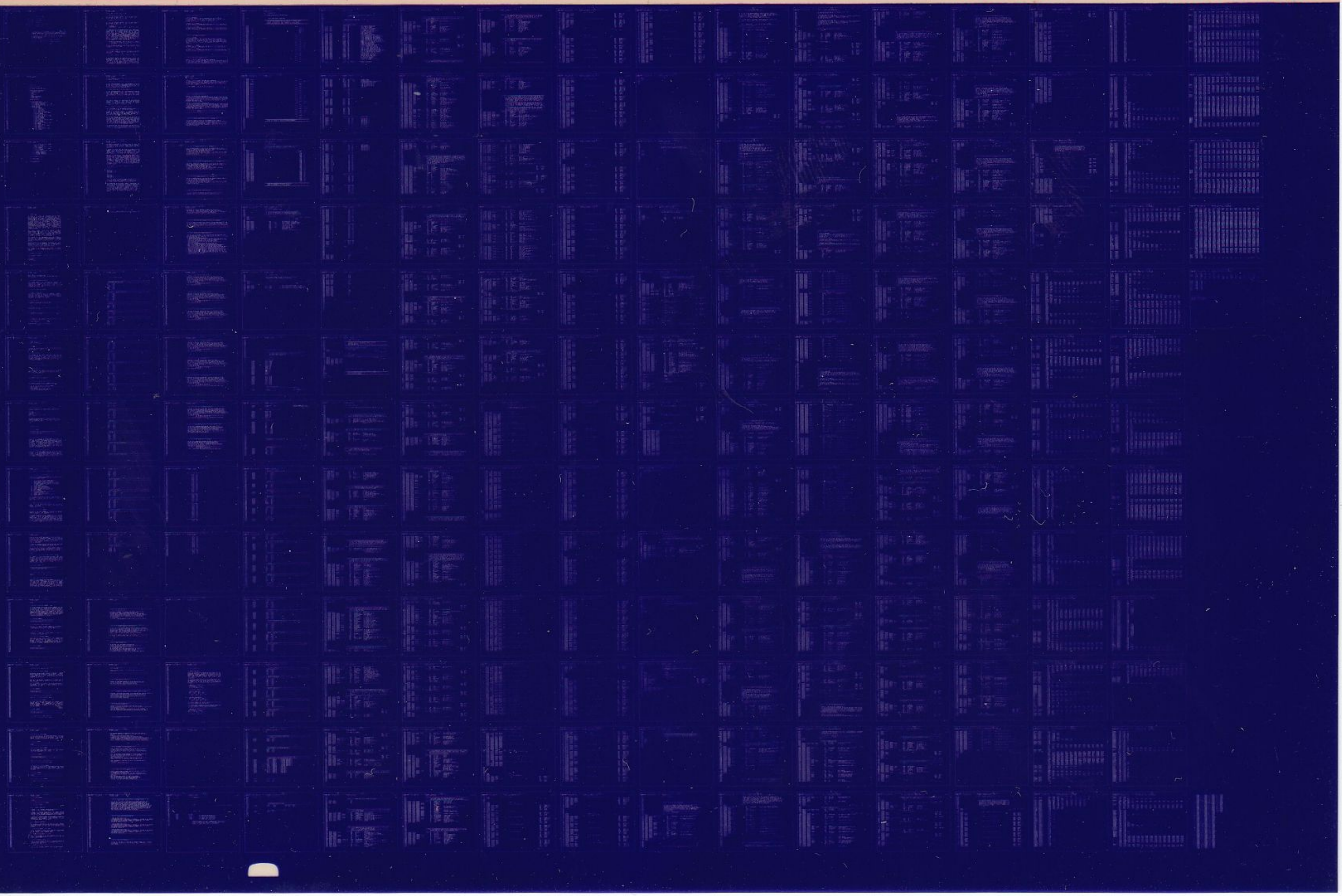


M8203, LUNT

M8203 STATIC DIAG. #2
CZDMSA0

AH-E235A-MC
COPYRIGHT © 1979
FICHE 1 OF 1

SEP 1979
digital
MADE IN USA



3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400

.REM @

IDENTIFICATION

PRODUCT CODE: AC-E234A-MC
PRODUCT NAME: CZDMSA0 M8203 STATIC DIAG #2
PRODUCT DATE: JUNE 1979
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: DAVID 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 IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

