGIN AGAIN
10166 026520 012705 024706          MOV      #BAUD,R5     ;JUMP TO BAUD THRU R5
10167 026524 000402          BR       3$
10168 026526 011137 026654      2$:     MOV      (R1),SPEED  ;SET UP BAUD RATE
10169 026532 000205          3$:     RTS      R5
10170
10171
10172

```

```

10173                                     ;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR)
10174 026534 000062 TABLE2: .WORD 50. ;50 BAUD
10175 026536 010070 .WORD 10070 ;
10176 026540 000113 .WORD 75. ;75 BAUD
10177 026542 010470 .WORD 10470 ;
10178 026544 000156 .WORD 110. ;110 BAUD
10179 026546 011070 .WORD 11070 ;TWO STOP BITS
10180 026550 000207 .WORD 135. ;134.5 BAUD
10181 026552 011470 .WORD 11470 ;TWO STOP BITS
10182 026554 000226 .WORD 150. ;150 BAUD
10183 026556 012070 .WORD 12070 ;TWO STOP BITS
10184 026560 000454 .WORD 300. ;300 BAUD
10185 026562 012430 .WORD 12430 ;ONE STOP BIT
10186 026564 001130 .WORD 600. ;600 BAUD
10187 026566 013030 .WORD 13030 ;ONE STOP BIT
10188 026570 002260 .WORD 1200. ;1200 BAUD
10189 026572 013430 .WORD 13430 ;ONE STOP BIT
10190 026574 003410 .WORD 1800. ;1800 BAUD
10191 026576 014030 .WORD 14030 ;ONE STOP BIT
10192 026600 003720 .WORD 2000. ;2000 BAUD
10193 026602 014430 .WORD 14430 ;ONE STOP BIT
10194 026604 004540 .WORD 2400. ;2400 BAUD
10195 026606 015030 .WORD 15030 ;ONE STOP BIT
10196 026610 007020 .WORD 3600. ;3600 BAUD
10197 026612 015430 .WORD 15430 ;ONE STOP BIT
10198 026614 011300 .WORD 4800. ;4800 BAUD
10199 026616 016030 .WORD 16030 ;ONE STOP BIT
10200 026620 016040 .WORD 7200. ;7200 BAUD
10201 026622 016430 .WORD 16430 ;ONE STOP BIT
10202 026624 022600 .WORD 9600. ;9600 BAUD
10203 026626 017070 .WORD 17070 ;
10204 026630 177777 000000 .WORD -1,0 ;TABLE TERMINATOR
10205
10206
10207 026634 000000 WCHFLG:0 ;ECHO OR CABLE FLAG
10208 026636 000000 STFLG: 0 ;PROGRAM START FLAG
10209 026640 000000 LOCKUP: 0 ;TIMEOUT FLAG
10210 026642 000000 LAST: 0 ;LAST ERROR PC
10211 026644 000000 TDATA: 0
10212 026646 000000 RDATA: 0
10213 026650 000000 BYTCNT: 0
10214 026652 000156 LINESP: 110. ;DEFAULT BAUD RATE
10215 026654 006307 SPEED: 6307 ;DEFAULT 110 BAUD, 8 BITS/CHAR,
10216 ;FDX, 2 STOP BITS
10217 026656 000100 NUMLIN: 100 ;DEFAULT VALUE, REC. INTERRUPT ENABLED
10218
10219 026660 000001 NUMTCR: 1 ;DEFAULT VALUE, TCR BIT 0
10220 026662 000240 PRIO: 240 ;DEFAULT DEVICE PRIORITY 5
10221 026664 000000 RECDAT: 0
10222 026666 000000 TBUF: 0
10226 026670 053200 041505 047524 MVECTO: .ASCIZ <200>/VECTOR ADDRESS- /
10227 026712 041600 047117 051124 MREGAD: .ASCIZ <200>/CONTROL REGISTER ADDRESS- /
10228 026746 050200 051501 020123 MPASS: .ASCIZ <200>/PASS DONE./
10229 026762 044600 053116 046101 MINVAL: .ASCIZ <200>/INVALID BAUD RATE - /
10230 027010 046200 047111 035105 MLINE: .ASCIZ <200>/LINE: /
10231 027020 041200 052501 020104 MSPEED: .ASCIZ <200>/BAUD RATE - /

```

