

DZ11

DZ11 ASYCH MUX TST
CZDZAI0

AH-8783I-MC

1 OF 1 OCT 1985

COPYRIGHT© 1976-85

digital

MADE IN USA

T 1' W
A ?;
CZDZA-IO
CZDZAI.P11

MACY11 30A(1052) 02-JUL-85 :6:19 PAGE 59
02-JUL-85 16:17

SEQ 1

5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642

.REM *

IDENTIFICATION

PRODUCT CODE: AC-8781I-MC
PRODUCT NAME: CZDZAI0 DZ11 LN ASYNC MUX TSTS
PRODUCT DATE: JUNE 1985
MAINTAINER: NAC SOFTWARE ENGINEERING
AUTHOR: DAVE HOFFMAN

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES

COPYRIGHT (C) 1976,1981,1985 BY DIGITAL EQUIPMENT CORPORATION

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.

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
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724

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 (SW00=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)

5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780

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: RUN BAUD RATE TEST
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.

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
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830

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.

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
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877

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)

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
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927

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 'SCOP1'). 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.

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
5960
5961
5962
5963
5964
5965
5966

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.

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
6000
6001
6002
6003
6004
6005
6006
6007

"# 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.

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
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060

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.

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
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110

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.

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
6147
6148
6149
6150
6151
6152
6153

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 '\$TSTNM' (BYTE 1122) 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.

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

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-I 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.

6188
6189
6190
6191
6192
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214

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.

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 100000 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.

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 BITS (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.

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
6353
6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368

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., \$DEVM, \$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

\$DEVM - 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 TABLE. THE USER IS THEN FREE TO MONITOR UNDER APT AS NORMAL.


```
HT=      11      ;;CODE FOR HORIZONTAL TAB
```

000011

```
(2)      000012      LF=      12      ;;CODE FOR LINE FEED
(2)      000015      CR=      15      ;;CODE FOR CARRIAGE RETURN
(2)      000200      CRLF=    200      ;;CODE FOR CARRIAGE RETURN-LINE FEED
(2)      177776      PS=    177776    ;;PROCESSOR STATUS WORD
(2)      .EQUIV      PS,PSW
(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) 040000 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)
(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)
(1)
(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 POP2SP=22626 ;INCREMENT STACK TWICE
```

```

(1)                                     ;DZ11 CONTROL AND STATUS REGISTER DEFINITIONS
(1)                                     ;(DZCSR)          BIT DEFINITIONS
(1)                                     ;-----
(1)
(1)      000010      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)
(1)      000021      $XON=21
(1)      000023      $XOFF=23
(1)
(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)
(1)                                     ;DZRBUF BIT DEFINITIONS
(1)                                     ;-----
(1)
(1)      010000      PARER=BIT12      ;PARITY ERROR
(1)      020000      FRMERR=BIT13     ;FRAME ERROR
(1)      040000      OVRUN=BIT14      ;OVERRUN ERROR
(1)      100000      DVALID=BIT15     ;DATA VALID
(1)
(1)                                     ;DZRBUF WORD DEFINITIONS
(1)                                     ;-----
(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)
(1)                                     ;DZLPR WORD DEFINITIONS
(1)                                     ;-----
(1)
(1)      000000      LP0=0            ;LINE PARAMETER 0
(1)      000001      LP1=BIT0        ;LINE PARAMETER 1
(1)      000002      LP2=BIT1        ;LINE PARAMETER 2

```


GENERAL DEFINITIONS AND EQUIVALENCES

(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)	000020	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)	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)	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
(1)		;DZTCR BIT DEFINITIONS	
(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
(1)
(1)
(1)      ;DZMSR BIT DEFINITIONS
(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
(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
(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
(1)
(1)
(1)      ;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)
```


TRAPCATCHER FOR UNEXPECTED INTERRUPTS

```
(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 011430 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 010310 $PWRDN ;POWER FAIL HANDLER
(1) 000026 000340 340 ;SERVICE AT PRIORITY LEVEL 7
(1) 000030 007220 $ERROR ;ERROR HANDLER
(1) 000032 000340 340 ;SERVICE AT PRIORITY LEVEL 7
(1) 000034 007112 .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) 000040 . = $SVPC ;: RESTORE PC
(1)
(1) 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 025244 JMP XSTART ;GOTO CABLE TEST/ECHO TEST
(1)
(2)
(2) 001000 . =1000
(2) 001000 005200 055103 055104 MTITLE: .ASCIZ <200><12>/CZDZA-IO/<200>/CZDZAI0 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 000000 $CMTAG: ;:START OF COMMON TAGS
(3) 001122 000 $TSTNM: .WORD 0 ;:CONTAINS THE TEST NUMBER
(3) 001123 000 $ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
(3) 001124 000000 $ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
(3) 001126 000000 $LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
(3) 001130 000000 $LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
(3) 001132 000000 $ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
(3) 001134 000 $ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
(3) 001135 001 $ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
(3) 001136 000000 $ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
(3) 001140 000000 $GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
(3) 001142 000000 $BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
(3) 001144 000000 $GDDAT: .WORD 0 ;:CONTAINS 'GOOD' DATA
(3) 001146 000000 $BDDAT: .WORD 0 ;:CONTAINS 'BAD' DATA
(3) 001150 000000 .WORD 0 ;:RESERVED--NOT TO BE USED
(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 $REGO: .WORD 0 ;:CONTAINS (($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)

```



```

(5) ;*****
(4) .EVEN
(4) 001234 $MAIL: ;:APT MAILBOX
(4) 001234 000000 $MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
(4) 001236 000000 $FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
(4) 001240 000000 $TESTN: .WORD ATESTN ;:TEST NUMBER
(4) 001242 000000 $PASS: .WORD APASS ;:PASS COUNT
(4) 001244 000000 $DEVCT: .WORD ADEVCT ;:DEVICE COUNT
(4) 001246 000000 $UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
(4) 001250 000000 $MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS
(4) 001252 000000 $MSGLG: .WORD AMSGLG ;:MESSAGE LENGTH
(4) 001254 $ETABLE: ;:APT ENVIRONMENT TABLE
(4) 001254 000 $ENV: .BYTE AENV ;:ENVIRONMENT BYTE
(4) 001255 000 $ENVH: .BYTE AENVH ;:ENVIRONMENT MODE BITS
(4) 001256 000000 $SWREG: .WORD ASWREG ;:APT SWITCH REGISTER
(4) 001260 000000 $USWR: .WORD AUSWR ;:USER SWITCHES
(4) 001262 000000 $CPUOP: .WORD ACPUOP ;:CPU TYPE,OPTIONS
(4) ;* BITS 15-11=CPU TYPE
(4) ;* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(4) ;* 11/70=06,PDQ=07,Q=10
(4) ;* BIT 10=REAL TIME CLOCK
(4) ;* BIT 9=FLOATING POINT PROCESSOR
(4) ;* BIT 8=MEMORY MANAGEMENT
(4) 001264 000 $MAMS1: .BYTE AMAMS1 ;:HIGH ADDRESS,M.S. BYTE
(4) 001265 000 $MTYP1: .BYTE AMTYP1 ;:MEM. TYPE,BLK#1
(4) ;* MEM.TYPE BYTE -- (HIGH BYTE)
(4) ;* 900 NSEC CORE=001
(4) ;* 300 NSEC BIPOLAR=002
(4) ;* 500 NSEC MOS=003
(4) 001266 000000 $MADR1: .WORD AMADR1 ;:HIGH ADDRESS,BLK#1
(4) ;* MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(4) 001270 000 $MAMS2: .BYTE AMAMS2 ;:HIGH ADDRESS,M.S. BYTE
(4) 001271 000 $MTYP2: .BYTE AMTYP2 ;:MEM.TYPE,BLK#2
(4) 001272 000000 $MADR2: .WORD AMADR2 ;:MEM.LAST ADDRESS,BLK#2
(4) 001274 000 $MAMS3: .BYTE AMAMS3 ;:HIGH ADDRESS,M.S.BYTE
(4) 001275 000 $MTYP3: .BYTE AMTYP3 ;:MEM.TYPE,BLK#3
(4) 001276 000000 $MADR3: .WORD AMADR3 ;:MEM.LAST ADDRESS,BLK#3
(4) 001300 000 $MAMS4: .BYTE AMAMS4 ;:HIGH ADDRESS,M.S.BYTE
(4) 001301 000 $MTYP4: .BYTE AMTYP4 ;:MEM.TYPE,BLK#4
(4) 001302 000000 $MADR4: .WORD AMADR4 ;:MEM.LAST ADDRESS,BLK#4
(4) 001304 000000 $VECT1: .WORD AVECT1 ;:INTERRUPT VECTOR#1,BUS PRIORITY#1
(4) 001306 000000 $VECT2: .WORD AVECT2 ;:INTERRUPT VECTOR#2BUS PRIORITY#2
(4) 001310 160010 $BASE: .WORD ABASE ;:BASE ADDRESS OF EQUIPMENT UNDER TEST
(4) 001312 000000 $DEVH: .WORD ADEVH ;:DEVICE MAP
(4) 001314 000000 $CDW1: .WORD ACDW1 ;:CONTROLLER DESCRIPTION WORD#1
(4) 001316 000000 $CDW2: .WORD ACDW2 ;:CONTROLLER DESCRIPTION WORD#2
(4) 001320 000000 $DDW0: .WORD ADDW0 ;:DEVICE DESCRIPTOR WORD#0
(4) 001322 000000 $DDW1: .WORD ADDW1 ;:DEVICE DESCRIPTOR WORD#1
(4) 001324 000000 $DDW2: .WORD ADDW2 ;:DEVICE DESCRIPTOR WORD#2
(4) 001326 000000 $DDW3: .WORD ADDW3 ;:DEVICE DESCRIPTOR WORD#3
(4) 001330 000000 $DDW4: .WORD ADDW4 ;:DEVICE DESCRIPTOR WORD#4
(4) 001332 000000 $DDW5: .WORD ADDW5 ;:DEVICE DESCRIPTOR WORD#5
(4) 001334 000000 $DDW6: .WORD ADDW6 ;:DEVICE DESCRIPTOR WORD#6
(4) 001336 000000 $DDW7: .WORD ADDW7 ;:DEVICE DESCRIPTOR WORD#7
(4) 001340 000000 $DDW8: .WORD ADDW8 ;:DEVICE DESCRIPTOR WORD#8
(4) 001342 000000 $DDW9: .WORD ADDW9 ;:DEVICE DESCRIPTOR WORD#9

```

```

(4) 001344 000000 $DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
(4) 001346 000000 $DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
(4) 001350 000000 $DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
(4) 001352 000000 $DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
(4) 001354 000000 $DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
(4) 001356 000000 $DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15
(4)
(4)
(4) 001360 $ETEND:
(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) 001360 $ERRTB:
(2) ;PROGRAM CONTROL PARAMETERS
(2) ;-----
(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)
(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 000001 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)                                ;PROGRAM CONTROL FLAGS
(2)                                ;-----
(2)
(2) 001414      000      EIAFLG: .BYTE 0      ;0=EIA 100000=20MA
(2) 001415      000      INIFLG: .BYTE 0      ;PROGRAM INITIALIZATION FLAG
(2) 001416      000      HDRFLG: .BYTE 0      ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
(2) 001417      000      MNTFLG: .BYTE 0      ;MAINTENANCE BIT SET FLAG
(2) 001420      000      DONFLG: .BYTE 0      ;TRANSMISSION COMPLETION FLAG
(2)              001422      .EVEN
(2)              ;DATA VARIABLES
(2) 001422      000000      TD0: .WORD 0
(2) 001424      000000      TD1: .WORD 0
(2) 001426      000000      TD2: .WORD 0
(2) 001430      000000      TD3: .WORD 0
(2) 001432      000000      TD4: .WORD 0
(2) 001434      000000      TD5: .WORD 0
(2) 001436      000000      TD6: .WORD 0
(2) 001440      000000      TD7: .WORD 0
(2) 001442      000000      TR0: .WORD 0
(2) 001444      000000      TR1: .WORD 0
(2) 001446      000000      TR2: .WORD 0
(2) 001450      000000      TR3: .WORD 0
(2) 001452      000000      TR4: .WORD 0
(2) 001454      000000      TR5: .WORD 0
(2) 001456      000000      TR6: .WORD 0
(2) 001460      000000      TR7: .WORD 0
(2) 001462      STOP:
(2)              ; -- END 0 MACRO =====
(2)              .SBTTL APT PARAMETER BLOCK
(2)
(3)              ;*****
(2)              ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(3)              ;*****
(2)              . $X= .      ;;SAVE CURRENT LOCATION
(2)              . =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(2) 000024      000200      200      ;;FOR APT START UP
(2)              . =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
(2) 000044      001462      $APTHDR ;;POINT TO APT HEADER BLOCK
(2)              . = $X      ;;RESET LOCATION COUNTER
(3)              ;*****
(2)              ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(2)              ;INTERFACE SPEC.
(2)
(2) 001462      $APTHD:
(2) 001462      000000      $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(2) 001464      001234      $MBADR: .WORD $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(2) 001466      000132      $TSTM: .WORD 90.      ;;RUN TIM OF LONGEST TEST
(2) 001470      000137      $PASTM: .WORD 95.      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(2) 001472      000137      $UNITM: .WORD 95.      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(2) 001474      000052      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
(1)              ;DZ11 STATUS TABLE AND ADDRESS ASSIGNMENTS
(1)              ;-----
(1)
(1)              . =1500
(1) 001500      DZ.MAP:
```


(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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)			; -- END 0 MACRO =====	
(3)			; -\$JUNK-----	
(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) ; -- END 0 MACRO =====
(3) ; -$JUNK-----
(3)
(3) 001734 000001 DZCR15: .BLKW 1 ;CONTROL STATUS REGISTER FOR DZ11 NUMBER 15
(3) 001736 000001 DZVC15: .BLKW 1 ;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 15
(3) 001740 000001 DZLV15: .BLKW 1 ;PRIORITY LEVEL AND EIA FLAG SELECTOR
(3) 001742 000001 LINE15: .BLKW 1 ;ALL LINES SELECTED
(3) 001744 000001 PAR15: .BLKW 1 ;PARAMETERS
(3) 001746 000001 MANT15: .BLKW 1 ;MAINTENANCE MODE FOR THIS DEVICE
(3) ; -- END 0 MACRO =====
(3) ; -$JUNK-----
(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) ; -- END 0 MACRO =====
(3) ; -$JUNK-----
(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
(3) ; -- END 0 MACRO =====
(1)
(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 .TRPTAB:
(3) 104400 ADVANCE=TRAP+0 ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
(2) 002002 007206 .ADVANCE
(2) ; -- END 0 MACRO =====
(3) 104401 SCOP1=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
(2) 002004 005376 .SCOP1
(2) ; -- END 0 MACRO =====
(3) 104402 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
(2) 002006 005422 .TYPE
(2) ; -- END 0 MACRO =====
(3) 104403 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
(2) 002010 006242 .INSTR
(2) ; -- END 0 MACRO =====
(3) 104404 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
(2) 002012 006416 .INSTER
(2) ; -- END 0 MACRO =====
(3) 104405 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
(2) 002014 006436 .PARAM
(2) ; -- END 0 MACRO =====
(3) 104406 SETFLG=TRAP+6 ;CALL TO SET FLAG ROUTINE
(2) 002016 011142 .SETFLG
(2) ; -- END 0 MACRO =====
(3) 104407 SAV05=TRAP+7 ;CALL TO REGISTER SAVE ROUTINE
(2) 002020 006636 .SAV05
(2) ; -- END 0 MACRO =====
(3) 104410 RES05=TRAP+10 ;CALL TO REGISTER RESTORE ROUTINE
(2) 002022 006676 .RES05
(2) ; -- END 0 MACRO =====
(3) 104411 CONVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE
(2) 002024 006730 .CONVRT
(2) ; -- END 0 MACRO =====
(3) 104412 CNVRT=TRAP+12 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
(2) 002026 006734 .CNVRT
(2) ; -- END 0 MACRO =====
(3) 104413 DEVICE.CLR=TRAP+13 ;CALL TO ISSUE A DEVICE CLEAR
(2) 002030 007134 .DEVICE.CLR
(2) ; -- END 0 MACRO =====
(3) 104414 DELAY=TRAP+14 ;CALL TO DELAY FOR FAST CPU'S
(2) 002032 007166 .DELAY
(2) ; -- END 0 MACRO =====
(3) 104415 PARMD=TRAP+15 ;CONVERT DECIMAL STRING TO OCTAL
(2) 002034 026772 .PARMD
(2) ; -- END 0 MACRO =====
(3) 104416 PAWCH=TRAP+16 ;SET FLAG ECHO OR CABLE
(2) 002036 027166 .PAWCH
(2) ; -- END 0 MACRO =====
(3) 104417 DCLASM=TRAP+17 ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2) 002040 007154 .DCLASM
(2) ; -- END 0 MACRO =====
(1)

```


CZDZA-IO
CZDZAI.P11

MACY11 30A(1052)
02-JUL-85 16:17

02-JUL-85 16:19 PAGE 83-15
APT PARAMETER BLOCK

SEQ 34

(1)
(1)


```
(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)
```


(1)
(1)
(1)
(1)
(1) 002102
(1) 002102 000000
(1) 002104 000000
(1) 002106 000000
(1) 002110 000000
(1) 002112 000000
(1) 002114 000000
(1) 002116 000000
(1) 002120 000000
(1) 002122 000000
(1) 002124 000000
(1) 002126 000000
(1) 002130 000000
(1) 002132 000000
(1) 002134 000000
(1) 002136 000000
(1) 002140 000000
(1) 002142 000000
(1) 002144 000000
(1) 002146 000000

TIME TABLE FOR RELATIVE TIMING TESTS

TMTBL:
T50: 0
T75: 0
T110: 0
T134: 0
T150: 0
T300: 0
T600: 0
T1200: 0
T1800: 0
T2000: 0
T2400: 0
T3600: 0
T4800: 0
T7200: 0
T9600: 0
TEIGHT: 0
TSEVEN: 0
TSIX: 0
TFIVE: 0

```

(1)                                     ;PROGRAM INITIALIZATION
(1)                                     ;LOCK OUT INTERRUPTS
(1)                                     ;SET UP PROCESSOR STACK
(1)                                     ;SET UP POWER FAIL VECTOR
(1)                                     ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
(1)                                     ;TYPE TITLE MESSAGE
(1)
(1) 002150                               .START:
(1) 002150 000005                       RESET                               ;CLEAR THE WORLD. START NEW ENVIRONMENT
(1) 002152 012706 001120                 MOV      #STACK,SP                ;SET UP STACK
(1) 002156 106427 000340                 MTPS     #PR7                    ;LOCK OUT INTERRUPTS
(1) 002162 012737 010310 000024         MOV      #PWRDN,#24                ;SET UP POWER FAIL VECTOR
(1) 002170 012737 005122 000020         MOV      #SCOPE,#IOTVEC          ;SET VECTOR FOR SCOPE ROUTINE
(1) 002176 012737 000340 000022         MOV      #340,#IOTVEC+2          ;LEVEL 7
(1) 002204 012737 007220 000030         MOV      #ERROR,#EMTVEC          ;SET VECTOR FOR ERROR ROUTINE
(1) 002212 012737 000340 000032         MOV      #340,#EMTVEC+2          ;LEVEL 7
(1) 002220 012737 007112 000034         MOV      #.TRPSR,#TRAPVEC        ;SET VECTOR FOR TRAP CALLS
(1) 002226 012737 000340 000036         MOV      #340,#TRAPVEC+2        ;LEVEL 7
(1) 002234 012737 010310 000024         MOV      #PWRDN,#PWRVEC          ;SET VECTOR FOR POWER FAIL ROUTINE
(1) 002242 012737 000340 000026         MOV      #340,#PWRVEC+2        ;LEVEL 7
(1) 002250 113737 001410 001411         MOVVB    DZNUM,SAVNUM          ;SAVE NUMBER OF DEVICES IN SYSTEM.
(1) 002256 005037 001242                 CLR      $PASS                    ;CLEAR PASS COUNT
(1) 002262 105037 001123                 CLRB     $ERFLG                  ;CLEAR ERROR FLAG
(1) 002266 012737 001500 001412         MOV      #DZ.MAP,ACTIVE        ;GET MAP POINTER.
(1) 002274 012737 000001 001406         MOV      #1,RUN               ;POINT POINTER TO FIRST DEVICE.
(1) 002302 005037 001132                 CLR      $ERTTL                  ;CLEAR ERROR COUNT
(1) 002306 005037 001136                 CLR      $ERRPC                  ;CLEAR LAST ERROR POINTER
(1) 002312 005037 001122                 CLR      $TSTNM                 ;SET UP FOR TEST 1
(1) 002316 012737 002150 001126         MOV      #.START,$LPADR        ;SET UP FOR POWER FAIL BEFORE
(1)                                     ;TESTING STARTS
(1)                                     ;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
(1) 002324 013746 000006                 MOV      6,-(SP)                ;SAVE BUS ERROR PS
(1) 002330 013746 000004                 MOV      4,-(SP)                ;SAVE BUS ERROR PC
(1) 002334 012737 002354 000004         MOV      #20$,4                ;SET UP TO TRAP TO THIS ROUTINE
(1) 002342 022777 177777 176610         CMP      #-1,$SWR              ;CAN 177570 BE REFERENCED?
(1) 002350 001402                         BEQ      22$                      ;IF SO AND IT IS -1, TREAT LIKE SWITCHLESS
(1) 002352 000407                         BR       21$                      ;IF YES, SKIP AROUND THE SETUP
(1) 002354 022626 20$: POP2SP            ;REMOVE THE TRAP FROM THE STACK
(1) 002356 012737 000176 001160 22$: MOV      #SWREG,SWR          ;IF NO, TRAP COMES HERE. POINT TO SOFTWARE SWR
(1) 002364 012737 000174 001162         MOV      #DISPREG,DISPLAY      ;POINT TO SOFTWARE DISPLAY REGISTER
(1) 002372 012637 000004 21$: MOV      (SP)+,4                ;RESTORE THE BUS ERROR VECTOR
(1) 002376 012637 000006         MOV      (SP)+,6
(1) 002402 105737 001415                 TSTB     INIFLG                  ;TITLE ALREADY PRINTED?
(1) 002406 001010                         BNE      29$                      ;BRANCH IF YES
(1) 002410 023727 000042 005056         CMP      #42,$ENDAD           ;RUNNING UNDER ACT?
(1) 002416 001402                         BEQ      31$                      ;IF YES DONT PRINT TITLE
(1) 002420 104402 001000                 TYPE     ,MTITLE                ;PRINT THE DIAGNOSTIC'S TITLE
(1) 002424 105337 001415 31$: DECB     INIFLG          ;SET THE ONCE ONLY FLAG
(1) 002430 105737 001255 29$: TSTB     $ENVM                ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
(1) 002434 100006                         BPL      30$                      ;IF NOT, GO CHECK FOR AUTO-SIZING
(1) 002436 004737 012102                 JSR      PC,SETAPT              ;OTHERWISE, GO DO APT SIZING FROM ETABLE
(1)                                     ;REPLACE "CLRB HDRFLG" WITH NOP'S
(1) 002442 000240                         NOP
(1) 002444 000240                         NOP
(1)                                     ;
(1)                                     ;CLRB     HDRFLG                ;MAKE SURE STATUS TABLE IS PRINTED

```



```

(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!
(1)
(1) ;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
(1) ;TABLE AND SET UP THE DIAGNOSTIC.
(1)
(1) ;GET THE BASE ADDRESS OF THE DZ11'S
(1)
(1) 002530 33$:
(2) ; --$GETPAR-----
(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
(2) ; -- END 0 MACRO -----
(1) 002546 013737 001500 001310 MOV DZCRO,$BASE ;COPY BASE ADDRESS TO ETABLE
(1)
(1) ;GET THE BASE VECTOR ADDRESS
(1)
(1) 002554 34$:
(2) ; --$GETPAR-----
(2) 002554 104403 INSTR ;CALL THE STRING INPUT ROUTINE
(2) 002556 003530 67$ ;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
(2) ; -- END 0 MACRO -----
(1) 002572 013737 001502 001304 MOV DZVCO,$VECT1 ;COPY VECTOR TO ETABLE
(1)
(1) ;GET THE BUS REQUEST LEVEL
(1)
(2) ; --$GETPAR-----
(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

```

CZDZA-IO MACY11 30A(1052) 02-JUL-85 16:19 PAGE 83-20
 CZDZAI.P11 02-JUL-85 16:17 PROGRAM INITIALIZATION AND START UP.

```

(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
(2)      ; -- END 0 MACRO -----
(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
(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    $VECT1+1 ;ALIGN THE BITS PROPERLY
(1)      ;FIND OUT IF MODULE IS EIA OR 20 MA.
(1)
(1) 002650 104402 004260      TYPE    ,74$      ;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 CHARACTER 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,DZLVO ;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,DZLVO ;IF SO, SET FLAG
(1) 003002 000726      BR     80$      ;GET MORE INPUT
(1) 003004      81$:
(1)      ;GET THE MODE OF OPERATION (E,I,S)
(1)
(2)      ; -GETFLG-----
(2) 003004 104403      INSTR      ;CALL THE STRING INPUT ROUTINE
(2) 003006 004002      72$      ;POINTER TO THE MESSAGE TO BE PRINTED
(2) 003010 104406      SETFLG      ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
(2) 003012 001512      MANTO      ;THIS IS THE FLAG BEING SETUP
(2)      ; -- END 0 MACRO -----
(1)      ;GET THE NUMBER OF DZ11'S RUNNING
(1)
(2)      ; -GETPAR-----
(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 $TMP1 ;POINTER TO MAP LOCATION TO BE FILLED
(2) 003030 000 .BYTE 0 ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 003031 001 .BYTE 1 ;NUMBER OF PARAMETERS TO STORE
(2) ; -- END O MACRO -----
(1)
(1) 003032 012737 000377 001506 MOV #377,LINE0 ;SET UP DEFAULT LINES
(1) 003040 012737 017070 001510 MOV #17070,PAR0 ;SET UP DEFAULT LPR PARAMETER
(1)
(1) 003046 012737 000001 007204 MOV #1,DLYCNT ;RECEIVER ON; 9600 BAUD; 2STOP BITS; 8 BIT/CHAR
(1) 003054 032777 000010 176076 BIT #SW03,@SWR ;INITIALIZE DELAY COUNT
(1) 003062 001402 BEQ 40$ ;DO YOU WANT PARAMETERS?
(1) 003064 004737 003274 JSR PC,23$ ;IF NO, SKIP THE PARAMETER CALL
(1) 003070 012737 000001 001312 40$: MOV #1,$DEVM ;GET PARAMETERS
(1) 003076 113737 001220 001410 MOV $TMP1,DZNUM ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
(1) 003104 113737 001220 001411 MOV $TMP1,SAVNUM ;COPY THE NUMBER OF DEVICES
(1) 003112 005337 001220 62$: DEC $TMP1 ;COPY A BACKUP NUMBER
(1) 003116 001404 BEQ 61$ ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
(1) 003120 000261 SEC ; SELECTED DEVICES
(1) 003122 006137 001312 ROL $DEVM ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
(1) 003126 000771 BR 62$ ;POINT TO THE NEXT DEVICE
(1) 003130 013737 001312 001222 61$: MOV $DEVM,$TMP2 ;GO DO THIS PROCEDURE AGAIN
(1) 003136 013737 001312 001404 MOV $DEVM,DZACTV ;# OF TIMES
(1) 003144 012700 001500 MOV #DZCR0,R0 ;COPY THE ACTIVE DEVICE PARAMETER
(1) 003150 012701 001514 MOV #DZCR1,R1 ;SET A POINTER TO THE SPECIFIED INFORMATION
(1) 003154 012702 001320 MOV #DDW0,R2 ;POINT R1 TO THE REST OF THE MAP TABLE
(1) 003160 000241 CLC ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
(1) 003162 006037 001222 ROR $TMP2 ;INITIALIZE THE "C" BIT FOR A ROTATION
(1) 003166 006237 001222 64$: ASR $TMP2 ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
(1) 003172 103404 BCS 41$ ;ISOLATE A SELECTION FLAG IN THE "C" BIT
(1) 003174 012711 177777 MOV #-1,(R1) ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
(1) 003200 000137 004374 JMP 63$ ;TERMINATE THE LIST
(1) 003204 012011 41$: MOV (R0)+,(R1) ;GO TO THE NEXT BLOCK
(1) 003206 062721 000010 ADD #10,(R1)+ ;ADDRESS
(1) 003212 012011 MOV (R0)+,(R1) ;POINT TO THE NEXT DZ11 ADDRESS VALUE
(1) 003214 062721 000010 ADD #10,(R1)+ ;VECTOR
(1) 003220 012021 MOV (R0)+,(R1)+ ;POINT TO THE NEXT VECTOR VALUE
(1) 003222 012021 MOV (R0)+,(R1)+ ;LEVEL
(1) 003224 016012 177774 MOV -4(R0),(R2) ;LINES
(1) 003230 042712 077777 BIC #77777,(R2) ;GET THE EIA FLAG FROM THE PRIORITY WORD
(1) 003234 051022 BIS (R0),(R2)+ ;ISOLATE THAT FLAG
(1) 003236 012021 MOV (R0)+,(R1)+ ;ADD PARAMETERS TO DEVICE DESCRIPTOR WORD
(1) 003240 012021 MOV (R0)+,(R1)+ ;PARAMETERS
(1) 003242 000751 BR 64$ ;MAINTENANCE MODE
(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)
(1) ;GET THE ACTIVE LINES PARAMETER

```

```

(1) 003274          23$:
(2)                                     ; -$GETPAR-----
(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      1             ;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
(2)                                     ; -- END 0 MACRO -----
(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    , $QUES ;THIS IS AN INCORRECT PARAMETER
(1) 003346 104402 011065      TYPE    , MBADLN  ;LET THE USER KNOW ABOUT IT
(1) 003352 000750      BR      23$              ;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$:
(2)                                     ; -$GETPAR-----
(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      PARO          ;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
(2)                                     ; -- END 0 MACRO -----
(1) 003404 012702 001506      MOV     #LINE0,R2   ;POINT TO THE LINE SELECTION PARAMETER
(1) 003410 012703 001510      MOV     #PARO,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 032444 007204      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$: .ASCIZ <200>/ENTER DELAY PARAMETER: /
(1)      004374              .EVEN
(1) 004374              63$:
(1) 004374 122737 000377 001415      CMPB   #377,INIFLG    ;ONLY DO AUTO SIZE ON 1ST START
(1) 004402 001006              BNE    16$           ;
(1) 004404 032777 000200 174546      BIT    #BIT7,@SWR    ;BIT7=1??
(1) 004412 001002              BNE    16$           ;BR IF NO AUTO SIZE
(1) 004414 004737 012254              JSR    PC,AUTO.SIZE ;GO DO THE AUTO SIZE
(1) 004420 105737 001416              TSTB   HDRFLG        ;HAS THE TABLE BEEN TYPED YET?
(1) 004424 001021              BNE    1$           ;IF SO, DON'T TYPE IT AGAIN
(1) 004426 105337 001416      DECB   HDRFLG        ;INDICATE THAT THE TABLE WILL BE TYPED
(1) 004432 104402 011040      TYPE   ,XHEAD        ;TYPE MAP HEADER
(1) 004436 012700 001500      MOV    #DZ.MAP,RO    ;SET POINTER
(1) 004442 010037 001220      MOV    RO,$TMP1        ;POINT TO THE MAP LOCATION
(1) 004446 012037 001222      MOV    (RO)+,$TMP2      ;SET DATA
(1) 004452 022737 177777 001222      CMP    #-1,$TMP2    ;END OF LIST?
(1) 004460 001403              BEQ     1$           ;BR IF YES
(1) 004462 104411              17$: CONVRT          ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
(1) 004464 011130              XSTATQ        ;CONVERT THE DATA AT THIS ADDRESS
(1) 004466 000765              BR     5$           ;GO PRINT THE NEXT PARAMETER
(1) 004470 005737 000042              TST    @#42         ;IS PROGRAM RUNNING UNDER MONITOR
(1) 004474 001026              BNE    3$           ;YES
(1) 004476 032777 000100 174454      BIT    #SW06,@SWR    ;DESELECT SPECIFIC DEVICES??
(1) 004504 001422              BEQ     3$           ;BR IF NO.
(1) 004506 104402 010761      TYPE   ,MNEW        ;TYPE THE MESSAGE.
(1) 004512 005000              CLR     RO           ;ZERO DATA DISPLAY
(1) 004514 000000              HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
(1) 004516 027737 174436 001312      CMP    @SWR,$DEVM    ;IS THE NUMBER VALID?
(1) 004524 101404              BLOS   2$           ;BR IF NUMBER IS OK.
(1) 004526 104402 010633      TYPE   ,MERR3        ;TELL USER OF INVALID NUMBER.
(1) 004532 000000              9$: HALT                    ;STOP EVERY THING.
(1) 004534 000776              BR     9$           ;RESTART THE PROGRAM AGAIN.
(1) 004536 017737 174416 001404 2$: MOV    @SWR,DZACTV    ;GET NEW DEVICE PATTERN
(1) 004544 013700 001404      MOV    DZACTV,RO      ;SHOW THE USER WHAT HE SELECTED.
(1) 004550 000000              HALT                    ;CONTINUE DYNAMIC SWITCHES.
(1) 004552 032777 000020 174400 3$: BIT    #SW04,@SWR    ;CHECK TO SEE IF DELAY COUNT CHANGES
(1) 004560 001407              BEQ     18$          ;IF NOT, GO CLEAR VECTOR AREA
(2)                                ; -$GETPAR-----
(2) 004562 104403              INSTR          ;CALL THE STRING INPUT ROUTINE
(2) 004564 004342              75$          ;POINTER TO MESSAGE TO BE PRINTED
```

```
(2) 004566 104405          PARAM          ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 004570 000001          1              ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 004572 177777          177777         ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2) 004574 007204          DLYCNT         ;POINTER TO MAP LOCATION TO BE FILLED
(2) 004576 000            .BYTE 0        ;MASK OF INVALID BITS FOR THIS PARAMETER
(2) 004577 001            .BYTE 1        ;NUMBER OF PARAMETERS TO STORE
(2)          ; -- END 0 MACRO =====
(1) 004600 012700 000300 18$: MOV #300,R0 ;PREPARE TO CLEAR THE FLOATING
(1) 004604 012701 000302      MOV #302,R1 ;VECTOR AREA. 300-776
(1) 004610 010120      4$: MOV R1,(R0)+ ;START PUTTING "PC+2 - HALT"
(1) 004612 005021      CLR (R1)+ ;IN VECTOR AREA.
(1) 004614 022021      CMP (R0)+,(R1)+ ;POP POINTERS
(1) 004616 022700 001000      CMP #1000,R0 ;ALL DONE??
(1) 004622 001372      BNE 4$ ;BR IF NO.
(1)          ;TEST START AND RESTART
(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 010657      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$: MOV BRW,TTST ;PREPARE NORMAL SCOPE ROUTINE
(1) 004674 012737 011532 001126 2$: MOV #CYCLE,$LPADR ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
(1) 004702 104402 010550      TYPE ,MR ;TYPE "RUNNING"
(1) 004706 000177 174214      JMP @LPADR ;START TESTING
(1)          ; -- END 0 MACRO =====
3644          ; -PRGEND-=====
```



```
(2) ;END OF PASS
(2) ;TYPE NAME OF TEST
(2) ;UPDATE PASS COUNT
(2) ;CHECK FOR EXIT TO ACT-11
(2) ;RESTART TEST
(3) .SBTTL END OF PASS ROUTINE
(3)
(4) ;*****
(3) ;*INCREMENT THE PASS NUMBER ($PASS)
(3) ;*IF THERES A MONITOR GO TO IT
(3) ;*IF THERE ISN'T JUMP TO CYCLE
(3)
(3) 004712 $EOP:
(5) ; -PASEND-----
(5) 004712 000004 SCOPE
(5) 004714 005037 001136 CLR $ERRPC ;CLEAR LAST ERROR PC
(5) 004720 105037 001123 CLR $ERFLG ;CLEAR ERROR FLAG
(5) 004724 104402 010525 TYPE ,MEPASS ;TYPE END PASS
(5) 004730 104402 010706 TYPE ,MCSRX ;TYPE CSR
(5) 004734 104412 005072 CNVRT ,XCSR ;SHOW IT
(5) 004740 104402 010714 TYPE ,MVECX ;TYPE VECTOR
(5) 004744 104412 005100 CNVRT ,XVEC ;SHOW IT
(5) 004750 005237 001242 INC $PASS ;RAISE PASS COUNT
(5) 004754 104402 010722 TYPE ,MPASSX ;TYPE PASSES
(5) 004760 104412 005106 CNVRT ,XPASS ;SHOW IT
(5) 004764 005337 001242 DEC $PASS ;RESTORE PASS COUNT
(5) 004770 104402 010733 TYPE ,MERRX ;TYPE ERRORS
(5) 004774 104412 005114 CNVRT ,XERR ;SHOW IT
(5) 005000 105337 001411 DECB SAVNUM ;ARE ALL DEVICES TESTED?
(5) 005004 001030 BNE $DOAGN ;BR IF NO.
(5) 005006 113737 001410 001411 MOV $DZNUM, SAVNUM ;RESTORE THE COUNT
(3) 005014 005037 001226 CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS.
(3) 005020 005237 001242 INC $PASS ;INCREMENT THE PASS NUMBER
(3) 005024 042737 100000 001242 BIC #100000, $PASS ;DON'T ALLOW A NEG. NUMBER
(3) 005032 005327 DEC (PC)+ ;LOOP?
(3) 005034 000001 $EOPCT: .WORD 1
(3) 005036 003013 BGT $DOAGN ;YES
(3) 005040 012737 MOV (PC)+, @ (PC)+ ;RESTORE COUNTER
(3) 005042 000001 $ENDCT: .WORD 1
(3) 005044 005034 $EOPCT
(3) 005046 013700 000042 $GET42: MOV @#42, R0 ;GET MONITOR ADDRESS
(3) 005052 001405 BEQ $DOAGN ;BRANCH IF NO MONITOR
(3) 005054 000005 RESET ;CLEAR THE WORLD
(3) 005056 004710 $ENDAD: JSR PC, (R0) ;GO TO MONITOR
(3) 005060 000240 NOP ;SAVE ROOM
(3) 005062 000240 NOP ;FOR
(3) 005064 000240 NOP ;ACT11
(3) 005066 $DOAGN: JMP @ (PC)+ ;RETURN
(3) 005070 011532 $RTNAD: .WORD CYCLE
(2)
(2) 005072 000001 XCSR: 1
(2) 005074 006 002 .BYTE 6,2
(2) 005076 002042 DZCSR
(2) 005100 000001 XVEC: 1
(2) 005102 003 002 .BYTE 3,2
```

```
(2) 005104 002072 DZRIV
(2) 005106 000001 XPASS: 1 .BYTE 6,2 $PASS
(2) 005110 006 002 XERR: 1 .BYTE 6,2 $ERTTL
(2) 005112 001242
(2) 005114 000001
(2) 005116 006 002
(2) 005120 001132
(2)
(2) ;SCOPE LOOP AND ITERATION HANDLER
(2) ;-----
(2)
(3) .SBTTL SCOPE HANDLER ROUTINE
(3)
(4) ;;*****
(3) ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(3) ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(3) ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(3) ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(3) ;*SW14=1 LOOP ON TEST
(3) ;*SW11=1 INHIBIT ITERATIONS
(3) ;*CALL
(3) ;* SCOPE ;;SCOPE=IOT
(3) OC5122 $SCOPE:
(5) (5) 005122 004737 007642 .SCOPE: JSR PC,SERV.G ; -SC-=====
(5) 005126 005037 001136 CLR $ERRPC ;FIND OUT IF <+G> WAS HIT
(5) 005132 022716 013040 CMP #TST1+2,(SP) ;CLEAR LAST ERROR PC.
(5) 005136 001417 BEQ $XTSTR ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
(5) 005140 000412 TTST: BR 1$ ;IF SO, DON'T LOOP ON IT
(5) 005142 105777 174016 TSTB @TKS ;GOTO 1$ (IF LOCK SW02=1; THIS LOC =240)
(5) 005146 100073 BPL $OVER ;KEYBOARD DONE?
(5) 005150 127727 174012 000021 CMPB @TKB,$XON ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
(5) 005156 001467 BEQ $OVER ;IS CHAR A RANDOM XON ? ;;DSH
(5) 005160 017766 174002 177776 MOV @TKB,-2(SP) ;BR IF YES ;;DSH
(5) ; -- END O MACRO =====
(3) 005166 032777 040000 173764 1$: BIT #BIT14,@SWR ;;LOOP ON PRESENT TEST?
(3) 005174 001060 BNE $OVER ;;YES IF SW14=1
(3) 005176 000416 ;####START OF CODE FOR THE XOR TESTER####
(3) $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3) MOV @ERRVEC,-(SP) ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3) 005200 013746 000004 MOV #5$,@ERRVEC ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3) 005204 012737 005224 000004 TST @177060 ;;SET FOR TIMEOUT
(3) 005212 005737 177060 MOV (SP)+,@ERRVEC ;;TIME OUT ON XOR?
(3) 005216 012637 000004 BR $SVLAD ;;RESTORE THE ERROR VECTOR
(3) 005222 000436 5$: CMP (SP)+,(SP)+ ;;GO TO THE NEXT TEST
(3) 005224 022626 MOV (SP)+,@ERRVEC ;;CLEAR THE STACK AFTER A TIME OUT
(3) 005226 012637 000004 BR $OVER ;;RESTORE THE ERROR VECTOR
(3) 005232 000441 6$::#####END OF CODE FOR THE XOR TESTER#### ;;LOOP ON THE PRESENT TEST
(3) 005234 105737 001123 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
(3) 005240 001404 BEQ 3$ ;;BR IF NO
(3) 005242 105037 001123 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
(3) 005246 005037 001226 3$: CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3) 005252 032777 004000 173700 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
(3) 005260 001011 BNE 1$ ;;BR IF YES
```



```

(3) 005262 005737 001242      TST      $PASS      ;;IF FIRST PASS OF PROGRAM
(3) 005266 001406              BEQ      1$          ;;      INHIBIT ITERATIONS
(3) 005270 005237 001124      INC      $ICNT      ;;INCREMENT ITERATION COUNT
(3) 005274 023737 001226 001124  CMP      $TIMES,$ICNT  ;;CHECK THE NUMBER OF ITERATIONS MADE
(3) 005302 002015              BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
(3) 005304 012737 000001 001124 1$:      MOV      #1,$ICNT  ;;REINITIALIZE THE ITERATION COUNTER
(3) 005312 013737 005374 001226      MOV      $MXCNT,$TIMES  ;;SET NUMBER OF ITERATIONS TO DO
(3) 005320 105237 001122      $SVLAD: INCB     $TSTNM      ;;COUNT TEST NUMBERS
(3) 005324 113737 001122 001240      MOV      $TSTNM,$TESTN  ;;SET TEST NUMBER IN APT MAILBOX
(3) 005332 011637 001126      MOV      (SP),$LPADR  ;;SAVE SCOPE LOOP ADDRESS
(3) 005336 013777 001122 173616 $OVER: MOV      $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER
(3) 005344 013716 001126      MOV      $LPADR,(SP)  ;;FUDGE RETURN ADDRESS
(5)                                ; -SC1-----
(5) 005350 105037 001417      3$:      CLRB     MNTFLG      ;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
(5) 005354 005737 001370      TST      MODE          ;HAS THE MODE BEEN CHANGED?
(5) 005360 001003              BNE      4$          ;IF NOT INTERNAL , GO DO A TEST
(5) 005362 112737 000010 001417      MOV      #MAINT,MNTFLG  ;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
(5) 005370 000002      4$:      RTI                    ;GO DO THE TEST
(5) 005372 000406      BRW:      406
(5)                                ; -- END O MACRO -----
(3) 005374 000005      $MXCNT: 5                      ;;MAX. NUMBER OF ITERATIONS
(1)
(1)                                ;CHECK FOR FREEZE ON CURRENT DATA
(1)                                ;-----
(1)
(1) 005376 032777 001000 173554 .SCOP1: BIT      #SW09,@SWR      ;IS SW09=1(SET)?
(1) 005404 001405              BEQ      1$          ;BR IF NOT SET.
(1) 005406 005737 001362      TST      LOCK          ;IS THER A TIGHT LOOP SPECIFIED?
(1) 005412 001402              BEQ      1$          ;IF NO, RETURN
(1) 005414 013716 001362      MOV      LOCK,(SP)      ;IF YES, GOTO THE ADDRESS IN LOCK.
(1) 005420 000002      1$:      RTI                    ;GO BACK.
(1)
(1) 005422 032777 010000 173530 .TYPE:  BIT      #SW12,@SWR      ;INHIBIT ALL PRINTOUT??
(1) 005430 001403              BEQ      1$          ;IF NOT, GO TYPE
(1) 005432 062716 000002      ADD      #2,(SP)      ;SKIP OVER MESSAGE POINTER
(1) 005436 000002      RTI                    ;RETURN TO WHERE PROCEDURE WAS INVOKED
(1) 005440
(2) 1$:
(2) .SBTTL  TYPE ROUTINE
(3)
(2) ;;*****
(2) ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(2) ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(2) ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(2) ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(2) ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(2) ;*
(2) ;*CALL:
(2) ;*1) USING A TRAP INSTRUCTION
(2) ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(2) ;*OR
(2) ;*      TYPE
(2) ;*      MESADR
(2) ;*
(2) 005440 105737 001177      $TYPE:  TSTB     $TPFLG      ;;IS THERE A TERMINAL?
(2) 005444 100002              BPL      1$          ;;BR IF YES