CONTENTS

3334	
3335	
3336	
3337	
3338	
3339	
3340	
3341	1.0 INTRODUCTION
3342	
3343	2.0 HARDWARE REQUIREMENTS
3344	
3345	3.0 PRELIMINARY PROGRAM REQUIREMENTS
3346	
3347	4.0 GENERAL PROGRAM CONSIDERATIONS
3348	4.1 DIAGNOSTIC SUPERVISOR
3349	4.2 EXECUTION TIME
3350	4.3 XXDP+
3351	4.4 ACT/SLIDE
3352	4.5 APT
3353	4.6 MEMORY MANAGEMENT
3354	4.7 MEMORY PARITY OPTION
3355	4.8 ERROR LOGGING
3356	
3357	5.0 PROGRAM LOAD MEDIA
3358	
3359	6.0 OPERATING INSTRUCTIONS
3360	6.1 LOADING AND STARTING PROCEDURES
3361	6.1.1 LOADING PROCEDURES
3362	6.1.2 STARTING PROCEDURES
3363	6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION
3364	6.2 INITIAL DIALOGUE
3365	6.3 PROGRAM OPTIONS
3366	6.3.1 START COMMAND
3367	6.3.1.1 TESTS SWITCH
3368	6.3.1.2 PASS SWITCH
3369	6.3.1.3 FLAGS SWITCH
3370	6.3.1.4 END OF PASS SWITCH
3371	6.3.1.5 EFFECT OF START COMMAND
3372	6.3.2 RESTART COMMAND
3373	6.3.2.1 TESTS, PASS, AND FLAG SWITCHES
3374	6.3.2.2 UNITS SWITCH
3375	6.3.2.3 EFFECT OF RESTART COMMAND
3376	6.3.3 CONTINUE COMMAND
3377	6.3.3.1 PASS SWITCH
3378	6.3.3.2 FLAGS SWITCH
3379	6.3.3.3 EFFECT OF CONTINUE COMMAND
3380	6.3.4 PROCEED COMMAND
3381	6.3.4.1 FLAGS SWITCH
3382	6.3.4.2 EFFECT OF PROCEED COMMAND
3383	6.3.5 ADD COMMAND
3384	6.3.5.1 UNITS SWITCH
3385	6.3.5.2 EFFECT OF ADD COMMAND
3386	6.3.6 DROP COMMAND
3387	6.3.6.1 UNITS SWITCH
3388	6.3.6.2 EFFECT OF DROP COMMAND
3389	6.3.7 PRINT COMMAND

3390	6.3.7.1 EFFECT OF PRINT COMMAND
3391	6.3.8 DISPLAY COMMAND
3392	6.3.8.1 UNITS SWITCH
3393	6.3.8.2 EFFECT OF DISPLAY COMMAND
3394	6.3.9 FLAGS COMMAND
3395	6.3.9.1 EFFECT OF FLAGS COMMAND
3396	6.3.10 ZFLAGS COMMAND
3397	6.3.10.1 EFFECT OF ZFLAGS COMMAND
3398	6.3.11 CONTROL CHARACTERS
3399	6.3.12 HARDWARE PARAMETERS
3400	6.3.13 SOFTWARE PARAMETERS
3401	6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE
3402	
3403	7.0 DEVICE INFORMATION TABLES
3404	
3405	8.0 TEST DESCRIPTIONS
3406	8.1 DATA PATTERNS USED
3407	
3408	9.0 ERROR INFORMATION
3409	9.1 ERROR REPORTING