```

10232 027036 052200 050131 020105 MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZ11 TERMINAL /
10233 027076 053600 044510 044103 MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /
10234 027144 052200 051105 044515 MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /
10235 027171 200 040503 046102 MCABLE: .ASCIZ <200>/CABLE TEST /
10236 027206 006777 177777 177412 MQUICK: .ASCII <377><15><377><377><12><377><377>
10237 027215 124 042510 050440 .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/
10238 027312 006777 177777 177412 .ASCII <377><15><377><377><12><377><377><377><0>
10239 027324 .EVEN

```

```

10243 :*****
10244 :UTILITIES
10245 :*****
10246
10247

```

```

10248 027324 006337 027532 ;THIS UTILITY CALCULATES PRIORITY LEVEL,SETS UP CSR'S,SETS UP VECTORS.
10249 027330 006337 027532 DZLEV: ASL DZPRT ;BUILD PRIORITY IN THIS LOCATION
10250 027334 006337 027532 ASL DZPRT ;USING ARITHMETIC SHIFTS, ROTATE
10251 027340 006337 027532 ASL DZPRT ; THE PRIORITY LEVEL PAST
10252 027344 006337 027532 ASL DZPRT ; THE BIT POSITIONS CORRE-
10253 027350 013737 027532 027534 MOV DZPRT,LESS1 ;MOVE THIS TO LESS1
10254 027356 162737 000001 027534 SUB #1,LESS1 ;CREATE THE NEXT LOWEST PRIORITY
10255 027364 042737 000037 027534 BIC #37,LESS1 ;INSURE THAT THE TNZVC BITS ARE CLEAR
10256 027372 013700 002072 MOV DZRIV,RO ;PLACE THE BASE VECTOR ADDRESS IN RO
10257 027376 062700 000002 ADD #2,RO ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
10258 027402 010037 002074 MOV RO,DZRRIS ;STORE IT HERE
10259 027406 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
10260 027412 010037 002076 MOV RO,DZTIV ;STORE IT HERE
10261 027416 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
10262 027422 010037 002100 MOV RO,DZTIS ;STORE IT HERE
10263
10264
10265

```

```

10266 027426 013700 001310 ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZ11. $BASE IS THE BASE ADDRESS
10267 027432 010037 002042 ;OF THE DEVICE
10268 027436 005200 MOV $BASE,RO ;COPY THE ADDRESS BEING LOADED
10269 027440 010037 002044 MOV RO,DZCSR ;XXX0
10270 027444 005200 INC RO
10271 027446 010037 002046 MOV RO,HDZCSR ;XXX1
10272 027452 010037 002052 INC RO
10273 027456 005200 MOV RO,DZRBUF ;XXX2
10274 027460 010037 002050 MOV RO,HDZRBUF ;XXX3
10275 027464 010037 002054 MOV RO,HDZLPR ;XXX3
10276 027470 005200 INC RO
10277 027472 010037 002056 MOV RO,DZTCR ;XXX4
10278 027476 005200 INC RO
10279 027500 010037 002060 MOV RO,HDZTCR ;XXX5
10280 027504 005200 INC RO
10281 027506 010037 002062 MOV RO,DZMSR ;XXX6
10282 027512 010037 002066 MOV RO,DZTDR ;XXX6
10283 027516 005200 INC RO
10284 027520 010037 002064 MOV RO,HDZMSR ;XXX7
10285 027524 010037 002070 MOV RO,HDZTDR ;XXX7
10286 027530 000207 RTS PC
10287 027532 000240 DZPRT: PR5
10288 027534 000200 LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS
10289