```

(2)	005446	000000			HALT		::HALT HERE IF NO TERMINAL	
(2)	005450	000430			BR	3\$	::LEAVE	
(2)	005452	010046		1\$:	MOV	R0,-(SP)	::SAVE R0	
(2)	005454	017600	000002		MOV	@2(SP),R0	::GET ADDRESS OF ASCIZ STRING	
(2)	005460	122737	000001	001254	CMPB	#APTENV,\$ENV	::RUNNING IN APT MODE	
(2)	005466	001011			BNE	62\$	::NO,G0 CHECK FOR APT CONSOLE	
(2)	005470	132737	000100	001255	BITB	#APTSPOOL,\$ENVM	::SPOOL MESSAGE TO APT	
(2)	005476	001405			BEQ	62\$	::NO,G0 CHECK FOR CONSOLE	
(2)	005500	010037	005510		MOV	R0,61\$	::SETUP MESSAGE ADDRESS FOR APT	
(2)	005504	004737	006002		JSR	PC,\$ATY3	::SPOOL MESSAGE TO APT	
(2)	005510	000000		61\$:	.WORD	0	::MESSAGE ADDRESS	
(2)	005512	132737	000040	001255	62\$:	BITB	#APTC SUP,\$ENVM	::APT CONSOLE SUPPRESSED
(2)	005520	001003			BNE	60\$	::YES,SKIP TYPE OUT	
(2)	005522	112046		2\$:	MOVB	(R0)+,-(SP)	::PUSH CHARACTER TO BE TYPED ONTO STACK	
(2)	005524	001005			BNE	4\$	::BR IF IT ISN'T THE TERMINATOR	
(2)	005526	005726			TST	(SP)+	::IF TERMINATOR POP IT OFF THE STACK	
(2)	005530	012600		60\$:	MOV	(SP)+,R0	::RESTORE R0	
(2)	005532	062716	000002	3\$:	ADD	#2,(SP)	::ADJUST RETURN PC	
(2)	005536	000002			RTI		::RETURN	
(2)	005540	122716	000011	4\$:	CMPB	#HT,(SP)	::BRANCH IF <HT>	
(2)	005544	001430			BEQ	8\$		
(2)	005546	122716	000200		CMPB	#CRLF,(SP)	::BRANCH IF NOT <CRLF>	
(2)	005552	001006			BNE	5\$		
(2)	005554	005726			TST	(SP)+	::POP <CR><LF> EQUIV	
(2)	005556	104402			TYPE		::TYPE A CR AND LF	
(2)	005560	001231			\$CRLF			
(2)	005562	105037	005770		CLRB	\$CHARCNT	::CLEAR CHARACTER COUNT	
(2)	005566	000755			BR	2\$	::GET NEXT CHARACTER	
(2)	005570	004737	005652	5\$:	JSR	PC,\$TYPEC	::GO TYPE THIS CHARACTER	
(2)	005574	123726	001176	6\$:	CMPB	\$FILLC,(SP)+	::IS IT TIME FOR FILLER CHARS.?	
(2)	005600	001350			BNE	2\$	::IF NO GO GET NEXT CHAR.	
(2)	005602	013746	001174		MOV	\$NULL,-(SP)	::GET # OF FILLER CHARS. NEEDED	
(2)							::AND THE NULL CHAR.	
(2)	005606	105366	000001	7\$:	DECB	1(SP)	::DOES A NULL NEED TO BE TYPED?	
(2)	005612	002770			BLT	6\$	::BR IF NO--GO POP THE NULL OFF OF STACK	
(2)	005614	004737	005652		JSR	PC,\$TYPEC	::GO TYPE A NULL	
(2)	005620	105337	005770		DECB	\$CHARCNT	::DO NOT COUNT AS A COUNT	
(2)	005624	000770			BR	7\$	::LOOP	
(2)								
(2)					;HORIZONTAL TAB PROCESSOR			
(2)								
(2)	005626	112716	000040	8\$:	MOVB	#' ,(SP)	::REPLACE TAB WITH SPACE	
(2)	005632	004737	005652	9\$:	JSR	PC,\$TYPEC	::TYPE A SPACE	
(2)	005636	132737	000007	005770	BITB	#7,\$CHARCNT	::BRANCH IF NOT AT	
(2)	005644	001372			BNE	9\$	::TAB STOP	
(2)	005646	005726			TST	(SP)+	::POP SPACE OFF STACK	
(2)	005650	000724			BR	2\$	::GET NEXT CHARACTER	
(2)	005652			\$TYPEC:				
(2)	005652	105777	173306		TSTB	@\$TKS	::CHAR IN KYBD BUFFER?	MJD001
(2)	005656	100022			BPL	10\$	::BR IF NOT	MJD001
(2)	005660	017746	173302		MOV	@\$TKB,-(SP)	::GET CHAR	MJD001
(2)	005664	042716	177600		BIC	#177600,(SP)	::STRIP EXTRANEIOUS BITS	MJD001
(2)	005670	122716	000023		CMPB	#XOFF,(SP)	::WAS CHAR XOFF	MJD001
(2)	005674	001012			BNE	102\$	::BR IF NOT	MJD001
(2)	005676			101\$:				MJD001
(2)	005676	105777	173262		TSTB	@\$TKS	::WAIT FOR CHAR	MJD001

CZDZA-IO MACY11 30A(1052) 02-JUL-85 16:19 PAGE 83-29
 CZDZAI.P11 02-JUL-85 16:17 TYPE ROUTINE

```

(2) 005702 100375      BPL      101$      ;MJD001
(2) 005704 117716 173256  MOVB    @TKB,(SP)  ;;GET CHAR      ;MJD001
(2) 005710 042716 177600  BIC     #177600,(SP)  ;;STRIP IT      ;MJD001
(2) 005714 122716 000021  CMPB    #$XON,(SP)  ;;WAS IT XON?   ;MJD001
(2) 005720 001366      BNE     101$      ;MJD001
(2) 005722      102$:      TST     (SP)+    ;;FIX STACK    ;MJD001
(2) 005722 005726      10$:      TSTB   @TPS      ;MJD001
(2) 005724 105777 173240  BPL     10$      ;MJD001
(2) 005730 100375      MOVB    2(SP),@TPB  ;;WAIT UNTIL PRINTER IS READY
(2) 005732 116677 000002 173232  CMPB    #CR,2(SP)  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2) 005740 122766 000015 000002  BNE     1$      ;;IS CHARACTER A CARRIAGE RETURN?
(2) 005746 001003      CLRB    $CHARCNT  ;;BRANCH IF NO
(2) 005750 105037 005770      BR     $TYPEX  ;;YES--CLEAR CHARACTER COUNT
(2) 005754 000406      CMPB    #LF,2(SP)  ;;EXIT
(2) 005756 122766 000012 000002 1$:      BEQ     $TYPEX  ;;IS CHARACTER A LINE FEED?
(2) 005764 001402      INCB    (PC)+    ;;BRANCH IF YES
(2) 005766 105227      $CHARCNT: .WORD 0  ;;COUNT THE CHARACTER
(2) 005770 000000      $TYPEX: RTS    PC  ;;CHARACTER COUNT STORAGE
(2) 005772 000207

(2)
(2)
(2)
(3)
(2) 005774 112737 000001 006240  ;;*****
(2) 006002 112737 000001 006236  $ATY1: MOVB   #1,$FFLG  ;;TO REPORT FATAL ERROR
(2) 006010 000403      $ATY3: MOVB   #1,$MFLG  ;;TO TYPE A MESSAGE
(2) 006012 112737 000001 006240  BR     $ATYC
(2) 006020      $ATY4: MOVB   #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
(2) 006020 010046      $ATYC:  MOV    R0,-(SP)  ;;PUSH R0 ON STACK
(2) 006022 010146      MOV    R1,-(SP)  ;;PUSH R1 ON STACK
(2) 006024 105737 006236      TSTB   $MFLG  ;;SHOULD TYPE A MESSAGE?
(2) 006030 001450      BEQ     5$      ;;IF NOT: BR
(2) 006032 122737 000001 001254  CMPB    #APTENV,$ENV  ;;OPERATING UNDER APT?
(2) 006040 001031      BNE     3$      ;;IF NOT: BR
(2) 006042 132737 000100 001255  BITB    #APTPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?
(2) 006050 001425      BEQ     3$      ;;IF NOT: BR
(2) 006052 017600 000004      MOV    @4(SP),R0  ;;GET MESSAGE ADDR.
(2) 006056 062766 000002 000004  ADD     #2,4(SP)  ;;BUMP RETURN ADDR.
(2) 006064 005737 001234      1$:      TST     $MSGTYPE  ;;SEE IF DONE W/ LAST XMISSION?
(2) 006070 001375      BNE     1$      ;;IF NOT: WAIT
(2) 006072 010037 001250      MOV    R0,$MSGAD  ;;PUT ADDR IN MAILBOX
(2) 006076 105720      2$:      TSTB   (R0)+    ;;FIND END OF MESSAGE
(2) 006100 001376      BNE     2$
(2) 006102 163700 001250      SUB     $MSGAD,R0  ;;SUB START OF MESSAGE
(2) 006106 006200      ASR     R0      ;;GET MESSAGE LGTH IN WORDS
(2) 006110 010037 001252      MOV    R0,$MSGLGT  ;;PUT LENGTH IN MAILBOX
(2) 006114 012737 000004 001234  MOV     #4,$MSGTYPE  ;;TELL APT TO TAKE MSG.
(2) 006122 000413      BR     5$
(2) 006124 017637 000004 006150 3$:      MOV    @4(SP),4$  ;;PUT MSG ADDR IN JSR LINKAGE
(2) 006132 062766 000002 000004  ADD     #2,4(SP)  ;;BUMP RETURN ADDRESS
(2) 006140 013746 177776      MOV    177776,-(SP)  ;;PUSH 177776 ON STACK
(2) 006144 004737 005440      JSR     PC,$TYPE  ;;CALL TYPE MACRO
(2) 006150 000000      4$:      .WORD 0
(2) 006152      5$:
(2) 006152 105737 006240      10$:      TSTB   $FFLG  ;;SHOULD REPORT FATAL ERROR?

```

Line	Address	Offset	Label	Instruction	Comment
(2)	006156	001416		BEQ 12\$	::IF NOT: BR
(2)	006160	005737	001254	TST \$ENV	::RUNNING UNDER APT?
(2)	006164	001413		BEQ 12\$	::IF NOT: BR
(2)	006166	005737	001234	11\$: TST \$MSGTYPE	::FINISHED LAST MESSAGE?
(2)	006172	001375		BNE 11\$	::IF NOT: WAIT
(2)	006174	017637	000004	MOV @4(SP), \$FATAL	::GET ERROR #
(2)	006202	062766	000002	ADD #2, 4(SP)	::BUMP RETURN ADDR.
(2)	006210	005237	001234	INC \$MSGTYPE	::TELL APT TO TAKE ERROR
(2)	006214	105037	006240	12\$: CLRB \$FFLG	::CLEAR FATAL FLAG
(2)	006220	105037	006237	CLRB \$LFLG	::CLEAR LOG FLAG
(2)	006224	105037	006236	CLRB \$MFLG	::CLEAR MESSAGE FLAG
(4)	006230	012601		MOV (SP)+, R1	::POP STACK INTO R1
(4)	006232	012600		MOV (SP)+, R0	::POP STACK INTO R0
(2)	006234	000207		RTS PC	::RETURN
(2)	006236	000	\$MFLG:	.BYTE 0	::MESSG. FLAG
(2)	006237	000	\$LFLG:	.BYTE 0	::LOG FLAG
(2)	006240	000	\$FFLG:	.BYTE 0	::FATAL FLAG
(2)		006242		.EVEN	
(2)		000200		APTSIZE=200	
(2)		000001		APTENV=001	
(2)		000100		APTSPool=100	
(2)		000040		APTCsUP=040	
(1)					
(1)					
(1)					
(1)					
(1)					
(1)	006242	010346		.INSTR: MOV R3, -(SP)	;SAVE R3 ON STACK
(1)	006244	010446		MOV R4, -(SP)	;SAVE R4 ON STACK
(1)	006246	017637	000004	MOV @4(SP), .MSG	;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1)	006254	062766	000002	ADD #2, 4(SP)	;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1)	006262	104402		.INST1: TYPE	;PRINT THE MESSAGE
(1)	006264	000000		.MSG: 0	;MESSAGE IS POINTED TO FROM HERE
(1)	006266	012704	011262	MOV #INBUF, R4	;POINT R4 TO THE INPUT BUFFER
(1)	006272	012703	000007	MOV #7, R3	;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1)	006276	105777	172662	1\$: TSTB @TKS	;HAS A CHARACTER BEEN RECEIVED?
(1)	006302	100375		BPL 1\$	;IF NO, KEEP WAITING FOR IT
(1)	006304	117714	172656	MOVB @TKB, (R4)	;IF YES, SAVE IT IN THE INPUT BUFFER
(1)	006310	142714	000200	BICB #200, (R4)	;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1)	006314	122714	000023	CMPB #\$XOFF, (R4)	;IS IT A XOFF?
(1)	006320	001014		BNE 83\$	;BR IF NOT
(1)	006322	105777	172636	101\$: TSTB @TKS	;WAIT FOR A CHARACTER
(1)	006326	100375		BPL 101\$	
(1)	006330	117714	172632	MOVB @TKB, (R4)	;GET CHARACTER
(1)	006334	142714	000200	BICB #200, (R4)	;STRIP DOWN CHARACTER
(1)	006340	122714	000021	CMPB #\$XON, (R4)	;WAIT FOR A XON?
(1)	006344	001366		BNE 101\$	;GET NEXT CHAR IF NOT
(1)	006346	105724		TSTB (R4)+	;POP STACK
(1)	006350	000752		BR 1\$	;WAIT FOR A CHAR
(1)	006352	122714	000021	83\$: CMPB #\$XON, (R4)	;IS IT A RANDOM XON
(1)	006356	001002		BNE 102\$	;BR IF NO
(1)	006360	105724		TSTB (R4)+	;ELSE, POP STACK
(1)	006362	000745		BR 1\$	;GO GET NEXT CHAR
(1)	006364	122724	000015	102\$: CMPB #15, (R4)-	;IS IT <CR> ?
(1)	006370	001417		BEQ INSTR2	;IF SO, TERMINATE THE INPUT SEQUENCE
(1)	006372	105777	172572	2\$: TSTB @TPS	;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1)	006376	100375		BPL 2\$	;IF WE CAN'T, WAIT UNTIL WE CAN

[illegible]

```

(1)
(1) 006600 013704 006632      MOV    DEVADR,R4      ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1) 006604 010524      1$:  MOV    R5,(R4)+      ;STORE THE RESULT
(1) 006606 062705 000002      ADD    #2,R5      ;CALCULATE THE NEXT DATUM
(1) 006612 105337 006635      DECB   ADCNT      ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1) 006616 001372      BNE     1$      ;IF NOT, GO STORE THE NEXT DATUM
(1) 006620 012604      MOV     (SP)+,R4      ;RESTORE R4
(1) 006622 012605      MOV     (SP)+,R5      ;RESTORE R5
(1) 006624 000002      RTI      ;RETURN TO THE MAIN PROGRAM
(1)
(1) 006626 000000      LOLIM:  0      ;LOWEST ACCEPTABLE VALUE
(1) 006630 000000      HILIM:  0      ;HIGHEST ACCEPTABLE
(1) 006632 000000      DEVADR:  0      ;LOCATION WHERE RESULT WILL BE STORED
(1) 006634      000      LOBITS: .BYTE 0      ;INCORRECT BITS MASK
(1) 006635      000      ADCNT:  .BYTE 0      ;COUNT OF ITEMS TO BE STORED
(1)
(1)      ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1)      ;-----
(1) 006636 016637 000004 001402 .SAV05: MOV    4(SP),SAVPC      ;SAVE R7 (PC)
(1)
(1)      ;SAVE R0-R5
(1)
(1) 006644 010537 001214      SV05:  MOV    R5,$REG5      ;SAVE R5
(1) 006650 010437 001212      MOV    R4,$REG4      ;SAVE R4
(1) 006654 010337 001210      MOV    R3,$REG3      ;SAVE R3
(1) 006660 010237 001206      MOV    R2,$REG2      ;SAVE R2
(1) 006664 010137 001204      MOV    R1,$REG1      ;SAVE R1
(1) 006670 010037 001202      MOV    R0,$REG0      ;SAVE R0
(1) 006674 000002      RTI      ;LEAVE.
(1)
(1)      ;RESTORE R0-R5
(1)
(1) 006676 013700 001202      .RES05: MOV    $REG0,R0      ;RESTORE R0
(1) 006702 013701 001204      MOV    $REG1,R1      ;RESTORE R1
(1) 006706 013702 001206      MOV    $REG2,R2      ;RESTORE R2
(1) 006712 013703 001210      MOV    $REG3,R3      ;RESTORE R3
(1) 006716 013704 001212      MOV    $REG4,R4      ;RESTORE R4
(1) 006722 013705 001214      MOV    $REG5,R5      ;RESTORE R5
(1) 006726 000002      RTI      ;LEAVE
(1)
(1)      ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(1)      ;-----
(1)
(1) 006730 104402 001231      .CONVR: TYPE    ,#CRLF      ;PRINT A CARRIAGE RETURN
(1) 006734 010046      .CNVRT: MOV    R0,-(SP)      ;SAVE R0
(1) 006736 010146      MOV    R1,-(SP)      ;SAVE R1
(1) 006740 010346      MOV    R3,-(SP)      ;SAVE R3
(1) 006742 010446      MOV    R4,-(SP)      ;SAVE R4
(1) 006744 010546      MOV    R5,-(SP)      ;SAVE R5
(1) 006746 017601 000012      MOV    @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
(1) 006752 062766 000002 000012      ADD    #2,12(SP)      ;POINT TO WHERE MAIN PROGRAM WILL RESUME
(1) 006760 012137 007104      MOV    (R1)+,WRDCNT      ;GET NUMBER OF WORDS TO BE PRINTED
(1) 006764 112105      1$:  MOVB   (R1)+,R5      ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
(1) 006766 112100      MOVB   (R1)+,R0      ;GET THE NUMBER OF SPACES TO PRINT
(1) 006770 013104      MOV    @R1)+,R4      ;COPY THE WORD TO BE CONVERTED

```



```

(1) 006772 110537 007106      MOV      R5,CHRCNT      ;COPY THE CHARACTER COUNT
(1) 006776 010403      3$:    MOV      R4,R3      ;COPY THE ARGUMENT WORD AGAIN
(1) 007000 042703 177770      BIC      #C<7>,R3      ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
(1) 007004 062703 000060      ADD      #060,R3      ;MAKE AN ASCII CHARACTER OUT OF THEM
(1) 007010 110346      MOV      R3,-(SP)      ;SAVE THAT CHARACTER
(1) 007012 006004      ROR      R4      ;MOVE THE NEXT THREE BITS INTO PLACE
(1) 007014 006204      ASR      R4      ;MOVE THEM AGAIN
(1) 007016 006204      ASR      R4      ;AND FINALLY A THIRD TIME
(1) 007020 005305      DEC      R5      ;REDUCE CHARACTER COUNT. ARE ALL CHARACTERS
(1)                                ;BUILT?
(1) 007022 001365      BNE      3$      ;IF NO, GO BUILD THE NEXT ONE.
(1) 007024 012703 011366      MOV      #MDATA,R3      ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1) 007030 112623      4$:    MOV      (SP)+,(R3)+      ;STORE THE CHARACTER, STARTING WITH THE MOST
(1) 007032 105337 007106      DECB     CHRCNT      ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1) 007036 001374      BNE      4$      ;IF NO, GO TRANSFER ANOTHER
(1) 007040 105700      TSTB     R0      ;ARE ANY SPACES TO BE PRINTED?
(1) 007042 001404      BEQ      6$      ;IF NO, DON'T SET UP ANY
(1) 007044 112723 000040      5$:    MOV      #040,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
(1) 007050 105300      DECB     R0      ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1) 007052 001374      BNE      5$      ;IF YES, GO ADD ANOTHER SPACE
(1) 007054 105013      6$:    CLRB     (R3)      ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
(1) 007056 104402 011366      TYPE     ,MDATA      ;PRINT THE STRING WE JUST BUILT
(1) 007062 005337 007104      DEC      WRDCNT      ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1) 007066 001336      BNE      1$      ;IF YES, GO CONVERT THEM
(1) 007070 012605      MOV      (SP)+,R5      ;RESTORE R5
(1) 007072 012604      MOV      (SP)+,R4      ;RESTORE R4
(1) 007074 012603      MOV      (SP)+,R3      ;RESTORE R3
(1) 007076 012601      MOV      (SP)+,R1      ;RESTORE R1
(1) 007100 012600      MOV      (SP)+,R0      ;RESTORE R0
(1) 007102 000002      RTI      ;RETURN TO THE MAIN PROGRAM
(1) 007104 000000      WRDCNT: 0
(1) 007106 000      CHRCNT: .BYTE
(1) 007107 000      SPACNT: .BYTE 0      ;NUMBER OF CHARACTERS TO PRINT
(1)                                ;NUMBER OF SPACES TO PRINT
(1) 007110 000000      BINWRD: 0
(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) 007112 010046      .TRPSR: MOV      R0,-(SP)      ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
(1) 007114 016600 000002      MOV      2(SP),R0      ;GET TRAP ADDRESS
(1) 007120 005740      TST      -(R0)      ;GET TRAP
(1) 007122 111000      MOV      (R0),R0      ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
(1) 007124 006300      ASL      R0      ;POSITION OFFSET FOR TABLE INDEXING
(1) 007126 016000 002002      MOV      .TRPTAB(R0),R0      ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1) 007132 000200      RTS      R0      ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                                ;DEVICE CLEAR ROUTINE
(1)                                ;ISSUE A DEVICE CLEAR
(1)                                ;-----
(1) 007134      .DEVICE.CLR:
(1) 007134 052777 000020 172700      BIS      #DCLR,#DZCSR      ;SET DCLR
(1) 007142 032777 000020 172672      1$:    BIT      #DCLR,#DZCSR      ;DID IT CLEAR?

```

```

(1) 007150 001374      BNE      1$      ;BR IF NO
(1) 007152 000002      RTI          ;EXIT ROUTINE
(1)
(1)                      ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                      ;-----
(1) 007154 104413      .DCLASM:DEVICE.CLR      ;ISSUE A DEVICE CLEAR
(1) 007156 153777 001417 172656      BISB      MNTFLG,@DZCSR      ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1) 007164 000002      RTI          ;RETURN TO CALLING ROUTINE
(1)
(1) 007166      .DELAY:
(1) 007166 010046      MOV      R0,-(SP)      ;SAVE R0
(1) 007170 013700 007204      MOV      DLYCNT,R0      ;SET COUNT
(1) 007174 005300      1$:      DEC      R0      ;DELAY
(1) 007176 001376      BNE      1$      ;
(1) 007200 012600      MOV      (SP)+,R0      ;RESTORE R0
(1) 007202 000002      RTI          ;LEAVE ROUTINE
(1) 007204 000001      DLYCNT: .WORD      1      ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                      ;ADVANCE TO NEXT TEST HANDLER
(1)                      ;-----
(1) 007206 013716 001360      .ADVANCE:MOV      NEXT,(SP)      ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
(1) 007212 005037 001362      CLR      LOCK      ;RESET TIGHT LOOP ADDRESS
(1) 007216 000002      RTI          ;CHECK TO SEE IF OLD TEST GETS REPEATED
(1)
(1)                      ;ERROR HANDLER
(1)                      ;-----
(1) 007220 004737 007642      $ERROR: JSR      PC,SERV.G      ;FIND OUT IF <+G> WAS HIT
(1) 007224 032777 010000 171726      BIT      @SW12,@SWR      ;BELL ON ERROR?
(1) 007232 001406      BEQ      XBX      ;BR IF NO BELL
(1) 007234 105777 171730      TSTB      @+TPS      ;TTY READY.
(1) 007240 100003      BPL      XBX      ;DON'T WAIT IF TTY NOT READY.
(1) 007242 112777 000207 171722      MOV      @207,@+TPB      ;PUSH A BELL AT THE TTY.
(1) 007250 032777 020000 171702      XB$: BIT      @SW13,@SWR      ;DELETE ERROR PRINT OUT?
(1) 007256 001113      BNE      HALTS      ;BR IF NO PRINT OUT WANTED.
(1) 007260 021637 001136      CMP      (SP), $ERRPC      ;WAS THIS ERROR FOUND LAST TIME?
(1) 007264 001404      BEQ      1$      ;BR IF YES
(1) 007266 011637 001136      MOV      (SP), $ERRPC      ;RECORD BEING HERE
(1) 007272 105037 001123      CLR      $ERFLG      ;PREPARE HEADER
(1) 007276 104407      1$:      SAVO5      ;SAVE ALL PROC REGISTERS
(1) 007300 011605      MOV      (SP),R5      ;GET THE PC OF ERROR
(1) 007302 162705 000002      SUB      @2,R5      ;GET ADDRESS OF TRAP CALL
(1) 007306 011504      MOV      (R5),R4      ;GET ERROR INSTRUCTION
(1) 007310 110437 001134      MOV      R4,$ITEMB      ;COPY TEST NUMBER FOR APT HANDLING
(1) 007314 006304      ASL      R4      ;MULT BY TWO
(1) 007316 061504      ADD      (R5),R4      ;DOUBLE IT
(1) 007320 006304      ASL      R4      ;MULT AGAIN
(1) 007322 042704 177001      BIC      @177001,R4      ;CLEAR JUNK
(1) 007326 062704 030340      ADD      @.ERRTAB,R4      ;GET POINTER
(1) 007332 012437 007456      MOV      (R4)+,ERRMSG      ;GET ERROR MESSAGE
(1) 007336 012437 007470      MOV      (R4)+,DATAHD      ;GET DATA HEADRER
(1) 007342 011437 007502      MOV      (R4),DATABP      ;GET DATA TABLE
(1) 007346 105737 001123      TSTB      $ERFLG      ;TYPE HEADER
(1) 007352 001403      BEQ      TYPMSG      ;BR IF YES
(1) 007354 005737 007502      TST      DATABP      ;DOES DATA TABLE EXIST?

```


Line	Address	Offset	Label	Operation	Comments
(1)	007360	001044		BNE	TYPDAT
(1)	007362	104402	001231	TYPE	, \$CRLF
(1)	007366	104402	001231	TYPE	, \$CRLF
(1)	007372	005737	001362	TST	LOCK
(1)	007376	001402		BEQ	1\$
(1)	007400	104402	010756	TYPE	, MASTEK
(1)	007404	104402	010744	1\$: TYPE	, MTSTN
(1)	007410	104412	007634	CNVRT	, XTSTN
(1)	007414	104402	011033	TYPE	, MERRPC
(1)	007420	104412	007626	CNVRT	, ERTABO
(1)	007424	104402	010706	TYPE	, MCSRX
(1)	007430	104412	005072	CNVRT	, XCSR
(1)	007434	104402	001231	TYPE	, \$CRLF
(1)	007440	112737	177777	001123	MOVB
(1)	007446	005737	007456	TST	ERRMSG
(1)	007452	001402		BEQ	WTBS.FM
(1)	007454	104402		TYPE	
(1)	007456	000000		ERRMSG: 0	
(1)	007460			WTBS.FM:	
(1)	007460	005737	007470	TST	DATAHD
(1)	007464	001402		BEQ	TYPDAT
(1)	007466	104402		TYPE	
(1)	007470	000000		DATAHD: 0	
(1)	007472	005737	007502	TYPDAT: TST	DATABP
(1)	007476	001402		BEQ	RESREG
(1)	007500	104411		CNVRT	
(1)	007502	000000		DATABP: 0	
(1)	007504	104410		RESREG: RES05	
(1)	007506	122737	000001	001254	HALTS: CMPB
(1)	007514	001007		BNE	\$APTENV, \$ENV
(1)	007516	113737	001134	007530	MOVB
(1)	007524	004737	006012	JSR	\$ITEMB, 7\$
(1)	007530	000000		7\$: .WORD	0
(1)	007532	000777		8\$: BR	8\$
(1)	007534	022737	005056	000042	2\$: CMP
(1)	007542	001403		BEQ	\$ENDAD, @42
(1)	007544	005777	171410	TST	\$SWR
(1)	007550	100004		BPL	EXITER
(1)	007552	016677	000002	171402	1\$: MOV
(1)	007560	000000		HALT	2(SP), @DISPLAY
(1)	007562	005237	001132	EXITER: INC	\$ERTTL
(1)	007566	032777	000400	171364	BIT
(1)	007574	001007		BNE	\$SW08, @SWR
(1)	007576	032777	002000	171354	BIT
(1)	007604	001407		BEQ	2\$
(1)	007606	013737	001360	001126	MOV
(1)	007614	012706	001120	1\$: MOV	NEXT, \$LPADR
(1)	007620	000177	171302	JMP	\$STACK, SP
(1)	007624	000002		2\$: RTI	@\$LPADR
(1)	007626	000001		ERTABO: 1	
(1)	007630	006	002	.BYTE	6,2
(1)	007632	001402		SAVPC	
(1)	007634	000001		XTSTN: 1	
(1)	007636	002	002	.BYTE	2,2
(1)	007640	001122		\$TSTNM	
(1)	007642	022737	177570	001160	SERV.G: CMP

(1)	007650	001002			BNE	99\$		;IF SO, IGNORE +G	
(1)	007652	000137	010260		JMP	6\$			::DSH-BHL
(1)	007656	017746	171304	99\$:	MOV	@TKB, -(SP)		;OTHERWISE, GET THE LAST CHARACTER TYPED	
(1)	007662	042716	177600		BIC	#177600, (SP)		;STRIP CHAR	::DSH-BHL
(1)	007666	122716	000023		CMPB	#XOFF, (SP)		;IS IT A XOFF	::DSH-BHL
(1)	007672	001012			BNE	102\$		;BR IF NO	::DSH-BHL
(1)									
(1)	007674	105777	171264	101\$:	TSTB	@TKS		;WAIT FOR A CHAR	::DSH-BHL
(1)	007700	100375			BPL	101\$			::DSH-BHL
(1)	007702	117716	171260		MOVB	@TKB, (SP)		;GET THE CHAR	::DSH-BHL
(1)	007706	042716	177600		BIC	#177600, (SP)		;STRIP CHAR	::DSH-BHL
(1)	007712	122716	000021		CMPB	#XON, (SP)		;IS IT A XON	::DSH-BHL
(1)	007716	001366			BNE	101\$		;BR IF NO	::DSH-BHL
(1)									
(1)	007720	122716	000021	102\$:	CMPB	#XON, (SP)		;IS IT RANDOM XON ?	::DSH
(1)	007724	001002			BNE	7\$		;BR IF NOT	::DSH
(1)	007726	005726			TST	(SP)+		;POP STACK	::DSH
(1)	007730	000553			BR	6\$		;IGNORE XON CHAR	::DSH
(1)									
(1)	007732	122726	000007	7\$:	CMPB	#7, (SP)+		;IS IT +G?	
(1)	007736	001150			BNE	6\$		;IF NOT, IGNORE INPUT	
(1)	007740	032777	004000	171216	BIT	#4000, @TKS		;RX BUSY?	
(1)	007746	001335			BNE	SERV.G		;BR IF YES	
(1)	007750	017737	171204	010302	MOV	@SWR, 90\$		;SAVE (SWR).	
(1)	007756	013777	010302	171174	1\$:	MOV	90\$, @SWR		
(1)	007764	104402	010262		TYPE	, 89\$			
(1)	007770	104412	010274		CNVRT	, 88\$			
(1)	007774	104402	010304		TYPE	, 91\$			
(1)	010000	105777	171160	9\$:	TSTB	@TKS		;WAIT FOR DONE.	
(1)	010004	100375			BPL	, -4			
(1)	010006	017746	171154		MOV	@TKB, -(SP)			
(1)	010012	042716	177600		BIC	#177600, (SP)		;STRIP CHAR	::DSH-BHL
(1)	010016	122716	000023		CMPB	#XOFF, (SP)		;IS IT A XOFF	::DSH-BHL
(1)	010022	001012			BNE	112\$		;BR IF NO	::DSH-BHL
(1)									
(1)	010024	105777	171134	111\$:	TSTB	@TKS		;WAIT FOR A CHAR	::DSH-BHL
(1)	010030	100375			BPL	111\$			::DSH-BHL
(1)	010032	117716	171130		MOVB	@TKB, (SP)		;GET THE CHAR	::DSH-BHL
(1)	010036	042716	177600		BIC	#177600, (SP)		;STRIP CHAR	::DSH-BHL
(1)	010042	122716	000021		CMPB	#XON, (SP)		;IS IT A XON	::DSH-BHL
(1)	010046	001366			BNE	111\$		;BR IF NO	::DSH-BHL
(1)									
(1)	010050	122716	000021	112\$:	CMPB	#XON, (SP)		;IS IT RANDOM XON ?	::DSH
(1)	010054	001002			BNE	8\$		;BR IF NOT	::DSH
(1)	010056	005726			TST	(SP)+		;POP STACK	::DSH
(1)	010060	000747			BR	9\$		;IGNORE XON CHAR	::DSH
(1)									
(1)	010062	122726	000015	8\$:	CMPB	#15, (SP)+			
(1)	010066	001472			BEQ	5\$			
(1)	010070	005077	171064		CLR	@SWR			
(1)	010074	105777	171070	2\$:	TSTB	@TPS			
(1)	010100	100375			BPL	, -4			
(1)	010102	016677	177776	171062	MOV	-2(SP), @TPB			
(1)	010110	000241			CLC				
(1)	010112	006177	171042		ROL	@SWR			
(1)	010116	006177	171036		ROL	@SWR			


```

(1) 010122 006177 171032 ROL @SWR ;
(1) 010126 103713 BCS 1$ ;ERROR
(1) 010130 026627 177776 000060 CMP -2(SP),#60 ;
(1) 010136 002707 BLT 1$ ;
(1) 010140 026627 177776 000067 CMP -2(SP),#67 ;
(1) 010146 003303 BGT 1$ ;
(1) 010150 042766 177770 177776 BIC #C<7>,-2(SP) ;
(1) 010156 056677 177776 170774 BIS -2(SP),@SWR ;
(1) 010164 105777 170774 12$: TSTB @TKS ;
(1) 010170 100375 BPL -4 ;
(1) 010172 017746 170770 MOV @TKB,-(SP) ;
(1) 010176 042716 177600 BIC #177600,(SP) ;STRIP CHAR ;:DSH-BHL
(1) 010202 122716 000023 CMPB @XOFF,(SP) ;IS IT A XOFF ;:DSH-BHL
(1) 010206 001012 BNE 122$ ;BR IF NO ;:DSH-BHL
(1)
(1) 010210 105777 170750 121$: TSTB @TKS ;WAIT FOR A CHAR ;:DSH-BHL
(1) 010214 100375 BPL 121$ ;:DSH-BHL
(1) 010216 117716 170744 MOV @TKB,(SP) ;GET THE CHAR ;:DSH-BHL
(1) 010222 042716 177600 BIC #177600,(SP) ;STRIP CHAR ;:DSH-BHL
(1) 010226 122716 000021 CMPB @XON,(SP) ;IS IT A XON ;:DSH-BHL
(1) 010232 001366 BNE 121$ ;BR IF NO ;:DSH-BHL
(1)
(1) 010234 122716 000021 122$: CMPB @XON,(SP) ;IS IT RANDOM XON ? ;:DSH
(1) 010240 001002 BNE 10$ ;BR IF NOT ;:DSH
(1) 010242 005726 TST (SP)+ ;POP STACK ;:DSH
(1) 010244 000747 BR 12$ ;IGNORE XON CHAR ;:DSH
(1)
(1) 010246 122726 000015 10$: CMPB #15,(SP)+ ;
(1) 010252 001310 BNE 2$ ;
(1) 010254 104402 001231 5$: TYPE ,CRLF ;
(1) 010260 000207 6$: RTS PC ;
(1)
(1) 010262 020200 051450 051127 89$: .ASCIZ <200>? (SWR)=/?
(1) 010270 036451 000057 .EVEN
(1) 010274 000001 88$: 1
(1) 010276 006 000 .BYTE 6,0
(1) 010300 010302 90$:
(1) 010302 000000 .WORD 0
(1) 010304 036457 000057 91$: .ASCIZ ?/?
(1) .EVEN
(2) .SBTTL POWER DOWN AND UP ROUTINES
(2)
(3) ;*****
(2) ;POWER DOWN ROUTINE
(2) 010310 012737 010454 000024 $PWRDN: MOV @ILLUP,@PWRVEC ;:SET FOR FAST UP
(2) 010316 012737 000340 000026 MOV #340,@PWRVEC+2 ;:PRIO:7
(4) 010324 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
(4) 010326 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
(4) 010330 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
(4) 010332 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
(4) 010334 010446 MOV R4,-(SP) ;:PUSH R4 ON STACK
(4) 010336 010546 MOV R5,-(SP) ;:PUSH R5 ON STACK
(4) 010340 017746 170614 MOV @SWR,-(SP) ;:PUSH @SWR ON STACK
(2) 010344 010637 010460 MOV SP,$SAVR6 ;:SAVE SP
(2) 010350 012737 010362 000024 MOV @PWRUP,@PWRVEC ;:SET UP VECTOR