3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466

1.0 INTRODUCTION

THE M8203 IS A SINGLE-LINE SYNCHRONOUS LINE UNIT MODULE WHICH SUPPORTS BOTH CHARACTER ?-ORIENTED (DDCMP, BSC, ETC.) AND BIT-ORIENTED (SDLC, HDLC, ETC.) PROTOCOLS, AND WHICH IS CURRENTLY EMPLOYED IN THE DMP-11 DDCMP MULTIDROP PROJECT. THE PURPOSE OF THIS PROGRAM IS TO PERFORM DIAGNOSTIC TESTING OF ALL M8203 LOGIC IN A RELATIVELY STATIC MANNER. THE FOLLOWING FUNCTIONS WILL BE PERFORMED: LINE UNIT REGISTER ADDRESSING, USYRT ADDRESSING, STATIC BIT INTERACTION AND READ/WRITE LOGIC TESTS, BASIC TRANSMITTER AND RECEIVER SEQUENCING AND DATA BUFFERING AND STATIC OPERATIONS IN CHARACTER AND BIT-STUFFING MODES. IN ADDITION DATA MESSAGES WILL BE SENT AT SPEEDS OF 2400 BAUD TO 1 MEGABAUD, WITH LOOPBACK IN THE USYRT, ON THE LINE UNIT AT TTL LEVEL, OR THROUGH AN EXTERNAL TEST CONNECTOR WITH A SPECIFIC MODEM INTERFACE SELECTED.

THE STATIC LOGIC TESTS WILL PROVIDE EXTENSIVE TROUBLESHOOTING CAPABILITIES, SUCH AS TIGHT SCOPE LOOPS, SWITCH OPTIONS, AND ABILITY TO 'LOCK' ONTO INTERMITTENT ERRORS. IN ADDITION TESTS WILL BE DESIGNED AND STRUCTURED TO ACHIEVE MAXIMUM FAULT RESOLUTION AND FACILITATE REPLACEMENT OF THE SMALLEST FIELD REPLACEABLE UNIT.

THIS PROGRAM WILL BE IMPLEMENTED USING THE DIAGNOSTIC SUPERVISOR AND A STRUCTURED PROGRAMMING APPROACH. BECAUSE THE DESIGN WILL CONFORM TO THE SUPERVISOR (STANDALONE VERSION) THE PROGRAM WILL BE COMPATIBLE WITH ACT, APT, XXDP+, AND SLIDE.

THROUGH DIALOGUE WITH THE OPERATOR, THE PROGRAM WILL ALLOW MODIFICATION OF DEVICE PARAMETERS, SUCH AS UNIBUS ADDRESS, VECTOR ADDRESSES AND DEVICE PRIORITY. IN ADDITION, THE OPERATOR CAN SPECIFY PARTICULAR TESTS TO BE RUN AND A VARIETY OF LOOPING, RUNNING, AND REPORTING MODES.

DEVICE ERRORS WILL BE REPORTED AS THEY OCCUR. THE REPORT WILL INCLUDE A TEST NUMBER AND DESCRIPTION OF THE ERROR, GOOD AND BAD TEST DATA, AND APPLICABLE DEVICE REGISTER CONTENTS.

2.0 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS:

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

3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522

DMC-11 OR KMC-11 MICROPROCESSOR
M8203 LINE UNIT AND BC08S-1 CABLE AND BERG CONNECTORS
H3254 AND H3255 TEST CONNECTORS (IF NOT PRESENT, SOME TESTS
WILL BE SKIPPED)

3.0 PRELIMINARY PROGRAM REQUIREMENTS

THIS PROGRAM OPERATES THE MICROPROCESSOR EXTENSIVELY IN
ORDER TO TEST THE LINE INIT. FOR THIS REASON, THE
MICROPROCESSOR DIAGNOSTIC AND SUBSYSTEM FUNCTIONAL TESTS
SHOULD BE RUN FIRST, AND ANY FAULTS FOUND IN THE
MICROPROCESSOR MODULE SHOULD BE REPAIRED, PRIOR TO RUNNING
THE M8203 STATIC LOGIC TESTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 DIAGNOSTIC SUPERVISOR

THIS PROGRAM IS COMPATIBLE WITH THE STANDALONE DIAGNOSTIC
SUPERVISOR, AND MUST BE LOADED TO BE CO-RESIDENT WITH THE
SUPERVISOR, OR BE PREVIOUSLY COMBINED WITH THE SUPERVISOR
AND LOADED AS A SINGLE FILE. IN EITHER CASE, THE COMBINED
PROGRAM WILL NOT EXCEED 16K OF MEMORY.