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 85-4

CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0110

			:ERROR ERROR TABLE	
	.ERRTAB:			:ERROR
10291				
10292	027536	000000	0	:ERROR 0
10293	027540	000000	0	
10294	027542	000000	0	
10295				
10296	027544	027756	EM1	:ERROR
10297	027546	031226	DH1	
10298	027550	031424	DT1	
10299				
10300	027552	030031	EM2	:ERROR 2
10301	027554	031251	DH2	
10302	027556	031436	DT2	
10303				
10304	027560	030057	EM3	:ERROR 3
10305	027562	031304	DH3	
10306	027564	031454	DT3	
10307				
10308	027566	030116	EM4	:ERROR 4
10309	027570	031304	DH3	
10310	027572	031454	DT3	
10311				
10312	027574	030145	EM5	:ERROR 5
10313	027576	031316	DH4	
10314	027600	031462	DT4	
10315				
10316	027602	030174	EM6	:ERROR 6
10317	027604	031316	DH4	
10318	027606	031462	DT4	
10319				
10320	027610	030232	EM7	:ERROR 7
10321	027612	031304	DH3	
10322	027614	031454	DT3	
10323				
10324	027616	030273	EM8	:ERROR 10
10325	027620	031304	DH3	
10326	027622	031454	DT3	
10327				
10328	027624	030335	EM9	:ERROR 11
10329	027626	031304	DH3	
10330	027630	031454	DT3	
10331				
10332	027632	030373	EM10	:ERROR 12
10333	027634	031304	DH3	
10334	027636	031454	DT3	
10335				
10336	027640	030432	EM13	:ERROR 13
10337	027642	031304	DH3	
10338	027644	031454	DT3	
10339				
10340	027646	030463	EM14	:ERROR 14
10341	027650	031304	DH3	
10342	027652	031454	DT3	
10343				
10344	027654	030515	EM15	:ERROR 15
10345	027656	000000	0	
10346	027660	000000	0	

10347				
10348	027662	030557	EM16	
10349	027664	031304	DH3	
10350	027666	031454	DT3	
10351				
10352	027670	030630	EM17	:ERROR 17
10353	027672	031304	DH3	
10354	027674	031454	DT3	
10355				
10356	027676	030666	EM20	
10357	027700	031304	DH3	
10358	027702	031454	DT3	
10359				
10360	027704	030727	EM21	:ERROR 21
10361	027706	031345	DH5	
10362	027710	031500	DT5	
10363				
10364	027712	030757	EM22	:ERROR 22
10365	027714	031316	DH4	
10366	027716	031462	DT4	
10367				
10368	027720	031021	EM23	:ERROR 23
10369	027722	031304	DH3	
10370	027724	031454	DT3	
10371				
10372	027726	031051	EM24	
10373	027730	031304	DH3	
10374	027732	031454	DT3	
10375				
10376	027734	031077	EM25	
10377	027736	031304	DH3	
10378	027740	031454	DT3	
10379				
10380	027742	031127	EM26	
10381	027744	031304	DH3	
10382	027746	031454	DT3	
10383				
10384	027750	031156	EM27	
10385	027752	031304	DH3	
10386	027754	031454	DT3	

```

10388      :ERROR MESSAGES
10392 027756 047200 020117 046123 EM1: .ASCIZ <200>/NO SLAVE SYNC RESPONSE FROM DZ11 REGISTER/
10393 030031      200 042522 044507 EM2: .ASCIZ <200>?REGISTER R/W FAILURE?
10394 030057      200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
10395 030116 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
10396 030145      200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
10397 030174 042200 030532 020061 EM6: .ASCIZ <200>/DZ11 *RECEIVER BUFFER* ERROR/
10398 030232 052200 040522 051516 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
10399 030273      200 047125 054105 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
10400 030335      200 042522 042503 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
10401 030373      200 047125 054105 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
10402 030432 051600 046111 020117 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
10403 030463      200 044523 047514 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
10404 030515      200 041501 044524 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
10405 030557      200 042522 042101 EM16: .ASCIZ <200>/READING DZRBUF DID NOT CLEAR SILO ALARM/
10406 030630 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
10407 030666 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
10408 030727      200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
10409 030757      200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
10410 031021      200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
10411 031051      200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
10412 031077      200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
10413 031127      200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
10414 031156 043200 046125 020114 EM27: .ASCIZ <200>/FULL BINARY COUNT PATTERN NOT RECEIVED/
10415
10416 031226 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZ11 REG/
10417 031251      200 054105 042520 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
10418 031304 046200 047111 020105 DH3: .ASCIZ <200>/LINE NO./
10419 031316 042600 050130 041505 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
10420 031345      200 054124 046040 DH5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/