```

```

(2) 010356 000000          HALT
(2) 010360 000776          BR      .-2          ;;HANG UP
(2)
(3) ;;*****
(2) ;POWER UP ROUTINE
(2) 010362 012737 010454 000024 $PWRUP: MOV    $$ILLUP,@PWRVEC ;;SET FOR FAST DOWN
(2) 010370 013706 010460          MOV    $SAVR6,SP      ;;GET SP
(2) 010374 005037 010460          CLR     $SAVR6        ;;WAIT LOOP FOR THE TTY
(2) 010400 005237 010460          1$: INC     $SAVR6      ;;WAIT FOR THE INC
(2) 010404 001375          BNE     1$                   ;;OF WORD
(4) 010406 012677 170546          MOV    (SP)+,@SWR      ;;POP STACK INTO @SWR
(4) 010412 012605          MOV    (SP)+,R5              ;;POP STACK INTO R5
(4) 010414 012604          MOV    (SP)+,R4              ;;POP STACK INTO R4
(4) 010416 012603          MOV    (SP)+,R3              ;;POP STACK INTO R3
(4) 010420 012602          MOV    (SP)+,R2              ;;POP STACK INTO R2
(4) 010422 012601          MOV    (SP)+,R1              ;;POP STACK INTO R1
(4) 010424 012600          MOV    (SP)+,R0              ;;POP STACK INTO R0
(2) 010426 012737 010310 000024 MOV    $$PWRDN,@PWRVEC ;;SET UP THE POWER DOWN VECTOR
(2) 010434 012737 000340 000026 MOV    #340,@PWRVEC+2 ;;PRIO:7
(2) 010442 104402          TYPE
(2) 010444 010462          $PWRMG: .WORD  MPFAIL        ;;REPORT THE POWER FAILURE
(2) 010446 012716          MOV    (PC)+,(SP)           ;;POWER FAIL MESSAGE POINTER
(2) 010450 012076          $PWRAD: .WORD  RESTART       ;;RESTART AT RESTART
(2) 010452 000002          RTI                          ;;RESTART ADDRESS
(2) 010454 000000          $ILLUP: HALT                ;;THE POWER UP SEQUENCE WAS STARTED
(2) 010456 000776          BR      .-2                  ;; BEFORE THE POWER DOWN WAS COMPLETE
(2) 010460 000000          $SAVR6: 0                    ;;PUT THE SP HERE
(2) 010462 050200 051127 043040 MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 010525      200 047105 020104 MEPASS: .ASCIZ  <200>/END PASS CZDZA-I /
(2) 010550 051200 047125 044516 MR: .ASCIZ  <200>/RUNNING /
(2) 010564 050200 047522 051107 MERR2: .ASCIZ  <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 010633      200 047111 052523 MERR3: .ASCIZ  <200>/INSUFFICIENT DATA!/
(2) 010657      200 047514 045503 MLOCK: .ASCIZ  <200>/LOCK ON SELECTED TEST/
(2) 010706 051503 035122 000040 MCSRX: .ASCIZ  /CSR: /
(2) 010714 042526 035103 000040 MVECX: .ASCIZ  /VEC: /
(2) 010722 040520 051523 051505 MPASSX: .ASCIZ  /PASSES: /
(2) 010733      105 051122 051117 MERRX: .ASCIZ  /ERRORS: /
(2) 010744 042524 052123 047040 MTSTN: .ASCIZ  /TEST NO: /
(2) 010756 020052      000 MASTEK: .ASCIZ  /* /
(2) 010761      200 042523 020124 MNEW: .ASCIZ  <200>/SET SWITCH REG TO DZ11'S DESIRED ACTIVE./
(2) 011033      120 035103 000040 MERRPC: .ASCIZ  /PC: /
(2) 011040 046600 050101 047440 XHEAD: .ASCIZ  <200>/MAP OF DZ11 STATUS/<200>
(2) 011065      200 046111 042514 MBADLN: .ASCIZ  <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2) 011130 011130          .EVEN
(2) 011130 000002          XSTATQ: 2
(2) 011132      006      003          .BYTE  6,3
(2) 011134 001220          $TMP1
(2) 011136      006      002          .BYTE  6,2
(2) 011140 001222          $TMP2
(1)
(2) .EVEN
(2)
(2) ; - $SETFLG-----
(2) ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
(2) ;-----
(2) ;E=EXTERNAL LOOP BACK
(2) ;I=INTERNAL LOOP BACK
(2) ;S=STAGGERED LOOP BACK

```



```
(2) 011142 017605 000000 .SETFLG:MOV 8(SP),R5 ;PICK UP ADDRESS OF TAG
(2) 011146 042737 000040 011262 BIC 40,INBUF ;STRIP LOWER CASE
(2) 011154 122737 000105 011262 CMPB #'E,INBUF ;IS IT EXTERNAL LOOP BACK ?
(2) 011162 001005 BNE 4$ ;NO
(2) 011164 013715 011254 MOV 1$,(R5) ;YES STORE INFO
(2) 011170 105037 001417 CLRB MNTFLG ;SET MAINT BIT =0
(2) 011174 000422 BR 7$ ;GET OUT
(2) 011176 122737 000111 011262 4$: CMPB #'I,INBUF ;IS IT INTERNAL LOOP BACK ?
(2) 011204 001006 BNE 5$ ;NO
(2) 011206 013715 011256 MOV 2$,(R5) ;YES STORE INFO
(2) 011212 112737 000010 001417 MOVB #MAINT,MNTFLG ;SET UP THE MAINTENANCE FLAG LOADER
(2) 011220 000410 BR 7$ ;GET OUT
(2) 011222 122737 000123 011262 5$: CMPB #'S,INBUF ;IS IT STAGGERED LOOP BACK ?
(2) 011230 001007 BNE 6$ ;WHAT ?
(2) 011232 013715 011260 MOV 3$,(R5) ;YES STORE INFO
(2) 011236 105037 001417 CLRB MNTFLG ;ZERO BITS
(2) 011242 062716 000002 7$: ADD #2,(SP) ;POP AROUND
(2) 011246 000002 RTI
(2) 011250 104404 6$: INSTER ;RETRY
(2) 011252 000733 BR .SETFLG ;DITTO
(2) 011254 000200 1$: .WORD 200 ;EXTERNAL = E
(2) 011256 000000 2$: .WORD 0 ;INTERNAL = I
(2) 011260 100000 3$: .WORD 100000 ;STAGGERED = S
(2) ; -- END O MACRO =====
(2) ; -$BUFFER-----
(2)
(2) ;BUFFERS FOR INPUT-OUTPUT
(2) 011262 000000 INBUF: 0
(2) 011324 . = .+40
(2) 011324 000000 TEMP: 0
(2) 011366 . = .+40
(2) 011366 000000 MDATA: 0
(2) 011430 . = .+40
(2) 011430 011637 011526 SET.PS: MOV (SP),3$
(2) 011434 162737 000002 011526 SUB #2,3$
(2) 011442 017737 000060 011530 MOV #3$,4$
(2) 011450 022737 106427 011530 CMP #106427,4$
(2) 011456 001003 BNE 1$
(2) 011460 011637 011526 MOV (SP),3$
(2) 011464 000412 BR 2$
(2) 011466 022737 106437 011530 1$: CMP #106437,4$
(2) 011474 001401 BEQ .+4
(2) 011476 000000 HALT ;RESERVED INSTRUCTION NOT "MTPS"
(2) 011500 011637 011526 MOV (SP),3$
(2) 011504 017737 000016 011526 MOV #3$,3$
(2) 011512 062716 000002 2$: ADD #2,(SP)
(2) 011516 017766 000004 000002 MOV #3$,2(SP)
(2) 011524 000002 RTI
(2) 011526 000000 3$: 0
(2) 011530 000000 4$: 0
```

```
(2) ; -- END O MACRO =====
(2) ; -$CYCLE-=====
(2) ;
(2) ;ROUTINE USED TO "CYCLE" THROUGH UP TO SIXTEEN DZ11'S
(2) ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
(2) ;AND RUNS THE SPECIFIED DZ11'S. THIS ROUTINE *MUST*
(2) ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
(2) ;SETUP NECESSARY.
(2) ;
(2) 011532 005737 001404 CYCLE: TST DZACTV ;ARE ANY DZ11'S TO BE TESTED?
(2) 011536 001004 BNE 1$ ;BR IF OK.
(2) 011540 104402 010564 TYPE ,MERR2 ;NO DZ11'S SELECTED!!
(2) 011544 000000 HALT ;STOP THE SHOW.
(2) 011546 000776 BR -.2 ;DISQUALIFY CONT. SW.
(2) 011550 013737 005374 001226 1$: MOV $MXCNT,$TIMES ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
(2) 011556 033737 001406 001404 BIT RUN,DZACTV ;IS THIS ONE "ACTIVE"
(2) 011564 001020 BNE 2$ ;BR IF GOOD ONE FOUND.
(2) 011566 000241 CLC
(2) 011570 006137 001406 ROL RUN ;UPDATE POINTER
(2) 011574 005537 001406 ADC RUN ;CATCH CARRY FROM RUN
(2) 011600 062737 000014 001412 ADD #14,ACTIVE ;UPDATE ADDRESS POINTER.
(2) 011606 022737 002000 001412 CMP #DZ.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
(2) 011614 001355 BNE 1$ ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
(2) 011616 012737 001500 001412 MOV #DZ.MAP,ACTIVE ;RESET ADDRESS POINTER.
(2) 011624 000751 BR 1$ ;KEEP LOOKING FOR ACTIVE DZ11
(2) 011626 000241 2$: CLC
(2) 011630 006137 001406 ROL RUN ;UPDATE POINTER.
(2) 011634 005537 001406 ADC RUN ;CATCH CARRY.
(2) 011640 013700 001412 MOV ACTIVE,RO ;GET ADDRESS POINTER.
(2) 011644 062737 000014 001412 ADD #14,ACTIVE ;UPDATE.
(2) 011652 022737 002000 001412 CMP #DZ.END,ACTIVE
(2) 011660 001003 BNE 3$ ;ALL DONE?
(2) 011662 012737 001500 001412 MOV #DZ.MAP,ACTIVE ;BR IF NO.
(2) 011670 012037 001310 3$: MOV (RO)+,$BASE ;RESTORE POINTER.
(2) 011674 012037 002072 MOV (RO)+,DZRIV ;LOAD SYSTEM CTRL. REG
(2) 011700 012037 030334 MOV (RO)+,DZPRT ;LOAD VECTOR
(2) 011704 113737 030335 001414 MOV DZPRT+1,EIAFLG ;LOAD PRIORITY
(2) 011712 042737 100000 030334 BIC #BIT15,DZPRT ;EIA OR 20MA
(2) 011720 012037 001364 MOV (RO)+,LINE ;CLEAR FLAG
(2) 011724 012037 001366 MOV (RO)+,PAR ;SET UP LINE DZ LINES ACTIVE
(2) 011730 012037 001370 MOV (RO)+,PAR ;SET UP PARAMETERIZATION
(2) 011734 004737 030126 MOV (RO)+,MODE ;SET UP MAINTENANCE MODE
(2) 011740 005737 000042 JSR PC,DZLEV ;SET UP
(2) 011744 001051 TST @#42 ;ARE WE UNDER MONITOR CONTROL?
(2) 011746 032777 000002 167204 BNE 4$ ;IF YES, SKIP THIS SETUP
(2) 011754 001445 BIT #SW01,@SWR ;IF SW01=1, GET STARTING TEST #
(2) 011756 104402 001231 7$: BEQ 4$ ;BR IF NO TEST IS TO BE INPUTTED
(2) 011762 104403 ; -$GETPAR-=====
(2) 011764 010744 INSTR ;CALL THE STRING INPUT ROUTINE
(2) 011766 104405 MTSTN ;POINTER TO MESSAGE TO BE PRINTED
(2) 011770 000001 PARAM ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2) 011772 001000 1 ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1000 ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
```



```

(3) 011774 001122          $TSTNM
(3) 011776      000          .BYTE 0          ;POINTER TO MAP LOCATION TO BE FILLED
(3) 011777      001          .BYTE 1          ;MASK OF INVALID BITS FOR THIS PARAMETER
(3)                                     ;NUMBER OF PARAMETERS TO STORE
(3)                                     ; -- END O MACRO -----
(2) 012000 012700 013036      MOV      #TST1,R0
(2) 012004 022710 000004      5$:      CMP      #4,(R0)
(2) 012010 001020          BNE      6$
(2) 012012 022760 012737 000002      CMP      #12737,2(R0)
(2) 012020 001014          BNE      6$
(2) 012022 023760 001122 000004      CMP      $TSTNM,4(R0)      ;IS THIS THE TEST ?
(2) 012030 001010          BNE      6$      ;IF NOT, DON'T PROCESS NUMBER
(2) 012032 010037 001126      MOV      R0,$LPADR      ;SAVE PC
(2) 012036 062737 000002 001126      ADD      #2,$LPADR      ;POP OVER SCOPE
(2) 012044 104402 001231      TYPE      , $CRLF
(2) 012050 000412          BR      8$
(2) 012052 005720          6$:      TST      (R0)+
(2) 012054 020027 023730      CMP      R0,#TLAST+10
(2) 012060 001351          BNE      5$
(2) 012062 104402 001230      TYPE      , $QUES
(2) 012066 000733          BR      7$
(2) 012070 012737 013036 001126 4$:      MOV      #TST1,$LPADR      ;PREPARE TEST ADDRESS
(2) 012076          8$:
(2) 012076 000177 167024      RESTART:JMP      @ $LPADR      ;GO START TESTING.***WARNING!****
(2)                                     ;THIS JUMP IS USED BY POWER UP ROUTINE!!!!
  
```

```
(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)
(2) 012102 012700 001500      SETAPT: MOV    #DZ.MAP,R0      ;POINT TO THE DEVICE MAP TABLE
(2) 012106 013701 001310      MOV    $BASE,R1      ;BUILD DEVICE ADDRESSES IN R1
(2) 012112 013702 001304      MOV    $VECT1,R2     ;BUILD DEVICE VECTORS IN R2
(2) 012116 042702 177007      BIC     #C<770>,R2    ;STRIP AWAY OTHER INFORMATION
(2)
(2) 012122 113703 001305      MOVB    $VECT1+1,R3   ;LOAD THE INTERRUPT PRIORITY FROM R3
(2) 012126 106003             RORB    R3            ;ALIGN THE NUMBER
(2) 012130 106003             RORB    R3            ;ALIGN THE NUMBER
(2) 012132 106003             RORB    R3            ;ALIGN THE NUMBER
(2) 012134 106003             RORB    R3            ;ALIGN THE NUMBER
(2) 012136 106003             RORB    R3            ;ALIGN THE NUMBER
(2) 012140 042703 177770      BIC     #C<7>,R3      ;REMOVE ALL BUT BUS LEVEL NUMBER
(2) 012144 012704 001320      MOV     #$DDW0,R4     ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
(2) 012150 013705 001312      MOV     $DEVM,R5     ;GET THE MAP OF ACTIVE DEVICES
(2) 012154 010537 001404      MOV     R5,DZACTV     ;SAVE THE BIT MAP
(2) 012160 006005             1$: ROR     R5         ;GET A DEVICE SELECTION BIT
(2) 012162 103407             BCS     3$           ;IF IT IS SELECTED, GO SET UP A MAP
(2) 012164 001425             BEQ     5$           ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
(2) 012166 005724             TST     (R4)+        ;POINT TO NEXT DEVICE DESCRIPTOR
(2) 012170 062701 000010      2$: ADD     #10,R1     ;SET UP THE NEXT ADDRESS
(2) 012174 062702 000010      ADD     #10,R2     ;SET UP THE NEXT VECTOR GROUP
(2) 012200 000767             BR      1$          ;GO SEE IF MORE DEVICES REMAIN
(2) 012202 010120             3$: MOV     R1,(R0)+  ;LOAD DEVICE ADDRESS
(2) 012204 010220             MOV     R2,(R0)+  ;LOAD THE VECTOR ADDRESS
(2) 012206 010320             MOV     R3,(R0)+  ;LOAD THE INTERRUPT PRIORITY LEVEL
(2) 012210 013720 001314      MOV     $CDW1,(R0)+ ;GET THE NUMBER OF LINES IN OPERATION
(2) 012214 012420             MOV     (R4)+,(R0)+ ;LOAD DEVICE PARAMETERS
(2) 012216 100006             BPL     4$          ;IF 20MA MODE SELECTED, SET IT UP
(2) 012220 052760 100000 177772 BIS     #100000,-6(R0) ;SET THE 20MA FLAG IN DZLVN
(2) 012226 042760 100000 177776 BIC     #100000,-2(R0) ;CLEAR THE FLAG IN DZPARN
(2) 012234 005020             4$: CLR     (R0)+    ;DEFAULT OPERATION TO INTERNAL MAINTENANCE MODE
(2) 012236 000754             BR      2$          ;GO BUILD THE NEXT ADDRESS
(2) 012240 012710 177777      5$: MOV     #-1,(R0) ;TERMINATE THE DEVICE MAP
(2) 012244 012737 001256 001160 MOV     #$SWREG,SWR ;SET TO SOFTWARE APT SWITCH REGISTER
(2) 012252 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) 012254             AUTO.SIZE:
(2) 012254 000005             RESET
(2) 012256 105337 001415      CSRMAP: DECB    INIFLG    ;INSURE A BUS INIT.
(2) 012262 012702 001500      MOV     #DZ.MAP,R2    ;SHOW THAT I WAS HERE
(2) 012266 012703 001320      MOV     #$DDW0,R3    ;LOAD MAP POINTER.
(2) 012272 005022             1$: CLR     (R2)+    ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
(2) 012274 022702 002000      CMP     #DZ.END,R2   ;ZERO ENTIRE MAP
(2)                               ;ALL DONE?
```


(2)	012300	001374		BNE	1\$	;BR IF NO
(2)	012302	105037	001410	CLRB	DZNUM	;SET OCTAL NUMBER OF DZ11'S TO 0
(2)	012306	012702	001500	MOV	#DZ.MAP,R2	
(2)	012312	012701	160000	MOV	#160000,R1	;SET FOR FIRST ADDRESS TO BE TESTED
(2)	012316	012737	012636	MOV	#6\$,R4	;SET FOR NON-EXISTENT DEVICE TIME OUT
(2)	012324	052711	000040	BIS	#BIT5,(R1)	;TRY TO SET MASTER SCAN ENABLE
(2)	012330	052761	000200	BIS	#BIT7,4(R1)	;TRY TO TRANSMIT ON LINE 7
(2)	012336	005000		CLR	R0	;USE R0 AS A COUNTER
(2)	012340	005711		TST	(R1)	;HAS TRANSMITTER READY COME UP?
(2)	012342	100403		BMI	8\$	;IF SO, GO GET A FINAL CHECK
(2)	012344	005300		DEC	R0	;REDUCE COUNT. TIME UP?
(2)	012346	001374		BNE	7\$	;IF NOT, KEEP WAITING
(2)	012350	000463		BR	3\$	;ASSUME IT'S NOT A DZ11
(2)	012352	032761	000200	BIT	#BIT7,4(R1)	;IS LINE 7 ENABLE STILL SET? IT SHOULD BE
(2)	012360	001457		BEQ	3\$	;IF IT'S NOT, ASSUME IT'S NOT A DZ11
(2)	012362	032711	000040	BIT	#BIT5,(R1)	;IS MASTER SCAN ENABLE STILL SET?
(2)	012366	001454		BEQ	3\$	;IF NOT, ASSUME IT'S NOT A DZ11
(2)	012370	005000		CLR	R0	
(2)	012372	052711	000020	BIS	#20,(R1)	;SET DEVICE CLEAR
(2)	012376	032711	000020	BIT	#20,(R1)	;SHOULD STAY SET FOR A WHILE IF DZ
(2)	012402	001446		BEQ	3\$	;BR IF NOT DZ11
(2)	012404	032711	000020	BIT	#20,(R1)	;WAIT FOR BIT TO CLEAR
(2)	012410	001404		BEQ	..+12	;BR WHEN CLEARED
(2)	012412	104414		DELAY		
(2)	012414	005200		INC	R0	
(2)	012416	001372		BNE	..-12	
(2)	012420	000437		BR	3\$	;BIT NOT CLEARED! MUST NOT BE DZ11
(2)	012422	005011		CLR	(R1)	;GET RID OF MASTER SCAN ENABLE
(2)	012424	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)	012430	010122		MOV	R1,(R2)+	;STORE CSR IN CORE TABLE.
(2)	012432	005722		TST	(R2)+	;POP OVER VECTOR STORE AREA
(2)	012434	012722	000005	MOV	#5,(R2)+	;SET THE DEFAULT BUS LEVEL
(2)	012440	052761	177400	BIS	#177400,4(R1)	;TRY TO SET ALL DTR BITS
(2)	012446	032761	177400	BIT	#177400,4(R1)	;IF ANY SET ASSUME EIA BOARD
(2)	012454	001003		BNE	9\$	;IF NONE SET ASSUME BOARD IS
(2)	012456	052762	100000	BIS	#BIT15,-2(R2)	;20 MA, SET 20 MA FLAG
(2)	012464	012722	000377	MOV	#377,(R2)+	;SET THE DEFAULT LINE SELECTION PARAMETER
(2)	012470	012712	017070	MOV	#17070,(R2)	;SET THE DEFAULT PARAMETERS
(2)	012474	012223		MOV	(R2)+,(R3)+	;COPY PARAMETERS INTO ETABLE DESCRIPTOR
(2)	012476	005022		CLR	(R2)+	;SET THE DEFAULT MODE OF OPERATION
(2)	012500	012712	177777	MOV	#-1,(R2)	;TERMINATE LIST
(2)	012504	105237	001410	INCB	DZNUM	;UPDATE DEVICE COUNTER
(2)	012510	122737	000020	CMPB	#20,DZNUM	;ARE MAX. NO. OF DEV FOUND?
(2)	012516	001405		BEQ	100\$	;YES DON'T LOOK FOR ANY MORE.
(2)	012520	062701	000010	ADD	#10,R1	;UPDATE CSR POINTER ADDRESS
(2)	012524	022701	163700	CMP	#163700,R1	
(2)	012530	001275		BNE	2\$	;BR IF MORE ADDRESS TO CHECK.
(2)	012532					
(2)	012532	105737	001410	TSTB	DZNUM	;WERE ANY DZ11'S FOUND AT ALL?
(2)	012536	001432		BEQ	5\$	;ERROR AUTO SIZER FOUND NO DZ11'S IN THIS SYS.
(2)	012540	113701	001410	MOVB	DZNUM,R1	
(2)	012544	110137	001411	MOVB	R1,SAVNUM	;SAVE NUMBER OF DEVICES
(2)	012550	012737	000001	MOV	#1,DZACTV	
(2)	012556	005301		DEC	R1	
(2)	012560	001404		BEQ	98\$	

```

(2) 012562 000261 SEC
(2) 012564 006137 001404 ROL DZACTV
(2) 012570 000772 BR 4$
(2) 012572 013737 001500 001310 98$: MOV DZCRO,$BASE ;POINT TO THE ADDRESS OF FIRST DEVICE
(2) 012600 013737 001512 001314 MOV MANTO,$CDW1 ;INDICATE TO ETABLE WHAT MODE IS BEING USED
(2) 012606 012737 000006 000004 99$: MOV #6,$#4 ;RESTORE TRAP VECTOR
(2) 012614 013737 001404 001312 MOV DZACTV,$DEVM ;SAVE ACTIVE REGISTER
(2) 012622 000410 BR VECMAP ;GO FIND THE VECTOR NOW.
(2) 012624 104402 010564 5$: TYPE ,MERR2 ;NOTIFY OPR THAT NO DZ11'S FOUND.
(2) 012630 005000 CLR R0 ;MAKE DATA DISPLAY ZERO
(2) 012632 000000 HALT ;STOP THE SHOW
(2) 012634 000776 BR -.2 ;DISABLE CONT. SW.
(2) 012636 012716 012520 6$: MOV #3$(,SP) ;ENTERED BY NON-EXISTENT TIME-OUT
(2) 012642 000002 RTI ;RETURN TO MAINSTREAM
(2)
(2) 012644 012737 000340 000022 VECMAP: MOV #340,$#22 ;SET IOT TRAP PRIORITY TO 7
(2) 012652 012737 012770 000020 MOV #4$, $#20 ;SET IOT TRAP VECTOR
(2) 012660 012702 001500 MOV #DZ.MAP,R2 ;SET SOFTWARE POINTER
(2) 012664 012700 000300 MOV #300,R0 ;FLOATING VECTORS START HERE.
(2) 012670 012701 000302 MOV #302,R1 ;PC OF IOT INSTR.
(2) 012674 010120 1$: MOV R1,(R0)+ ;START FILLING VECTOR AREA
(2) 012676 012721 000004 MOV #4,(R1)+ ;WITH .+2; IOT
(2) 012702 022021 CMP (R0)+,(R1)+ ;ADD 2 TO R0 +R1
(2) 012704 020127 001000 CMP R1,#1000 ;HAS THE VECTOR AREA BEEN EXCEEDED?
(2) 012710 101771 BLOS 1$ ;BR IF MORE TO FILL
(2) 012712 013704 001404 MOV DZACTV,R4 ;STORE TEMPORARILY
(2) 012716 000241 2$: CLC
(2) 012720 006004 ROR R4 ;BRING OUT A BIT
(2) 012722 103036 BCC 5$ ;BR IF ALL DONE
(2) 012724 106427 000000 MTPS #0 ;ZERO CPU PRIO
(2) 012730 012772 040040 000000 MOV #BIT14+BIT5,$(R2)
(2) 012736 011201 MOV (R2),R1 ;GET CSR
(2) 012740 112761 000200 000004 MOVB #BIT7,4(R1) ;SET THE TCR BIT!
(2) INC R0 ;ATTEMPT TO FORCE AN INTERRUPT
(2) BNE -.2 ;STALL
(2) 012750 001376 -.2 ; FOR TIME TO INTERRUPT
(2) 012752 012762 000300 000002 MOV #300,2(R2) ;NO INTERRUPT ASSUME 300 AND FIX DZ11 LATER
(2) 012760 000005 3$: RESET ;INIT
(2) 012762 062702 000014 ADD #14,R2 ;POP SOFTWARE POINTER
(2) 012766 000753 BR 2$ ;KEEP GOING
(2) 012770 011662 000002 4$: MOV (SP),2(R2) ;GET VECTOR ADDRESS
(2) 012774 162762 000010 000002 SUB #10,2(R2) ;POINT BACK TO THE CORRECT VECTOR
(2) 013002 042762 000007 000002 BIC #7,2(R2) ;CLEAR JUNK
(2) 013010 022626 POP2SP ;POP IOT JUNK OFF STACK
(2) 013012 012716 012760 MOV #3$(,SP) ;SET FOR RETURN
(2) 013016 000002 RTI
(2) 013020 013737 001502 001304 5$: MOV DZVCO,$VECT1 ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
(2) 013026 012737 005122 000020 MOV #.SCOPE,IOTVEC ;RESTORE THE SCOPE TRAP
(2) 013034 000207 RTS PC ;ALL DONE WITH "AUTO SIZING"
(2)
(2) ; -- END O MACRO *****

```



```

8648                                     ; - $UNIBUS-----
(2)                                     ; - $XZ-----
(3)                                     ;***** TEST 1 *****
(1)                                     ;*THIS TEST PROVES THE SLAVE SYNC RESPONSE
(1)                                     ;*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
(1)                                     ;*      DZCSR, DZRBUF, DZTCR, DZMSR
(2)                                     ; - $XZ-----
(3)                                     ;:* TEST 1
(6)                                     ;*****
(5) 013036 000004 TST1: SCOPE
(3) 013040 012737 000001 001122      MOV      #1,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)                                     ; -- END 0 MACRO -----
(3) 013046 012737 013226 001360      MOV      #TST2,NEXT      ;POINT TO THE START OF THE NEXT TEST
(1) 013054 012737 013214 000004      MOV      #5$,4          ;SET TRAP VECTOR
(1) 013062 012737 000340 000006      MOV      #PR7,6        ;SET PRIORITY TO LEVEL 7
(1) 013070 012737 013076 001362      MOV      #1$,LOCK       ;SET RETURN IF SW09=11
(1) 013076 013700 002042 1$:         MOV      DZCSR,R0        ;SET ADDRESS TO TEST
(1) 013102 011001      MOV      (R0),R1      ;READ THE ADDRESS
(1) 013104 000240      NOP                    ;WASTE TIME
(1) 013106 005010      CLR      (R0)         ;WRITE THE ADDRESS
(1) 013110 000240      NOP                    ;WASTE TIME
(1) 013112 012737 013120 001362      MOV      #2$,LOCK       ;SET RETURN ADDRESS FOR SW09
(1) 013120 013700 002046 2$:         MOV      DZRBUF,R0       ;SET ADDRESS TO TEST
(1) 013124 011001      MOV      (R0),R1      ;READ THE ADDRESS
(1) 013126 000240      NOP                    ;
(1) 013130 005010      CLR      (R0)         ;WRITE THE ADDRESS
(1) 013132 000240      NOP                    ;WASTE TIME
(1) 013134 012737 013142 001362      MOV      #3$,LOCK       ;SET RETURN ADDRESS FOR SW09
(1) 013142 013700 002056 3$:         MOV      DZTCR,R0        ;SET ADDRESS TO TEST
(1) 013146 011001      MOV      (R0),R1      ;READ THE ADDRESS
(1) 013150 000240      NOP                    ;
(1) 013152 005010      CLR      (R0)         ;WRITE THE ADDRESS
(1) 013154 000240      NOP                    ;
(1) 013156 012737 013164 001362      MOV      #4$,LOCK       ;SET RETURN ADDRESS
(1) 013164 013700 002062 4$:         MOV      DZMSR,R0        ;SET ADDRESS TO TEST
(1) 013170 011001      MOV      (R0),R1      ;READ FROM ADDRESS
(1) 013172 000240      NOP                    ;
(1) 013174 005010      CLR      (R0)         ;WRITE THE ADDRESS
(1) 013176 000240      NOP                    ;
(1) 013200 012737 000006 000004      MOV      #6,4          ;SET TRAP CATCHER BACK TO NORMAL
(1) 013206 005037 000006      CLR      6          ;
(1) 013212 104400      ADVANCE                ;SCOPE THIS TEST
(1) 013214 011601 5$:         MOV      (SP),R1      ;SAVE PC OF TRAP
(1) 013216 022626      CMP      (SP)+,(SP)+      ;POP TRAP OFF STACK
(1) 013220 104001      ERROR      1          ;*NO SLAVE SYNC RESPONSE.
(1) 013222 104401      SCOP1                ;SW09=1?
(1) 013224 000111      JMP      (R1)         ;RTI
(1)                                     ; -- END 0 MACRO -----
8650                                     ; - $XZ-----
(2)                                     ;***** TEST 2 *****
8651                                     ;*THIS TEST PROVES THAT BIT "DCLR"
8652                                     ;*CAN BE SET AND THAT IT WILL CLEAR
8653                                     ;*BY ITSELF AFTER A PERIOD OF TIME.
8654                                     ; - $XZ-----
8655                                     ;:* TEST 2

```

```
(5) ;*****
(4) 013226 000004 TST2: SCOPE
(2) 013230 012737 000002 001122 MOV #2,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(2) ; -- END 0 MACRO -----
(2) 013236 012737 013312 001360 MOV #TST3,NEXT ;POINT TO THE START OF THE NEXT TEST
8656 013244 013700 002042 MOV DZCSR,R0 ;SET POINTER
8657 013250 012705 000020 MOV #DCLR,R5 ;SET DCLR
8658 013254 010510 MOV R5,(R0) ;WRITE DCLR INTO DZCSR
8659 013256 011004 MOV (R0),R4 ;READ BACK DZCSR
8660 013260 020504 CMP R5,R4 ;DZCSR OK?
8661 013262 001401 BEQ 1$ ;IF IT IS SET SKIP THE ERROR CALL
8662 013264 104002 ERROR 2 ;*DCLR SHOULD BE SET..MOMENTARILY
8663 ;NOW LETS WATCH IT DISAPPEAR
8664 013266 005002 1$: CLR R2 ;SET COUNTER TO 0
8665 013270 005005 CLR R5 ;SET EXPECTED TO 0
8666 013272 005003 CLR R3 ;DUAL LOOP COUNTER
8667 013274 011004 2$: MOV (R0),R4 ;IS DCLR CLEAR?
8668 013276 001405 BEQ 3$ ;IF YES , GO TO THE NEXT TEST
8669 013300 005203 INC R3 ;IF NO,COUNT 1 OF 65535 TICKS
8670 ;THE WORD CREATED BY THE IMMEDIATE 0 WILL BE
8671 ;THE COUNTER
8672 013302 001374 BNE 2$ ;HAS THE TIME EXPIRED? IF NO, GO TEST BIT AGAIN
8673 013304 005302 DEC R2 ;HAS THE TOTAL TIME EXPIRED?
8674 013306 001372 BNE 2$ ;IF NO, CHECK THE BIT AGAIN
8675 013310 104002 ERROR 2 ;*DCLR FAILED TO CLEAR
8676 013312 3$:
8680 ; -MRRW-----
(3) ; -XZ-----
(4) ;***** TEST 3 *****
(2) ;*TEST TO VERIFY THAT BIT "MAINT" CAN
(2) ;*BE SET. THEN VERIFY THAT BIT "MAINT" CAN
(2) ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
(2) ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
(2) ;*CLEARED BY A "DEVICE CLEAR"
(3) ; -XZ-----
(4) ;:* TEST 3
(7) ;*****
(6) 013312 000004 TST3: SCOPE
(4) 013314 012737 000003 001122 MOV #3,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(4) ; -- END 0 MACRO -----
(4) 013322 012737 013404 001360 MOV #TST4,NEXT ;POINT TO THE START OF THE NEXT TEST
(2) 013330 013700 002042 MOV DZCSR,R0 ;GET BASE ADDRESS
(2) 013334 012705 000010 MOV #MAINT,R5 ;SET BIT
(2) 013340 010510 MOV R5,(R0) ;SET SET IN DEVICE
(2) 013342 011004 MOV (R0),R4 ;READ THE BIT FROM DEVICE
(2) 013344 020504 CMP R5,R4 ;WAS BIT SET?
(2) 013346 001401 BEQ 1$ ;BR IF YES
(2) 013350 104002 ERROR 2 ;*BIT R/W FAILURE
(2) 013352 040510 1$: BIC R5,(R0) ;CLEAR THE BIT.
(2) 013354 011004 MOV (R0),R4 ;READ DEVICE
(2) 013356 001404 BEQ 2$ ;BR IF BITS WERE CLEARED.
(2) 013360 010546 MOV R5,-(SP) ;SAVE THE BIT
(2) 013362 005005 CLR R5 ;SET EXPECTED RESULTS TO 0
(2) 013364 104002 ERROR 2 ;*BIT FAILED TO CLEAR
(2) 013366 012605 MOV (SP)+,R5 ;RESTORE THE BIT.
(2) 013370 010510 2$: MOV R5,(R0) ;SET THE BIT AGAIN
```



```

(2) 013372 104413      DEVICE.CLR      ;ISSUE DEVICE CLEAR
(2) 013374 011004      MOV      (R0),R4    ;READ THE BIT.
(2) 013376 001402      BEQ      3$         ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013400 005005      CLR      R5         ;SET EXPECTED TO ZERO
(2) 013402 104002      ERROR    2         ;*BIT NOT CLEARED BY DEVICE CLEAR
(2) 013404
(2)
(1)
(2)
(3)
(4)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(3)
(4)
(7)
(6) 013404 000004
(4) 013406 012737 000004 001122
(4)
(4) 013414 012737 013476 001360
(2) 013422 013700 002042
(2) 013426 012705 000040
(2) 013432 010510
(2) 013434 011004
(2) 013436 020504
(2) 013440 001401
(2) 013442 104002
(2) 013444 040510
(2) 013446 011004
(2) 013450 001404
(2) 013452 010546
(2) 013454 005005
(2) 013456 104002
(2) 013460 012605
(2) 013462 010510
(2) 013464 104413
(2) 013466 011004
(2) 013470 001402
(2) 013472 005005
(2) 013474 104002
(2) 013476
(2)
(1)
(2)
(3)
(4)
(2)
(2)
(2)
(2)
(2)
(2)
(3)
(4)

      ; -- END 0 MACRO =====
      ; -- END 0 MACRO =====
      ; --MRRW-----
      ; --XZ-----
      ;***** TEST 4 *****
      ;*TEST TO VERIFY THAT BIT "MSENAB" CAN
      ;*BE SET. THEN VERIFY THAT BIT "MSENAB" CAN
      ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
      ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
      ;*CLEARED BY A "DEVICE CLEAR"
      ; --XZ-----
      ;:* TEST 4
      ;:*****
      TST4:  SCOPE
      MOV      #4,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
      ; -- END 0 MACRO =====
      MOV      #TST5,NEXT     ;POINT TO THE START OF THE NEXT TEST
      MOV      DZCSR,R0       ;GET BASE ADDRESS
      MOV      #MSENAB,R5     ;SET BIT
      MOV      R5,(R0)        ;SET SET IN DEVICE
      MOV      (R0),R4        ;READ THE BIT FROM DEVICE
      CMP      R5,R4          ;WAS BIT SET?
      BEQ      1$             ;BR IF YES
      ERROR    2              ;*BIT R/W FAILURE
      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
      MOV      (SP)+,R5       ;RESTORE THE BIT.
      MOV      R5,(R0)       ;SET THE BIT AGAIN
      2$:      DEVICE.CLR     ;ISSUE DEVICE CLEAR
      MOV      (R0),R4        ;READ THE BIT.
      BEQ      3$             ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
      CLR      R5             ;SET EXPECTED TO ZERO
      ERROR    2              ;*BIT NOT CLEARED BY DEVICE CLEAR
      3$:
      ; -- END 0 MACRO =====
      ; -- END 0 MACRO =====
      ; --MRRW-----
      ; --XZ-----
      ;***** TEST 5 *****
      ;*TEST TO VERIFY THAT BIT "SILOEN" CAN
      ;*BE SET. THEN VERIFY THAT BIT "SILOEN" CAN
      ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
      ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
      ;*CLEARED BY A "DEVICE CLEAR"
      ; --XZ-----
      ;:* TEST 5