4.2 EXECUTION TIME

THE MAXIMUM TIME REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS
IS ABOUT 45 SECONDS PER PASS FOR EACH UNIT.

4.3 XXDP+

THIS PROGRAM MAY BE LOADED UNDER XXDP+, AND MAY BE RUN IN
DUMP MODE OR CHAIN MODE.

4.4 ACT/SLIDE

THIS PROGRAM MAY BE LOADED UNDER ACT OR SLIDE AND MAY BE RUN
IN DUMP MODE OR CHAIN MODE.

4.5 APT

THIS PROGRAM MAY BE LOADED BY THE APT SYSTEM (INCLUDING
APT-RD) AND RUN IN PROGRAM MODE OR SCRIPT MODE.

4.6 MEMORY MANAGEMENT

MEMORY MANAGEMENT IS NOT UTILIZED IN THIS PROGRAM. IF IT IS
INSTALLED, IT IS DISABLED BY THE PROGRAM.

3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578

4.7 MEMORY PARITY OPTION

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

4.8 ERROR LOGGING

AT THE END OF EACH PASS ON ALL UNITS, THE PROGRAM PRINTS OUT THE CUMULATIVE TOTAL NUMBER OF ERRORS SINCE THE LAST START OR RESTART COMMAND.

5.0 PROGRAM LOAD MEDIA

THIS PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM ACT, SLIDE, OR APT SYSTEMS, OR FROM ANY MEDIA SUPPORTED BY XXDP+. WHEN USING THE PAPER TAPE ABSOLUTE LOADER, THE PROGRAM SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC SUPERVISOR. WHEN USING XXDP+, THE DIAGNOSTIC SUPERVISOR SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC PROGRAM.

6.0 OPERATING INSTRUCTIONS

6.1 LOADING AND STARTING PROCEDURES

6.1.1 LOADING PROCEDURES

THIS PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER. IT MAY ALSO BE LOADED FROM ANY XXDP+ LOAD MEDIA. WHEN LOADED UNDER XXDP+, THE DIAGNOSTIC SUPERVISOR WILL BE LOADED AUTOMATICALLY.

6.1.2 STARTING PROCEDURES

THE PROGRAM STARTS AT LOCATION 200. USE STANDARD DEC PROCEDURES TO START THE PROGRAM.

6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION

THE DIAGNOSTIC CAN BE EXECUTED STANDALONE UNDER XXDP+, WITHOUT READING THE REMAINDER OF THIS DOCUMENT, AS FOLLOWS:

- A) LOAD AND START DIAGNOSTIC USING RUN COMMAND
- B) RECEIVE DIAGNOSTIC SUPERVISOR IDENTIFICATION AND PROMPT (DRS-C>)
- C) ENTER STA<CR>
- D) ANSWER HARDWARE AND SOFTWARE QUESTIONS
- E) GET END OF PASS MESSAGES OR ERROR MESSAGES
- F) TO END EXECUTION, ENTER CONTROL/C

3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634

6.2 INITIAL DIALOGUE

AFTER THE PROGRAM AND THE SUPERVISOR ARE LOADED AND THE PROGRAM IS STARTED, THE FOLLOWING IDENTIFICATION IS TYPED :

```
DRS LOADED  
DIAG. RUN-TIME SERVICES  
CZDMS-A-0  
M8203 STATIC LOGIC TESTS - PART 2 OF 2  
UNIT IS M8203  
DR>
```

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

6.3 PROGRAM OPTIONS

6.3.1 START COMMAND

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

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

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

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

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

3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690

SET. THE EXIT IS A RETURN TO COMMAND MODE. SEE EXAMPLE AT
END OF 6.3.1.5.