```

```

10421
10422      .EVEN
10426      :DATA TABLES FOR ERROR MESSAGES
10427 031424 000002      DT1: 2
10428 031426      006      003      .BYTE 6,3
10429 031430 001204      $REG1
10430 031432      006      001      .BYTE 6,1
10431 031434 001202      $REG0
10432
10433 031436 000003      DT2: 3
10434 031440      006      004      .BYTE 6,4
10435 031442 001214      $REG5
10436 031444      006      001      .BYTE 6,1
10437 031446 001212      $REG4
10438 031450      006      001      .BYTE 6,1
10439 031452 001202      $REG0
10440
10441 031454 000001      DT3: 1
10442 031456      003      001      .BYTE 3,1
10443 031460 001372      SAVLIN
10444
10445 031462 000003      DT4: 3
10446 031464      006      004      .BYTE 6,4
10447 031466 001214      $REG5
10448 031470      006      001      .BYTE 6,1
10449 031472 001212      $REG4

```

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 86-1
CZDZA DZ11 DEVICE DIAGNOSTICS.

SEQ 0113

10450 031474 003 001
 10451 031476 001372
 10452
 10453 031500 000004
 10454 031502 003 005
 10455 031504 001372
 10456 031506 006 011
 10457 031510 001214
 10458 031512 006 007
 10459 031514 001220
 10460 031516 006 001
 10461 031520 001400

.BYTE 3.1
 SAVLIN
 DT5: 4
 .BYTE 3.5
 SAVLIN
 .BYTE 6.9.
 \$REG5
 .BYTE 6.7
 \$TMP1
 .BYTE 6.1
 REGIST

:TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

:-----

10471 031522 002450
 10472 031524 001560
 10473 031526 001120
 10474 031530 000750
 10475 031532 000660
 10476 031534 000330
 10477 031536 000150
 10478 031540 000060
 10479 031542 000040
 10480 031544 000030
 10481 031546 000020
 10482 031550 000010
 10483 031552 000001
 10484 031554 000001
 10485 031556 000001
 10486 031560 000001
 10487
 10488
 10489
 10490
 10491
 10492 031562
 10496 001512 001512
 10497 001512 100000
 10498 000001

DLYTBL: 2450 :TIME FOR 50 BAUD
 1560 :TIME FOR 75 BAUD
 1120 :TIME FOR 110 BAUD
 750 :TIME FOR 134 BAUD
 660 :TIME FOR 150 BAUD
 330 :TIME FOR 300 BAUD
 150 :TIME FOR 600 BAUD
 60 :TIME FOR 1200 BAUD
 40 :TIME FOR 1800 BAUD
 30 :TIME FOR 2000 BAUD
 20 :TIME FOR 2400 BAUD
 10 :TIME FOR 3600 BAUD
 1 :TIME FOR 4800 BUAD
 1 :TIME FOR 7200 BAUD
 1 :TIME FOR 9600 BAUD
 1 :TIME OF DELAY FOR 19200 BAUD

:DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
 :FOR ALL TESTS TO FUNCTION CORRECTLY ON A PDP11/45 WITH BIPOLAR
 :MEMORY. THE TIMES WERE ALSO TESTED ON AN 11/40 AND 11/10.