```

```
(7) ;*****
(6) 013476 000004 TST5: SCOPE ;*****
(4) 013500 012737 000005 001122 MOV #5,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(4) ; -- END 0 MACRO -----
(4) 013506 012737 013570 001360 MOV #TST6,NEXT ;POINT TO THE START OF THE NEXT TEST
(2) 013514 013700 002042 MOV DZCSR,R0 ;GET BASE ADDRESS
(2) 013520 012705 010000 MOV #SILOEN,R5 ;SET BIT
(2) 013524 010510 MOV R5,(R0) ;SET SET IN DEVICE
(2) 013526 011004 MOV (R0),R4 ;READ THE BIT FROM DEVICE
(2) 013530 020504 CMP R5,R4 ;WAS BIT SET?
(2) 013532 001401 BEQ 1$ ;BR IF YES
(2) 013534 104002 ERROR 2 ;*BIT R/W FAILURE
(2) 013536 040510 1$: BIC R5,(R0) ;CLEAR THE BIT.
(2) 013540 011004 MOV (R0),R4 ;READ DEVICE
(2) 013542 001404 BEQ 2$ ;BR IF BITS WERE CLEARED.
(2) 013544 010546 MOV R5,-(SP) ;SAVE THE BIT
(2) 013546 005005 CLR R5 ;SET EXPECTED RESULTS TO 0
(2) 013550 104002 ERROR 2 ;*BIT FAILED TO CLEAR
(2) 013552 012605 MOV (SP)+,R5 ;RESTORE THE BIT.
(2) 013554 010510 2$: MOV R5,(R0) ;SET THE BIT AGAIN
(2) 013556 104413 DEVICE.CLR ;ISSUE DEVICE CLEAR
(2) 013560 011004 MOV (R0),R4 ;READ THE BIT.
(2) 013562 001402 BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
(2) 013564 005005 CLR R5 ;SET EXPECTED TO ZERO
(2) 013566 104002 ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
(2) 013570 3$:
(2) ; -- END 0 MACRO -----
(1) ; -- END 0 MACRO -----
(2) ; -$MRRW-----
(3) ; -$XZ-----
(4) ;***** TEST 6 *****
(2) ;*TEST TO VERIFY THAT BIT "RIE" CAN
(2) ;*BE SET. THEN VERIFY THAT BIT "RIE" CAN
(2) ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
(2) ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
(2) ;*CLEARED BY A "DEVICE CLEAR"
(3) ; -$XZ-----
(4) ;: * TEST 6
(7) ;*****
(6) 013570 000004 TST6: SCOPE ;*****
(4) 013572 012737 000006 001122 MOV #6,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(4) ; -- END 0 MACRO -----
(4) 013600 012737 013662 001360 MOV #TST7,NEXT ;POINT TO THE START OF THE NEXT TEST
(2) 013606 013700 002042 MOV DZCSR,R0 ;GET BASE ADDRESS
(2) 013612 012705 000100 MOV #RIE,R5 ;SET BIT
(2) 013616 010510 MOV R5,(R0) ;SET SET IN DEVICE
(2) 013620 011004 MOV (R0),R4 ;READ THE BIT FROM DEVICE
(2) 013622 020504 CMP R5,R4 ;WAS BIT SET?
(2) 013624 001401 BEQ 1$ ;BR IF YES
(2) 013626 104002 ERROR 2 ;*BIT R/W FAILURE
(2) 013630 040510 1$: BIC R5,(R0) ;CLEAR THE BIT.
(2) 013632 011004 MOV (R0),R4 ;READ DEVICE
(2) 013634 001404 BEQ 2$ ;BR IF BITS WERE CLEARED.
(2) 013636 010546 MOV R5,-(SP) ;SAVE THE BIT
(2) 013640 005005 CLR R5 ;SET EXPECTED RESULTS TO 0
(2) 013642 104002 ERROR 2 ;*BIT FAILED TO CLEAR
```



```
(2) 013644 012605
(2) 013646 010510
(2) 013650 104413
(2) 013652 011004
(2) 013654 001402
(2) 013656 005005
(2) 013660 104002
(2) 013662
(2)
(1)
(2)
(3)
(4)
(2)
(2)
(2)
(2)
(2)
(2)
(3)
(4)
(7)
(6) 013662 000004
(4) 013664 012737 000007 001122
(4)
(4) 013672 012737 013754 001360
(2) 013700 013700 002042
(2) 013704 012705 040000
(2) 013710 010510
(2) 013712 011004
(2) 013714 020504
(2) 013716 001401
(2) 013720 104002
(2) 013722 040510
(2) 013724 011004
(2) 013726 001404
(2) 013730 010546
(2) 013732 005005
(2) 013734 104002
(2) 013736 012605
(2) 013740 010510
(2) 013742 104413
(2) 013744 011004
(2) 013746 001402
(2) 013750 005005
(2) 013752 104002
(2) 013754
(2)
(1)
8681
(2)
(3)
(1)
(1)
(1)
(1)
(2)
```

```
MOV (SP)+,R5 ;RESTORE THE BIT.
2$: MOV R5,(R0) ;SET THE BIT AGAIN
DEVICE.CLR ;ISSUE DEVICE CLEAR
MOV (R0),R4 ;READ THE BIT.
BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
CLR R5 ;SET EXPECTED TO ZERO
ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR

3$:
; -- END O MACRO =====
; -- END O MACRO =====

; -$MRRW-----
; -$XZ-----

;***** 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"
; -$XZ-----

;:* TEST 7
;*****
TST7: SCOPE
MOV #7,$TSTNM ;LOAD THE NUMBER OF THIS TEST
; -- END O MACRO =====
MOV #TST10,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;GET BASE ADDRESS
MOV #TIE,R5 ;SET BIT
MOV R5,(R0) ;SET SET IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 1$ ;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
CLR R5 ;SET EXPECTED RESULTS TO 0
ERROR 2 ;*BIT FAILED TO CLEAR
MOV (SP)+,R5 ;RESTORE THE BIT.
2$: MOV R5,(R0) ;SET THE BIT AGAIN
DEVICE.CLR ;ISSUE DEVICE CLEAR
MOV (R0),R4 ;READ THE BIT.
BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
CLR R5 ;SET EXPECTED TO ZERO
ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR

3$:
; -- END O MACRO =====
; -- END O MACRO =====

; -$TCR-----
; -$XZ-----

;***** TEST 10 *****
;*THIS TESTS THAT ALL OF THE FOLLOWING
;*BITS CAN BE: SET, CLEARED, CLEARED BY "DEVICE CLEAR "
;*BITS TESTED ARE:
;* TCR0, TCR1, TCR2, TCR3, TCR4, TCR5, TCR6, TCR7
; -$XZ-----
```

```
(3)          ::* TEST 10
(6)          ;*****
(5) 013754 000004 TST10: SCOPE
(3) 013756 012737 000010 001122      MOV    #10,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)          ; -- END 0 MACRO =====
(3) 013764 012737 014112 001360      MOV    #TST11,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 013772 013700 002056              MOV    DZTCR,R0      ;SET DEVICE ADDRESS
(1) 013776 012705 000001              MOV    #TCR0,R5      ;SET EXPECTED RESULTS
(1) 014002 012737 014010 001362      MOV    #1$,LOCK      ;SET FOR SW09
(1) 014010 010510      1$: MOV    R5,(R0)      ;SET THE BIT
(1) 014012 011004      MOV    (R0),R4      ;READ THE BIT FROM THE DEVICE
(1) 014014 042704 177400      BIC    #C<377>,R4      ;CLEAR HIGH BYTE
(1) 014020 020504      CMP    R5,R4      ;WAS BIT OK?
(1) 014022 001401      BEQ    2$          ;BR IF YES
(1) 014024 104002      ERROR 2          ;*BIT FAILED TO SET.
(1) 014026 040510      2$: BIC    R5,(R0)      ;CLEAR THE BIT
(1) 014030 011004      MOV    (R0),R4      ;READ THE REGISTER
(1) 014032 042704 177400      BIC    #C<377>,R4      ;CLEAR HIGH BYTE
(1) 014036 005704      TST    R4          ;BITS CLEAR?
(1) 014040 001404      BEQ    3$          ;BR IF YES
(1) 014042 010546      MOV    R5,-(SP)      ;SAVE GOOD RESULTS
(1) 014044 005005      CLR    R5          ;SET EXPECTED TO 0
(1) 014046 104002      ERROR 2          ;*REPORT BIT NOT CLEAR
(1) 014050 012605      MOV    (SP)+,R5      ;RESTORE R5
(1) 014052 010510      3$: MOV    R5,(R0)      ;SET THE BIT AGAIN.
(1) 014054 104413      DEVICE.CLR      ;ISSUE DEVICE CLEAR
(1) 014056 011004      MOV    (R0),R4      ;READ THE REGISTER
(1) 014060 042704 177400      BIC    #C<377>,R4      ;CLEAR HIGH BYTE
(1) 014064 005704      TST    R4          ;BITS CLEAR?
(1) 014066 001404      BEQ    4$          ;BR IF YES
(1) 014070 010546      MOV    R5,-(SP)      ;SAVE GOOD RESULTS
(1) 014072 005005      CLR    R5          ;SET EXPECTED TO 0
(1) 014074 104002      ERROR 2          ;*REPORT BIT NOT CLEAR
(1) 014076 012605      MOV    (SP)+,R5      ;RESTORE R5
(1) 014100 104401      4$: SCOP1      ;LOCK ON BIT? SET SW09=1
(1) 014102 106305      ASLB    R5          ;CHANGE TO NEXT BIT
(1) 014104 001341      BNE    1$          ;CONTINUE TESTING
(1) 014106 005037 001362      CLR    LOCK      ;MAKE SURE TIGHT LOOP IS CLEANED UP
(1)          ; -- END 0 MACRO =====
8682          ; -$TCR-----
(2)          ; -$XZ-----
(3)          ;***** TEST 11 *****
(1)          ;*THIS TESTS THAT ALL OF THE FOLLOWING
(1)          ;*BITS CAN BE: SET, CLEARED, CLEARED BY "RESET INSTR *NOT* DEVICE CLEAR "
(1)          ;*BITS TESTED ARE:
(1)          ;* DTR0, DTR1, DTR2, DTR3, DTR4, DTR5, DTR6, DTR7
(1)          ;*THIS TEST IS NOT DONE IF MODULE IS 20MA VERSION
(2)          ; -$XZ-----
(3)          ::* TEST 11
(6)          ;*****
(5) 014112 000004 TST11: SCOPE
(3) 014114 012737 000011 001122      MOV    #11,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)          ; -- END 0 MACRO =====
(3) 014122 012737 014274 001360      MOV    #TST12,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 014130 013700 002056              MOV    DZTCR,R0      ;SET DEVICE ADDRESS
(1) 014134 012705 000400              MOV    #DTR0,R5      ;SET EXPECTED RESULTS
```



```

(1) 014140 012737 014156 001362 MOV #1$,LOCK ;SET FOR SW09
(1) 014146 105737 001414 TSTB EIAFLG ;20MA OR EIA
(1) 014152 100001 BPL 1$ ;BR IF EIA
(1) 014154 104400 ADVANCE ;EXIT TEST
(1) 014156 010510 1$: MOV R5,(R0) ;SET THE BIT
(1) 014160 011004 MOV (R0),R4 ;READ THE BIT FROM THE DEVICE
(1) 014162 105004 CLRB R4 ;CLEAR LOW BYTE
(1) 014164 020504 CMP R5,R4 ;WAS BIT OK?
(1) 014166 001401 BEQ 2$ ;BR IF YES
(1) 014170 104002 ERROR 2 ;*BIT FAILED TO SET.
(1) 014172 040510 2$: BIC R5,(R0) ;CLEAR THE BIT
(1) 014174 011004 MOV (R0),R4 ;READ THE REGISTER
(1) 014176 105004 CLRB R4 ;CLEAR LOW BYTE
(1) 014200 005704 TST R4 ;BITS CLEAR?
(1) 014202 001404 BEQ 3$ ;BR IF YES
(1) 014204 010546 MOV R5,-(SP) ;SAVE GOOD RESULTS
(1) 014206 005005 CLR R5 ;SET EXPECTED TO 0
(1) 014210 104002 ERROR 2 ;*REPORT BIT NOT CLEAR
(1) 014212 012605 MOV (SP)+,R5 ;RESTORE R5
(1) 014214 010510 3$: MOV R5,(R0) ;SET THE BIT AGAIN.
(1) 014216 104413 DEVICE.CLR ;ISSUE DEVICE CLEAR
(1) 014220 011004 MOV (R0),R4 ;READ THE REGISTER
(1) 014222 105004 CLRB R4 ;CLEAR LOW BYTE
(1) 014224 030510 BIT R5,(R0) ;WAS BIT CLEARED BY DEVICE.CLR?
(1) 014226 001001 BNE 4$ ;BR IF NO (IT SHOULDN'T BE CLEAR)
(1) 014230 104002 ERROR 2 ;*BIT CLEARED BY DEVICE.CLR
(1) 014232 104401 4$: SCOP1 ;LOCK ON BIT? SW09=1
(1) 014234 006305 ASL R5 ;CHANGE TO NEXT BIT
(1) 014236 001347 BNE 1$ ;IF NOT DONE LOOP
(1) 014240 012710 177400 MOV #177400,(R0) ;SET ALL DTR BITS
(1) 014244 005005 CLR R5 ;CLEAR LOCATION FOR ERROR PRINTOUT
(1) 014246 005227 000000 5$: INC #0 ;ACT DELAY LOOP FOR
(1) 014252 001375 BNE 5$ ;RESET INSTRUCTION
(1) 014254 000005 RESET ;ISSUE A BUS INIT
(1) 014256 011004 MOV (R0),R4 ;READ REGISTER
(1) 014260 105004 CLRB R4 ;CLEAR LOW BYTE
(1) 014262 005704 TST R4 ;DTR BITS CLEAR?
(1) 014264 001401 BEQ .+4 ;IF YES CONTINUE
(1) 014266 104002 ERROR 2 ;IF NO PRINT ERROR
(1) 014270 005037 001362 CLR LOCK ;MAKE SURE TIGHT LOOP IS CLEANED UP
(1) ; -- END 0 MACRO =====
8683 ; $MRR-----
(2) ; -$XZ-----
(3) ;***** 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) ;*
(2) ; -$XZ-----
(3) ;*: TEST 12
(6) ;*****
(5) 014274 000004 TST12: SCOPE
(3) 014276 012737 000012 001122 MOV #12,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3) ; -- END 0 MACRO =====

```

```

(3) 014304 012737 014412 001360      MOV    #TST13,NEXT      ;POINT TO THE START OF THE NEXT TEST
(1) 014312 013700 002042              MOV    DZCSR,R0        ;SET ADDRESS TO R0
(1) 014316 005005                      CLR     R5            ;SET EXPECTED TO 0
(1) 014320 012710 027607              MOV    #RDONE+BIT11+BIT10+BIT9+BIT8+BIT2+BIT1+BIT0+SILOAL,(R0)
(1)                                     ;WRITE THE BITS
(1) 014324 011004                      MOV    (R0),R4        ;READ BACK THE BITS
(1) 014326 001401                      BEQ     1$           ;BR IF NONE ARE SET.
(1) 014330 104002                      ERROR   2            ;*BITS WERE SET.
(1) 014332 012710 100000 1$:          MOV    #TRDY,(R0)      ;ATTEMPT TO WRITE TRDY
(1) 014336 011004                      MOV    (R0),R4        ;READ TRDY
(1) 014340 001401                      BEQ     2$           ;BR IF NOT SET
(1) 014342 104002                      ERROR   2            ;*
(1) 014344 012705 100000 2$:          MOV    #TRDY,R5        ;SET EXPECTED BIT
(1) 014350 005077 165476              CLR     @DZLPR        ;LOAD LINE 0
(1) 014354 052777 000001 165474      BIS     #TCR0,@DZTCR  ;SET TCR BIT
(1) 014362 052710 000040              BIS     #MSENAB,(R0) ;
(1) 014366 052705 000040              BIS     #MSENAB,R5   ;SET SCAN ENABLE
(1) 014372 005002                      CLR     R2            ;SET COUNTER TO ZERO
(1) 014374 011004 3$:                MOV    (R0),R4        ;READ THE REGISTER
(1) 014376 020504                      CMP     R5,R4        ;BIT SET?
(1) 014400 001404                      BEQ     4$           ;BR IF YES
(1) 014402 104414                      DELAY                    ;STALL TIME
(1) 014404 005202                      INC     R2            ;UPDATE COUNTER
(1) 014406 001372                      BNE     3$           ;BR IF COUNTER NOT DONE.
(1) 014410 104002                      ERROR   2            ;*TRDY NOT SET!
(1) 014412                          4$:
(1)                                     ; -- END 0 MACRO =====
8684                                     ; -XZ-----
(2)                                     ;***** TEST 13 *****
8685                                     ;*THIS TEST PERFORMS RESET TESTING AND
8686                                     ;*TESTING OF READ ONLY AND WRITE ONLY BITS
8687                                     ;* IN REGISTER DZCSR
8688                                     ;*VERIFY THAT "TIE", "SILOEN", "RIE", "MSENAB", "MAINT"
8689                                     ;*ARE THE ONLY R/W BITS IN THE DZCSR.
8690                                     ;*THEN VERIFY THAT A RESET WILL CLEAR THESE BITS
8691                                     ;*THIS TEST ALSO CHECKS BYTE OPERATIONS ON THE CSR
8692                                     ; -XZ-----
8693                                     ;:* TEST 13
(5)                                     ;*****
(4) 014412 000004 TST13: SCOPE
(2) 014414 012737 000013 001122      MOV    #13,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(2)                                     ; -- END 0 MACRO =====
(2) 014422 012737 014542 001360      MOV    #TST14,NEXT    ;POINT TO THE START OF THE NEXT TEST
8694 014430 104413 DEVICE.CLR
8695 014432 013700 002042      MOV    DZCSR,R0        ;SET UP FOR ERROR MESSAGE
8696 014436 012710 177757      MOV    #C<DCLR>,(R0)  ;TRY TO WRITE
8697 014442 012705 050150      MOV    #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
8698 014446 011004      MOV    (R0),R4        ;ACTUAL
8699 014450 020405      CMP     R4,R5        ;CMP EXPECTED VS ACTUAL
8700 014452 001401      BEQ     1$           ;YES
8701 014454 104002      ERROR   2            ;*NO
8702 014456 105010 1$:          CLRB    (R0)        ;CLEAR LOWER BYTE OF CSR
8703 014460 105005      CLRB    R5            ;SET EXPECTED
8704 014462 011004      MOV    (R0),R4        ;READ CSR BITS
8705 014464 020405      CMP     R4,R5        ;COMPARE ACTUAL TO EXPECTED
8706 014466 001401      BEQ     3$           ;BRANCH IF SAME

```



```

8707 014470 104002
8708 014472 012710 177757
8709 014476 105077 165342
8710 014502 012705 000150
8711
8712 014506 011004
8713 014510 020405
8714 014512 001401
8715 014514 104002
8716 014516 012710 177757
8717 014522 005005
8718 014524 005227 000000
8719 014530 001375
8720 014532 000005
8721 014534 011004
8722 014536 001401
8723 014540 104002
8724 014542
8725
(2)
(3)
(1)
(1)
(1)
(2)
(3)
(6)
(5) 014542 000004
(3) 014544 C12737 000014 001122
(3)
(3) 014552 012737 014632 001360
(1) 014560 104413
(1) 014562 013700 002046
(1) 014566 011005
(1) 014570 012777 177777 165254
(1) 014576 011004
(1) 014600 042705 104000
(1) 014604 020405
(1) 014606 001401
(1) 014610 104002
(1) 014612 010403
(1) 014614 005103
(1) 014616 010377 165230
(1) 014622 011004
(1) 014624 020405
(1) 014626 001401
(1) 014630 104002
(1) 014632
(1)
8726
(2)
(3)
(1)
(1)
(1)
(2)

```

```

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
MOV #C<DCLR>,(R0) ; RESET CSR BITS
CLR R5 ; SET R5 TO EXPECTED RESULTS
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

2$:
; - $MRWD-----
; - $XZ-----
; ***** TEST 14 *****
; *THIS TEST PERFORMS RESET TESTING AND
; *TESTING OF READ ONLY REGISTER DZRBUF
; *AND TESTING OF WRITE ONLY REGISTER DZLPR
; - $XZ-----

::* TEST 14
; *****
TST14: SCOPE
MOV #14,$TSTNM ; LOAD THE NUMBER OF THIS TEST
; -- END 0 MACRO =====
MOV #TST15,NEXT ; POINT TO THE START OF THE NEXT TEST
DEVICE.CLR ; CLEAR DZ11
MOV DZRBUF,R0 ; SET UP FOR ERROR MESSAGE
MOV (R0),R5 ; SET EXPECTED
MOV #-1,@DZLPR ; TRY TO WRITE ALL 1'S
MOV (R0),R4 ; ACTUAL
BIC #DVALID!BIT11,R5 ; DITTO
CMP R4,R5 ; CMP ACTUAL VS EXPECTED
BEQ 1$ ; 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$:
; -- END 0 MACRO =====
; - $MRWD-----
; - $XZ-----
; ***** TEST 15 *****
; *THIS TEST PERFORMS RESET TESTING AND
; *TESTING OF READ ONLY REGISTER DZMSR
; *AND TESTING OF WRITE ONLY REGISTER DZTOR
; - $XZ-----

```

```
(3)          ;:* TEST 15
(6)          ;*****
(5) 014632 000004 TST15: SCOPE
(3) 014634 012737 000015 001122      MOV #15,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)          ; -- END 0 MACRO =====
(3) 014642 012737 014716 001360      MOV #TST16,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 014650 104413      DEVICE.CLR      ;CLEAR DZ11
(1) 014652 013700 002062      MOV DZMSR,R0      ;SET UP FOR ERROR MESSAGE
(1) 014656 011005      MOV (R0),R5      ;SET EXPECTED
(1) 014660 012777 177777 165200      MOV #-1,@DZTDR    ;TRY TO WRITE ALL 1'S
(1) 014666 011004      MOV (R0),R4      ;ACTUAL
(1) 014670 020405      CMP R4,R5      ;CMP ACTUAL VS EXPECTED
(1) 014672 001401      BEQ 1$          ;IF YES,GO CONTINUE PROCESSING
(1) 014674 104002      ERROR 2          ;*ERROR- BIT PATTERN NOT CORRECT
(1) 014676 010403      1$: MOV R4,R3      ;GET A COPY OF THE ACTUAL BIT PATTERN
(1) 014700 005103      COM R3          ;GET THE LOGICAL INVERSE OF THE BIT PATTERN
(1) 014702 010377 165160      MOV R3,@DZTDR    ;TRY TO WRITE
(1) 014706 011004      MOV (R0),R4      ;ACTUAL
(1) 014710 020405      CMP R4,R5      ;CMP ACTUAL VS EXPECTED
(1) 014712 001401      BEQ 2$          ;IF YES, GET OUT OF THIS TEST
(1) 014714 104002      ERROR 2          ;*NO
(1) 014716          2$:
(1)          ; -- END 0 MACRO =====

8727
8728
8729          ; - $XZ-----
(2)          ;***** TEST 16 *****
8730          ;*VERIFY THAT IF WE ARE IN "STAGGERED" MODE
8731          ;*THAT SETTING "DTR" FOR A LINE WILL
8732          ;*BRING UP "RING" AND "CARRIER" FOR THE
8733          ;*ASSOCIATED LINE IN WHICH WE ARE STAGGERED!
8734          ;* LINE0 DTR= LINE1 RING AND CARRIER
8735          ;* LINE1 DTR= LINE0 RING AND CARRIER
8736          ;* LINE2 DTR= LINE3 RING AND CARRIER
8737          ;* LINE3 DTR= LINE 4 RING AND CARRIER
8738          ;*
8739          ;* ETC...
8740          ; - $XZ-----
8741
8742          ;:* TEST 16
(5)          ;*****
(4) 014716 000004 TST16: SCOPE
(2) 014720 012737 000016 001122      MOV #16,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(2)          ; -- END 0 MACRO =====
(2) 014726 012737 015112 001360      MOV #TST17,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 014734 012737 015006 001362      MOV #1$,LOCK      ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8742 014742 105737 001414      TSTB EIAFLG      ;EIA OR 20MA?
8743 014746 100001      BPL 10$          ;BR IF EIA
8744 014750 104400      ADVANCE          ;EXIT TEST
8745 014752 013700 002062      10$: MOV DZMSR,R0      ;SET REGISTER
8746 014756 104413      DEVICE.CLR      ;INIT DZ11
8747 014760 005003      CLR R3          ;ZERO LINE NUMBER
8748 014762 012702 000001      MOV #1,R2      ;SET POINTER
8749 014766 005737 001370      TST MODE      ;ARE WE IN STAGGERED MODE?
8750 014772 100405      BMI 1$          ;YES WE ARE!
8751 014774 013737 001360 001126      MOV NEXT,$LPADR    ;LEAVE THIS TEST! NOT STAGGERED
8752 015002 000177 164120      JMP @LPADR      ;EXIT
```



```

8753 015006 130237 001364      1$: BITB R2,LINE      ;TEST THIS LINE?
8754 015012 001004              BNE 3$      ;YES
8755 015014 005203      2$: INC R3      ;LINE #
8756 015016 106302        ASLB R2      ;GET NEXT LINE
8757 015020 103372        BCC 1$      ;KEEP TESTING
8758 015022 104400        ADVANCE      ;ADVANCE THIS TEST
8759 015024 010204      3$: MOV R2,R4      ;SAVE BINARY BIT FOR LINE #
8760 015026 032703 000001      BIT #BIT0,R3 ;GET STAGGERED COMPANION LINE
8761 015032 001402        BEQ 4$      ;BR IF LINE EVEN
8762 015034 006204        ASR R4      ;ADJUST LINE
8763 015036 000401        BR 5$      ;
8764 015040 006304      4$: ASL R4      ;ADJUST LINE
8765 015042 005005      5$: CLR R5      ;SET EXPECTED
8766 015044 150405        BISB R4,R5      ;
8767 015046 000305        SWAB R5      ;
8768 015050 150405        BISB R4,R5      ;
8769 015052 150277 165002      BISB R2,@HDZTCR ;SET DTR
8770 015056 104414        DELAY      ;CABLE DELAY
8771 015060 011004        MOV (R0),R4 ;READ MSR REGISTER
8772 015062 020504        CMP R5,R4      ;OK?
8773 015064 001401        BEQ 6$      ;YES
8774 015066 104002      6$: ERROR 2      ;*ERROR IN RING OR CARRIER
8775 015070 140277 164764      BICB R2,@HDZTCR ;CLEAR DTR
8776 015074 104414        DELAY      ;CABLE DELAY
8777 015076 011004        MOV (R0),R4 ;READ MSR
8778 015100 001402        BEQ 7$      ;BR IF THEY CLEARED
8779 015102 005005        CLR R5      ;SET EXPECTED TO 0
8780 015104 104002      7$: ERROR 2      ;*BITS NOT CLEARED
8781 015106 104401      SCOPE 1$      ;LOCK ON SIGNAL?
8782 015110 000741        BR 2$      ;CONTINUE TEST
8783
8784                                ; -$XZ-----
(2)                                ;***** TEST 17 *****
8785                                ;*TEST TO VERIFY THAT IF IN "EXTERNAL"
8786                                ;*MODE; SETTING DTR FOR SELECTED LINES
8787                                ;*WILL BRING UP "CARRIER" AND "RING"
8788                                ;*FOR THAT SAME LINE. NOTE: IF YOU HAVE
8789                                ;*SELECTED MODE AS "EXTERNAL"; THE H325 TEST CONNECTER
8790                                ;*MUST BE USED ON ALL SPECIFIED LINES.
8791                                ;*LINES MAY BE SPECIFIED BY SWR03=1
8792                                ;*AND SWR00=1 AT START TIME OR ALTERING
8793                                ;*STATUS MAP.
8794                                ; -$XZ-----
8795                                ;:* TEST 17
(5)                                ;*****
(4) 015112 000004      TST17: SCOPE
(2) 015114 012737 000017 001122      MOV #17,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(2)                                ; -- END O MACRO -----
(2) 015122 012737 015250 001360      MOV #TST20,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 015130 012737 015164 001362      MOV #3$,LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8796 015136 105737 001370      TSTB MODE ;EXTERNAL?
8797 015142 100401      BMI 2$      ;BR IF YES
8798 015144 104400      1$: ADVANCE ;EXIT TEST
8799 015146 105737 001414      2$: TSTB EIAFLG ;YOU BETTER BE IN
8800 015152 100774      BMI 1$      ;EIA MODE FOR THIS TEST.
8801 015154 013700 002062      MOV DZMSR,R0 ;SET REGISTER

```

```

8802 015160 012702 000001      MOV    #1,R2      ;SET LINE POINTER
8803 015164 130237 001364      3$:  BITB    R2,LINE    ;LINE SELECTED?
8804 015170 001003              BNE     5$      ;BR IF YES
8805 015172 106302              4$:  ASLB    R2      ;NEXT LINE
8806 015174 103373              BCC     3$      ;CONTINUE TEST
8807 015176 104400              ADVANCE ;ADVANCE THIS TEST
8808 015200 005005              5$:  CLR     R5      ;SET EXPECTED
8809 015202 150205              BISB    R2,R5      ;
8810 015204 000305              SWAB    R5      ;
8811 015206 150205              BISB    R2,R5      ;
8812 015210 150277 164644      BISB    R2,@HDZTCR ;SET DTR
8813 015214 104414              DELAY   ;CABLE DELAY
8814 015216 011004              MOV     (R0),R4 ;READ MSR
8815 015220 020504              CMP     R5,R4 ;BITS OK?
8816 015222 001401              BEQ     6$      ;BR IF YES
8817 015224 104002              ERROR   2      ;CARRIER OR RING ERROR
8818 015226 140277 164626      6$:  BICB    R2,@HDZTCR ;CLEAR DTR
8819 015232 104414              DELAY   ;CABLE DELAY
8820 015234 011004              MOV     (R0),R4 ;READ MSR
8821 015236 001402              BEQ     7$      ;BR IF BITS CLEARED
8822 015240 005005              CLR     R5      ;CLEAR EXPECTED LOC.
8823 015242 104002              ERROR   2      ;BITS NOT CLEARED.
8824 015244 104401              7$:  SCOP1   2      ;LOCK ON LINE?
8825 015246 000751              BR      4$      ;CONTINUE TEST
8826
8827                                ; -$TLINE-----
(2)                                ; -$XZ-----
(3)                                ;*****
(1)                                ;***** TEST 20 *****
(1)                                ;* THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
(1)                                ;* IS READY TO BE LOADED, AND THAT THE LINE SPECI-
(1)                                ;* FIED IN BITS 8-10 OF DZCSR CORRESPOND
(1)                                ;* TO THE LINE SELECTED IN DZTCR
(2)                                ; -$XZ-----
(3)                                ;: * TEST 20
(6)                                ;:*****
(5) 015250 000004      TST20: SCOPE
(3) 015252 012737 000020 001122      MOV     #20,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3)                                ; -- END 0 MACRO -----
(3) 015260 012737 015374 001360      MOV     #TST21,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 015266 104413      DEVICE.CLR ;ISSUE A "DEVICE CLEAR" (RESET)
(1) 015270 013700 002042      MOV     DZCSR,R0 ;SET POINTER
(1) 015274 012705 100040      MOV     #MSENAB!TRDY,R5 ;START THE EXPECTED LINE NUMBER AT 0
(1) 015300 005037 001372      CLR     SAVLIN ;SET UP FOR ERROR PRINTOUTS
(1) 015304 012702 000001      MOV     #1,R2 ;USING R2 AS A BIT POINTER, POINT TO LINE 0
(1) 015310 130237 001364      1$:  BITB    R2,LINE    ;IS THIS LINE SELECTED?
(1) 015314 001420              BEQ     5$      ;IF NO, SKIP THE STARTUP
(1) 015316 050277 164534      2$:  BIS     R2,@DZTCR ;SET THE GO BIT FOR THIS LINE
(1) 015322 052710 000040      BIS     #MSENAB,(R0) ;START THE SCANNER
(1) 015326 005004      CLR     R4      ;SET FOR DELAY
(1) 015330 032710 100000      3$:  BIT     #TRDY,(R0) ;TX READY?
(1) 015334 001004      BNE     4$      ;BR IF YES
(1) 015336 104414      DELAY   ;DELAY
(1) 015340 005204      INC     R4      ;COUNTER
(1) 015342 001372      BNE     3$      ;BR IF <>0!
(1) 015344 104003      ERROR   3      ;*TX NOT READY!
(1) 015346 011004      4$:  MOV     (R0),R4 ;GET THE LINE POINTED TO BY THE SCANNER

```



```

(1) 015350 020405      CMP      R4,R5      ;IS THE LINE NUMBER WHAT IT SHOULD BE?
(1) 015352 001401      BEQ      5$      ;IF YES,GO WORK ON THE NEXT LINE
(1) 015354 104002      ERROR    2      ;*LINE NUMBER DID NOT MATCH TCR BIT
(1) 015356 062705 000400 5$: ADD      #400,R5      ;POINT TO THE NEXT EXPECTED LINE
(1) 015362 104413      DEVICE.CLR      ;ISSUE A "DEVICE CLEAR" (RESET)
(1) 015364 005237 001372 INC      SAVLIN      ;ADJUST FOR NEXT LINE
(1) 015370 106302      ASLB      R2      ;POINT TO THE NEXT LINE.ARE ALL LINES TESTED?
(1) 015372 103346      BCC      1$      ;IF NOT, GO DO THE NEXT LINE
(1) 015374
(1)
8828 ; -- END O MACRO -----
(2) ; - $XZ-----
8829 ;***** TEST 21 *****
8830 ;*TEST TO TRANSMIT ONE CHAR AND
8831 ;*RECEIVE ONE CHAR ON ONE LINE
8832 ;*AT A TIME. THE CHAR IS "252" AND
8833 ;*ALL SELECTED LINES WILL BE TURNED ON
8834 ;*ONE AT A TIME. THIS IS THE FIRST TIME ANY
8835 ;*DATA IS CHECKED IN THE RECEIVER.
8836 ;*USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
8837 ;*WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.
8838 ; - $XZ-----
(5) ;:* TEST 21
(4) 015374 000004      ;*****
(2) 015376 012737 000021 001122 TST21: SCOPE
(2) MOV      #21,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(2) 015404 012737 015732 001360 ; -- END O MACRO -----
(1) 015412 012737 015710 001362 MOV      #TST22,NEXT      ;POINT TO THE START OF THE NEXT TEST
MOV      #16$,LOCK      ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8839 ;$LINEUP-----
(2) ; - $MRESET-----
(2) 015420 104417      DCLASM      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2) ; -- END O MACRO -----
(1) 015422 013701 001366      MOV      PAR,R1      ;PICK UP PARAMETERS
(1) 015426 012702 000001      MOV      #1,R2      ;PICK UP INIT POINTER
(1) 015432 030237 001364      1$: BIT      R2,LINE      ;SHOULD THIS LINE BE SET UP ?
(1) 015436 001402      BEQ      2$      ;NO
(1) 015440 010177 164406      MOV      R1,$DZLPR      ;SET UP LINE PARAMETERS
(1) 015444 005201      2$: INC      R1      ;POSITION POINTER TO THE NEXT LINE
(1) 015446 106302      ASLB      R2      ;GOT 'EM ALL ?
(1) 015450 103370      BCC      1$      ;IF NO, GO SET UP THE NEXT LINE
(1) 015452 005037 001372      CLR      SAVLIN      ;CLEAR LINE # INDICATOR
(1) ; -- END O MACRO -----
8840 015456 012702 000001      MOV      #1,R2      ;LINE POINTER
8841 015462 052777 000040 164352 BIS      #MSENAB,$DZCSR      ;START SCANNER
8842 015470 030237 001364      3$: BIT      R2,LINE      ;VALID LINE ?
8843 015474 001462      BEQ      14$      ;NO SET UP NEXT LINE
8844 015476 010277 164354      MOV      R2,$DZTCR      ;SET TCR BIT
8845 015502 032777 000200 164332 4$: BIT      #RDONE,$DZCSR      ;IS REC DONE = 0 ?
8846 015510 001401      BEQ      5$      ;IF YES, ALLOW TIME FOR TRDY TO SET
8847 015512 104020      ERROR    20      ;*REC DONE SHOULD = 0
8848 015514 005005      5$: CLR      R5
8849 015516 032777 100000 164316 6$: BIT      #TRDY,$DZCSR
8850 015524 001004      BNE      7$
8851 015526 104414      DELAY
8852 015530 105205      INCB      R5
8853 015532 001371      BNE      6$

```

[illegible]


```
8897      ;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
8898      ;*CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
8899      ;*(ONE LINE AT A TIME   BASED UPON VALID LINES)
8900      ;*THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
8901      ; - $XZ-----
8902      ;: * TEST 22
      ;:*****
      (5) TST22: SCOPE
      (4) 015732 000004
      (2) 015734 012737 000022 001122      MOV    #22,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
      (2) ; -- END 0 MACRO -----
      (2) 015742 012737 016260 001360      MOV    #TST23,NEXT    ;POINT TO THE START OF THE NEXT TEST
      (1) 015750 012737 016064 001362      MOV    #4$,LOCK      ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
8903      ;$LINEUP-----
      (2) ; - $MRESET-----
      (2) 015756 104417      DCLASH      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
      (2) ; -- END 0 MACRO -----
      (1) 015760 013701 001366      MOV    PAR,R1      ;PICK UP PARAMETERS
      (1) 015764 012702 000001      MOV    #1,R2      ;PICK UP INIT POINTER
      (1) 015770 030237 001364      1$: BIT    R2,LINE      ;SHOULD THIS LINE BE SET UP ?
      (1) 015774 001402      BEQ    2$      ;NO
      (1) 015776 010177 164050      MOV    R1,$DZLPR      ;SET UP LINE PARAMETERS
      (1) 016002 005201      2$: INC    R1      ;POSITION POINTER TO THE NEXT LINE
      (1) 016004 106302      ASLB   R2      ;GOT 'EM ALL ?
      (1) 016006 103370      BCC    1$      ;IF NO, GO SET UP THE NEXT LINE
      (1) 016010 005037 001372      CLR    SAVLIN      ;CLEAR LINE # INDICATOR
      (1) ; -- END 0 MACRO -----
8904 016014 012700 001422      MOV    #TDO,R0      ;POINT TO THE DATA AREA
8905 016020 005020      CLR    (R0)+      ;CLEAR A DATA WORD
8906 016022 022700 001462      CMP    #STOP,R0      ;FINISHED ?
8907 016026 001374      BNE    .-6      ;NO
8908 016030 005000      CLR    R0      ;CLEAR OFFSET
8909 016032 013737 002046 001400      MOV    DZRBUF,REGIST ;SAVE FOR ERROR MSG
8910 016040 012702 000001      MOV    #1,R2      ;LINE POINTER
8911 016044 052777 000040 163770      BIS    #MSENAB,$DZCSR ;START SCANNER
8912 016052 030237 001364      3$: BIT    R2,LINE      ;VALID LINE ?
8913 016056 001465      BEQ    14$      ;NO SET UP NEXT LINE
8914 016060 010277 163772      MOV    R2,$DZTCR      ;SET TCR BIT
8915 016064 032777 000200 163750 4$: BIT    #RDONE,$DZCSR ;IS REC DONE = 0 ?
8916 016072 001401      BEQ    5$      ;IF YES, ALLOW TIME FOR TRDY TO SET
8917 016074 104020      ERROR   20      ;*REC DONE SHOULD = 0
8918 016076 005005      5$: CLR    R5
8919 016100 032777 100000 163734 6$: BIT    #TRDY,$DZCSR
8920 016106 001004      BNE    7$
8921 016110 104414      DELAY
8922 016112 105205      INCB   R5
8923 016114 001371      BNE    6$
8924 016116 104003      ERROR   3
8925 016120 116077 001422 163740 7$: MOVB   TDO(R0),$DZTDR ;*TRDY FAILED TO SET!
8926 016126 013705 001372      MOV    SAVLIN,R5      ;LOAD CHARACTER
      ;MAKE EXPECTED LINE #
      ; - $STAG-----
      (1) 016132 105737 001371      TSTB   MODE+1      ;IS THIS TEST IN STAGGERED MODE?
      (1) 016136 001406      BEQ    10$      ;IF NOT, SKIP STAGGERED SETUP
      (1)
      (1) ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
      (1)
      (1) ASR    R5      ;GET THE LAST BIT INTO THE CARRY BIT
```

```

(1) 016142 103402      BCS      8$      ;IF IT IS SET, GO CLEAR IT
(1) 016144 000261      SEC              ;IF IT IS CLEAR SET IT HERE
(1) 016146 000401      BR        9$      ;SKIP THE CLEARING
(1) 016150 000241      8$: CLC              ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 016152 006105      9$: ROL      R5      ;GET THE NEW BIT BACK INTO R5
(1)      ; -- END 0 MACRO -----
8928 016154 000305      10$: SWAB     R5      ;MOVE THE LINE NUMBER TO THE UPPER BYTE
8929 016156 156005 001422      BISB    TDO(R0),R5 ;ADD CHARACTER
8930 016162 052705 100000      BIS     #DVALID,R5 ;ADD DATA VALID
8931 016166 005003      CLR      R3
8932 016170 032777 000200 163644 11$: BIT     #RDONE,#DZCSR
8933 016176 001004      BNE      12$
8934 016200 104414      DELAY
8935 016202 005203      INC      R3
8936 016204 001371      BNE      11$
8937 016206 104004      ERROR     4      ;*RDONE FAILED TO SET!
8938 016210 017704 163632      12$: MOV     #DZRBUF,R4 ;LOAD THE VALUE ACTUALLY RECEIVED
8939 016214 020405      CMP      R4,R5      ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
8940 016216 001401      BEQ      13$      ;IF YES, GO DO THE NEXT LINE
8941 016220 104006      ERROR     6      ;*NO DATA/CONTENTS DID NOT COMPARE
8942 016222 104401      13$: SCOP1      ;CHECK TO SEE IF SWITCH NINE IS SET
8943 016224 105260 001422      INCB     TDO(R0) ;INCREMENT BINARY PATTERN FOR THIS LINE
8944 016230 001315      BNE      4$      ;GO 'ROUND AGAIN FOR NEXT CHARACTER
8945 016232 040277 163620      14$: BIC     R2,#DZTCR ;CLEAR TCR BIT FOR THAT LINE.
8946 016236 005237 001372      15$: INC     SAVLIN ;INC EXPECTED LINE
8947 016242 013700 001372      MOV     SAVLIN,R0 ;SET UP CHARACTER OFFSET
8948 016246 006300      ASL      R0      ;MAKE THE OFFSET A POWER OF TWO
8949 016250 106302      ASLB     R2      ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
8950 016252 103277      BCC      3$      ;IF NO, GO AROUND AGAIN FOR NEXT LINE
8951 016254 005037 001362      CLR      LOCK ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST
8952
8953
8954      ; - $XZ-----
(2)      ;***** TEST 23 *****
8955      ;*THIS TEST WILL PROVE THAT EACH RECEIVING LINE CAN
8956      ;*BE DISABLED BY SETTING THE RCVON BIT TO ZERO
8957      ;*FOR EACH LINE IN THE LPR REGISTER. IT ALSO
8958      ;*VERIFIES THAT MASTER CLEAR WILL ZERO DVALID FOR
8959      ;*CHARACTERS STORED IN THE SILO.
8960      ; - $XZ-----
8961      ;:* TEST 23
(5)      ;*****
(4) 016260 000004      TST23: SCOPE
(2) 016262 012737 000023 001122      MOV     #23,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(2)      ; -- END 0 MACRO -----
(2) 016270 012737 016612 001360      MOV     #TST24,NEXT ;POINT TO THE START OF THE NEXT TEST
8962 016276 105037 001420      CLRB     DONFLG ;INITIALIZE FOR FIRST TEST LOOP
8963 016302 005037 001372      CLR      SAVLIN ;ZERO LINE NO. FOR ERROR REPORT
8964 016306 104417      DCLASM      ;EXECUTE MASTER CLEAR
8965 016310 013701 001366      MOV     PAR,R1 ;STORE DEFAULT PARAMETERS
8966 016314 042701 010000      BIC     #RCVON,R1 ;CLEAR RCVON BIT
8967 016320 012702 000001      1$: MOV     #1,R2 ;INIT LINE POINTER
8968 016324 010177 163522      2$: MOV     R1,#DZLPR ;LOAD LINE PARAMETER REGISTER
8969 016330 005201      INC      R1 ;SET R1 FOR NEXT LINE
8970 016332 106302      ASLB     R2 ;SHIFT R2 TO NEXT LINE
8971 016334 103373      BCC      2$      ;ALL LINES LOADED?