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

<FLAG-LIST> IS A SEQUENCE OF ELEMENTS OF THE FORM <FLAG>,
<FLAG=1>, OR <FLAG=0>, SEPARATED BY COLONS, WHERE <FLAG> HAS
ONE OF THE FOLLOWING VALUES:

HOE	HALT ON ERROR, CAUSING COMMAND MODE TO BE ENTERED WHEN AN ERROR IS ENCOUNTERED
LOE	LOOP ON ERROR, CAUSING THE DIAGNOSTIC TO LOOP CONTINUOUSLY WITHIN THE SMALLEST DEFINED BLOCK OF CODING (SEGMENT, SUBTEST, OR TEST) CONTAIN- ING THE ERROR
IER	INHIBIT ERROR REPORTING
IBE	INHIBIT BASIC ERROR REPORTS
IXE	INHIBIT EXTENDED ERROR REPORTS
PRI	DIRECT ALL MESSAGES TO A LINE PRINTER
PNT	PRINT NUMBER OF TEST BEING EXECUTED
BOE	BELL ON ERROR
UAM	RUN IN UNATTENDED MODE, BYPASSING MANUAL INTERVENTION TESTS
ISR	INHIBIT STATISTICAL REPORTS
IDU	INHIBIT DROPPING OF UNITS BY DIAGNOSTIC
LOT	LOOP ON TEST

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

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

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

6.3.1.5 EFFECT OF START COMMAND

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

THE HARDWARE PARAMETER DIALOGUE COMMENCES WITH THE QUESTION
"# UNITS?" TO WHICH THE OPERATOR REPLIES WITH A DECIMAL
NUMBER N FROM 1 TO 16. THE TERM 'UNIT' REFERS TO THE DEVICE
TO WHICH THIS SERIES OF DIAGNOSTICS IS DEDICATED. FOLLOWING
THIS ARE THE QUESTIONS WHEREBY THE P-TABLES THEMSELVES WILL
BE BUILT. EACH P-TABLE IS A CORE-RESIDENT TABLE CONTAINING
ALL THE HARDWARE INFORMATION FOR ONE UNIT. THE OPERATOR
MUST SUPPLY N (NUMBER OF UNITS) VALUES FOR EACH QUESTION.

3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746

HE MAY DO THIS BY GIVING ONE ANSWER TO EACH QUESTION (IN WHICH CASE THE SERIES OF QUESTIONS WILL BE POSED N TIMES) OR BY GIVING N VALUES, SEPARATED BY COMMAS, TO EACH QUESTION (SERIES WILL BE POSED ONCE). EACH QUESTION IS FOLLOWED BY THE RESPONSE RADIX (D FOR DECIMAL, B FOR BINARY, O FOR OCTAL, L FOR YES/NO) IN PARENTHESES AND THE DEFAULT VALUE AFTER THE PARENTHESES.

FOLLOWING THE HARDWARE QUESTIONS ARE THE SOFTWARE QUESTIONS TO BUILD THE SOFTWARE TABLES, WHICH DEFINE THE MODE (QUICK VERIFY ETC.) THAT THE DIAGNOSTIC WILL EXECUTE IN.

WHEN THE QUESTION '# UNITS?' IS ANSWERED, MEMORY STORAGE IS ALLOCATED FOR THE P-TABLES, AND IF THERE IS NOT ENOUGH TO ACCOMMODATE THEM THE MESSAGE 'TOO MANY UNITS' IS ISSUED. IN THIS CASE THE DIAGNOSTIC MUST BE EXECUTED MORE THAN ONCE TO TEST ALL UNITS.