CORMAX:
 .MANTO
 100000
 .END

AVECT1=	000000	8616				
AVECT2=	000000	8616				
BAUD	024706	9845#	10166			
BINWRD	007120	8617#				
BIT0 =	000001	8616#	8655	8732	9609	9610
BIT00 =	000001	8616#				
BIT01 =	000002	8616#				
BIT02 =	000004	8616#				
BIT03 =	000010	8616#				
BIT04 =	000020	8616#				
BIT05 =	000040	8616#				
BIT06 =	000100	8616#				
BIT07 =	000200	8616#				
BIT08 =	000400	8616#				
BIT09 =	001000	8616#				
BIT1 =	000002	8616#	8655			
BIT10 =	002000	8616#	8655			
BIT11 =	004000	8616#	8617	8655	8697	
BIT12 =	010000	8616#	10031			
BIT13 =	020000	8616#	9935	10028		
BIT14 =	040000	8616#	8617	10025		
BIT15 =	100000	8616#	8617			
BIT2 =	000004	8616#	8655			
BIT3 =	000010	8616#	9602	9604		
BIT4 =	000020	8616#	9602			
BIT5 =	000040	8616#	8617			
BIT6 =	000100	8616#				
BIT7 =	000200	8616#	8617			
BIT8 =	000400	8616#	8655			
BIT9 =	001000	8616#	8655			
BPTVEC=	000014	8616#				
BRK0 =	000400	8616#				
BRK1 =	001000	8616#				
BRK2 =	002000	8616#				
BRK3 =	004000	8616#				
BRK4 =	010000	8616#				
BRK5 =	020000	8616#				
BRK6 =	040000	8616#				
BRK7 =	100000	8616#				
BRW	005372	8616	8617#			
BUILD	024200	9688#	9728			
BYTCNT	026650	10213#				
CHRCNT	007116	8617#*				
CNVRT =	104412	8616#	8617			
CONVRT=	104411	8616#	8617			
CORMAX	031562	10492#	10493			
COUNT0	024056	9609*	9610*	9640*	9648#	
COUNT1	024060	9609*	9610*	9641*	9649#	
CO0 =	000400	8616#				
CO1 =	001000	8616#				
CO2 =	002000	8616#				
CO3 =	004000	8616#				
CO4 =	010000	8616#				
CO5 =	020000	8616#				
CO6 =	040000	8616#				
CO7 =	100000	8616#				

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 87-9

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0123

STKLMT=	177774	8616#							
STOP	001462	8616#	8878	9204	9312	9393	9544	9574	9609 9610
SV05	006654	8617#							
SWR	001160	8616#*	8617*	9804	9839	9905			
SWREG	000176	8616#							
SW0 =	000001	8616#							
SW00 =	000001	8616#	9804						
SW01 =	000002	8616#	8617	9839					
SW02 =	000004	8616#							
SW03 =	000010	8616#							
SW04 =	000020	8616#							
SW05 =	000040	8616#							
SW06 =	000100	8616#							
SW07 =	000200	8616#							
SW08 =	000400	8616#	8617						
SW09 =	001000	8616#	8617						
SW1 =	000002	8616#							
SW10 =	002000	8616#	8617						
SW11 =	004000	8616#							
SW12 =	010000	8616#	8617						
SW13 =	020000	8616#	8617						
SW14 =	040000	8616#							
SW15 =	100000	8616#							
SW2 =	000004	8616#							
SW3 =	000010	8616#							
SW4 =	000020	8616#							
SW5 =	000040	8616#							
SW6 =	000100	8616#							
SW7 =	000200	8616#							
SW8 =	000400	8616#							
SW9 =	001000	8616#							
S110 =	001000	8616#							
S1200 =	003400	8616#							
S134 =	001400	8616#							
S150 =	002000	8616#							
S1800 =	004000	8616#							
S19200 =	007400	8616#							
S2000 =	004400	8616#							
S2400 =	005000	8616#							
S300 =	002400	8616#							
S3600 =	005400	8616#							
S4800 =	006000	8616#							
S50 =	000000	8616#	9538	9541					
S600 =	003000	8616#							
S7200 =	006400	8616#							
S75 =	000400	8616#							
S9600 =	007000	8616#							
TABLE2	026534	10159	10174#						
TBITVE=	000014	8616#							
TBUF	026666	9939*	9960	10222#					
TCR0 =	000001	8616#	8653	8655					
TCR1 =	000002	8616#							
TCR2 =	000004	8616#							
TCR3 =	000010	8616#							
TCR4 =	000020	8616#							
TCR5 =	000040	8616#							