```


8972	016336	012701	000252		MOV	#252,R1	;LOAD TRANSMITTING CHARACTER
8973	016342	013702	001364		MOV	LINE,R2	;COPY ACTIVE LINE BITS
8974	016346	010277	163504		MOV	R2,@DZTCR	;LOAD TCR BITS
8975	016352	052777	000040	163462	BIS	#MSENAB,@DZCSR	;SET SCANNER
8976	016360	005005		3\$:	CLR	R5	;INIT DELAY COUNTER
8977	016362	005777	163454	4\$:	TST	@DZCSR	;TRDY SET?
8978	016366	100404			BMI	5\$	;IF YES BRANCH
8979	016370	104414			DELAY		;IF NOT THEN WAIT
8980	016372	005205			INC	R5	;INCREMENT DELAY COUNTER
8981	016374	001372			BNE	4\$	;DELAY DONE?
8982	016376	104003			ERROR	3	;IF YES TRDY FAILED TO SET
8983	016400	117705	163440	5\$:	MOVB	@HDZCSR,R5	;MOVE LINE NO. INTO R5
8984	016404	012703	000001		MOV	#1,R3	;INIT TCR POINTER
8985	016410	042705	177770		BIC	#C<7>,R5	;ISOLATE LINE NO.
8986	016414	001403			BEQ	21\$	;IF LINE 0 GO TEST TRANSM. FLAG
8987	016416	106303		20\$:	ASLB	R3	;POINT R3 TO NEXT TCR BIT
8988	016420	005305			DEC	R5	;DECREMENT R5 UNTIL R3 POINTS
8989	016422	001375			BNE	20\$	;TO CORRECT TCR BIT
8990	016424	030302		21\$:	BIT	R3,R2	;HAS THIS LINE BEEN SERVICED?
8991	016426	001007			BNE	6\$	;IF NOT GO SEND CHARACTER
8992	016430	140377	163422		BICB	R3,@DZTCR	;IF YES CLEAR TCR BIT
8993	016434	001351			BNE	3\$	;IF MORE LINES SET BRANCH
8994	016436	105737	001420		TSTB	DONFLG	;IF ALL LOADED IS THIS SECOND PASS
8995	016442	001040			BNE	12\$	;IF YES BRANCH TO SECOND PART OF TEST
8996	016444	000404			BR	7\$	;OTHERWISE CONTINUE WITH FIRST PART
8997	016446	110177	163414	6\$:	MOVB	R1,@DZTDR	;TRANSMIT CHARACTER
8998	016452	040302			BIC	R3,R2	;CLEAR FLAG FOR THIS LINE
8999	016454	000741			BR	3\$	;GO WAIT FOR NEXT LINE
9000	016456	005077	163374	7\$:	CLR	@DZTCR	;CLEAR TCR BITS
9001	016462	005005			CLR	R5	;CLEAR DELAY COUNTER
9002	016464	104414		8\$:	DELAY		;WAIT FOR LAST CHARACTER
9003	016466	005205			INC	R5	;INCREMENT DELAY COUNTER
9004	016470	001375			BNE	8\$	;IF NOT FINISHED CONTINUE WAITING
9005	016472	105777	163344		TSTB	@DZCSR	;RDONE BIT SET?
9006	016476	100003			BPL	10\$	;IF NO CONTINUE
9007	016500	005037	001372		CLR	SAVLIN	;IF YES SET LINE NO. TO ZERO
9008	016504	104020			ERROR	20	;AND PRINT ERROR
9009	016506	017704	163334	10\$:	MOV	@DZRBUR,R4	;READ SILO
9010	016512	100007			BPL	11\$	;IF DVALID IS ZERO BRANCH
9011	016514	000304			SWAB	R4	;IF SET THEN
9012	016516	042704	177770		BIC	#C<7>,R4	;ISOLATE LINE NO. IN R4
9013	016522	010437	001372		MOV	R4,SAVLIN	;SET SAVLIN FOR ERROR REPORT
9014	016526	104017			ERROR	17	;DATA VALID SHOULD NOT BE SET
9015	016530	000766			BR	10\$	;GO READ SILO AGAIN
9016	016532	105237	001420	11\$:	INCB	DONFLG	;PREPARE FOR SECOND PART OF TEST
9017	016536	013701	001366		MOV	PAR,R1	;MOVE DEFAULT PARAMETERS TO R1
9018	016542	000666			BR	1\$	;GO LOAD LPR REGISTER
9019	016544	005005		12\$:	CLR	R5	;INIT DELAY COUNTER
9020	016546	104414		13\$:	DELAY		;WAIT FOR LAST CHARACTER
9021	016550	005205			INC	R5	;TO BE RECEIVED
9022	016552	001375			BNE	13\$	;DELAY FINISHED?
9023	016554	104413			DEVICE.CLR		;IF YES EXECUTE MASTER CLEAR
9024	016556	000240			NOP		
9025	016560	000240			NOP		
9026	016562	105777	163254		TSTB	@DZCSR	;RDONE SET?
9027	016566	100003			BPL	14\$	;IF NOT BRANCH

```

9028 016570 005037 001372      CLR      SAVLIN      ;IF YES THEN PRINT OUT
9029 016574 104020      ERROR    20      ;REPORT
9030 016576 017704 163244      14$:  MOV     @DZRBUR,R4    ;READ SILO
9031 016602 100003      BPL      15$      ;DATA VALID SET?
9032 016604 005037 001372      CLR      SAVLIN      ;IF YES THEN PRINT OUT
9033 016610 104017      ERROR    17      ;ERROR REPORT
9034 016612      15$:
9035
9036                                ; -$XZ-----
          ;***** 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
          ; -$XZ-----
9045  (5)  ;:* TEST 24
          ;*****
          TST24: SCOPE
          MOV     #24,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
          ; -- END 0 MACRO -----
          (2) 016612 000004      MOV     #TST25,NEXT    ;POINT TO THE START OF THE NEXT TEST
          (2) 016614 012737 000024 001122      MOV     #3$,LOCK    ;SET FOR LOOP
          (2) 016622 012737 017070 001360      TST      MODE      ;ARE WE RUNNING IN INTERNAL MODE?
9046 016630 012737 016726 001362      BEQ      12$      ;IF SO, SKIP THIS TEST
9047 016636 005737 001370      DCLASH      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
9048 016642 001510      ; -- END 0 MACRO -----
9049  (1) 016644 104417      MOV     PAR,R1      ;PICK UP PARAMETERS
          (1)      BIS      #ODDPAR!PARITY,R1 ;FORCE ODD PARITY
9050 016646 013701 001366      MOV     #1,R0      ;PICK UP INIT POINTER
9051 016652 052701 000300      1$:  BIT      R0,LINE    ;SHOULD THIS LINE BE SET UP ?
9052 016656 012700 000001      BEQ      2$      ;IF NOT,DON'T SET IT UP
9053 016662 030037 001364      MOV     R1,@DZLPR    ;OTHERWISE, SET UP LINE PARAMETERS
9054 016666 001402      2$:  INC      R1
9055 016670 010177 163156      ASLB     R0      ;GOT 'EM ALL ?
9056 016674 005201      BCC      1$      ;NO
9057 016676 106300      CLR      SAVLIN      ;CLEAR LINE #
9058 016700 103370      MOV     #1,R2      ;LINE POINTER
9059 016702 005037 001372      BIS      #MSENAB,@DZCSR ;SET MASTER SCAN ENABLE
9060 016706 012702 000001      MOV     DZRBUR,REGIST ;SAVE FOR ERRR MESSAGE
9061 016712 052777 000040 163122      3$:  BIT      R2,LINE
9062 016720 013737 002046 001400      BEQ      11$
9063 016726 030237 001364      MOV     R2,@DZTCR    ;SET TCR BIT
9064 015732 001446      MOV     R2,@HDZTDR    ;SET BREAK BIT
9065 016734 010277 163116      MOVB    #377,@DZTDR ;LOAD CHARACTER
9066 016740 110277 163124      MOV     SAVLIN,R5    ;MAKE EXPECTED DATA
9067 016744 112777 000377 163114 4$:
9068 016752 013705 001372      TSTB     MODE+1    ;IS THIS TEST IN STAGGERED MODE?
9069      BEQ      7$      ;IF NOT, SKIP STAGGERED SETUP
          (1)
          (1)      ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
          (1)
          (1)
          (1) 016756 105737 001371
          (1) 016762 001406
          (1)
          (1)
          (1) 016764 006205      ASR      R5      ;GET THE LAST BIT INTO THE CARRY BIT

```



```

(1) 016766 103402      BCS      5$      ;IF IT IS SET, GO CLEAR IT
(1) 016770 000261      SEC              ;IF IT IS CLEAR SET IT HERE
(1) 016772 000401      BR        6$      ;SKIP THE CLEARING
(1) 016774 000241      5$: CLC              ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 016776 006105      6$: ROL      R5      ;GET THE NEW BIT BACK INTO R5
(1)      ; -- END O MACRO -----
9070 017000 000305      7$: SWAB      R5      ;PUT LINE NUMBER IN UPPER BYTE
9071 017002 052705 130000      BIS      #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
9072 017006 005004      CLR      R4
9073 017010 032777 000200 163024 8$: BIT      #RDONE,@DZCSR
9074 017016 001004      BNE      9$
9075 017020 104414      DELAY
9076 017022 005204      INC      R4
9077 017024 001371      BNE      8$
9078 017026 104004      ERROR      4      ;*RDONE FAILED TO SET!
9079 017030 017704 163012      9$: MOV      @DZRBUF,R4      ;ACTUAL
9080 017034 020405      CMP      R4,R5      ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
9081 017036 001401      BEQ      10$      ;IF YES, GO CLEAN UP
9082 017040 104006      ERROR      6      ;*DATA/CONTENTS FAILED TO COMPARE
9083 017042 105077 163022      10$: CLRB      @HDZTDR      ;CLEAR BREAK BITS
9084 017046 104401      SCOP1
9085 017050 005237 001372      11$: INC      SAVLIN      ;LOOP?
9086 017054 040277 162776      BIC      R2,@DZTCR      ;INC LINE #
9087 017060 106302      ASLB      R2      ;CLEAR TCR BIT
9088 017062 103321      BCC      3$
9089 017064 005037 001362      12$: CLR      LOCK      ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST
9090      ; -- $LVLST-----
(2)      ; -- $XZ-----
(3)      ;***** 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.
(1)      ;*DEFAULT PRIORITY IS AT 5 (240).
(2)      ; -- $XZ-----
(3)      ;:* TEST 25
(6)      ;*****
(5) 017070 000004      TST25: SCOPE
(3) 017072 012737 000025 001122      MOV      #25,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)      ; -- END O MACRO -----
(3) 017100 012737 017400 001360      MOV      #TST26,NEXT      ;POINT TO THE START OF THE NEXT TEST
(2)      ;$LINEUP-----
(3)      ; -- $MRESET-----
(3) 017106 104417      DCLASM      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(3)      ; -- END O MACRO -----
(2) 017110 013701 001366      MOV      PAR,R1      ;PICK UP PARAMETERS
(2) 017114 012702 000001      MOV      #1,R2      ;PICK UP INIT POINTER
(2) 017120 030237 001364      1$: BIT      R2,LINE      ;SHOULD THIS LINE BE SET UP ?
(2) 017124 001402      BEQ      2$      ;NO
(2) 017126 010177 162720      MOV      R1,@DZLPR      ;SET UP LINE PARAMETERS
(2) 017132 005201      2$: INC      R1      ;POSITION POINTER TO THE NEXT LINE
(2) 017134 106302      ASLB      R2      ;GOT 'EM ALL ?
(2) 017136 103370      BCC      1$      ;IF NO, GO SET UP THE NEXT LINE
(2) 017140 005037 001372      CLR      SAVLIN      ;CLEAR LINE # INDICATOR
(2)      ; -- END O MACRO -----
(1) 017144 106437 030334      MTPS      @#DZPRT      ;SET CPU STATUS TO DZ11 PRIO,
(1) 017150 113777 001364 162700      MOV      LINE,@DZTCR      ;ENABLE THE VALID LINES

```

```

(1) 017156 3$:
(2)
(2) 017156 012777 017246 162712 MOV #6$, @DZTIV ; - $INTSET-----
(2) 017164 012777 017254 162700 MOV #7$, @DZRIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017172 013777 030334 162674 MOV DZPRT, @DZ RIS ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017200 013777 030334 162672 MOV DZPRT, @DZTIS ;SET THE INTERRUPT VECTOR STATUS
(2) 017206 052777 040040 162626 BIS #TIE! MSENAB, @DZCSR ;SET TRANSMITTER INTERRUPT PRIORITY
; -- END O MACRO -----
(1) 017214 005005 CLR R5
(1) 017216 032777 100000 162616 4$: BIT #TRDY, @DZCSR
(1) 017224 001403 BEQ 5$
(1) 017226 000240 NOP
(1) 017230 000240 NOP
(1) 017232 000412 BR 8$
(1) 017234 104414 5$: DELAY
(1) 017236 005205 INC R5
(1) 017240 001366 BNE 4$
(1) 017242 104003 ERROR 3 ;*TRDY NOT SET!
(1) 017244 000405 BR 8$
(1) 017246 104010 6$: ERROR 10 ;*TRANSMITTER SHOULD NOT INTERRUPT
(1) 017250 022626 CMP (SP)+, (SP)+ ;POP FOR FAKE RTI
(1) 017252 000402 BR 8$ ;CONTINUE TEST
(1) 017254 104012 7$: ERROR 12 ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017256 022626 CMP (SP)+, (SP)+ ;POP FOR FAKE RTI
(1) 017260 042777 040000 162554 8$: BIC #TIE, @DZCSR ;RESET TRANSMITTER INTERRUPT ENABLE
; - $INTSET-----
(2) 017266 012777 017364 162602 MOV #11$, @DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017274 012777 017372 162570 MOV #12$, @DZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017302 013777 030334 162564 MOV DZPRT, @DZ RIS ;SET THE INTERRUPT VECTOR STATUS
(2) 017310 013777 030334 162562 MOV DZPRT, @DZTIS ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017316 052777 000140 162516 BIS #RIE! MSENAB, @DZCSR ;ENABLE THE DEVICE
; -- END O MACRO -----
(1) 017324 113777 001422 162534 MOV B TD0, @DZTDR ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
(1) 017332 005005 CLR R5
(1) 017334 032777 000200 162500 9$: BIT #RDONE, @DZCSR
(1) 017342 001403 BEQ 10$
(1) 017344 000240 NOP
(1) 017346 000240 NOP
(1) 017350 000412 BR 13$
(1) 017352 104414 10$: DELAY
(1) 017354 005205 INC R5
(1) 017356 001366 BNE 9$
(1) 017360 104004 ERROR 4 ;*NO RX DONE! (NOT SET)
(1) 017362 000405 BR 13$ ;CONTINUE TEST
(1) 017364 104010 11$: ERROR 10 ;*TRANSMITTER SHOULD NOT INTERRUPT
(1) 017366 022626 CMP (SP)+, (SP)+ ;POP FOR FAKE RTI
(1) 017370 000402 BR 13$ ;CONT TEST
(1) 017372 104012 12$: ERROR 12 ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017374 022626 CMP (SP)+, (SP)+ ;POP FOR FAKE RTI
(1) 017376 13$:
(2) 017376 104413 DEVICE.CLR ;ISSUE DEVICE CLEAR (RESET)
(2) ; -- END O MACRO -----
(1) ; -- END O MACRO -----
9091 ; - $LVLTST-----
(2) ; - $XZ-----
(3) ;***** 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.
(2) ; - $XZ-----
(3) ;:* TEST 26
(6) ;*****
(5) 017400 000004 TST26: SCOPE
(3) 017402 012737 000026 001122 MOV #26,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3) ; -- END 0 MACRO -----
(3) 017410 012737 017726 001360 MOV #TST27,NEXT ;POINT TO THE START OF THE NEXT TEST
(2) ;$LINEUP-----
(3) ; - $MRESET-----
(3) 017416 104417 DCLASH ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(3) ; -- END 0 MACRO -----
(2) 017420 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
(2) 017424 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
(2) 017430 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
(2) 017434 001402 BEQ 2$ ;NO
(2) 017436 010177 162410 MOV R1,@DZLPR ;SET UP LINE PARAMETERS
(2) 017442 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
(2) 017444 106302 ASLB R2 ;GOT 'EM ALL ?
(2) 017446 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
(2) 017450 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
(2) ; -- END 0 MACRO -----
(1) 017454 106437 030336 MTPS @#LESS1 ;MAKE CPU ONE LEVEL LOWER THAN DZ11
(1) 017460 113777 001364 162370 MOVB LINE,@DZTCR ;ENABLE THE VALID LINES
(1) 017466 3$: ; - $INTSET-----
(2) 017466 012777 017560 162402 MOV #6$,@DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 017474 012777 017576 162370 MOV #7$,@DZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017502 013777 030334 162364 MOV DZPRT,@DZCRIS ;SET THE INTERRUPT VECTOR STATUS
(2) 017510 013777 030334 162362 MOV DZPRT,@DZTIS ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017516 052777 040040 162316 BIS #TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(2) ; -- END 0 MACRO -----
(1) 017524 005005 CLR R5
(1) 017526 032777 100000 162306 4$: BIT #TRDY,@DZCSR
(1) 017534 001404 BEQ 5$
(1) 017536 000240 NOP
(1) 017540 000240 NOP
(1) 017542 104007 ERROR 7 ;*TRANSMITTER FAILED TO INTERRUPT
(1) 017544 000416 BR 8$
(1) 017546 104414 5$: DELAY
(1) 017550 005205 INC R5
(1) 017552 001365 BNE 4$
(1) 017554 104003 ERROR 3 ;*TRDY NOT SET!
(1) 017556 000411 BR 8$
(1) 017560 022626 6$: POP2SP
(1) 017562 042777 040000 162252 BIC #TIE,@DZCSR ;REMOVE THE INTERRUPT FROM THE STACK
(1) 017570 106437 030336 MTPS @#LESS1 ;DON'T LET ANY MORE INTERRUPTS OCCUR
(1) 017574 000402 BR 8$ ;MAKE CPU ONE LEVEL LOWER THAN DZ11
(1) 017576 104012 7$: ERROR 12 ;RETURN TO THE NORMAL FLOW
(1) 017600 022626 CMP (SP)+,(SP)+ ;*RECEIVER SHOULD NOT INTERRUPT
(1) 017602 042777 040000 162232 8$: BIC #TIE,@DZCSR ;POP FOR FAKE RTI
(2) ;RESET TRANSMITTER INTERRUPT ENABLE
(2) ; - $INTSET-----
(2) 017610 012777 017710 162260 MOV #11$,@DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
```

```

(2) 017616 012777 017716 162246      MOV    #12$,@DZRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 017624 013777 030334 162242      MOV    DZPRT,@DZRI5    ;SET THE INTERRUPT VECTOR STATUS
(2) 017632 013777 030334 162240      MOV    DZPRT,@DZTI5    ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 017640 052777 000140 162174      BIS     #RIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(2)                                     ; -- END 0 MACRO =====
(1) 017646 113777 001422 162212      MOVB   TDO,@DZTDR      ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
(1) 017654 005005                                     CLR     R5
(1) 017656 032777 000200 162156 9$:   BIT     #RDONE,@DZCSR
(1) 017664 001404                                     BEQ     10$
(1) 017666 000240                                     NOP
(1) 017670 000240                                     NOP
(1) 017672 104011      ERROR    11                                     ;*RECEIVER FAILED TO INTERRUPT
(1) 017674 000413      BR       13$
(1) 017676 104414      10$:    DELAY
(1) 017700 005205      INC      R5
(1) 017702 001365      BNE      9$
(1) 017704 104004      ERROR    4                                     ;*NO RX DONE! (NOT SET)
(1) 017706 000406      BR       13$                                     ;CONTINUE TEST
(1) 017710 104010      11$:    ERROR    10                             ;*TRANSMITTER SHOULD NOT INTERRUPT
(1) 017712 022626      CMP      (SP)+,(SP)+                         ;POP FOR FAKE RTI
(1) 017714 000403      BR       13$                                     ;CONT TEST
(1) 017716 022626      12$:    POP2SP                               ;REMOVE THE INTERRUPT FROM THE STACK
(1) 017720 005077 162116      CLR     @DZCSR                         ;DON'T ALLOW ANY MORE INTERRUPTS
(1) 017724                                     13$:
(2) 017724 104413      DEVICE.CLR                                ;ISSUE DEVICE CLEAR (RESET)
(2)                                     ; -- END 0 MACRO =====
(1)                                     ; -- END 0 MACRO =====
9092
9093                                     ; -$XZ-----
(2)                                     ;***** TEST 27 *****
9094                                     ;*THIS TEST VERIFIES THAT THE RECEIVER WILL
9095                                     ;*INTERRUPT BEFORE THE TRANSMITTER EVEN
9096                                     ;*THOUGH THE TRANSMITTER WAS ENABLED
9097                                     ;*FIRST. SET PS TO LEVEL 7;
9098                                     ;*GET RDONE AND TRDY TO SET;
9099                                     ;*SET TX IE AND RX IE;
9100                                     ;*CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
9101                                     ; -$XZ-----
9102                                     ;:* TEST 27
(5)                                     ;*****
(4) 017726 000004      TST27:  SCOPE
(2) 017730 012737 000027 001122      MOV     #27,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(2)                                     ; -- END 0 MACRO =====
(2) 017736 012737 020360 001360      MOV     #TST30,NEXT    ;POINT TO THE START OF THE NEXT TEST
9103                                     ;$LINEUP-----
(2)                                     ; -$MRESET-----
(2) 017744 104417      DCLASM
(2)                                     ; -- END 0 MACRO =====
(1) 017746 013701 001366      MOV     PAR,R1      ;PICK UP PARAMETERS
(1) 017752 012702 000001      MOV     #1,R2      ;PICK UP INIT POINTER
(1) 017756 030237 001364      1$:   BIT     R2,LINE      ;SHOULD THIS LINE BE SET UP ?
(1) 017762 001402      BEQ     2$      ;NO
(1) 017764 010177 162062      MOV     R1,@DZLPR      ;SET UP LINE PARAMETERS
(1) 017770 005201      2$:   INC      R1      ;POSITION POINTER TO THE NEXT LINE
(1) 017772 106302      ASLB     R2      ;GOT 'EM ALL ?
(1) 017774 103370      BCC     1$      ;IF NO, GO SET UP THE NEXT LINE

```


CZDZA-IO MACY11 30A(1052) 02-JUL-85 16:19 PAGE 83-67
 CZDZAI.P11 02-JUL-85 16:17 CZDZA DZ11 DEVICE DIAGNOSTICS.

```

(1) 017776 005037 001372          CLR      SAVLIN          ;CLEAR LINE # INDICATOR
(1)          ; -- END 0 MACRO =====
9104 020002 012777 020232 162062  MOV      #8$,@DZRIV      ;SETUP INTERRUPT STUFF
9105 020010 013777 030334 162056  MOV      DZPRT,@DZRI$      ;
9106 020016 012777 020322 162052  MOV      #12$,@DZTIV      ;
9107 020024 013777 030334 162046  MOV      DZPRT,@DZTIS      ;
9108 020032 052777 000040 162002  BIS      #MSENAB,@DZCSR      ;
9109 020040 012702 000001          MOV      #1,R2          ;LINE POINTER
9110 020044 030237 001364          3$:  BIT      R2,LINE      ;VALID LINE ?
9111 020050 001004          BNE      4$
9112 020052 005237 001372          INC      SAVLIN
9113 020056 106302          ASLB      R2
9114 020060 000771          BR       3$
9115 020062 106427 000340          4$:  MTPS      #PR7
9116 020066 000240          NOP
9117 020070 000240          NOP
9118 020072 110277 161760  MOVB      R2,@DZTCR      ;SET TCR BIT
9119 020076 005777 161744  TST      @DZRBUF      ;VALID DATA?
9120 020102 100001          BPL      .+4          ;IT BETTER NOT BE SET
9121 020104 104017          ERROR     17          ;DATA VALID SHOULD NOT BE SET
9122 020106 105777 161730          5$:  TSTB      @DZCSR      ;RECEIVER DONE ?
9123 020112 100001          BPL      .+4
9124 020114 104020          ERROR     20          ;RECEIVER DONE BIT SHOULD NOT BE SET
9125 020116 005005          CLR      R5
9126 020120 005004          CLR      R4
9127 020122 005777 161714          99$:  TST      @DZCSR      ;WAIT FOR TRDY
9128 020126 100404          BMI      100$      ;BR IF READY
9129 020130 104414          DELAY
9130 020132 005204          INC      R4          ;STALL TIME
9131 020134 001372          BNE      99$
9132 020136 104003          ERROR     3
9133 020140 105077 161722          100$:  CLR      @DZTDR      ;TRDY FAILED TO SET
9134 020144 005004          CLR      R4
9135 020146 032777 000200 161666  6$:  BIT      #RDONE,@DZCSR
9136 020154 001004          BNE      7$
9137 020156 104414          DELAY
9138 020160 005204          INC      R4
9139 020162 001371          BNE      6$
9140 020164 104004          ERROR     4          ;*RDONE FAILED TO SET!
9141 020166 005777 161650          7$:  TST      @DZCSR      ;TRANS DONE BIT = 1 ?
9142 020172 100401          BMI      .+4          ;YES
9143 020174 104003          ERROR     3          ;*NO TRANS DONE FAILED TO SET
9144          ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
9145          ;SET INTERRUPT ENABLES AND WATCH THE FUR FLY
9146 020176 052777 040000 161636  BIS      #TIE,@DZCSR
9147 020204 052777 000100 161630  BIS      #RIE,@DZCSR
9148 020212 106427 000000          MTPS      #0
9149 020216 000240          NOP
9150 020220 000240          NOP
9151 020222 104007          ERROR     7          ;*TRANSMITTER FAILED TO INTERRUPT
9152 020224 104011          ERROR     11         ;*RECEIVER FAILED TO INTERRUPT
9153          ;CHECK BR LEVEL
9154 020226 000137 020326          JMP      13$      ;GET OUT
9155
9156          ;RECEIVER INTERRUPT ROUTINE
9157 020232 017704 161610          8$:  MOV      @DZRBUF,R4          ;ACTUAL

```

```

9158 020236 010403      MOV    R4,R3
9159 020240 000303      SWAB   R3
9160 020242 042703 177770 BIC    #+C<7>,R3      ;STRIP JUNK
9161                                     ; - $STAG-----
(1) 020246 105737 001371 TSTB   MODE+1      ;IS THIS TEST IN STAGGERED MODE?
(1) 020252 001406      BEQ    11$      ;IF NOT, SKIP STAGGERED SETUP
(1)                                     ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1)
(1) 020254 006203      ASR     R3      ;GET THE LAST BIT INTO THE CARRY BIT
(1) 020256 103402      BCS     9$      ;IF IT IS SET, GO CLEAR IT
(1) 020260 000261      SEC          ;IF IT IS CLEAR SET IT HERE
(1) 020262 000401      BR      10$      ;SKIP THE CLEARING
(1) 020264 000241      9$: CLC          ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 020266 006103      10$: ROL    R3      ;GET THE NEW BIT BACK INTO R3
(1)                                     ; -- END O MACRO -----
9162 020270 020337 001372 11$: CMP    R3,SAVLIN      ;IS THIS A VALID LINE
9163 020274 001401      BEQ     .+4      ;YES
9164 020276 104015      ERROR   15      ;*INVALID LINE
9165 020300 042704 177400 BIC    #+C<377>,R4      ;STRIP JUNK
9166 020304 120504      CMPB   R5,R4      ;DATA COMPARE ?
9167 020306 001401      BEQ     .+4      ;YES
9168 020310 104005      ERROR   5       ;*DATA DOES NOT COMPARE
9169 020312 040277 161540 BIC    R2,@DZTCR      ;CLEAR TCR BIT
9170 020316 022626      POP2SP      ;REMOVE HE INTERRUPT VECTOR FROM THE STACK
9171 020320 000402      BR      13$      ;GO GET OUT OF INTERRUPT MODE
9172                                     ;TRANSMITTER INTERRUPT SVC ROUTINE
9173 020322 104011      12$: ERROR 11      ;THE RECEIVER INTERRUPT FAILED
9174                                     ;TO OVERRIDE THE TRANSMITTER
9175 020324 022626      POP2SP      ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
9176 020326 042777 040100 161506 13$: BIC    #TIE!RIE,@DZCSR ;CLEAR INTERRUPT ENABLES
9177 020334 013777 002074 161530      MOV    DZRI5,@DZRIV ;RESTORE TRAPCATCHER
9178 020342 005077 161526      CLR     @DZRIS
9179 020346 013777 002100 161522      MOV    DZTI5,@DZTIV
9180 020354 005077 161520      CLR     @DZTI5
9181                                     ; - $XZ-----
(2)                                     ;***** TEST 30 *****
9182                                     ;*TEST TO VERIFY THAT 'RDONE DOES NOT SET
9183                                     ;*IF THE SCANNER IS DISABLED.
9184                                     ;*TURN ON SCANNER, WAIT FOR TRDY.
9185                                     ;*TURN OFF SCANNER, TRANSMIT A CHARACTER
9186                                     ;*'RDONE SHOULD NOT SET.
9187                                     ; - $XZ-----
9188                                     ;:* TEST 30
(5)                                     ;*****
(4) 020360 000004      TST30: SCOPE
(2) 020362 012737 000030 001122      MOV    #30,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(2)                                     ; -- END O MACRO -----
(2) 020370 012737 020546 001360      MOV    #TST31,NEXT      ;POINT TO THE START OF THE NEXT TEST
9189                                     ;$LINEUP-----
(2)                                     ; - $MRESET-----
(2) 020376 104417      DCLASM      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2)                                     ; -- END O MACRO -----
(1) 020400 013701 001366      MOV    PAR,R1      ;PICK UP PARAMETERS
(1) 020404 012702 000001      MOV    #1,R2      ;PICK UP INIT POINTER
(1) 020410 030237 001364      1$: BIT    R2,LINE      ;SHOULD THIS LINE BE SET UP ?

```



```

(1) 020414 001402      BEQ      2$      ;NO
(1) 020416 010177 161430  MOV      R1, @DZLPR ;SET UP LINE PARAMETERS
(1) 020422 005201      2$: INC      R1      ;POSITION POINTER TO THE NEXT LINE
(1) 020424 106302      ASLB     R2      ;GOT 'EM ALL ?
(1) 020426 103370      BCC      1$      ;IF NO, GO SET UP THE NEXT LINE
(1) 020430 005037 001372  CLR      SAVLIN ;CLEAR LINE # INDICATOR
(1)      ; -- END O MACRO =====
9190 020434 052777 000040 161400  BIS      #MSENAB, @DZCSR ;TURN ON SCANNER
9191 020442 012702 000001      MOV      #1, R2 ;INIT LINE COUNTER
9192 020446 030237 001364      3$: BIT      R2, LINE ;FIND A VALID LINE
9193 020452 001004      BNE      4$      ;IF WE FOUND ONE GO TO TEST
9194 020454 005237 001372      INC      SAVLIN ;IF NOT
9195 020460 106302      ASLB     R2      ;KEEP LOOKING
9196 020462 000771      BR       3$
9197 020464 110277 161366      4$: MOVB     R2, @DZTCR ;SET TCR BIT
9198 020470 005005      CLR      R5
9199 020472 005777 161344      5$: TST      @DZCSR ;IS TRDY SET
9200 020476 100404      BMI      6$      ;CON'T TESTING IF IT IS
9201 020500 104414      DELAY     ;IF IT NOT WAIT A WHILE
9202 020502 005205      INC      R5
9203 020504 001372      BNE      5$
9204 020506 104003      ERROR     3 ;WE WAITED LONG ENOUGH-ERROR
9205 020510 042777 000040 161324 6$: BIC      #MSENAB, @DZCSR ;TURN OFF SCANNER
9206 020516 105077 161344      CLRB     @DZTCR ;TRANSMIT A CHARACTER
9207 020522 005005      CLR      R5 ;CLEAR COUNTER
9208 020524 104414      7$: DELAY     ;WAIT SUFFICIENT TIME FOR
9209 020526 005205      INC      R5 ;RDONE TO SET
9210 020530 001375      BNE      7$
9211 020532 032777 000200 161302  BIT      #RDONE, @DZCSR ;RDONE SET
9212 020540 001401      BEQ      8$      ;IT SHOULDN'T BE-CONTINUE
9213 020542 104020      ERROR     20 ;IF IT IS THERE'S AN ERROR
9214 020544 104400      8$: ADVANCE
9215      ; -$XZ-----
(2)      ;***** TEST 31 *****
9216      ;*THIS TEST VERIFIES OVERRUN AND SILO ALARM
9217      ;*ONE LINE AT A TIME - BASED UPON VALID LINES
9218      ;*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
9219      ;*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
9220      ;*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
9221      ;*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
9222      ;*CHAR PULLED OUT OUT THE SILO.
9223      ;*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
9224      ;*ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
9225      ;*USED TO SCOPE SILO ALARM PULSES, ETC.
9226      ; -$XZ-----
9227      ;:* TEST 31
(5)      ;*****
(4) 020546 000004      TST31: SCOPE
(2) 020550 012737 000031 001122  MOV      #31, $TSTNM ;LOAD THE NUMBER OF THIS TEST
(2)      ; -- END O MACRO =====
(2) 020556 012737 021274 001360  MOV      #TST32, NEXT ;POINT TO THE START OF THE NEXT TEST
9228 020564 012737 021200 001362  MOV      #18$, LOCK ;SET FOR LOOP
9229      ;$LINEUP-----
(2)      ; -$MRESET-----
(2) 020572 104417      DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2)      ; -- END O MACRO =====

```

(1)	020574	013701	001366		MOV	PAR,R1		;PICK UP PARAMETERS
(1)	020600	012702	000001		MOV	#1,R2		;PICK UP INIT POINTER
(1)	020604	030237	001364	1\$:	BIT	R2,LINE		;SHOULD THIS LINE BE SET UP ?
(1)	020610	001402			BEQ	2\$		;NO
(1)	020612	010177	161234		MOV	R1,@DZLPR		;SET UP LINE PARAMETERS
(1)	020616	005201		2\$:	INC	R1		;POSITION POINTER TO THE NEXT LINE
(1)	020620	106302			ASLB	R2		;GOT 'EM ALL ?
(1)	020622	103370			BCC	1\$		;IF NO, GO SET UP THE NEXT LINE
(1)	020624	005037	001372		CLR	SAVLIN		;CLEAR LINE # INDICATOR
(1)					; -- END O MACRO =====			
9230	020630	012700	001422		MOV	#TDO,R0		;POINT TO THE DATA AREA
9231	020634	005020			CLR	(R0)+		;CLEAR A DATA WORD
9232	020636	022700	001462		CMP	#STOP,R0		;FINISHED ?
9233	020642	001374			BNE	.-6		;NO
9234	020644	005000			CLR	R0		;CLEAR OFFSET
9235	020646	012702	000001		MOV	#1,R2		;LINE POINTER
9236	020652	052777	010040	161162	BIS	#MSENAB!SILOEN,@DZCSR		;START SCANNER & SET SILO ENABLE
9237	020660	030237	001364	3\$:	BIT	R2,LINE		;VALID LINE?
9238	020664	001002			BNE	+.6		;YES
9239	020666	000137	021162		JMP	22\$		;TRY NEXT LINE
9240	020672	013700	001372		MOV	SAVLIN,R0		;MAKE OFFSET
9241	020676	006300			ASL	R0		;MAKE POWER OF TWO
9242	020700	010277	161152		MOV	R2,@DZTCR		;SET TCR BIT
9243	020704	105777	161132	4\$:	TSTB	@DZCSR		;REC DONE = 1 ?
9244	020710	100001			BPL	+.4		
9245	020712	104020			ERROR	20		;REC DONE SHOULD NOT = 1
9246	020714	005003			CLR	R3		;SET CHARACTER COUNT
9247	020716	005004		5\$:	CLR	R4		
9248	020720	032777	100000	161114	6\$:	BIT	#TRDY,@DZCSR	
9249	020726	001004			BNE	7\$		
9250	020730	104414			DELAY			
9251	020732	105204			INCB	R4		
9252	020734	001371			BNE	6\$		
9253	020736	104003			ERROR	3		;*TRDY FAILED TO SET
9254	020740	116077	001422	161120	7\$:	MOVB	TDO(R0),@DZTDR	;LOAD A CHARACTER
9255	020746	005260	001422		INC	TDO(R0)		;SET UP NEXT CHARACTER
9256	020752	020327	000017		CMP	R3,#15.		;16 CHARACTERS ?
9257	020756	103006			BHIS	8\$		
9258	020760	032777	020000	161054	BIT	#SILOAL,@DZCSR		;SILO ALARM = 0 ?
9259	020766	001401			BEQ	+.4		;YES
9260	020770	104013			ERROR	13		;*SILO ALARM SHOULD NOT = 1
9261								;UNTIL 16. DATA CHARACTERS
9262	020772	000411			BR	10\$		
9263	020774	005004			CLR	R4		
9264	020776	032777	020000	161036	8\$:	BIT	#SILOAL,@DZCSR	
9265	021004	001004			9\$:	BNE	10\$	
9266	021006	104414			DELAY			
9267	021010	005204			INC	R4		
9268	021012	001371			BNE	9\$		
9269	021014	104014			ERROR	14		;*SILO ALARM FAILED TO SET!
9270								;SILO ALARM SHOULD =1 AFTER 16.
9271								;DATA CHARACTERS
9272	021016	005203			10\$:	INC	R3	;INC CHAR COUNT
9273	021020	022703	000102		CMP	#66.,R3		;FINISHED SENDING CHARACTERS ?
9274	021024	001334			BNE	5\$		
9275	021026	005004			CLR	R4		;NO


```

9276 021030 104414      DELAY
9277 021032 105204      INCB    R4
9278 021034 001375      BNE     .-4
9279                      ;NOW LETS READ THE SILO
9280 021036 013705 001372 MOV     SAVLIN,R5      ;MAKE EXPECTED LINE #
9281                      ; - $STAG-----
(1) 021042 105737 001371 TSTB    MODE+1      ;IS THIS TEST IN STAGGERED MODE?
(1) 021046 001406      BEQ     13$      ;IF NOT, SKIP STAGGERED SETUP
(1)
(1)                      ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
(1) 021050 006205      ASR     R5      ;GET THE LAST BIT INTO THE CARRY BIT
(1) 021052 103402      BCS     11$      ;IF IT IS SET, GO CLEAR IT
(1) 021054 000261      SEC          ;IF IT IS CLEAR SET IT HERE
(1) 021056 000401      BR      12$      ;SKIP THE CLEARING
(1) 021060 000241      11$: CLC          ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
(1) 021062 006105      12$: ROL     R5      ;GET THE NEW BIT BACK INTO R5
(1)                      ; -- END O MACRO -----
9282 021064 000305      13$: SWAB    R5      ;PUT IN UPPER BYTE
9283 021066 052705 100000 BIS     @DVALID,R5      ;ADD DATA VALID
9284 021072 017704 160750 14$: MOV     @DZRBUFF,R4      ;ACTUAL
9285 021076 020405      CMP     R4,R5      ;ACTUAL VS. EXPECTED
9286 021100 001401      BEQ     15$      ;YES
9287 021102 104006      ERROR    6      ;*DATA/CONTENTS DID NOT COMPARE
9288 021104 032777 020000 160730 15$: BIT     @SILOAL,@DZCSR      ;SILO ALARM= 0 ?
9289 021112 001401      BEQ     16$      ;YES
9290 021114 104016      ERROR    16      ;READING DZRBUFF DID NOT CLEAR SILO ALARM
9291 021116 005205      16$: INC     R5      ;UP CHARACTER
9292 021120 120527 000077 CMPB    R5,#63.      ;LAST SILO CHAR ?....64TH CHAR
9293 021124 101762      BLOS    14$
9294 021126 005205      INC     R5      ;ADD 1 MORE FOR THE CLOBBED CHAR
9295 021130 052705 040000 BIS     @OVRUN,R5      ;ADD OVERRUN TO EXPECTED
9296 021134 120527 000101 CMPB    R5,#65.      ;LAST CHARACTER ?
9297 021140 001754      BEQ     14$
9298 021142 017704 160700 MOV     @DZRBUFF,R4      ;FOR GOOD MEASURE
9299 021146 005704      TST     R4      ;DATA VALID SHOULD = 0
9300 021150 100001      BPL     17$      ;YES
9301 021152 104017      ERROR    17      ;DATA VALID SHOULD = 0
9302 021154 040277 160676 17$: BIC     R2,@DZTCR      ;CLR TCR BIT
9303 021160 104401      SCOP1
9304 021162 005237 001372 22$: INC     SAVLIN      ;INC EXPECTED LINE
9305 021166 106302      ASLB    R2      ;NEXT LINE
9306 021170 103402      BCS     .+6      ;NO
9307 021172 000137 020660 JMP     3$      ;YES
9308 021176 104400      ADVANCE      ;GO TO NEXT TEST
9309
9310                      ;TIGHT SCOPE LOOP FOR THIS TEST. SENDS 20. CHARACTERS
9311                      ;ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
9312                      ;USED TO SCOPE SILO ALARM PULSES, ETC.
9313
9314 021200 052777 010040 160634 18$: BIS     @MSENAB!SILOEN,@DZCSR      ;SETUP DEVICE
9315 021206 012777 021264 160662 MOV     @20$,@DZTIV      ;SETUP TRANSMITTER VECTOR
9316 021214 012737 000024 001216 MOV     @20$,@TMP0      ;TEMPORARY COUNT OF CHARACTER BURST
9317 021222 050277 160630      BIS     R2,@DZTCR      ;ENABLE LINE
9318 021226 052777 040000 160606 BIS     @TIE,@DZCSR      ;ENABLE INTERRUPTS
9319 021234 106427 000000 MTPS    @0      ;LOWER PRIORITY