EXAMPLE:

STA/TESTS:1:2-4:6:8-10/PASS:3/FLAGS:IER:HOE=1:UAM:LOE

THIS COMMAND WILL CAUSE THREE PASSES TO BE MADE, EACH PASS CONSISTING OF TESTS 1,2,3,4,6,8,9, AND 10 EXECUTED AGAINST ALL UNITS. THERE IS NO DIFFERENCE BETWEEN SAYING <FLAG> AND SAYING <FLAG=1>. THE NOTATION <FLAG=0> IS MEANINGFUL ONLY ON A COMMAND OTHER THAN START TO CLEAR A FLAG THAT WAS PREVIOUSLY SET. NOTE THAT ON ALL COMMANDS ONLY THE FIRST THREE LETTERS ARE SCANNED.

6.3.2 RESTART COMMAND

```
*****  
RES(TART)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:  
  <FLAG-LIST>/UNITS:<UNIT-LIST>  
*****
```

6.3.2.1 TESTS, PASS, AND FLAGS SWITCHES

<TEST-LIST>, <PASS-CNT>, AND <FLAG-LIST> ARE AS IN THE START COMMAND.

6.3.2.2 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (0,1 ETC.) OR RANGES OF DECIMAL NUMBERS (0-5, 8-10 ETC.) THAT SPECIFY THE UNITS TO BE TESTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS MAY RANGE FROM 0 THRU N-1 (N IS THE NUMBER OF UNITS SPECIFIED IN THE PREVIOUS START COMMAND). THE NUMBER INDICATES THE POSITION OF THE P-TABLE AS THE DATA WAS ENTERED DURING THE HARDWARE DIAGLOGUE. THE UNITS WHICH ARE SELECTED MUST NOT HAVE BEEN DROPPED BY THE DROP COMMAND. SEE THE DISCUSSION OF ADD AND DROP COMMANDS BELOW. DEFAULT

3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802

IS TO TEST ALL UNITS WHICH HAVE NOT BEEN DROPPED BY A DROP
COMMAND.

6.3.2.3 EFFECT OF RESTART COMMAND

THE RESTART COMMAND DIFFERS FROM THE START COMMAND IN THAT
THE P-TABLES FROM THE PREVIOUS START COMMAND (THERE MUST
HAVE BEEN ONE) ARE USED, INSTEAD OF NEW ONES BEING BUILT.
THE UNITS SWITCH GIVES THE ABILITY TO SELECT A SUBSET OF
THESE. THE SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED
(OPERATOR WILL BE ASKED). THE COMMAND CAN BE USED AFTER
COMMAND MODE HAS BEEN REENTERED IN ANY OF THE THREE NORMAL
WAYS: A) THE REQUESTED NUMBER OF PASSES HAVE BEEN MADE B)
AN ERROR WAS ENCOUNTERED WITH THE HALT ON ERROR FLAG SET C)
A CONTROL/C WAS ENTERED BY THE OPERATOR.

6.3.3 CONTINUE COMMAND

CON(TINUE)/PASS:<PASS-CNT/FLAGS:<FLAG-LIST>

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

<PASS-CNT> IS SAME AS IN START COMMAND, BUT THE DEFAULT IS
THE UNSATISFIED PASS-CNT FROM THE PREVIOUS START OR RESTART.
IF NONE REMAINS, THE DEFAULT IS NON-ENDING EXECUTION.

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

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

6.3.3.3 EFFECT OF CONTINUE COMMAND

CONTINUE MUST FOLLOW A START OR RESTART, AND COMMAND MODE
MUST HAVE BEEN ENTERED DUE TO A HALT ON ERROR OR A
CONTROL/C. THE EFFECT OF THE COMMAND IS TO GO TO THE
BEGINNING OF THE TEST THAT WAS BEING EXECUTED WHEN THE HALT
OR CONTROL/C TOOK PLACE. SOFTWARE DIALOGUE MAY OPTIONALLY
BE REEXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

6.3.4 PROCEED COMMAND

PRO(CEED)/FLAGS:<FLAG-LIST>

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

3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858

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

6.3.4.2 EFFECT OF PROCEED COMMAND

PROCEED MUST FOLLOW A START, RESTART, OR CONTINUE. COMMAND
MODE MUST HAVE BEEN ENTERED VIA A HALT ON ERROR. THE EFFECT
OF THE COMMAND IS TO BEGIN EXECUTION AT THE LOCATION
FOLLOWING THE ERROR CALL. NEITHER HARDWARE NOR SOFTWARE
PARAMETERS MAY BE ALTERED.