CZDZA-GO
CZDZAG.P11

MACY11 30A(1052)
24-JUN-81 09:45

24-JUN-81 09:46 PAGE 87-11

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0125

TST21	015404	8799	8810#						
TST22	015742	8810	8874#						
TST23	016270	8874	8933#						
TST24	016622	8933	9017#						
TST25	017100	9017	9062#						
TST26	017410	9062	9063#						
TST27	017736	9063	9074#						
TST3	013322	8628	8652#						
TST30	020370	9074	9160#						
TST31	020556	9160	9199#						
TST32	021304	9199	9307#						
TST33	021666	9307	9380#						
TST34	022474	9380	9532#						
TST35	023204	9532	9609#						
TST36	023444	9609	9610#	10466					
TST37	= ***** U	9610							
TST4	013414	8652#							
TST5	013506	8652#							
TST6	013600	8652#							
TST7	013672	8652#							
TTST	005140	8616*	8617#						
TWOSTO=	000040	8616#	9538	9541					
TXSVC	022116	9397	9422#						
TYPDAT	007502	8617#							
TYPE =	104402	8616#	8617	9874	9880	9920	10065	10165	
TYPMSG	007372	8617#							
T110	002106	8616#							
T1200	002120	8616#							
T134	002110	8616#							
T150	002112	8616#							
T1800	002122	8616#							
T2000	002124	8616#							
T2400	002126	8616#							
T300	002114	8616#							
T3600	002130	8616#							
T4800	002132	8616#							
T50	002102	8616#							
T600	002116	8616#							
T7200	002134	8616#							
T75	002104	8616#							
T9600	002136	8616#							
VECMAP	012654	8617#							
VEC1	024504	9804#							
VEC2	024514	9806#							
WCHFLG	026634	9844	9868	10207#					
WRDCNT	007114	8617#*							
WTBS.F	007470	8617#							
XBEGIN	024754	9840	9865#	10069					
XBX	007260	8617#							
XCSR	005072	8617#							
XEOP	026150	9909	9951	9971	10017	10065#			
XERR	005114	8617#							
XHEAD	011050	8616	8617#						
XMTCNT	001376	8616#	9580*	9609*	9610*	9669			
XMTLIN	001374	8616#	9536*	9546*	9548	9570			
XMTSRV	024062	9581	9609	9610	9655#				

CZDZA-GO MACY11 30A(1052) 24-JUN-81 09:46 PAGE 88-2
CZDZAG.P11 24-JUN-81 09:45 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0132

.S40CA 955#
.1170 509#

. ABS. 031562 000

ERRORS DETECTED: 0

CZDZAG.BIC,CZDZAG.SEQ/CRF/DOC/NL:TOC=SYSMAC.C5,CZDZAG.P11
RUN-TIME: 16 22 1 SECONDS
RUN-TIME RATIO: 48/40=1.1
CORE USED: 51K (101 PAGES)

DOCUMENT PAGES: 132