```

```

9320 021240 000001          19: WAIT          ;ALLOW INTERRUPTS
9321 021242 005337 001216    DEC      $TMP0    ;REDUCE COUNT. ALL CHARACTERS SENT?
9322 021246 001374          BNE      19:        ;IF NO, WAIT FOR MORE
9323 021250 042777 050040 160564 BIC      $SILOEN!$SENAB!TIE,$DZCSR ;RESET SILO COUNTER, CLEAR STROBE
9324 021256 104401          SCOP1          ;LOOP AGAIN?
9325 021260 000137 021154    JMP      17:        ;IF NOT, RETURN TO WHERE YOU LEFT OFF
9326 021264 112777 000252 160574 20: MOV     $252,$DZTDR ;SEND A CHARACTER
9327 021272 000002          RTI              ;ALLOW MORE CHARACTERS TO COME
9328                                     ; -$XZ-----
9329                                     ;***** TEST 32 *****
9330                                     ;*THIS TEST THAT "SILO ENABLE" WILL INHIBIT
9331                                     ;*RECEIVER INTERRUPTS AND THAT ON THE
9332                                     ;*16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
9333                                     ;*INTERRUPT WITH "RIE" SET.
9334                                     ;*THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
9335                                     ; -$XZ-----
9336                                     ;:* TEST 32
9337                                     ;*****
9338 (5) 021274 000004          TST32: SCOPE
9339 (4) 021276 012737 000032 001122    MOV     $32,$TSTNM ;LOAD THE NUMBER OF THIS TEST
9340 (2) 021276 012737 000032 001122    ; -- END 0 MACRO -----
9341 (2) 021304 012737 021656 001360    MOV     $TST33,NEXT ;POINT TO THE START OF THE NEXT TEST
9342 021312 012737 021400 001362    MOV     $3$,LOCK ;SET FOR LOOP
9343                                     ;$LINEUP-----
9344 (2) 021320 104417          ; -$MRESET-----
9345 (2)                                     ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
9346 (2) 021322 013701 001366    ; -- END 0 MACRO -----
9347 (1) 021326 012702 000001    MOV     PAR,R1 ;PICK UP PARAMETERS
9348 (1) 021332 030237 001364    MOV     $1,R2 ;PICK UP INIT POINTER
9349 (1) 021336 001402          1: BIT     R2,LINE ;SHOULD THIS LINE BE SET UP ?
9350 (1) 021340 010177 160506    BEQ     2:        ;NO
9351 (1) 021344 005201          MOV     R1,$DZLPR ;SET UP LINE PARAMETERS
9352 (1) 021346 106302          2: INC     R1 ;POSITION POINTER TO THE NEXT LINE
9353 (1) 021350 103370          ASLB    R2 ;GOT 'EM ALL ?
9354 (1) 021352 005037 001372    BCC     1:        ;IF NO, GO SET UP THE NEXT LINE
9355                                     ;CLEAR LINE # INDICATOR
9356 (1) 021356 012700 001422    ; -- END 0 MACRO -----
9357 021362 005020          MOV     $TDO,R0 ;POINT TO THE DATA AREA
9358 021364 022700 001462    CLR      (R0)+ ;CLEAR A DATA WORD
9359 021370 001374          CMP     $STOP,R0 ;FINISHED ?
9360 021372 005000          BNE      .-6 ;NO
9361 021374 012702 000001    CLR      R0 ;CLEAR OFFSET
9362 021400 012777 021620 160464 3: MOV     $1,R2 ;LINE POINTER
9363 021406 012777 000340 160460    MOV     $11,$DZDRIV ;SET FOR UNEXPECTED INTER.
9364 021414 052777 010140 160420    MOV     $PR7,$DZDRIS ;SET PRIO.
9365                                     BIS     $SENAB!$SILOEN!RIE,$DZCSR ;START SCANNER & SET SILO ENABLE
9366 021422 030237 001364    BIT     R2,LINE ;VALID LINE?
9367 021426 001002          BNE      .+6 ;YES
9368 021430 000137 021636    JMP     22:        ;TRY NEXT LINE
9369 021434 005777 160406    TST     $DZRBUF ;EMPTY THE SILO
9370 021440 100775          BMI      .-4 ;BR IF DATA VALID IS SET!
9371 021442 106427 000000    MTPS    $0 ;SET PROCESSOR PRIORITY TO 0
9372 021446 013700 001372    MOV     SAVLIN,R0 ;MAKE OFFSET
9373 021452 006300          ASL      R0 ;MAKE POWER OF TWO
9374 021454 010277 160376    MOV     R2,$DZTCR ;SET TCR BIT

```


9357	021460	005004			5\$:	CLR	R4	
9358	021462	032777	100000	160352	6\$:	BIT	#TRDY, @DZCSR	
9359	021470	001004				BNE	7\$	
9360	021472	104414				DELAY		
9361	021474	005204				INC	R4	
9362	021476	001371				BNE	6\$	
9363	021500	104003				ERROR	3	; *TRDY FAILED TO SET
9364	021502	116077	001422	160356	7\$:	MOVB	TDO(R0), @DZTDR	; LOAD A CHARACTER
9365	021510	005260	001422			INC	TDO(R0)	; SET UP NEXT CHARACTER
9366	021514	022760	000017	001422		CMP	#15., TDO(R0)	; 15 CHARS YET?
9367	021522	001406				BEQ	8\$	
9368	021524	032777	020000	160310		BIT	#SILOAL, @DZCSR	; SILO ALARM = 0 ?
9369	021532	001401				BEQ	..4	; YES
9370	021534	104013				ERROR	13	; *SILO ALARM SHOULD NOT = 1
9371								; UNTIL 16. DATA CHARACTERS
9372	021536	000750				BR	5\$	
9373	021540	012777	021626	160324	8\$:	MOV	#12\$, @DZRIV	; SET NEW VECTOR
9374	021546	032777	100000	160266		BIT	#TRDY, @DZCSR	; READY FOR 16TH CHAR
9375	021554	001774				BEQ	..6	
9376	021556	016077	001422	160302		MOV	TDO(R0), @DZTDR	; LOAD THE 16TH CHAR.
9377	021564	005004				CLR	R4	
9378	021566	032777	020000	160246	9\$:	BIT	#SILOAL, @DZCSR	
9379	021574	001005				BNE	10\$	
9380	021576	104414				DELAY		
9381	021600	005204				INC	R4	
9382	021602	001371				BNE	9\$	
9383	021604	104014				ERROR	14	; *SILO ALARM FAILED TO SET!
9384	021606	000410				BR	17\$	; SILO ALARM SHOULD =1 AFTER 16.
9385								; DATA CHARACTERS
9386	021610	000240			10\$:	NOP		; STALL
9387	021612	000240				NOP		
9388	021614	104000				ERROR		; SILO ALARM NOT INTERRUPTING.
9389	021616	000404				BR	17\$	; CONTINUE TEST.
9390	021620	022626			11\$:	CMP	(SP)+, (SP)+	; FAKE RTI
9391	021622	104012				ERROR	12	; RX SHOULD NOT INTERRUPT
9392	021624	000401				BR	17\$	; CONTINUE
9393	021626	022626			12\$:	CMP	(SP)+, (SP)+	; GOOD INTERRUPT TO HERE.
9394	021630	040277	160222		17\$:	BIC	R2, @DZTCR	; CLR TCR BIT
9395	021634	104401				SCOP1		; LOOP?
9396	021636	005237	001372		22\$:	INC	SAVLIN	; INC EXPECTED LINE
9397	021642	106302				ASLB	R2	; NEXT LINE
9398	021644	103402				BCS	..6	; NO
9399	021646	000137	021400			JMP	3\$	; YES
9400	021652	005037	001362			CLR	LOCK	; CLEAR TIGHT LOOP FOR NEXT TEST

```
9402
9403
(2)
9404
9405
9406
9407
9408
9409
(5)
(4) 021656 000004
(2) 021660 012737 000033 001122
(2)
(2) 021666 012737 022464 001360
9410
(1) 021674 104417
(1)
9411 021676 013737 001364 022462
9412 021704 013701 001366
9413 021710 012700 000001
9414 021714 030037 001364
9415 021720 001402
9416 021722 010177 160124
9417 021726 005201
9418 021730 106300
9419 021732 103370
9420 021734 012700 001422
9421 021740 005020
9422 021742 022700 001462
9423 021746 001374
9424 021750 012777 022204 160114
9425 021756 012777 000340 160110
9426 021764 012777 022106 160104
9427 021772 012777 000340 160100
9428 022000 052777 000100 160034
9429 022006 052777 040000 160026
9430 022014 052777 000040 160020
9431 022022 113777 001364 160026
9432 022030 106437 030336
9433
9434
9435 022034 005037 022104
9436 022040 013727 007204
9437 022044 000000
9438 022046 005337 022044
9439 022052 001375
9440 022054 105737 022462
9441 022060 001002
9442 022062 000137 022362
9443 022066 005237 022104
9444 022072 001362
9445 022074 104007
9446 022076 104011
9447 022100 000137 022434
9448 022104 000000
9449

; - $XZ -
***** TEST 33 *****
; *THIS TEST RUNS ALL LINES FULL BORE
; *BASED UPON QUALIFIED LINES
; *...THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
; *TRANSMITTER
; - $XZ -

; : * TEST 33
; : *****
TST33: SCOPE
      MOV     #33,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
; -- END 0 MACRO
      MOV     #TST34,NEXT    ;POINT TO THE START OF THE NEXT TEST
; - $MRESET -
      DCLASM
; -- END 0 MACRO
      MOV     LINE,RXTCR     ;SET IMAGE OF TCR BITS
RSTART: MOV     PAR,R1       ;PICK UP PARAMETER
      MOV     #1,R0         ;PICK UP INIT POINTER
INIT:   BIT     R0,LINE      ;SHOULD THIS LINE BE SET UP
      BEQ     1$            ;NO
      MOV     R1,@DZLPR     ;SET UP LINE PARAM REGISTER
1$:     INC     R1
      ASLB    R0            ;GOT 'EM ALL ?
      BCC     INIT         ;NO
      MOV     #TDO,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,@DZRIS  ;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
      MOV     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
      BNE     3$
      JMP     OUT
3$:     INC     66$
      BNE     67$
      ERROR   7
      ERROR   11
      JMP     FINI
66$:    0
; *TRANSMITTER FAILED TO INTERRUPT
; *RECEIVER FAILED TO INTERRUPT
```


9450					;TRANS INTR SVC ROUTINE	
9451	022106	005777	157730		TXSVC: TST @DZCSR	;TRANS INTR ?
9452	022112	100401			BMI .+4	
9453	022114	104003			ERROR 3	;*TRANSMITTER FAILED
9454	022116	117703	157722		MOVB @HDZCSR,R3	;SAVE IT
9455					;NOW TEST FOR LINE # ETC	
9456	022122	042703	177770		BIC #C<7>,R3	;STRIP JUNK
9457	022126	010304			MOV R3,R4	;SAVE
9458	022130	010337	001372		MOV R3,SAVLIN	;ADJUST LOCATION FOR ERROR PRINTOUT
9459	022134	012702	000001		MOV #1,R2	;SET UP POSITION POINTER
9460	022140	105303		3\$:	DECB R3	;IS IT THIS LINE ?
9461	022142	100402			BMI 4\$	;YES
9462	022144	006302			ASL R2	;UP THE LINE #
9463	022146	000774			BR 3\$	;GO 'ROUND AGAIN
9464	022150	030237	001364	4\$:	BIT R2,LINE	;VALID LINE?
9465	022154	001001			BNE .+4	;YES
9466	022156	104010			ERROR 10	;NO,INVALID LINE!!!!
9467	022160	006304			ASL R4	;MAKE POWER OF 2
9468	022162	116477	001422 157676		MOVB TDO(R4),@DZTDR	;LOAD CHARACTER
9469	022170	105264	001422		INCB TDO(R4)	;SET UP NEXT CHARACTER
9470	022174	001002			BNE 5\$	;LAST CHARACTER ?
9471	022176	040277	157654		BIC R2,@DZTCR	;YES ,CLEAR TCR BIT
9472	022202	000002		5\$:	RTI	
9473						
9474						
9475					;REC INTR SVC ROUTINE	
9476	022204	105777	157632		RXSVC: TSTB @DZCSR	;REC DONE ?
9477	022210	100401			BMI .+4	;YES
9478	022212	104004			ERROR 4	;FALSE INTERRUPT
9479	022214	017704	157626		MOV @DZRBUR,R4	;SAVE IT
9480	022220	010403			MOV R4,R3	
9481	022222	000303			SWAB R3	
9482	022224	042703	177770		BIC #C<7>,R3	;STRIP JUNK
9483	022230	010337	001372		MOV R3,SAVLIN	;SAVE LINE NUMBER
9484	022234	032777	020000 157600		BIT #SILOAL,@DZCSR	;SILO ALARM?
9485	022242	001401			BEQ .+4	;NO
9486	022244	104000			ERROR	;SILO ALARM SHOULD NOT =1
9487	022246	005704			TST R4	;DATA VALID SET?
9488	022250	100401			BMI .+4	;YES
9489	022252	104023			ERROR 23	;YOU LOSE ...DATA VALID WAS'NT SET
9490	022254	032704	070000		BIT #OVRUN!FRMERR!PARER,R4	
9491	022260	001401			BEQ .+4	
9492	022262	104000			ERROR	;RECEIVER ERROR FLAG/S WERE SET
9493	022264	012702	000001		MOV #1,R2	;SET UP POSITION POINTER
9494	022270	105303		5\$:	DECB R3	
9495	022272	100402			BMI 6\$	
9496	022274	006302			ASL R2	;RE POSITION POINTER
9497	022276	000774			BR 5\$	;GO 'ROUND AGAIN
9498	022300	030237	001364	6\$:	BIT R2,LINE	;LINE VALID ?
9499	022304	001001			BNE .+4	;YES
9500	022306	104011			ERROR 11	;INVALID LINE #
9501	022310	013703	001372		MOV SAVLIN,R3	;GET THE LINE NUMBER AGAIN
9502	022314	006303			ASL R3	;USE R3 AS A POINTER IN THE DATA TABLE
9503	022316	126304	001442		CMPB TRO(R3),R4	;DOES THE DATA CHARACTER COMPARE ?
9504	022322	001405			BEQ 2\$	;YES
9505	022324	016305	001442		MOV TRO(R3),R5	;SAVE EXPECTED

```
9506 022330 042704 177400      BIC      #C<377>,R4      ;CLEAR JUNK
9507                                ;R2 = LINE # BY BIT POSITION
9508                                ;R4 = ACTUAL DATA
9509                                ;R5 = EXPECTED DATA
9510 022334 104005                ERROR      5              ;*NO, DATA DOES NOT COMPARE
9511 022336 005263 001442      2$: INC      TR0(R3)          ;SET UP FOR NEXT CHARACTER
9512 022342 105763 001442      TSTB      TR0(R3) ;ALL CHARS DONE?
9513 022346 001002                BNE      .+6
9514 022350 040237 022462      BIC      R2,RXTCR          ;ZERO LINE DONE INDICATOR.
9515 022354 012716 022034      MOV      #SNAP,(SP)        ;RESET THE BACKGROUND TIMING LOOP
9516 022360 000002                RTI
9517
9518
9519                                ;FINISH UP ROUTINE
9520 022362 106427 000340      OUT:  MTPS      #PR7              ;STOP ALL INTERRUPTS
9521 022366 104413                DEVICE.CLR          ;CLEAR ALL INTERRUPTS AWAY
9522 022370 005003                CLR      R3
9523 022372 005037 001372      CLR      SAVLIN
9524 022376 012702 000001      MOV      #1,R2
9525 022402 030237 001364      1$: BIT      R2,LINE          ;VALID LINE ?
9526 022406 001405                BEQ      2$              ;NO
9527 022410 022763 000400 001442 CMP      #400,TR0(R3)      ;RECEIVED A BINARY COUNT PATTERN ?
9528 022416 001401                BEQ      .+4              ;YES
9529 022420 104027                ERROR      27            ;THE LINE FAILED TO RECEIVE A FULL
9530                                ;BINARY COUNT PATTERN
9531 022422 005237 001372      2$: INC      SAVLIN          ;SET UP FOR NEXT LINE
9532 022426 005723                TST      (R3)+           ;ADD 2
9533 022430 106302                ASLB      R2              ;SET UP NEXT LINE POINTER
9534 022432 103363                BCC      1$              ;FINISHED ?
9535 022434                FINI:
9536 022434 013777 002074 157430 MOV      DZRI5,@DZRI5      ;RESTORE TRAPCATCHER
9537 022442 005077 157426      CLR      @DZRI5
9538 022446 013777 002100 157422 MOV      DZTI5,@DZTI5
9539 022454 005077 157420      CLR      @DZTI5
9540 022460 104400                ADVANCE
9541 022462 000000      RXTCR:  0              ;GO TO THE NEXT TEST
9542                                ;RX IMAGE OF TCR BITS
9543
9544
```

(2)

```
9545                                ; - $XZ - =====
9546                                ;***** TEST 34 *****
9547                                ;*DZ11 RELATIVE TIMING TEST.
9548                                ;*EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
9549                                ;*AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
9550                                ;*WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
9551                                ;*DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
9552                                ;*THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
9553                                ;* AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
9554                                ;*PARAMETERS ARE:
9555                                ;* EIGHT BITS/PER/CHAR - TWO STOP BITS AT
9556                                ;*      50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
9557                                ;*      2400, 3600, 4800, 7200, 9600 BAUD.
9558                                ;* THEN, 9600 BAUD - TWO STOP BITS AT
9559                                ;*      SEVEN, SIX, FIVE BITS/PER/CHAR.
9560                                ;*AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
9560                                ;*THE NEXT SELECTED LINE IS THE TESTED.
9560                                ; - $XZ - =====
```



```
9561      ;:* TEST 34
(5)      ;:*****
(4) 022464 000004
(2) 022466 012737 000034 001122 TST34: SCOPE
(2)      MOV      #34,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(1) 022474 012737 000002 001226 ; -- END O MACRO =====
(2) 022502 012737 023174 001360 MOV      #2,$TIMES
9562 022510 012737 022634 001362 MOV      #TST35,NEXT      ;POINT TO THE START OF THE NEXT TEST
9563 022516 005037 025242 MOV      #3$,LOCK      ;SET FOR LOOP
9564 022522 005037 001372 CLR      OFFSET      ;RESET THIS VARIABLE
9565 022526 005037 001374 CLR      SAVLIN      ;RESET LINE NUMBER INDICATOR
9566 022532 012737 000001 001216 CLR      XMTLIN      ;USE THIS WORD TO TELL WHAT LINE TRANSMITTED
9567 022540 012737 010070 023172 MOV      #1,$TMP0      ;USE $TMP0 AS A BIT POINTER
9568 022546 033737 001216 001364 1$: BIT      $TMP0,LINE      ;IS THIS LINE ACTIVE?
9569 022554 001027 BNE      3$      ;IF SO, GO GET STARTED
9570 022556 012737 010070 023172 2$: MOV      #RCVON!S50!EIGHT!TWOSTOP,7$ ;BUILD TEMPORARY PARAMETERS
9571 022564 012700 001422 MOV      #TDO,RO      ;POINT TO THE DATA AREA
9572 022570 005020 CLR      (RO)+      ;CLEAR A DATA WORD
9573 022572 022700 001462 CMP      #STOP,RO      ;FINISHED ?
9574 022576 001374 BNE      .-6      ;NO
9575 022600 005237 001374 INC      XMTLIN      ;POINT TO THE NEXT LINE TO TRANSMIT
9576 022604 042737 000007 023172 BIC      #7,7$      ;MAKE SURE TEMPORARY PARAMETERS POINT TO 0
9577 022612 053737 001374 023172 BIS      XMTLIN,7$      ;ADD DESIRED LINE NUMBER
9578 022620 005037 025242 CLR      OFFSET
9579 022624 106337 001216 ASLB      $TMP0      ;POINT TO THE NEXT LINE
9580 022630 103346 BCC      1$      ;PROCESS THE NEXT LINE
9581 022632 104400 ADVANCE      ;TEST TO SEE IF THIS TEST GETS REPEATED
9582 022634 3$:
(1)
(1) 022634 104417 ; - $MRESET-----
(1) DCLASH ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
; -- END O MACRO =====
9583 022636 042737 010000 023172 BIC      #RCVON,7$      ;ZERO PARAMTERS FOR TX LINE
9584 022644 013777 023172 157200 MOV      7$,@DZLPR      ;LOAD PARAMTERS FOR TX
9585 022652 005737 001370 TST      MODE      ;STAGGERED?
9586 022656 100011 BPL      100$      ;BR IF NO
9587 022660 000241 CLC      ;SET UP LINE
9588 022662 006037 023172 ROR      7$
9589 022666 103002 BCC      98$      ;BR IF LINE WAS EVEN
9590 022670 000241 CLC      ;PREPARE TO MKE LINE EVEN
9591 022672 000401 BR      99$      ;CONTINUE
9592 022674 000261 98$: SEC      ;PREPARE TO MAKE LINE ODD
9593 022676 006137 023172 99$: ROL      7$      ;SET ALTERED LINE
9594 022702 052737 010000 023172 100$: BIS      #RCVON,7$      ;SET RX ON
9595 022710 013777 023172 157134 MOV      7$,@DZLPR      ;LOAD RX PARAMETERS
9596 022716 013737 023172 001372 MOV      7$,SAVLIN      ;ADJUST LOCATION FOR ERROR PRINTOUT
9597 022724 042737 177770 001372 BIC      #+C<7>,SAVLIN      ;STRIP JUNK
9598 022732 042737 000007 023172 BIC      #7,7$      ;CLEAR OLD LINE #
9599 022740 053737 001374 023172 BIS      XMTLIN,7$      ;SET LINE UP AGAIN
9600 022746 013737 023172 001400 MOV      7$,REGIST      ;SAVE PARAMETERS FOR PRINTOUT
9601 022754 012700 001422 MOV      #TDO,RO      ;POINT TO THE DATA AREA
9602 022760 005020 CLR      (RO)+      ;CLEAR A DATA WORD
9603 022762 022700 001462 CMP      #STOP,RO      ;FINISHED ?
9604 022766 001374 BNE      .-6      ;NO
9605 022770 005002 CLR      R2      ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
9606 022772 005003 CLR      R3      ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
9607 022774 005037 001220 CLR      $TMP1      ;INITIALIZE THE TIMER
```

```

9608 023000 005037 001224 CLR $TMP3 ;INITIALIZE THESE BITS ALSO
9609 023004 012737 000020 001376 MOV #20,XMTCNT ;SET HOW MANY CHARACTERS TO TRANSMIT
9610 023012 012777 024664 :57055 MOV #XMTSRV,@DZTIV
9611 023020 012777 025030 157044 MOV #RXISR1,@DZRIV
9612 023026 013777 030334 157040 MOV DZPRT,@DZRIS
9613 023034 013777 030334 157036 MOV DZPRT,@DZTIS
9614 023042 113777 001216 157006 MOVB $TMP0,@DZTCR ;START THE VALID LINE
9615 023050 052777 040140 156764 BIS #TIE!RIE!MSENAB,@DZCSR
9616 023056 106427 000000 MTPS #0 ;LOWER THE PRIORITY TO ALLOW INTERRUPTS
9617 023062 032777 000100 156752 4$: BIT #RIE,@DZCSR ;IS ROUTINE DONE?
9618 023070 001407 BEQ 5$ ;WHEN ALL IS DONE RX IE IS CLEARED IN ISR.
9619 023072 005237 001220 INC $TMP1 ;COUNT TIME
9620 023076 001371 BNE 4$ ;CONTINUE TEST
9621 023100 105237 001224 INCB $TMP3 ;DOUBLE COUNT
9622 023104 001366 BNE 4$ ;CONTINUE TEST
9623 023106 104011 ERROR 11 ;INTERRUPTS NOT FINISHED
9624 023110 004737 007642 5$: JSR PC,SERV.G ;<*G>?
9625 023114 104401 SCOP1 ;LOOP?
9626 023116 062737 000002 025242 ADD #2,OFFSET
9627 023124 013700 023172 MOV 7$,R0
9628 023130 042700 170377 BIC #C<17*400>,R0
9629 023134 022700 007000 CMP #<16*400>,R0
9630 023140 001010 BNE 6$
9631 023142 032737 000030 023172 BIT #BIT4+BIT3,7$
9632 023150 001602 BEQ 2$
9633 023152 162737 000010 023172 SUB #BIT3,7$
9634 023160 000625 BR 3$
9635 023162 062737 000400 023172 6$: ADD #400,7$
9636 023170 000621 BR 3$
9637 023172 000000 7$: 0
9638 ; -$PARTST-----
(2) ; -$XZ-----
(3) ;***** TEST 35 *****
(1) ;* THIS TEST VERIFIES THAT EVEN PARITY WORKS
(1) ;* FOR ALL ODD LINES SELECTED AND THAT ODD PARITY WORKS FOR ALL
(1) ;* EVEN LINES SELECTED.
(1) ;*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
(1) ;*THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
(1) ;*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
(1) ;*YOU ARE IN "STAGGERED" MODE.
(1) ;*40(8) CHARS ARE USED FOR THIS TEST.
(1) ;*ALL SELECTED LINES WILL BE ENABLED
(1) ;*AT THE SAME TIME!
(2) ; -$XZ-----
(3) ;:* TEST 35
(6) ;*****
(5) 023174 000004 TST35: SCOPE
(3) 023176 012737 000035 001122 MOV #35,$TSTNM ;LOAD THE NUMBER OF THIS TEST
(3) ; -- END 0 MACRO =====
(3) 023204 012737 023446 001360 MOV #TST36,NEXT ;POINT TO THE START OF THE NEXT TEST
(1) 023212 005737 001370 TST MODE ;IS THIS STAGGERED MODE?
(1) 023216 100112 BPL 6$ ;IF NOT, DON'T DO THIS TEST
(2) ; -$MRESET-----
(2) 023220 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2) ; -- END 0 MACRO =====
(1) 023222 013701 001366 MOV PAR,R1 ;USE R1 TO BUILD PARAMETERS TO BE LOADED

```



```

(1) 023226 042701 000200      BIC      #ODDPAR,R1      ;MAKE SURE ODD PARITY ISN'T SET
(1) 023232 052701 000100      BIS      #PARITY,R1      ;MAKE SURE PARITY IS TURNED ON
(1) 023236 012702 000001      MOV      #1,R2      ;USE R2 AS A LINE POINTER
(1) 023242 030237 001364      1$:    BIT      R2,LINE      ;IS THIS A VALID LINE?
(1) 023246 001411              BEQ      3$      ;IF NOT, SKIP TO THE NEXT LINE
(1) 023250 032701 000001      BIT      #BIT0,R1      ;IS THIS LINE AN ODD LINE?
(1) 023254 001002              BNE      2$      ;IF IT'S ODD, USE EVEN PARITY
(1) 023256 052701 000200      BIS      #ODDPAR,R1      ;IF IT'S EVEN, USE ODD PARITY
(1) 023262 010177 156564      2$:    MOV      R1,#DZLPR      ;LOAD THE LINE PARAMETER REGISTER
(1) 023266 042701 000200      BIC      #ODDPAR,R1      ;SET UP THE NEXT PARITY TO EVEN
(1) 023272 005201      3$:    INC      R1      ;POINT TO THE NEXT LINE
(1) 023274 106302              ASLB     R2      ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
(1) 023276 103361              BCC      1$      ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
(1) 023300 005037 001372      CLR      SAVLIN      ;CLEAR THE LINE NUMBER INDICATOR
(1) 023304 005002              CLR      R2      ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
(1) 023306 005037 024660      CLR      COUNT0      ;;;2-10-84 ECB REV I
(1) 023312 005037 024662      CLR      COUNT1      ;;;2-10-84 ECB REV I
(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 012700 001422      MOV      #TD0,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)                                ; - $INTSET-----
(2) 023344 012777 024664 156524 MOV      #XMTSRV,#DZTIV      ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 023352 012777 024506 156512 MOV      #PARESE,#DZRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 023360 013777 030334 156506 MOV      DZPRT,#DZRRIS      ;SET THE INTERRUPT VECTOR STATUS
(2) 023366 013777 030334 156504 MOV      DZPRT,#DZTIS      ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 023374 052777 040140 156440 BIS      #RIE!TIE!MSENAB,#DZCSR ;ENABLE THE DEVICE
(2)                                ; -- END 0 MACRO -----
(1) 023402 113777 001364 156446 MOV      LINE,#DZTCR      ;ENABLE ALL SELECTED LINES
(1) 023410 106427 000000              MTPS     #0      ;ALLOW INTERRUPTS
(1) 023414 032777 000100 156420 5$:    BIT      #RIE,#DZCSR      ;WHEN RX DONE; RIE WILL =0
(1) 023422 001410              BEQ      6$      ;BR IF ALL DONE
(1) 023424 104414              DELAY     ;;; 2-10-84 ECB REV I
(1) 023426 005237 024660              INC      COUNT0
(1) 023432 102770              BVS      5$
(1) 023434 105237 024662              INCB     COUNT1
(1) 023440 100365              BPL      5$
(1) 023442 104011              ERROR     11      ;*RX FAILED TO FINISH (INTERRUPT)
(1) 023444 104400      6$:    ADVANCE      ;ADVANCE LOOP
(1)                                ; -- END 0 MACRO -----
9639                                ; - $PARTST-----
(2)                                ; - $XZ-----
(3)                                ;***** 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!
(2)                                ; - $XZ-----

```

```
(3)          ::* TEST 36
(6)          ;*****
(5) 023446 000004 TST36: SCOPE
(3) 023450 012737 000036 001122      MOV    #36,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
(3)          ; -- END 0 MACRO =====
(3) 023456 012737 023720 001360      MOV    #TST37,NEXT    ;POINT TO THE START OF THE NEXT TEST
(1) 023464 005737 001370      TST     MODE              ;IS THIS STAGGERED MODE?
(1) 023470 100112      BPL     6$                      ;IF NOT, DON'T DO THIS TEST
(2)          ; -$MRESET-----
(2) 023472 104417      DCLASM      ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
(2)          ; -- END 0 MACRO =====
(1) 023474 013701 001366      MOV    PAR,R1              ;USE R1 TO BUILD PARAMETERS TO BE LOADED
(1) 023500 042701 000200      BIC    #ODDPAR,R1          ;MAKE SURE ODD PARITY ISN'T SET
(1) 023504 052701 000100      BIS    #PARITY,R1          ;MAKE SURE PARITY IS TURNED ON
(1) 023510 012702 000001      MOV    #1,R2              ;USE R2 AS A LINE POINTER
(1) 023514 030237 001364      1$:  BIT    R2,LINE          ;IS THIS A VALID LINE?
(1) 023520 001411      BEQ    3$                      ;IF NOT, SKIP TO THE NEXT LINE
(1) 023522 032701 000001      BIT    #BIT0,R1           ;IS THIS LINE AN ODD LINE?
(1) 023526 001402      BEQ    2$                      ;IF IT'S EVEN, USE EVEN PARITY
(1) 023530 052701 000200      BIS    #ODDPAR,R1          ;IF IT'S ODD, USE ODD PARITY
(1) 023534 010177 156312      2$:  MOV    R1,@DZLPR        ;LOAD THE LINE PARAMETER REGISTER
(1) 023540 042701 000200      BIC    #ODDPAR,R1          ;SET UP THE NEXT PARITY TO EVEN
(1) 023544 005201      3$:  INC     R1                  ;POINT TO THE NEXT LINE
(1) 023546 106302      ASLB    R2                      ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
(1) 023550 103361      BCC    1$                      ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
(1) 023552 005037 001372      CLR    SAVLIN            ;CLEAR THE LINE NUMBER INDICATOR
(1) 023556 005002      CLR    R2                      ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
(1) 023560 005037 024660      CLR    COUNT0             ; ::2-10-84 ECB REV I
(1) 023564 005037 024662      CLR    COUNT1             ; ::2-10-84 ECB REV I
(1) 023570 005003      CLR    R3                      ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
(1) 023572 012737 000040 001376      MOV    #40,XMTCNT      ;TRANSMIT A BINARY COUNT PATTERN(00-40)
(1) 023600 012700 001422      MOV    #TD0,R0            ;POINT TO THE DATA AREA
(1) 023604 005020      CLR    (R0)+                   ;CLEAR A DATA WORD
(1) 023606 022700 001462      CMP    #STOP,R0           ;FINISHED ?
(1) 023612 001374      BNE    .-6                      ;NO
(1) 023614 005000      CLR    R0                      ;CLEAR OFFSET
(2)          ; -$INTSET-----
(2) 023616 012777 024664 156252      MOV    #XMTSRV,@DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
(2) 023624 012777 024506 156240      MOV    #PARESE,@DZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
(2) 023632 013777 030334 156234      MOV    DZPRT,@DZRI$ ;SET THE INTERRUPT VECTOR STATUS
(2) 023640 013777 030334 156232      MOV    DZPRT,@DZTIS ;SET TRANSMITTER INTERRUPT PRIORITY
(2) 023646 052777 040140 156166      BIS    #RIE!TIE!MSENAB,@DZCSR ;ENABLE THE DEVICE
(2)          ; -- END 0 MACRO =====
(1) 023654 113777 001364 156174      MOV    LINE,@DZTCR    ;ENABLE ALL SELECTED LINES
(1) 023662 106427 000000      MTPS    #0                ;ALLOW INTERRUPTS
(1) 023666 032777 000100 156146      5$:  BIT    #RIE,@DZCSR ;WHEN RX DONE; RIE WILL =0
(1) 023674 001410      BEQ    6$                      ;BR IF ALL DONE
(1) 023676 104414      DELAY    ; :: 2-10-84 ECB REV I
(1) 023700 005237 024660      INC     COUNT0
(1) 023704 102770      BVS     5$
(1) 023706 105237 024662      INCB    COUNT1
(1) 023712 100365      BPL     5$
(1) 023714 104011      ERROR    11                      ;*RX FAILED TO FINISH (INTERRUPT)
(1) 023716 104400      6$:  ADVANCE    ;ADVANCE LOOP
(1)          ; -- END 0 MACRO =====
```



```
9641                                     ; -$XZ-----
9642                                     ;***** TEST 37 *****
9643                                     ;*
9644                                     ;* ECB 21-FEB-84
9645                                     ;* BAUD RATE TIMING TEST. THIS TEST ADDED IN ORDER TO TEST CRYSTAL
9646                                     ;* SPEEDS. IT LOOKS FOR EITHER A KW11L OR KW11P. IF EITHER IS AVAIL.
9647                                     ;* THEN THE TEST IS RAN. IF NEITHER AVAILABLE, THEN TEST IS NOT RAN.
9648                                     ;*
9649                                     ;* KR 18-JUN-84
9650                                     ;* ADDED NUMBERS IN TTABLE AND NOW TEST ALL LINES AT ALL BAUD RATES
9651                                     ;* INSTEAD OF JUST LINE 0 AT ALL BAUD RATES.
9652                                     ;*
9653                                     ; -$XZ-----
9654                                     ;:* TEST 37
9655                                     ;*****
9656 (4) 023720 000004 TST37: SCOPE
9657 (2) 023722 012737 000037 001122 MOV #37,$TSTNM ;LOAD THE NUMBER OF THIS TEST
9658 (2) 023730 012737 004712 001360 ; -- END 0 MACRO =====
9659 (1) 023730 012737 004712 001360 MOV #$EOP,NEXT ;POINT TO THE END-OF-PASS HANDLER
9660 023736 032777 000040 155214 BIT #BITS, @SWR ;RUN THIS TEST?
9661 023744 001001 BNE 1$ ; BRANCH IF YES
9662 023746 104400 ADVANCE ; OTHERWISE, BYE
9663 023750 1$:
9664 023750 106427 000000 MTPS #0 ;DROP PRIORITY TO 0
9665 023754 013700 000004 MOV @#4,R0 ;SAVE CONTENTS OF LOCATION 4.
9666 023760 012737 024024 000004 MOV #10$,@#4 ;SET TO TRAP IF NOT KW11L
9667 023766 012701 177546 MOV #177546,R1 ;GET KW11L ADDRESS.
9668 023772 012737 024100 000100 MOV #CLKSRV,@#100 ;SET FOR INTERRUPT.
9669 024000 012737 000340 000102 MOV #340,@#102 ;PRIORITY 7 ON INTERRUPT.
9670 024006 005037 024110 CLR BCOUNT ;OUR OWN CLOCK TICK COUNTER
9671 024012 005037 024106 CLR CCOUNT ;CHAR SENT COUNTER
9672 024016 012711 000100 MOV #100,(R1) ;THIS WILL CAUSE A BOMB TO OCCUR IF
9673 024022 000473 BR TCCONT ;THE KW11L DOES NOT EXIST
9674 024024 022626 10$: CMP (SP)+,(SP)+ ;BUT IF ITS THERE, THEN WE'RE STARTED.
9675 024026 012737 024070 000004 MOV #20$,@#4 ;START TEST
9676 024034 012737 024100 000104 MOV #CLKSRV,@#104 ;READJUST STACK
9677 024042 012737 000340 000106 MOV #340,@#106 ;SET TO TRAP IF NO KW11P
9678 024050 012701 172540 MOV #172540,R1 ;SET FOR INTERRUPT.
9679 024054 012761 177777 000002 MOV #-1,2(R1) ;PROITY 7 ON INTERRUPT.
9680 024062 012711 000133 MOV #133,(R1) ;KW11P ADDRESS.
9681 024066 000451 BR TCCONT ;SET THE # OF COUNTS FOR INTERRUPT (1)
9682 024070 022626 20$: CMP (SP)+,(SP)+ ;NOTE, IF KW11P DOES NOT EXIST, THEN
9683 024072 010037 000004 MOV R0,@#4 ;WE WILL BOMB FROM THIS PLACE.
9684 024076 104400 ADVANCE ;SET INTERRUPT ENABLE,UP COUNT,REPEATED
9685 024100 005237 024106 CLKSRV: INC CCOUNT ;INTERRUPT,LF CLOCK, AND RUN.
9686 024104 000002 RTI ;START TEST.
9687
9688
9689
9690
```

```

9691 024106 000000
9692 024110 000000
9693 024112 000000
9694 024114 000000
9695
9696
9697
9698
9699
9700
9701
9702
9703
9704
9705
9706
9707
9708
9709
9710
9711
9712
9713
9714
9715
9716
9717
9718
9719
9720
9721
9722
9723
9724
9725
9726
9727 024116 000005 000007
9728 024122 000007 000014
9729 024126 000013 000021
9730 024132 000014 000024
9731 024136 000016 000025
9732
9733 024142 000035 000050
9734 024146 000072 000114
9735 024152 000164 000224
9736 024156 000255 000334
9737 024162 000300 000364
9738
9739 024166 000347 000444
9740 024172 000533 000664
9741 024176 000716 001104
9742 024202 001265 001544
9743 024206 001635 002204
9744
9745
9746

CCOUNT: .WORD 0 ;INCREMENTED BY CLKSRV. #CLOCK TICKS.
BCOUNT: .WORD 0 ;USED BY TCCONT AS TIMER.
DCOUNT: .WORD 0 ;USED BY TCCONT AS TIMER.
ECOUNT: .WORD 0 ;USED BY ERROR REPORTING.

;+
; THESE NUMBER WHERE WORKING GREAT ON A PDP-11/24 RUNNING AT 60 HERTZ
;
;TTABLE: .WORD 5., 7. ;COUNT OF CHARS OUT FOR 50 BAUD
; .WORD 7., 9. ;COUNT OF CHARS OUT FOR 75 BAUD
; .WORD 11., 13. ;COUNT OF CHARS OUT FOR 110 BAUD
; .WORD 13., 15. ;COUNT OF CHARS OUT FOR 134.5 BAUD
; .WORD 15., 17. ;COUNT OF CHARS OUT FOR 150 BAUD
;
; .WORD 29., 32. ;COUNT OF CHARS OUT FOR 300 BAUD
; .WORD 59., 62. ;COUNT OF CHARS OUT FOR 600 BAUD
; .WORD 119., 122. ;COUNT OF CHARS OUT FOR 1200 BAUD
; .WORD 179., 182. ;COUNT OF CHARS OUT FOR 1800 BAUD
; .WORD 199., 202. ;COUNT OF CHARS OUT FOR 2000 BAUD
;
; .WORD 239., 242. ;COUNT OF CHARS OUT FOR 2400 BAUD
; .WORD 359., 362. ;COUNT OF CHARS OUT FOR 3600 BAUD
; .WORD 479., 482. ;COUNT OF CHARS OUT FOR 4800 BAUD
; .WORD 719., 722. ;COUNT OF CHARS OUT FOR 7200 BAUD
; .WORD 959., 962. ;COUNT OF CHARS OUT FOR 9600 BAUD
;-

;+
; THESE NUMBER WHERE MODIFIED SO THEY WOULD WORK ON A
; 50 HERTZ MACHINE (TESTED ON PDP-11/70) AS WELL AS 60 HERTZ.
; AT 9600 BAUD THERE IS ABOUT 17% ERROR.
; THE LOWER THE BAUD RATE THE LESS THE ERROR.
;
; THIS TABLE WAS REDONE TO INCLUDE THE 2% ALLOWABLE ERROR RATE, PLUS THE
; POSSIBILITY OF AN INCOMPLETE TICK AT THE START OF THE TIMING.
;-

TTABLE: .WORD 5., 7. ;COUNT OF CHARS OUT FOR 50 BAUD
        .WORD 7., 12. ;COUNT OF CHARS OUT FOR 75 BAUD
        .WORD 11., 17. ;COUNT OF CHARS OUT FOR 110 BAUD
        .WORD 12., 20. ;COUNT OF CHARS OUT FOR 134.5 BAUD
        .WORD 14., 21. ;COUNT OF CHARS OUT FOR 150 BAUD
        .WORD 29., 40. ;COUNT OF CHARS OUT FOR 300 BAUD
        .WORD 58., 76. ;COUNT OF CHARS OUT FOR 600 BAUD
        .WORD 116., 148. ;COUNT OF CHARS OUT FOR 1200 BAUD
        .WORD 173., 220. ;COUNT OF CHARS OUT FOR 1800 BAUD
        .WORD 192., 244. ;COUNT OF CHARS OUT FOR 2000 BAUD
        .WORD 231., 292. ;COUNT OF CHARS OUT FOR 2400 BAUD
        .WORD 347., 436. ;COUNT OF CHARS OUT FOR 3600 BAUD
        .WORD 462., 580. ;COUNT OF CHARS OUT FOR 4800 BAUD
        .WORD 693., 868. ;COUNT OF CHARS OUT FOR 7200 BAUD
        .WORD 925., 1156. ;COUNT OF CHARS OUT FOR 9600 BAUD
ENDTTB =. ;END OF TTABLE