NOTE THAT IF THE MESSAGE 'TOO MANY UNITS' IS ISSUED, TWO OR
MORE CORE IMAGES MUST BE CREATED (WITH DIFFERENT NAMES) TO
TEST ALL UNITS.

NOTE THAT ALTHOUGH THE CHAINABLE IMAGE CAN BE EXECUTED ON A
16K MACHINE, THE ORIGINAL CCI CREATION MUST BE DONE ON A
LARGE MACHINE, THE EXACT SIZE BEING DEPENDENT ON WHICH
UPDATE UTILITY IS USED.

6.3.5 ADD COMMAND

ADD/UNITS:<UNIT-LIST>

6.3.5.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.5.2 EFFECT OF ADD COMMAND

THE UNITS SPECIFIED ARE ADDED TO THE TEST SEQUENCE. EACH
UNIT MUST HAVE A P-TABLE IN MEMORY DUE TO AN EARLIER
HARDWARE DIALOGUE. THIS COMMAND MUST BE FOLLOWED BY A
RESTART OR CONTINUE. THE UNITS SWITCH MUST BE SPECIFIED.
THE ADD COMMAND IS MEANINGFUL ONLY FOR UNITS THAT WERE
PREVIOUSLY DROPPED.

6.3.6 DROP COMMAND

DRO(P)/UNITS:<UNIT-LIST>

6.3.6.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914

6.3.6.2 EFFECT OF DROP COMMAND

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

6.3.7 PRINT COMMAND

PRI(NT)

6.3.7.1 EFFECT OF PRINT COMMAND

THE TOTAL NUMBER OF ERRORS FOR EACH UNIT SINCE THE LAST START OR RESTART COMMAND ARE PRINTED. THE ISR (INHIBIT STATISTICAL REPORTING) FLAG IS CLEARED.

6.3.8 DISPLAY COMMAND

'**
DIS(PLAY)/UNITS:<UNIT-LIST>

6.3.8.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.8.2 EFFECT OF DISPLAY COMMAND

THE HARDWARE P-TABLES FOR ALL UNITS UNDER TEST ARE PRINTED OUT IN THE FORMAT IN WHICH THEY WERE ENTERED. ANY UNITS THAT WERE DROPPED BY THE OPERATOR 'DROP' COMMAND ARE SO DESIGNATED.

6.3.9 FLAGS COMMAND

FLA(GS)

6.3.9.1 EFFECT OF FLAGS COMMAND

THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.

3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970

6.3.10 ZFLAGS COMMAND

ZFL(AGS)

6.3.10.1 EFFECT OF ZFLAGS COMMAND

ALL FLAGS ARE CLEARED.

6.3.11 CONTROL CHARACTERS

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

A CONTROL Z (Z) ENTERED DURING ONE OF THE THREE OPERATOR DIALOGUES- HARD CORE QUESTIONS (SEE 6.2), HARDWARE DIALOGUE (SEE 6.3.1.5), OR SOFTWARE DIALOGUE (SEE 6.3.1.5) CAUSES THE DEFAULTS TO BE TAKEN FOR THE REMAINDER OF THAT DIALOGUE.

A CONTROL O (O) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES ALL TELETYPE OUTPUT TO BE SURPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR UNTIL ANOTHER O IS TYPED, WHICH RESTORES NORMAL TELETYPE OUTPUT.