```



```

9747      ; CHECK OUT THE CLOCK
9748      ;
9749
9750 024212 010037 000004      TCCONT: MOV      R0,#4      ;RESTORE LOCATION 4.
9751 024216 005037 024106      CLR      CCOUNT      ;START WITH LINE CLOCK AT ZERO COUNT
9752 024222 012737 177750 024110      MOV      #-30,BCOUNT      ;30-18MS LOOPS TO WAIT
9753
9754 024230 005737 024106      10$:  TST      CCOUNT      ;WAIT FOR COUNT TO GO TO 1 TO AVOID
9755 024234 001010                BNE      20$      ;USING THAT DANGEROUS FIRST COUNT!
9756 024236 005237 024112      INC      DCOUNT      ;BUT AS A SAFE GUARD,
9757 024242 001372                BNE      10$      ;DON'T HANG HERE
9758 024244 005237 024110      INC      BCOUNT
9759 024250 001367                BNE      10$
9760 024252 005011      CLR      (R1)      ;CLEAR CLOCK
9761 024254 104400      ADVANCE      ;CLOCK NOT WORKING RIGHT- ABORT TEST
9762                                ;NOT THE FAULT OF THE DZ!
9763
9764      ;
9765      ; START XMITTING CHARACTERS TO ALL 8 LINES AT ALL BAUD RATES.
9766      ; IF THE NUMBER OF CHARACTERS TRANSMITTED IS BETWEEN THE
9767      ; TWO NUMBERS IN TTABLE THEN THE LINE IS OK, ELSE ERROR
9768      ;
9769
9770 024256 012777 000050 155556 20$:  MOV      #50,0DZCSR      ;SET MSE AND MAINT.
9771 024264 012700 000030      MOV      #30,R0      ;SET BAUD RATE 50, 8 BIT CHAR LEN
9772 024270 012703 000001      MOV      #1,R3      ;HOLDS CURRENT LINE, START AT LINE 0
9773
9774 024274 110377 155556      30$:  MOVB     R3,0DZTCR      ;ENABLE THE NEXT LINE TO XMIT
9775 024300 012702 024116      MOV      #TTABLE,R2      ;GET TABLE OF EXPECTED RESULTS
9776
9777 024304 010077 155542      40$:  MOV      R0,0DZLPR      ;UPDATE THE BAUD RATE AND NEXT LINE
9778 024310 005037 024110      CLR      BCOUNT      ;REUSE TIMER AS XMIT CHAR COUNTER
9779 024314 005037 024106      CLR      CCOUNT      ;CLEAR CLOCK INTERRUPT COUNTER
9780
9781 024320 022737 000074 024106 50$:  CMP      #60.,CCOUNT      ;LOOP FOR 60 CLOCK TICKS
9782 024326 001413                BEQ      60$      ;EXIT WHEN THAT OCCURS.
9783 024330 005777 155506      TST      0DZCSR      ;READY TO XMIT A CHAR?
9784 024334 100371                BPL      50$      ;NO, THEN WAIT
9785 024336 112777 000101 155522      MOVB     #101,0DZTDR      ;XMIT A CHAR
9786 024344 005237 024110      INC      BCOUNT      ;UPDATE XMIT COUNT, SHOULD NOT OVERFLOW.
9787 024350 102363                BVC      50$      ;IF NO OVERFLOW, CONTINUE
9788 024352 005011      CLR      (R1)      ;IF OVERFLOW, CLOCK COULD BE BAD, CLEAR IT
9789 024354 104400      ADVANCE      ;ABORT THIS TEST IF CLOCK 10-2.'
9790
9791 024356 005037 024106      60$:  CLR      CCOUNT
9792 024362 022737 000074 024106      CMP      #60.,CCOUNT      ;LOOP FOR 60 CLOCK TICKS
9793 024370 001403                BEQ      65$      ;EXIT WHEN THAT OCCURS.
9794 024372 005777 155444      TST      0DZCSR      ; WAIT FOR LAST CHARACTER TO GO
9795 024376 100367                BPL      60$      ; SPIN IF NOT DONE
9796 024400 023712 024110      65$:  CMP      BCOUNT,(R2)      ;IF COUNT < LOW
9797 024404 103422                BLO      70$      ; THEN ERROR
9798 024406 023762 024110 000002      CMP      BCOUNT,2(R2)      ;IF COUNT > HI
9799 024414 101016                BHI      70$      ; THEN ERROR
9800 024416 062700 000400      ADD      #400,R0      ;NEXT BAUD RATE
9801 024422 062702 000004      ADD      #4,R2      ;GET NEXT PAIR OF NUMBERS
9802 024426 022702 024212      CMP      #ENDTTB,R2      ;HIT END OF TTABLE?

```

```

9803 024432 001324          BNE      40$          ;NO, CONTINUE TESTING THIS LINE
9804 024434 042700 007400   BIC      #7400,R0       ;BRING THE BAUD RATE BACK TO 50
9805 024440 005200          INC      R0             ;AND GET NEXT LINE NUMBER
9806 024442 106303          ASLB     R3             ;MOVE ONTO THE NEXT LINE
9807 024444 103313          BCC      30$          ;IF NOT DONE WITH ALL LINES THEN CONTINUE
9808 024446 005011          CLR      (R1)          ;ELSE STOP CLOCK
9809 024450 104400          ADVANCE   ;AND DO THE NEXT TEST
9810
9811
9812          ;
9813          ; ERROR - COUNTER NOT IN THE RANGE OF THE NUMBERS IN TTABLE
9814          ;
9815 024452 005011          70$:   CLR      (R1)          ;STOP CLOCK.
9816 024454 011237 024114   MOV      (R2),ECOUNT      ;MOVE LOW COUNT FOR ERROR MESSAGE
9817 024460 016237 000002 024112 MOV      2(R2),DCOUNT    ;MOVE HIGH COUNT FOR ERROR MESSAGE
9818 024466 042700 177770   BIC      #+C<7>,R0       ;STRIP EVERYTHING BUT LINE NUMBER
9819 024472 010037 001372   MOV      R0,SAVLIN        ;MOVE LINE NUMBER FOR ERROR MESSAGE
9820 024476 104030          ERROR    30             ;COUNT TOO HIGH OR LOW
9821 024500 004737 007642   JSR      PC,SERV.G        ;<+G?>
9822 024504 104400          ADVANCE   ;DO NEXT TEST
9823
9824          ;
9825          ;RECEIVER SERVICE ROUTINE(PARITY TEST ONLY)
9826          ;
9827
9828 024506 017704 155334   PARESE: MOV      @DZRBUFF,R4      ;GET THE CHARACTER
9829 024512 010401          MOV      R4,R1             ;COPY THE RECEIVED INFORMATION
9830 024514 000301          SWAB     R1                 ;GET THE LINE NUMBER IN THE LOWER BYTE
9831 024516 042701 177770   BIC      #+C<7>,R1       ;ISOLATE THE LINE NUMBER
9832 024522 010137 001372   MOV      R1,SAVLIN        ;FILL LOC. FOR ERROR PRINTOUT
9833 024526 005704          TST      R4                 ;WAS DATA VALID?
9834 024530 100401          BMI      10$             ;BRANCH IF YES
9835 024532 104023          ERROR    23             ;ERROR - DATA VALID NOT SET!
9836 024534 006301          10$:   ASL      R1             ;ALIGN IT ON A WORD BOUNDARY
9837 024536 032704 010000   BIT      #PARER,R4        ;PARITY ERROR SHOULD BE SET. IS IT?
9838 024542 001013          BNE      11$             ;IF SO, GO CHECK CHARACTER
9839 024544 013737 002046 001400 MOV      DZRBUFF,REGIST ;SET UP FOR THE ERROR MESSAGE
9840 024552 010405          MOV      R4,R5
9841 024554 042705 000377   BIC      #377,R5
9842 024560 156105 001442   BISB     TRO(R1),R5      ;GET THE CORRECT CHARACTER
9843 024564 052705 110000   BIS      #DVALID!PARER,R5 ;BUILD WHAT WAS EXPECTED
9844 024570 104006          ERROR    6               ;*ERROR- DID NOT GET CORRECT INFORMATION
9845 024572 126104 001442   11$:   CMPB     TRO(R1),R4    ;CHECK THE CHARACTER. IS IT CORRECT?
9846 024576 001407          BEQ      12$             ;IF SO, GO SET UP NEXT CHARACTER
9847 024600 116105 001442   MOV      TRO(R1),R5      ;LOAD THE CHARACTER FOR ERROR REPORTING
9848 024604 042705 177400   BIC      #+C<377>,R5      ;CLEAR SIGN EXTEND
9849 024610 042704 177400   BIC      #+C<377>,R4      ;REMOVE THE JUNK FROM R4, THE ACTUAL CHARACTER
9850 024614 104005          ERROR    5               ;DATA ERROR
9851 024616 005261 001442   12$:   INC      TRO(R1)      ;SET UP THE NEXT CHARACTER
9852 024622 005203          INC      R3               ;ADD TO THE TOTAL RECEIVED COUNT
9853 024624 005037 024660   CLR      COUNT0      ;RESET COUNTERS TO NEXT
9854 024630 005037 024662   CLR      COUNT1      ;RECIEVER INTERRUPT
9855 024634 032777 040000 155200 BIT      #TIE,@DZCSR    ;ARE TRANSMISSIONS DONE?
9856 024642 001005          BNE      13$             ;IF NO, GO RECEIVE SOME MORE
9857 024644 020203          CMP      R2,R3            ;ARE ALL CHARACTERS RECEIVED?
9858 024646 001003          BNE      13$             ;IF NO, GO RECEIVE SOME MORE

```



```
9859 024650 042777 000100 155164      BIC      @RIE,@DZCSR      ;DISABLE RECEIVER INTERRUPTS
9860 024656 000002      13$: RTI      ;GO BACK TO RECEIVER WAIT LOOP
9861 024660 000000      COUNT0: 0
9862 024662 000000      COUNT1: 0
9863
9864
9865      ;TRANSMITTER INTERRUPT SERVICE
9866      ;-----
9867
9868 024664 117701 155154      XMTSRV: MOVB      @HDZCSR,R1      ;GET THE LINE NUMBER. IS THE TRANSMITTER
9869 024670 100411      BMI      1$      ;REALLY READY? IF SO, GO LOAD THE CHARACTER
9870 024672 013700 001372      MOV      SAVLIN,R0      ;ADJUST LOCATION SAVLIN
9871 024676 042701 177770      BIC      @C<7>,R1      ;ISOLATE THE LINE NUMBER
9872 024702 010137 001372      MOV      R1,SAVLIN      ;FOR ERROR PRINTOUT
9873 024706 104003      ERROR      3      ;*TRANSMITTER NOT READY- FALSE INTERRUPT
9874 024710 010037 001372      MOV      R0,SAVLIN      ;RESET SAVLIN TO PREVIOUS VALUE
9875 024714 042701 177770      1$: BIC      @C<7>,R1      ;ISOLATE THE LINE NUMBER
9876 024720 006301      ASL      R1      ;MAKE SURE IT REFERENCES A WORD BOUNDARY
9877 024722 116177 001422 155136      MOVB      TDO(R1),@DZTDR      ;LOAD THE CURRENT CHARACTER FOR THIS LINE
9878 024730 005261 001422      INC      TDO(R1)      ;SET UP NEXT CHARACTER FOR THIS LINE
9879      ;*****
9880      ;(REV. F0: DELETED INC
9881      ;*****
9882 024734 023761 001376 001422      CMP      XMTCNT,TDO(R1)      ;HAVE WE DONE ALL PATTERNS ON THIS LINE?
9883 024742 001015      BNE      4$      ;IF NOT, KEEP ON TRANSMITTING
9884 024744 012700 000001      MOV      @1,R0      ;SET UP A DESELECTION POINTER
9885 024750 006201      ASR      R1      ;GET THE LINE NUMBER AGAIN
9886 024752 005301      2$: DEC      R1      ;REDUCE THE COUNT. WAS THIS THE LINE?
9887 024754 100402      BMI      3$      ;IF SO, GO DISABLE THE ENABLE BIT FOR IT
9888 024756 006300      ASL      R0      ;MOVE THE POINTER TO THE NEXT LINE
9889 024760 000774      BR      2$      ;GO CHECK THE NEXT LINE
9890 024762 140077 155070      3$: BICB      R0,@DZTCR      ;DISABLE THE LINE POINTED TO BY R0
9891 024766 001003      BNE      4$      ;IF MORE LINES ARE ACTIVE , GO CONTINUE TRANSMIT
9892 024770 042777 040000 155044      BIC      @TIE,@DZCSR      ;IF NOT, DISABLE TRANSMITTER INTERRUPTS
9893      ;*****
9894 024776 005202      4$: INC      R2      ;UP THE NUMBER OF TRANSMISSIONS      (REV. F0)
9895      ;*****
9896 025000 000002      RTI      ;RETURN TO THE TIMING LOOP
9897
9898      ; RELATIVE TIME BUILDING ROUTINE
9899      ; -----
9900
9901 025002 012737 000004 001222      BUILD: MOV      @4,$TMP2      ;ROTATE 4 BITS BACK INTO $TMP1
9902 025010 006037 001224      1$: ROR      $TMP3      ;GET THE BITS FROM $TMP3, THE HIGH BYTE
9903 025014 006037 001220      ROR      $TMP1      ;OF THE RELATIVE TIME COUNTER. PUT THEM BACK
9904 025020 005337 001222      DEC      $TMP2      ;INTO $TMP1 USING THE CARRY BIT WITH
9905      ;ROTATE INSTRUCTIONS
9906 025024 001371      BNE      1$      ;REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
9907 025026 000207      RTS      PC      ;RETURN TO CALLING TEST
9908
```

```
9910                                     ;RECEIVER SERVICE ROUTINE
9911
9912 025030 105777 155006      RXISR1: TSTB      @DZCSR      ;IS THE RECEIVER REALLY READY?
9913 025034 100401              BMI      1$             ;IF SO, GO SERVICE IT
9914 025036 104004              ERROR     4             ;*ERROR- RECEIVER DONE FLAG ISN'T SET
9915 025040 017704 155002      1$:      MOV      @DZRBUF,R4 ;SAVE THE RECEIVER INFORMATION
9916 025044 100401              BMI      2$             ;IF IT WAS VALID, GO PROCESS IT
9917 025046 104023              ERROR     23            ;ERROR- DATA VALID WASN'T SET
9918 025050 032704 070000      2$:      BIT      @OVRRUN!FRMERR!PARER,R4 ;ARE ANY ERROR FLAGS SET?
9919 025054 001403              BEQ      3$             ;IF NOT, GO CONTINUE PROCESSING
9920 025056 013700 002046      MOV      DZRBUF,R0       ;SET UP FOR ERROR REPORTING
9921 025062 104002              ERROR     2             ;ERROR- RECEIVER ERROR FLAG SET
9922 025064 010401              3$:      MOV      R4,R1     ;COPY THE RECEIVER INFORMATION
9923 025066 000301              SWAB      R1             ;GET THE LINE NUMBER IN THE LOWER BYTE
9924 025070 042701 177770      BIC      @C<7>,R1        ;ISOLATE THE LINE NUMBER
9925 025074 006301              ASL      R1             ;ALIGN IT ON A WORD BOUNDARY
9926 025076 120461 001442      CMPB     R4,TRO(R1)      ;IS THE CHARACTER WHAT IT SHOULD BE?
9927 025102 001413              BEQ      4$             ;IF SO,GO CONTINUE PROCESSING
9928 025104 116105 001442      MOVB     TRO(R1),R5      ;GET WHAT WAS EXPECTED FOR ERROR REPORTING
9929 025110 042705 177400      BIC      @C<377>,R5      ;ELIMINATE PROPAGATED SIGN
9930 025114 042704 177400      BIC      @C<377>,R4      ;ISOLATE THE ACTUAL CHARACTER
9931 025120 010137 001372      MOV      R1,SAVLIN       ;GET THE LINE NUMBER OF THE RECEIVER ERROR
9932 025124 006237 001372      ASR      SAVLIN          ;ALIGN IT CORRECTLY FOR REPORTING
9933 025130 104005              ERROR     5             ;*DATA ERROR
9934 025132 005261 001442      4$:      INC      TRO(R1)  ;SET UP THE NEXT EXPECTED CHARACTER
9935 025136 005203              INC      R3             ;INCREMENT THE COUNT OF RECEIVED CHARACTERS
9936 025140 032761 000020 001442      BIT      @20,TRO(R1) ;HAVE ALL CHARACTERS BEEN RECEIVED?
9937 025146 001402              BEQ      5$             ;IF NOT, GO RECEIVE SOME MORE
9938 025150 020203              CMP      R2,R3          ;HAVE WE RECEIVED ALL CHARACTERS?
9939 025152 001401              BEQ      6$             ;IF SO,GO DETERMINE THE TIMING
9940 025154 000002              5$:      RTI              ;GO CONTINUE TIMING AND ALLOW INTERRUPTS
9941 025156 004737 025002      6$:      JSR      PC,BUILD ;GET THE RELATIVE TIME (SIGNIFICANT BITS)
9942
9943 025162 013700 025242      MOV      OFFSET,R0       ;GET POINTER
9944 025166 013760 001220 002102      MOV      $TMP1,TMTBL(R0) ;SAVE THIS TEST'S TIME
9945 025174 005737 025242      TST      OFFSET         ;FIRST TEST?
9946 025200 001414              BEQ      7$             ;IF NOT, GO CHECK THE TIME
9947 025202 005740              TST      -(R0)          ;POINT TO THE PREVIOUS TIME TAKEN
9948 025204 026037 002102 001220      CMP      TMTBL(R0),$TMP1 ;IS THIS TIME WHAT IT SHOULD BE?
9949 025212 101007              BHI      7$             ;IF SO, GO TO THE NEXT TEST
9950 025214 016005 002102      MOV      TMTBL(R0),R5    ;PLACE WHAT WAS EXPECTED IN R5
9951 025220 010137 001372      MOV      R1,SAVLIN       ;GET THE LINE NUMBER OF THE RECEIVER
9952 025224 006237 001372      ASR      SAVLIN          ;MAKE SURE IT'S THE LINE NUMBER
9953 025230 104021              ERROR     21            ;TIMING ERROR
9954 025232 042777 000140 154602 7$:      BIC      @RIE!MSENAB,@DZCSR ;DISABLE THE DEVICE
9955 025240 000002              RTI              ;RETURN TO THE PROGRAM
9956 025242 000000              OFFSET: 0
```



```
9958
9959      ;DZ11 ECHO/CABLE TEST
9960
9961      ;*STARTING PROCEDURE
9962      ;*LOAD PROGRAM
9963      ;*LOAD ADDRESS 000210
9964      ;*PRESS START
9965      ;*PROGRAM WILL TYPE DZ11 ECHO/CABLE TEST
9966      ;*PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
9967      ;*TYPE IN E OR C RESPECTIVELY
9968      ;*PROGRAM WILL TYPE "VECTOR ADDRESS-"
9969      ;*TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
9970      ;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
9971      ;*PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"
9972      ;*TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
9973      ;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
9974      ;*PROGRAM WILL TYPE "LINE NUMBER-"
9975      ;*TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
9976      ;*,FOLLOWED BY <CARRIAGE RETURN>
9977      ;*PROGRAM WILL TYPE "BAUD RATE-"
9978      ;*TYPE IN THE BAUD RATE OF THE DZ11 TERMINAL
9979      ;*,FOLLOWED BY <CARRIAGE RETURN>
9980      ;*THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL
9981      ;*      50
9982      ;*      75
9983      ;*      110
9984      ;*      135      (ROUNDED OFF 134.5)
9985      ;*      150
9986      ;*      300
9987      ;*      600
9988      ;*      1200
9989      ;*      1800
9990      ;*      2000
9991      ;*      2400
9992      ;*      3600
9993      ;*      4800
9994      ;*      7200
9995      ;*      9600
9996      ;*ALL OTHERS ARE REJECTED
9997
9998      ;*PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTE
10002
10003
10004      ;PROGRAM INITIALIZATION
10005      ;LOCK OUT INTERRUPTS
10006      ;SET UP PROCESSOR STACK
10007      ;SET UP POWER FAIL VECTOR
10008      ;CLEAR PROGRAM FLAGS AND COUNTS
10009
10010      025244 012706 001120      XSTART: MOV      #STACK,SP      ;SET UP PROCESSOR STACK
10011      025250 106427 000340      MTPS      #PR7      ;LOCK OUT INTERRUPTS
10012      025254 012737 025244 001126      MOV      @XSTART,$LPADR ;SET UP IN CASE OF POWER FAIL
10013      025262 005037 027440      CLR      STFLG      ;CLEAR TEST START FLAG
10014      025266 005037 001242      CLR      $PASS      ;CLEAR PASS COUNT
10015      025272 005037 001132      CLR      $ERTTL      ;CLEAR ERROR COUNT
10016      025276 105037 001123      CLRB      $ERFLG      ;CLEAR ERROR FLAG
```

10017	025302	005037	027444		CLR	LAST	;CLEAR LAST ERROR PC
10018	025306	032777	000001	153644	VEC1:	#SW00, @SWR	;IF SW00=1, GET NEW VECTOR
10019	025314	001465			BEQ	OTHER	;AND CSR
10020	025316	012701	000300		VEC2:	#300, R1	
10021	025322	012702	000302		MOV	#302, R2	
10022	025326	010221			1\$:	R2, (R1)+	;RESTORE TRAPCATCHER
10023	025330	005022			CLR	(R2)+	;IN FLOATING VECTOR AREA
10024	025332	022122			CMP	(R1)+, (R2)+	;UPDATE THE POINTERS
10025	025334	020127	001000		CMP	R1, #1000	
10026	025340	001372			BNE	1\$	
10027	025342	104403			INSTR		;INPUT ADDRESS OF DEVICE VECTOR
10028	025344	027472			MVECTOR		;MESSAGE "VECTOR ADDRESS-"
10029	025346	104405			PARAM		;CONVERT STRING TO OCTAL
10030	025350	000300			300		;LOW LIMIT
10031	025352	000770			770		;HIGH LIMIT
10032	025354	002072			DZRV		;LOCATIONS TO BE FILLED
10033	025356	003			.BYTE	3	;LSB MASK
10034	025357	004			.BYTE	4	;NUMBER OF LOCATIONS
10035	025360	104403			INSTR		;INPUT ADDRESS OF DEVICE CSR
10036	025362	027514			MREGAD		;MESSAGE "CONTROL REGISTER ADDRESS-"
10037	025364	104405			PARAM		;CONVERT STRING TO OCTAL
10038	025366	160000			160000		;LOW LIMIT
10039	025370	163700			163700		;HIGH LIMIT
10040	025372	002			DZCSR		;LOCATIONS TO BE FILLED
10041	025374	007			.BYTE	7	;LSB MASK
10042	025375	001			.BYTE	1	;NUMBER OF LOCATIONS
10043	025376	013737	002042	002046	MOV	DZCSR, DZRBUR	;BEGIN BUILDING DEVICE ADDRESSES
10044	025404	062737	000002	002046	ADD	#2, DZRBUR	;FORM THE READ BUFFER ADDRESS
10045	025412	013737	002046	002052	MOV	DZRBUR, DZLPR	;REMEMBER THAT THIS IS ALSO LINE PARAMETER REG.
10046	025420	013737	002046	002056	MOV	DZRBUR, DZTCR	;BEGIN BUILDING TRANSMITTER CONTROL REGISTER
10047	025426	062737	000002	002056	ADD	#2, DZTCR	;FORM THE TRANSMITTER CONTROL REGISTER POINTER
10048	025434	013737	002056	002060	MOV	DZTCR, HDZTCR	
10049	025442	005237	002060		INC	HDZTCR	
10050	025446	013737	002056	002066	MOV	DZTCR, DZTDR	;BEGIN FORMING TRANSMITTER DATA REGISTER
10051	025454	062737	000002	002066	ADD	#2, DZTDR	;FORM THE TRANSMITTER DATA REGISTER
10052	025462	013737	002066	002062	MOV	DZTDR, DZMSR	
10053	025470	032777	000002	153462	OTHER:	BIT	;RESELECT OF TEST?
10054	025476	001427			BEQ	XBEGIN	;IF NOT, SKIP ASKING WHICH ONE
10055	025500	104403			INSTR		;INPUT WHICH TEST YOU ARE RUNNING
10056	025502	027700			MWHICH		;ECHO OR CABLE
10057	025504	104416			PAWCH		;SET FLAG
10058	025506	027436			WCHFLG		;THIS FLAG
10059	025510	104403			BAUD:	INSTR	;INPUT BAUD RATE
10060	025512	027622			MSPEED		;MESSAGE "BAUD RATE-"
10061	025514	104415			PARMD		;CONVERT DECIMAL STRING TO OCTAL
10062	025516	000062			50.		;LOW LIMIT
10063	025520	022600			9600.		;HIGH LIMIT
10064	025522	027454			LINE		;LOCATION TO BE FILLED
10065	025524	000			.BYTE	0	;LSB MASK
10066	025525	001			.BYTE	1	;NUMBER OF LOCATIONS
10067	025526	104413			LINE:	DEVICE.CLR	;CLEAR DEVICE
10068	025530	005037	027440		CLR	STFLG	;CLEAR PROGRAM START FLAG
10069	025534	104403			INSTR		;INPUT LINE NUMBER
10070	025536	027612			MLINE		;MESSAGE "LINE NUMBER-"
10071	025540	104405			PARAM		;CONVERT STRING TO OCTAL
10072	025542	000000			0		;LOW LIMIT

ADDRESS	HEX	HEX	HEX	HEX	ASSEMBLY	COMMENT
10073	025544	000007			7	;HIGH LIMIT
10074	025546	001372			SAVLIN	;LOCATION TO BE FILLED
10075	025550	000			0	;LSB MASK
10076	025551	001			.BYTE	;NUMBER OF LOCATIONS
10077	025552	004537	027242		.BYTE JSR 1 R5,SET	
10078						
10079	025556	106427	000340		XBEGIN: MTPS #PR7	;LOCK OUT INTERRUPTS
10080	025562	012706	001120		MOV #STACK,SP	;SET UP PROCESSOR STACK
10081	025566	005037	027442		CLR LOCKUP	;CLEAR TIMEOUT
10082	025572	005737	027436		TST WCHFLG	;ECHO OR CABLE TEST ?
10083	025576	001413			BEQ 2\$	;ECHO
10084	025600	012737	026314	001126	MOV #TEST2,\$LPADR	;CABLE TEST
10085	025606	005737	027440		TST STFLG	;ARE YOU LOOPING ?
10086	025612	001017			BNE 1\$	;YES
10087	025614	005137	027440		COM STFLG	;NO
10088	025620	104402	027773		TYPE ,MCABLE	;TYPE CABLE TEST
10089	025624	000412			BR 1\$	
10090	025626	012737	025656	001126	2\$: MOV #TEST1,\$LPADR	;SET UP ECHO TEST
10091	025634	005737	027440		1ST STFLG	;ARE YOU LOOPING ?
10092	025640	001004			BNE 1\$	;YES
10093	025642	005137	027440		COM STFLG	;NO
10094	025646	104402	027746		TYPE ,MTERM	;TYPE ECHO TEST
10095	025652	000177	153250		1\$: JMP @LPADR	;START TESTING
10096						
10097						;THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
10098						; (IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
10099						; ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
10100	025656	104413			TEST1: DEVICE.CLR	;CLEAR DZ11
10101	025660	012737	000001	001122	MOV #1,\$TSTNM	
10102	025666	013777	027462	154162	MOV NUMTCR,@DZTCR	;SET TCR BIT
10103	025674	013737	027460	001366	MOV NUMLIN,PAR	;SET PARAMETERS
10104	025702	053737	027456	001366	BIS SPEED,PAR	;SET BAUD RATE
10105	025710	013777	001366	154134	MOV PAR,@DZLPR	;LOAD PARAM.
10106	025716	012777	000040	154116	MOV #MSENAB,@DZCSR	;SET SCANN ENABLE
10107	025724	005004			CLR R4	
10108	025726	012705	030010		MOV #MQUICK,R5	;SET MESSAGE BUFFER
10109	025732	005777	154104		3\$: TST @DZCSR	;TRDY?
10110	025736	100404			BMI 2\$	;BR IF YES
10111	025740	104414			DELAY	;WAIT
10112	025742	005304			DEC R4	
10113	025744	001372			BNE 3\$	
10114	025746	104003			ERROR 3	
10115	025750	005004			2\$: CLR R4	;NO TRDY SET! WHY?
10116	025752	112577	154110		MOVB (R5)+,@DZTDR	;RESET COUNTER TO 0
10117	025756	001365			BNE 3\$	;LOAD CHAR
10118	025760	004737	007642		JSR PC,SERV.G	
10119	025764	122777	000377	153166	CMPB #377,@SWR	; <+G>?
10120	025772	001731			BEQ TEST1	;RE-DO QUICK BROWN?
10121	025774	104413				;BR IF REPEAT PATTERN
10122	025776	106427	000340		DEVICE.CLR	
10123	026002	012737	026752	001360	MTPS #PR7	;LOCK OUT INTERRUPTS
10124	026010	104413			MOV #XEOP,NEXT	
10125	026012	013737	027460	001366	DEVICE.CLR	
10126	026020	053737	027456	001366	MOV NUMLIN,PAR	;SELECT LINE # & SET INTERRUPT ENABLE
10127					BIS SPEED,PAR	;SET LINE SPEED AND
10128	026026	052737	010000	001366	BIS #RCVON,PAR	;CHARACTER LENGTH (TRANS. & REC.)
						;MAKE SURE RECEIVER IS TURNED ON

```

10129 026034 013777 001366 154010      MOV    PAR,@DZLPR      ;LOAD THE LINE PARAMETER REGISTER
10130 026042 012777 026116 154022      MOV    #INTSVC,@DZRIV ;SET UP INTERRUPT SERVICE
10131 026050 013777 027464 154016      MOV    PRI0,@DZRI5    ;AND LEVEL
10132 026056 106437 030336              MTPS    @#LESS1       ;ALLOW INTERRUPTS
10133 026062 012777 000140 153752      MOV    #RIE!MSENAB,@DZCSR ;SET RECEIVER INTERRUPT ENABLE
10134 026070 104402 027640              TYPE    ,MCHAR      ;TYPE "ANY CHARACTER"
10135 026074 105777 153064      1$:    TSTB    @#TKS      ;IF SOMBODY HITS A KEY- GET NEW LINE #
10136 026100 100375                  BPL     1$             ;LOOP HERE
10137 026102 005777 153060      TST     @#TKB      ;CLEAR CHAR
10138 026106 004737 007642      JSR     PC,SERV.G    ;MAKE SURE IT WASN'T <+G>
10139 026112 000137 025526      JMP     LINEX        ;
10140
10141
10142
10143 026116 105777 153720      INTSVC: ;THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
10144 026122 100401              TSTB    @DZCSR      ;TEST REC. FLAG
10145 026124 104004              BMI     .+4
10146 026126 017737 153714 027466      ERROR  4             ;ERROR - INTERRUPT NOT CAUSED BY FLAG
10147 026134 100401              MOV     @DZRBUF,RECDAT
10148 026136 104023              BMI     .+4
10149 026140 032737 020000 027466      ERROR  23            ;NON- VALID CHARACTER
10150 026146 001401              BIT     #BIT13,RECDAT ;CHECK FOR FRAMING ERROR
10151 026150 104025              BEQ     .+4            ;BR IF NO ERROR
10152                                ERROR  25            ;EITHER SOMBODY HIT THE
10153 026152 113737 027466 027470      MOV     RECDAT,TBUF    ;"BREAK KEY" OR YOU HAVE AN ERROR!
10154 026160 113737 027466 011262      MOV     RECDAT,INBUF   ;MOVE CHARACTER TO OUTPUT AREA
10155 026166 042737 177600 011262      BIC     #+C<177>,INBUF ;MOVE CHARACTER TO CHECK FOR +C
10156 026174 042737 174377 027466      BIC     #174377,RECDAT ;STRIP JUNK PLUS PARITY
10157 026202 000337 027466              SWAB    RECDAT      ;SAVE ONLY LINE NUMBER
10158 026206 023737 001372 027466      CMP     SAVLIN,RECDAT ;DOES THE LINE # COMPARE?
10159 026214 001401              BEQ     .+4
10160 026216 104015              ERROR  15            ;*WRONG LINE NUMBER
10161 026220 012777 000040 153614      MOV     #MSENAB,@DZCSR ;START THE TRANSMITTERS SCANNER
10162 026226 123727 011262 000003      CMPB    INBUF,#3      ;IS IT A +C ?
10163 026234 001004              BNE     1$             ;NO
10164 026236 104413              DEVICE.CLR
10165 026240 012716 026752              MOV     #XEOP,(SP)    ;CRUNCH STACK
10166 026244 000002              RTI
10167 026246 005003      1$:    CLR     R3             ;INITIALIZE DELAY
10168 026250 013777 027462 153600      MOV     NUMTCR,@DZTCR ;ENABLE THE LINE
10169 026256 005777 153560      10$:   TST     @DZCSR      ;TRANSMITTER READY?
10170 026262 100403              BMI     2$             ;IF YES BRANCH
10171 026264 005203              INC     R3             ;INCREMENT DELAY
10172 026266 001373              BNE     10$            ;DELAY DONE?
10173 026270 104003              ERROR  3             ;TRANSMIT READY NOT SET!
10174 026272 113777 027470 153566      MOV     TBUF,@DZTDR    ;TRANSMIT THE -CHARACTER
10175 026300 012777 000140 153534      MOV     #RIE!MSENAB,@DZCSR ;RESTART THE RECEIVER
10176 026306 005077 153544              CLR     @DZTCR      ;CLEAR TCR BIT
10177 026312 000002              RTI
10178
10179
10180
10181
10182
10183 026314 106427 000340      ;THIS TEST TRANSMITS A BINARY COUNT PATTERN
10184 026320 012737 000002 001122      ;VIA INTERRUPT MODE TO THE RECEIVER
10184 026320 012737 000002 001122      ;...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
10184 026320 012737 000002 001122      TEST2: MTPS    #PR7      ;DISABLE INTERRUPTS
10184 026320 012737 000002 001122      MOV     #2,$TSTNM

```


10185	026326	012737	026752	001360		MOV	#XEOB,NEXT	
10186	026334	104413				DEVICE.CLR		
10187								
10188								
10189								
10190								
10191	026336	012737	026344	001362		MOV	#1\$,LOCK	;LOOP
10192	026344	113777	027462	153506	1\$:	MOVB	NUMTCR,@HDZTCR	;SET DTR
10193	026352	005005				CLR	R5	
10194	026354	153705	027462			BISB	NUMTCR,R5	;BUILD EXPECTED
10195	026360	000305				SWAB	R5	;PUT IN HIGH BYTE
10196	026362	153705	027462			BISB	NUMTCR,R5	
10197	026366	104414				DELAY		;WAIT FOR CABLE DELAY
10198	026370	017704	153466			MOV	@DZMSP,R4	;READY MODEM BITS
10199	026374	020504				CMP	R5,R4	;ARE THEY OK?
10200	026376	001401				BEQ	2\$	;BR IF YES
10201	026400	104022				ERROR	22	;IS THE TEST CONNECTOR ON?
10202								;HAS RIGHT LINE BEEN SELECTED?
10203								;IF SO- YOU HAVE A PROBLEM!
10204								;MODEM BITS NOT RIGHT
10205	026402	104401			2\$:	SCOP1		;LOOP
10206	026404	104413			3\$:	DEVICE.CLR		;INIT DZ11
10207	026406	013737	027456	001366		MOV	SPEED,PAR	;SET LINE SPEED
10208	026414	053737	027460	001366		BIS	NUMLIN,PAR	;SELECT LINE # & REC. INTERRUPT ENABLE
10209	026422	052737	010000	001366		BIS	#RCVON,PAR	;ENABLE THE RECEIVER FOR THIS LINE
10210	026430	052777	040140	153404		BIS	#TIE!RIE!MSENAB,@DZCSR	;SET TRANSMITTER INTERRUPT ENABLE
10211	026436	012777	026552	153426		MOV	#INTREC,@DZRIV	;SET UP INTR SERVICE
10212	026444	013777	027464	153422		MOV	PRI0,@DZRI5	;SET UP LEVEL
10213	026452	012777	026732	153416		MOV	#INTRAN,@DZTIV	;SET UP INTR SERVICE
10214	026460	013777	027464	153412		MOV	PRI0,@DZTIS	;SET UP LEVEL
10215	026466	005001				CLR	R1	;RX DATA POINTER- SET TO 0
10216	026470	005002				CLR	R2	;TX DATA POINTER- SET TO 0
10217	026472	013777	001366	153352		MOV	PAR,@DZLPR	;SET THE PARAMETERS AND TURN ON RECEIVER
10218	026500	106437	030336			MTPS	@#LESS1	;ALLOW INTERRUPTS
10219	026504	013777	027462	153344		MOV	NUMTCR,@DZTCR	;SET UP TCR BIT
10220								
10221								
10222	026512	105777	152446		SPIN:	TSTB	@\$TKS	;IF SOMEBODY HITS A KEY- GET A NEW LINE #
10223	026516	100006				BPL	1\$	;BR IF NO KEY HIT
10224	026520	005777	152442			TST	@\$TKB	;CLEAR CHAR
10225	026524	004737	007642			JSR	PC,SERV.G	;MAKE SURE IT WASN'T <+G>
10226	026530	000137	025526			JMP	LINEX	;SW02=1
10227	026534	005237	027442		1\$:	INC	LOCKUP	;INC TIMEOUT FLAG
10228	026540	001364				BNE	SPIN	;IF NOT 0 RETURN SPINNING
10229	026542	104011				ERROR	11	;*RECEIVER FAILED TO INTERRUPT CHECK CABLE/TERMINATOR
10230	026544	104413			QUITS:	DEVICE.CLR		
10231	026546	000137	026752			JMP	XEOB	;CALL FOR END OF PASS
10232	026552	005037	027442		INTREC:	CLR	LOCKUP	;CLEAR TIMEOUT FLAG
10233	026556	105777	153260			TSTB	@DZCSR	;TEST REC DONE
10234	026562	100401				BMI	+.4	;YES
10235	026564	104004				ERROR	4	;*FALSE INTERRUPT
10236	026566	017737	153254	027466		MOV	@DZRBUF,RECDAT	;SAVE WORD
10237	026574	100401				BMI	+.4	
10238	026576	104023				ERROR	23	;*NON VALID CHARACTER
10239	026600	032737	040000	027466		BIT	#BIT14,RECDAT	;DATA OVERRUN ?
10240	026606	001401				BEQ	+.4	;NO

10241	026610	104024		ERROR	24	;*YES
10242	026612	032737	020000 027466	BIT	#BIT13,RECDAT	;FRAMING ERROR ?
10243	026620	001401		BEQ	..+4	;NO
10244	026622	104025		ERROR	25	;*YES
10245	026624	032737	010000 027466	BIT	#BIT12,RECDAT	;PARITY ERROR ?
10246	026632	001401		BEQ	..+4	;NO
10247	026634	104026		ERROR	26	;*YES
10248	026636	110105		MOVB	R1,R5	;SET EXPECTED
10249	026640	042705	177400	BIC	#C<377>,R5	;CLEAR HIGH BYTE
10250	026644	113704	027466	MOVB	RECDAT,R4	;GET FOUND
10251	026650	042704	177400	BIC	#C<377>,R4	;CLEAR HIGH BYTE
10252	026654	020504		CMP	R5,R4 ;OK?	
10253	026656	001401		BEQ	..+4	
10254	026660	104005		ERROR	5	;DATA ERROR
10255	026662	042737	174377 027466	BIC	#174377,RECDAT	;SAVE ONLY LINE NUMBER
10256	026670	000337	027466	SWAB	RECDAT	
10257	026674	023737	001372 027466	CMP	SAVLIN,RECDAT	;DOES THE LINE # COMPARE ?
10258	026702	001401		BEQ	..+4	;YES
10259	026704	104015		ERROR	15	;*WRONG LINE #
10260	026706	120127	000377	CMPB	R1,#377	;LAST CHARACTER ?
10261	026712	001003		BNE	1\$	;NO
10262	026714	012716	026544	MOV	#QUITS,(SP)	;CRUNCH STACK
10263	026720	000403		BR	2\$	
10264	026722	105201		1\$: INCB	R1	;UPDATE EXPECTED DATA
10265	026724	012716	026512	MOV	#SPIN,(SP)	;CRUNCH STACK
10266	026730	000002		2\$: RTI		
10267						
10268	026732	005777	153104	INTRAN: TST	@DZCSR ;TEST TRANSMIT FLAG	
10269	026736	100401		BMI	..+4	
10270	026740	104003		ERROR	3	;*FALSE INTERRUPT
10271	026742	110277	153120	MOVB	R2,@DZTDR	;TRANSMIT A CHARACTER
10272	026746	105202		INCB	R2	;UPDATE TX DATA
10273	026750	000002		RTI	;RETURN	


```

10275
10276
10277
10278
10279 026752 104402
10280 026754 027550
10281 026756 005037 027444
10282 026762 105037 001123
10283 026766 000137 025556
10284
10285
10286 026772 011605
10287 026774 012537 027156
10288 027000 012537 027160
10289 027004 012537 027162
10290 027010 112537 027164
10291 027014 112537 027165
10292 027020 010516
10293 027022 005005
10294 027024 012704 011262
10295 027030 122714 000015
10296 027034 001424
10297 027036 121427 000060
10298 027042 002421
10299 027044 121427 000071
10300 027050 003016
10301 027052 142714 000060
10302 027056 005002
10303 027060 152402
10304 027062 060205
10305 027064 122714 000015
10306 027070 001410
10307 027072 006305
10308 027074 010502
10309 027076 006305
10310 027100 006305
10311 027102 060205
10312 027104 000754
10313 027106 104404
10314 027110 000744
10315
10316
10317
10318 027112 020537 027160
10319 027116 101373
10320 027120 020537 027156
10321 027124 103770
10322 027126 133705 027164
10323 027132 001365
10324
10325
10326
10327 027134 013704 027162
10328 027140 010524
10329 027142 062705 000002
10330 027146 105337 027165

;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$

```

```

10331 027152 001372      BNE      5$
10332 027154 000002      RTI
10333 027156 000000      6$:      0
10334 027160 000000      7$:      0
10335 027162 000000      8$:      0
10336 027164      000      9$:      .BYTE 0
10337 027165      000     10$:      .BYTE 0
10338
10339
10340
10341      ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
10342      ;BUFFER TO THE CHARACTERS "E" AND "C".
10343      ;IF THE CHARACTER IS "E" CLEAR THE FLAG
10344      ;IF THE CHARACTER IS "C" SET THE FLAG
10345 027166 017605 000000 .PAWCH:MOV      0(SP),R5
10346 027172 142737 000040 011262 BICB      #40,INBUF      ;SET FOR LOWER CASE INPUT
10347 027200 122737 000105 011262 CMPB      #'E,INBUF      ;IS IT "E" ?
10348 027206 001002      BNE      1$
10349 027210 105015      CLRB      (R5)      ;000
10350 027212 000406      BR      2$
10351 027214 122737 000103 011262 1$:      CMPB      #'C,INBUF      ;IS IT "C" ?
10352 027222 001005      BNE      3$
10353 027224 112715 177777      MOVB      #-1,(R5)      ;3177
10354 027230 062716 000002      2$:      ADD      #2,(SP)
10355 027234 000002      RTI
10356 027236 104404      3$:      INSTER      ;RETRY
10357 027240 000752      BR      .PAWCH
10358
10359
10360
10361      ;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
10362      ;LINE NUMBER (SAVLIN) FOR DZLPR, DZTCR AND DZCSR
10363      ;REGISTER USAGE.
10364
10365 027242 013737 001372 027460 SET:      MOV      SAVLIN,NUMLIN      ;SAVE SAVLIN
10366 027250 013700 001372      XTCR0:      MOV      SAVLIN,R0      ;COPY THE LINE NUMBER FOR LOOP CONTROL
10367 027254 005037 027462      CLRB      NUMTCR      ;SET A DEFAULT OF LINE 0 OR NO LINES
10368 027260 012702 000001      MOV      #1,R2      ;SET A BIT POINTER TO THE FIRST LINE
10369 027264 005300      XTCR1:      DEC      R0      ;REDUCE THE INDICATOR.IS IT MINUS YET?
10370 027266 100402      BMI      SET1      ;IF SO, R2 POINTS TO THE RIGHT LINE
10371 027270 006302      ASL      R2      ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
10372 027272 000774      BR      XTCR1      ;GO SEE IF THIS LINE IS THE ONE
10373 027274 012701 027336      SET1:      MOV      #TABLE2,R1
10374 027300 010237 027462      MOV      R2,NUMTCR      ;COPY THE CORRECT BIT POINTER
10375 027304 022137 027454      1$:      CMP      (R1)+,LINESP
10376 027310 001407      BEQ      2$
10377 027312 005721      TST      (R1)+      ;IS IT THE END OF TABLE?
10378 027314 001373      BNE      1$      ;NO
10379 027316 104402 027564      TYPE      ,MINVAL      ;INVALID BAUD RATE,BEGIN AGAIN
10380 027322 012705 025510      MOV      #BAUD,R5      ;JUMP TO BAUD THRU R5
10381 027326 000402      BR      3$
10382 027330 011137 027456      2$:      MOV      (R1),SPEED      ;SET UP BAUD RATE
10383 027334 000205      3$:      RTS      R5
10384
10385
10386

```


			;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR)			
10387			TABLE2:	.WORD	50.	;50 BAUD
10388	027336	000062		.WORD	10070	;
10389	027340	010070		.WORD	75.	;75 BAUD
10390	027342	000113		.WORD	10470	;
10391	027344	010470		.WORD	110.	;110 BAUD
10392	027346	000156		.WORD	11070	;TWO STOP BITS
10393	027350	011070		.WORD	135.	;134.5 BAUD
10394	027352	000207		.WORD	11470	;TWO STOP BITS
10395	027354	011470		.WORD	150.	;150 BAUD
10396	027356	000226		.WORD	12070	;TWO STOP BITS
10397	027360	012070		.WORD	300.	;300 BAUD
10398	027362	000454		.WORD	12430	;ONE STOP BIT
10399	027364	012430		.WORD	600.	;600 BAUD
10400	027366	001130		.WORD	13030	;ONE STOP BIT
10401	027370	013030		.WORD	1200.	;1200 BAUD
10402	027372	002260		.WORD	13430	;ONE STOP BIT
10403	027374	013430		.WORD	1800.	;1800 BAUD
10404	027376	003410		.WORD	14030	;ONE STOP BIT
10405	027400	014030		.WORD	2000.	;2000 BAUD
10406	027402	003720		.WORD	14430	;ONE STOP BIT
10407	027404	014430		.WORD	2400.	;2400 BAUD
10408	027406	004540		.WORD	15030	;ONE STOP BIT
10409	027410	015030		.WORD	3600.	;3600 BAUD
10410	027412	007020		.WORD	15430	;ONE STOP BIT
10411	027414	015430		.WORD	4800.	;4800 BAUD
10412	027416	011300		.WORD	16030	;ONE STOP BIT
10413	027420	016030		.WORD	7200.	;7200 BAUD
10414	027422	016040		.WORD	16430	;ONE STOP BIT
10415	027424	016430		.WORD	9600.	;9600 BAUD
10416	027426	022600		.WORD	17070	;
10417	027430	017070		.WORD	-1.0	;TABLE TERMINATOR
10418	027432	177777	000000			
10419						
10420						
10421	027436	000000	WCHFLG:	0		;ECHO OR CABLE FLAG
10422	027440	000000	STFLG:	0		;PROGRAM START FLAG
10423	027442	000000	LOCKUP:	0		;TIMEOUT FLAG
10424	027444	000000	LAST:	0		;LAST ERROR PC
10425	027446	000000	TDATA:	0		
10426	027450	000000	RDATA:	0		
10427	027452	000000	BYTCNT:	0		
10428	027454	000156	LINESP:	110.		;DEFAULT BAUD RATE
10429	027456	006307	SPEED:	6307		;DEFAULT 110 BAUD, 8 BITS/CHAR,
10430						;FDX, 2 STOP BITS
10431	027460	000100	NUMLIN:	100		;DEFAULT VALUE, REC. INTERRUPT ENABLED
10432						
10433	027462	000001	NUMTCR:	1		;DEFAULT VALUE, TCR BIT 0
10434	027464	000240	PRI0:	240		;DEFAULT DEVICE PRIORITY 5
10435	027466	000000	RECDAT:	0		
10436	027470	000000	TBUF:	0		
10440	027472	053200	041505	047524	MVECTO:	.ASCIZ <200>/VECTOR ADDRESS - /
10441	027514	041600	047117	051124	MREGAD:	.ASCIZ <200>/CONTROL REGISTER ADDRESS - /
10442	027550	050200	051501	020123	MPASS:	.ASCIZ <200>/PASS DONE. /
10443	027564	044600	053116	046101	MINVAL:	.ASCIZ <200>/INVALID BAUD RATE - /
10444	027612	046200	047111	035105	MLINE:	.ASCIZ <200>/LINE: /

```

10446 027640 052200 050131 020105 MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZ11 TERMINAL /
10447 027700 053600 044510 044103 MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /
10448 027746 052200 051105 044515 MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /
10449 027773 200 040503 046102 MCABLE: .ASCIZ <200>/CABLE TEST /
10450 030010 006777 177777 177412 MQUICK: .ASCII <377><15><377><377><12><377><377>
10451 030017 124 042510 050440 .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/
10452 030114 006777 177777 177412 .ASCII <377><15><377><377><12><377><377><377><0>
10453 030126
10457 .EVEN
10458 ;:*****
10459 ;UTILITIES
10460 ;:*****
10461 ;THIS UTILITY CALCULATES PRIORITY LEVEL,SETS UP CSR'S,SETS UP VECTORS.
10462 030126 006337 030334 DZLEV: ASL DZPRT ;BUILD PRIORITY IN THIS LOCATION
10463 030132 006337 030334 ASL DZPRT ;USING ARITHMETIC SHIFTS, ROTATE
10464 030136 006337 030334 ASL DZPRT ; THE PRIORITY LEVEL PAST
10465 030142 006337 030334 ASL DZPRT ; THE BIT POSITIONS CORRE-
10466 030146 006337 030334 ASL DZPRT ; SPONDING TO THE CONDITION CODES
10467 030152 013737 030334 030336 MOV DZPRT,LESS1 ;MOVE THIS TO LESS1
10468 030160 162737 000001 030336 SUB #1,LESS1 ;CREATE THE NEXT LOWEST PRIORITY
10469 030166 042737 000037 030336 BIC #37,LESS1 ;INSURE THAT THE TNZVC BITS ARE CLEAR
10470 030174 013700 002072 MOV DZRIV,R0 ;PLACE THE BASE VECTOR ADDRESS IN R0
10471 030200 062700 000002 ADD #2,R0 ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
10472 030204 010037 002074 MOV R0,DZRRIS ;STORE IT HERE
10473 030210 062700 000002 ADD #2,R0 ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
10474 030214 010037 002076 MOV R0,DZTIV ;STORE IT HERE
10475 030220 062700 000002 ADD #2,R0 ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
10476 030224 010037 002100 MOV R0,DZTIS ;STORE IT HERE
10477
10478 ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZ11. $BASE IS THE BASE ADDRESS
10479 ;OF THE DEVICE
10480 030230 013700 001310 MOV $BASE,R0 ;COPY THE ADDRESS BEING LOADED
10481 030234 010037 002042 MOV R0,DZCSR ;XXX0
10482 030240 005200 INC R0
10483 030242 010037 002044 MOV R0,HDZCSR ;XXX1
10484 030246 005200 INC R0
10485 030250 010037 002046 MOV R0,DZRBUF ;XXX2
10486 030254 010037 002052 MOV R0,DZLPR ;XXX2
10487 030260 005200 INC R0
10488 030262 010037 002050 MOV R0,HDZRBUF ;XXX3
10489 030266 010037 002054 MOV R0,HDZLPR ;XXX3
10490 030272 005200 INC R0
10491 030274 010037 002056 MOV R0,DZTCR ;XXX4
10492 030300 005200 INC R0
10493 030302 010037 002060 MOV R0,HDZTCR ;XXX5
10494 030306 005200 INC R0
10495 030310 010037 002062 MOV R0,DZMSR ;XXX6
10496 030314 010037 002066 MOV R0,DZTDR ;XXX6
10497 030320 005200 INC R0
10498 030322 010037 002064 MOV R0,HDZMSR ;XXX7
10499 030326 010037 002070 MOV R0,HDZTDR ;XXX7
10500 030332 000207 RTS PC
10501 030334 000240 DZPRT: PR5
10502 030336 000200 LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS
10503

```


			;ERROR ERROR TABLE	
			.ERRTAB:	
10505				
10506	030340	000000	0	;ERROR 0
10507	030342	000000	0	
10508	030344	000000	0	
10509				
10510	030346	030566	EM1	;ERROR
10511	030350	032073	DH1	
10512	030352	032324	DT1	
10513				
10514	030354	030641	EM2	;ERROR 2
10515	030356	032116	DH2	
10516	030360	032336	DT2	
10517				
10518	030362	030667	EM3	;ERROR 3
10519	030364	032151	DH3	
10520	030366	032354	DT3	
10521				
10522	030370	030726	EM4	;ERROR 4
10523	030372	032151	DH3	
10524	030374	032354	DT3	
10525				
10526	030376	030755	EM5	;ERROR 5
10527	030400	032163	DH4	
10528	030402	032362	DT4	
10529				
10530	030404	031004	EM6	;ERROR 6
10531	030406	032163	DH4	
10532	030410	032362	DT4	
10533				
10534	030412	031042	EM7	;ERROR 7
10535	030414	032151	DH3	
10536	030416	032354	DT3	
10537				
10538	030420	031103	EM8	;ERROR 10
10539	030422	032151	DH3	
10540	030424	032354	DT3	
10541				
10542	030426	031145	EM9	;ERROR 11
10543	030430	032151	DH3	
10544	030432	032354	DT3	
10545				
10546	030434	031203	EM10	;ERROR 12
10547	030436	032151	DH3	
10548	030440	032354	DT3	
10549				
10550	030442	031242	EM13	;ERROR 13
10551	030444	032151	DH3	
10552	030446	032354	DT3	
10553				
10554	030450	031273	EM14	;ERROR 14
10555	030452	032151	DH3	
10556	030454	032354	DT3	
10557				
10558	030456	031325	EM15	;ERROR 15
10559	030460	000000	0	
10560	030462	000000	0	

10561			
10562	030464	031367	EM16
10563	030466	032151	DH3
10564	030470	032354	DT3
10565			
10566	030472	031440	EM17 ;ERROR 17
10567	030474	032151	DH3
10568	030476	032354	DT3
10569			
10570	030500	031476	EM20
10571	030502	032151	- DH3
10572	030504	032354	DT3
10573			
10574	030506	031537	EM21 ;ERROR 21
10575	030510	032212	DH5
10576	030512	032400	DT5
10577			
10578	030514	031567	EM22 ;ERROR 22
10579	030516	032163	DH4
10580	030520	032362	DT4
10581			
10582	030522	031631	EM23 ;ERROR 23
10583	030524	032151	DH3
10584	030526	032354	DT3
10585			
10586	030530	031661	EM24
10587	030532	032151	DH3
10588	030534	032354	DT3
10589			
10590	030536	031707	EM25
10591	030540	032151	DH3
10592	030542	032354	DT3
10593			
10594	030544	031737	EM26
10595	030546	032151	DH3
10596	030550	032354	DT3
10597			
10598	030552	031766	EM27
10599	030554	032151	DH3
10600	030556	032354	DT3
10601			
10602	030560	032036	EM30
10603	030562	032271	DH6
10604	030564	032422	DT6
10605			


```

10607
10608
10612 030566 047200 020117 046123 EM1: .ASCIZ <200>/NO SLAVE SYNC RESPONSE FROM DZ11 REGISTER/
10613 030641 200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE?
10614 030667 200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
10615 030726 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
10616 030755 200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
10617 031004 042200 030532 020061 EM6: .ASCIZ <200>/DZ11 *RECEIVER BUFFER* ERROR/
10618 031042 052200 040522 051516 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
10619 031103 200 047125 054105 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
10620 031145 200 042522 042503 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
10621 031203 200 047125 054105 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
10622 031242 051600 046111 020117 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
10623 031273 200 044523 047514 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
10624 031325 200 041501 044524 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
10625 031367 200 042522 042101 EM16: .ASCIZ <200>/READING DZRBUFF DID NOT CLEAR SILO ALARM/
10626 031440 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
10627 031476 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
10628 031537 200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
10629 031567 200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
10630 031631 200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
10631 031661 200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
10632 031707 200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
10633 031737 200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
10634 031766 043200 046125 020114 EM27: .ASCIZ <200>/FULL BINARY COUNT PATTERN NOT RECEIVED/
10635 032036 041200 052501 020104 EM30: .ASCIZ <200>/BAUD RATE TIMING TEST ERROR/
10636 032073 200 051124 050101 DH1: .ASCIZ <200>/TRAP PC DZ11 REG/
10637 032116 042600 050130 041505 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
10638 032151 200 044514 042516 DH3: .ASCIZ <200>/LINE NO./
10639 032163 200 054105 042520 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
10640 032212 052200 020130 044514 DH5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/
10641 032271 200 044510 044107 DH6: .ASCIZ <200>/HIGH LOW COUNT LINE/
10642
10646 .EVEN
10647 032324 000002 DT1: 2
10648 032326 006 003 .BYTE 6,3
10649 032330 001204 $REG1
10650 032332 006 001 .BYTE 6,1
10651 032334 001202 $REG0
10652
10653 032336 000003 DT2: 3
10654 032340 006 004 .BYTE 6,4
10655 032342 001214 $REG5
10656 032344 006 001 .BYTE 6,1
10657 032346 001212 $REG4
10658 032350 006 001 .BYTE 6,1
10659 032352 001202 $REG0
10660
10661 032354 000001 DT3: 1
10662 032356 003 001 .BYTE 3,1
10663 032360 001372 SAVLIN
10664
10665 032362 000003 DT4: 3
10666 032364 006 004 .BYTE 6,4
10667 032366 001214 $REG5
10668 032370 006 001 .BYTE 6,1

```

10669	032372	001212		\$REG4	
10670	032374	003	001	.BYTE	3.1
10671	032376	001372		SAVLIN	
10672					
10673	032400	000004		DT5: 4	
10674	032402	003	005	.BYTE	3.5
10675	032404	001372		SAVLIN	
10676	032406	006	011	.BYTE	6.9.
10677	032410	001214		\$REG5	
10678	032412	006	007	.BYTE	6.7
10679	032414	001220		\$TMP1	
10680	032416	006	001	.BYTE	6.1
10681	032420	001400		REGIST	
10682	032422	000004		DT6: 4	
10683	032424	006	001	.BYTE	6.1
10684	032426	024112		DCOUNT	;FOR BAUD RATE ERROR MESSAGE
10685	032430	006	001	.BYTE	6.1
10686	032432	024114		ECOUNT	;HIGH COUNTER
10687	032434	006	001	.BYTE	6.1
10688	032436	024110		BCOUNT	;LOW COUNTER
10689	032440	001	001	.BYTE	6.1
10690	032442	001372		SAVLIN	;ACTUAL XMIT COUNT
10691					
10699					
10700					
10701					
10702	032444	002450			
10703	032446	001560			
10704	032450	001120			
10705	032452	000750			
10706	032454	000660			
10707	032456	000330			
10708	032460	000150			
10709	032462	000060			
10710	032464	000040			
10711	032466	000030			
10712	032470	000020			
10713	032472	000010			
10714	032474	000001			
10715	032476	000001			
10716	032500	000001			
10717	032502	000001			
10718					
10719					
10720					
10721					
10722					
10723	032504				
10727		001512			
10728	001512	100000			
10729		000001			

;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

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

ABASE = 160010
ACDW1 = 000000
ACDW2 = 000000
ACPUOP = 000000
ACTIVE 001412
ADDW0 = 000000
ADDW1 = 000000
ADDW10 = 000000
ADDW11 = 000000
ADDW12 = 000000
ADDW13 = 000000
ADDW14 = 000000
ADDW15 = 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT = 000000
ADEVM = 000000
ADRCNT 006635
ADVANC = 104400
AENV = 000000
AENVM = 000000
AFATAL = 000000
AMADR1 = 000000
AMADR2 = 000000
AMADR3 = 000000
AMADR4 = 000000
AMAMS1 = 000000
AMAMS2 = 000000
AMAMS3 = 000000
AMAMS4 = 000000
AMSGAD = 000000
AMSGLG = 000000
AMSGTY = 000000
AMTYP1 = 000000
AMTYP2 = 000000
AMTYP3 = 000000
AMTYP4 = 000000
APASS = 000000
APRIOR = 000000
APTCSU = 000040
APTENV = 000001
APTSIZ = 000200
APTSPO = 000100
ASWREG = 000000
ATESTN = 000000
AUNIT = 000000
AUSWR = 000000

AUTO.S 012254
AVECT = 000300
AVECT1 = 000000
AVECT2 = 000000
BAUD 025510
BCOUNT 024110
BINWRD 007110
BIT0 = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1 = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2 = 000004
BIT3 = 000010
BIT4 = 000020
BIT5 = 000040
BIT6 = 000100
BIT7 = 000200
BIT8 = 000400
BIT9 = 001000
BPTVEC = 000014
BRK0 = 000400
BRK1 = 001000
BRK2 = 002000
BRK3 = 004000
BRK4 = 010000
BRK5 = 020000
BRK6 = 040000
BRK7 = 100000
BRW 005372
BUILD 025002
BYTCNT 027452
CCOUNT 024106
CHRCNT 007106
CLKSRV 024100
CNVRT = 104412
CONVRT = 104411
CORMAX 032504
COUNT0 024660
COUNT1 024662

C00 = 000400
C01 = 001000
C02 = 002000
C03 = 004000
C04 = 010000
C05 = 020000
C06 = 040000
C07 = 100000
CR = 000015
CRLF = 000200
CSRMAP 012262
CYCLE 011532
DATABP 007502
DATAHD 007470
DCLASM = 104417
DCLR = 000020
DCOUNT 024112
DDISP = 177570
DELAY = 104414
DEVADR 006632
DEVICE = 104413
DH1 032073
DH2 032116
DH3 032151
DH4 032163
DH5 032212
DH6 032271
DISPLA 001162
DISPRE 000174
DLYCNT 007204
DLYTBL 032444
DONFLG 001420
DSWR = 177570
DTR0 = 000400
DTR1 = 001000
DTR2 = 002000
DTR3 = 004000
DTR4 = 010000
DTR5 = 020000
DTR6 = 040000
DTR7 = 100000
DT1 032324
DT2 032336
DT3 032354
DT4 032362
DT5 032400
DT6 032422
DVALID = 100000
DZACTV 001404
DZCR0 001500
DZCR1 001514
DZCR10 001640
DZCR11 001654

DZCR12 001670
DZCR13 001704
DZCR14 001720
DZCR15 001734
DZCR16 001750
DZCR17 001764
DZCR2 001530
DZCR3 001544
DZCR4 001560
DZCR5 001574
DZCR6 001610
DZCR7 001624
DZCSR 002042
DZLEV 030126
DZLPR 002052
DZLV0 001504
DZLV1 001520
DZLV10 001644
DZLV11 001660
DZLV12 001674
DZLV13 001710
DZLV14 001724
DZLV15 001740
DZLV16 001754
DZLV17 001770
DZLV2 001534
DZLV3 001550
DZLV4 001564
DZLV5 001600
DZLV6 001614
DZLV7 001630
DZMSR 002062
DZNUM 001410
DZPRT 030334
DZRBUF 002046
DZRIS 002074
DZRIV 002072
DZTCR 002056
DZTDR 002066
DZTIS 002100
DZTIV 002076
DZVC0 001502
DZVC1 001516
DZVC10 001642
DZVC11 001656
DZVC12 001672
DZVC13 001706
DZVC14 001722
DZVC15 001736
DZVC16 001752
DZVC17 001766
DZVC2 001532
DZVC3 001546

DZVC4 001562
DZVC5 001576
DZVC6 001612
DZVC7 001626
DZ.END 002000
DZ.MAP 001500
ECOUNT 024114
EIAFLG 001414
EIGHT = 000030
EIGHTS = 000070
EMTVEC = 000030
EM1 030566
EM10 031203
EM13 031242
EM14 031273
EM15 031325
EM16 031367
EM17 031440
EM2 030641
EM20 031476
EM21 031537
EM22 031567
EM23 031631
EM24 031661
EM25 031707
EM26 031737
EM27 031766
EM3 030667
EM30 032036
EM4 030726
EM5 030755
EM6 031004
EM7 031042
EM8 031103
EM9 031145
ENDTTB = 024212
ERRMSG 007456
ERRVEC = 000004
ERTAB0 007626
EVEPAR = 000000
EXITER 007562
FINI 022434
FIVE = 000000
FIVES = 000040
FRMERR = 020000
HALTS 007506
HDRFLG 001416
HDZCSR 002044
HDZLPR 002054
HDZMSR 002064
HDZRB0 002050
HDZTCR 002060
HDZTDR 002070

HILIM 006630	MANT12 001702	PARAM = 104405	RES05 = 104410	SWREG = 000176
HT = 000011	MANT13 001716	PARAM1 006476	RIE = 000100	SW0 = 000001
INBUF 011262	MANT14 001732	PARER = 010000	RING0 = 000001	SW00 = 000001
INIFLG 001415	MANT15 001746	PARERR 006552	RING1 = 000002	SW01 = 000002
INIT 021714	MANT16 001762	PARESE 024506	RING2 = 000004	SW02 = 000004
INIT1 021740	MANT17 001776	PARITY= 000100	RING3 = 000010	SW03 = 000010
INSTER= 104404	MANT2 001542	PARMD = 104415	RING4 = 000020	SW04 = 000020
INSTR = 104403	MANT3 001556	PAR0 001510	RING5 = 000040	SW05 = 000040
INSTR2 006430	MANT4 001572	PAR1 001524	RING6 = 000100	SW06 = 000100
INTRAN 026732	MANT5 001606	PAR10 001650	RING7 = 000200	SW07 = 000200
INTREC 026552	MANT6 001622	PAR11 001664	RL0 = 000000	SW08 = 000400
INTSVC 026116	MANT7 001636	PAR12 001700	RL1 = 000400	SW09 = 001000
IOTVEC= 000020	MASTEK 010756	PAR13 001714	RL2 = 001000	SW1 = 000002
LAST 027444	MBADLN 011065	PAR14 001730	RL3 = 001400	SW10 = 002000
LESS1 030336	MCABLE 027773	PAR15 001744	RL4 = 002000	SW11 = 004000
LF = 000012	MCHAR 027640	PAR16 001760	RL5 = 002400	SW12 = 010000
LIMITS 006556	MCSRX 010706	PAR17 001774	RL6 = 003000	SW13 = 020000
LINE 001364	MDATA 011366	PAR2 001540	RL7 = 003400	SW14 = 040000
LINESP 027454	MEPASS 010525	PAR3 001554	RSTART 021704	SW15 = 100000
LINEX 025526	MERRPC 011033	PAR4 001570	RSTRT 026766	SW2 = 000004
LINE0 001506	MERRX 010733	PAR5 001604	RUN 001406	SW3 = 000010
LINE1 001522	MERR2 010564	PAR6 001620	RXISR1 025030	SW4 = 000020
LINE10 001646	MERR3 010633	PAR7 001634	RXSVC 022204	SW5 = 000040
LINE11 001662	MINVAL 027564	PAWCH = 104416	RXTCR 022462	SW6 = 000100
LINE12 001676	MLINE 027612	PIRQ = 177772	R6 = 000006	SW7 = 000200
LINE13 001712	MLOCK 010657	PIRQVE= 000240	R7 = 000007	SW8 = 000400
LINE14 001726	MNEW 010761	POPRO = 012600	SAVLIN 001372	SW9 = 001000
LINE15 001742	MNTFLG 001417	POP1SP= 005726	SAVNUM 001411	S110 = 001000
LINE16 001756	MODE 001370	POP2SP= 022626	SAVPC 001402	S1200 = 003400
LINE17 001772	MPASS 027550	PRI0 027464	SAV05 = 104407	S134 = 001400
LINE2 001536	MPASSX 010722	PRO = 000000	SCOP1 = 104401	S150 = 002000
LINE3 001552	MPFAIL 010462	PR1 = 000040	SERV.G 007642	S1800 = 004000
LINE4 001566	MQUICK 030010	PR2 = 000100	SET 027242	S19200= 007400
LINE5 001602	MR 010550	PR3 = 000140	SETAPT 012102	S2000 = 004400
LINE6 001616	MREGAD 027514	PR4 = 000200	SETFLG= 104406	S2400 = 005000
LINE7 001632	MSENAB= 000040	PR5 = 000240	SET.PS 011430	S300 = 002400
LOBITS 006634	MSPEED 027622	PR6 = 000300	SET1 027274	S3600 = 005400
LOCK 001362	MTERM 027746	PR7 = 000340	SEVEN = 000020	S4800 = 006000
LOCKUP 027442	MTITLE 001000	PS = 177776	SEVENS= 000060	S50 = 000000
LOLIM 006626	MTSTN 010744	PSW = 177776	SIL0AL= 020000	S600 = 003000
LP0 = 000000	MVECTO 027472	PUSHRO= 010046	SIL0EN= 010000	S7200 = 006400
LP1 = 000001	MVECX 010714	PUSH1S= 005746	SIX = 000010	S75 = 000400
LP2 = 000002	MWHICH 027700	PUSH2S= 024646	SIXS = 000050	S9600 = 007000
LP3 = 000003	NEXT 001360	PWRVEC= 000024	SNAP 022034	TABLE2 027336
LP4 = 000004	NUMLIN 027460	QUITS 026544	SPACNT 007107	TBITVE= 000014
LP5 = 000005	NUMTCR 027462	RCVON = 010000	SPEED 027456	TBUF 027470
LP6 = 000006	ODDPAR= 000200	RDATA 027450	SPIN 026512	TCCONT 024212
LP7 = 000007	OFFSET 025242	RDONE = 000200	STACK = 001120	TCR0 = 000001
MAINT = 000010	ONESTO= 000000	RECDAT 027466	STFLG 027440	TCR1 = 000002
MANT0 001512	OTHER 025470	REGIST 001400	STKLMT= 177774	TCR2 = 000004
MANT1 001526	OUT 022362	RESREG 007504	STOP 001462	TCR3 = 000010
MANT10 001652	OVRRUN= 040000	RESTAR 012076	SV05 006644	TCR4 = 000020
MANT11 001666	PAR 001366	RESVEC= 000010	SWR 001160	TCR5 = 000040

TCR6 = 000100	TST21 015374	XHEAD 011040	\$E = 000041	\$NWTST = 000000
TCR7 = 000200	TST22 015732	XMTCNT 001376	\$ENDAD 005056	\$OVER 005336
TDATA 027446	TST23 016260	XMTLIN 001374	\$ENDCT 005042	\$PASS 001242
TD0 001422	TST24 016612	XMTSRV 024664	\$ENV 001254	\$PASTM 001470
TD1 001424	TST25 017070	XPASS 005106	\$ENVM 001255	\$PWRAD 010450
TD2 001426	TST26 017400	XSTART 025244	\$EOP 004712	\$PWRDN 010310
TD3 001430	TST27 017726	XSTATQ 011130	\$EOPCT 005034	\$PWRMG 010444
TD4 001432	TST3 013312	XTCRO 027250	\$ERFLG 001123	\$PWRUP 010362
TD5 001434	TST30 020360	XTCR1 027264	\$ERMAX 001135	\$QUES 001230
TD6 001436	TST31 020546	XTSTN 007634	\$ERROR 007220	\$REGAD 001200
TD7 001440	TST32 021274	XVEC 005100	\$ERRPC 001136	\$REG0 001202
TEIGHT 002140	TST33 021656	XX = 160210	\$ERRTB 001360	\$REG1 001204
TEMP 011324	TST34 022464	YY = 000500	\$ERTTL 001132	\$REG2 001206
TEST1 025656	TST35 023174	ZZ = 000020	\$ETABL 001254	\$REG3 001210
TEST2 026314	TST36 023446	\$APTHD 001462	\$ETEND 001360	\$REG4 001212
TFIVE 002146	TST37 023720	\$ATYC 006020	\$FATAL 001236	\$REG5 001214
TIE = 040000	TST4 013404	\$ATY1 005774	\$FFLG 006240	\$RTNAD 005070
TKVEC = 000060	TST5 013476	\$ATY3 006002	\$FILLC 001176	\$SAVR6 010460
TLAST = 023720	TST6 013570	\$ATY4 006012	\$FILLS 001175	\$SCOPE 005122
TL0 = 000000	TST7 013662	\$AUTOB 001154	\$GDADR 001140	\$SETUP = 000000
TL1 = 000400	TTABLE 024116	\$BASE 001310	\$GDDAT 001144	\$SVLAD 005320
TL2 = 001000	TTST 005140	\$BDADR 001142	\$GET42 005046	\$SVPC = 000040
TL3 = 001400	TWOSTO = 000040	\$BDDAT 001146	\$HD = 000001	\$SWR = 164000
TL4 = 002000	TXSVC 022106	\$CDW1 001314	\$HIBTS 001462	\$SWREG 001256
TL5 = 002400	TYPDAT 007472	\$CDW2 001316	\$ICNT 001124	\$SWRMK = 000000
TL6 = 003000	TYPE = 104402	\$CHARC 005770	\$ILLUP 010454	\$TESTN 001240
TL7 = 003400	TYPMSG 007362	\$CMTAG 001120	\$INTAG 001155	\$TIMES 001226
TMTBL 002102	T110 002106	\$CM1 = 000006	\$ITEMB 001134	\$TKB 001166
TPVEC = 000064	T1200 002120	\$CM2 = 000014	\$LF 001232	\$TKS 001164
TRAPVE = 000034	T134 002110	\$CM3 = 000006	\$LFLG 006237	\$TMP0 001216
TRDY = 100000	T150 002112	\$CM4 = 000004	\$LPADR 001126	\$TMP1 001220
TRTVEC = 000014	T1800 002122	\$CPUOP 001262	\$LPERR 001130	\$TMP2 001222
TR0 001442	T2000 002124	\$CRAP = 177777	\$MADR1 001266	\$TMP3 001224
TR1 001444	T2400 002126	\$CRLF 001231	\$MADR2 001272	\$TN = 000040
TR2 001446	T300 002114	\$DDW0 001320	\$MADR3 001276	\$TPB 001172
TR3 001450	T3600 002130	\$DDW1 001322	\$MADR4 001302	\$TPFLG 001177
TR4 001452	T4800 002132	\$DDW10 001344	\$MAIL 001234	\$TPS 001170
TR5 001454	T50 002102	\$DDW11 001346	\$MAMS1 001264	\$TSTM 001466
TR6 001456	T600 002116	\$DDW12 001350	\$MAMS2 001270	\$TSTNM 001122
TR7 001460	T7200 002134	\$DDW13 001352	\$MAMS3 001274	\$TYPE 005440
TSEVEN 002142	T75 002104	\$DDW14 001354	\$MAMS4 001300	\$TYPEC 005652
TSIX 002144	T9600 002136	\$DDW15 001356	\$MBADR 001464	\$TYPEX 005772
TST1 013036	VECMAP 012644	\$DDW2 001324	\$MFLG 006236	\$UNIT 001246
TST10 013754	VEC1 025306	\$DDW3 001326	\$MSGAD 001250	\$UNITM 001472
TST11 014112	VEC2 025316	\$DDW4 001330	\$MSGLG 001252	\$USWR 001260
TST12 014274	WCHFLG 027436	\$DDW5 001332	\$MSGTY 001234	\$VECT1 001304
TST13 014412	WRDCNT 007104	\$DDW6 001334	\$MTYP1 001265	\$VECT2 001306
TST14 014542	WTBS.F 007460	\$DDW7 001336	\$MTYP2 001271	\$XOFF = 000023
TST15 014632	XBEGIN 025556	\$DDW8 001340	\$MTYP3 001275	\$XON = 000021
TST16 014716	XBX 007250	\$DDW9 001342	\$MTYP4 001301	\$XTSTR 005176
TST17 015112	XCSR 005072	\$DEVCT 001244	\$MXCNT 005374	\$Y = 000020
TST2 013226	XEOP 026752	\$DEVN 001312	\$N = 000037	\$GET4 = 000000
TST20 015250	XERR 005114	\$DOAGN 005066	\$NULL 001174	. = 001514

.ADVAN	007206	.DELAY	007166	.INST1	006262	.RES05	006676	.START	002150
.BEGIN	004624	.DEVIC	007134	.MSG	006264	.SAV05	006636	.TRPSR	007112
.CNVRT	006734	.ERRTA	030340	.PARAM	006436	.SCOPE	005122	.TRPTA	002002
.CONVR	006730	.INSTE	006416	.PARMD	026772	.SCOP1	005376	.TYPE	005422
.DCLAS	007154	.INSTR	006242	.PAWCH	027166	.SETFL	011142	.#X	= 001462

. ABS. 032504 000

ERRORS DETECTED: 0

CZDZAI.BIN,CZDZAI.SEQ=SYSMAC.SML,CZDZAI.P11
RUN-TIME: 20.23 .7 SECONDS
RUN-TIME RATIO: 64/45=1.4
CORE USED: 52K (103 PAGES)