6.3.12 HARDWARE PARAMETERS

THE FOLLOWING 8 QUESTIONS WILL BE ASKED ON A START COMMAND. THE VALUE LOCATED TO THE LEFT OF THE QUESTION MARK IS THE DEFAULT VALUE THAT WILL BE TAKEN ON A CARRIAGE RETURN RESPONSE.

1. DEVICE CSR ADDRESS : (O) 160170?

THIS IS THE ADDRESS AT WHICH THE CSR REGISTERS (SELO) RESIDE ON THE UNIBUS. THE ALLOWABLE RANGE IS 160000-177776 (OCTAL), AND THE DEFAULT VALUE IS 160170.

2. DEVICE VECTOR ADDRESS : (O) 300 ?

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

3. DEVICE PRIORITY LEVEL : (O) 5 ?

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

4. MB203 SWITCH PACK #1 (REG 11) : (O) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #1, WHICH RESIDES IN INBUS REG 11. THE ALLOWABLE RANGE IS

3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026

000-053, AND THE DEFAULT VALUE IS 000.

5. M8203 SWITCH PACK #2 (REG 15) : (0) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #2, WHICH RESIDES IN INBUS REG 15. THE ALLOWABLE RANGE IS 000-377, AND THE DEFAULT VALUE IS 000.

6. M8203 SWITCH PACK #3 (REG 16) : (0) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #3, WHICH RESIDES IN INBUS REG 16. THE ALLOWABLE RANGE IS 000-377, AND THE DEFAULT VALUE IS 000.

7. TURNAROUND TYPE -
(0=H3254&H3255, 1=CABLE, 2=MOD LOC, 3=MOD REM, 4 NONE)
: (0) 0 ?

THIS INDICATOR TELLS THE PROGRAM WHETHER TEST CONNECTOR(S) WILL BE MOUNTED SO THAT CERTAIN TESTS CAN BE RUN IN EXTERNAL LOOPBACK MODE. THE ALLOWABLE ANSWERS ARE 0-4, AND THE DEFAULT VALUE IS 0. 0 MEANS THAT THE H3254 AND H3255 TEST CONNECTORS WILL BE USED. 1 MEANS THAT EXTERNAL TURNAROUND WILL BE PROVIDED AT THE FAR END OF A CABLE ATTACHED TO THE J1 OR J2 CONNECTOR. 2 MEANS THAT MODEM LOCAL LOOPBACK WILL BE USED, AND 3 MEANS THAT MODEM REMOTE LOOPBACK WILL BE USED. 4 MEANS THAT NO EXTERNAL TURNAROUND WILL BE PROVIDED. WHEN 0 IS SELECTED, ALL TESTS WILL BE RUN, AND IF 1-4 IS SELECTED, CERTAIN TESTS CANNOT BE RUN, AND THE PROGRAM WILL TYPE THE NUMBER(S) OF TEST(S) TO BE SKIPPED.

8. PLEASE SELECT BAUD RATE; TYPE '0' FOR 2.4K; '1' FOR 4.8K;
'2' FOR 9.6K; '3' FOR 19.2K; '4' FOR 56K; '5' FOR 250K;
'6' FOR 500K; OR '7' FOR 1 MEG BAUD : (0) 4?

THIS IS THE BAUD RATE WHICH IS SELECTED IN THE SWITCH PACK ON THE M8203. THE ALLOWABLE RANGE IS 0-7, AND THE DEFAULT VALUE IS 4 (FOR 56K).

6.3.13 SOFTWARE PARAMETERS

FOUR SOFTWARE PARAMETER QUESTIONS ARE ASKED BY THE M8203 STATIC LOGIC TESTS PROGRAM, PART 2. THESE QUESTIONS ARE THE FOLLOWING:

1. IS MAN. INTERVEN. DESIRED TO MOUNT TEST CONNECTOR(S)
(L) N?

IF THE OPERATOR ANSWERS THE QUESTION WITH Y (YES), THE PROGRAM WILL LATER PAUSE BEFORE TESTING EACH LOGICAL UNIT AND INFORM THE OPERATOR TO INSTALL THE APPROPRIATE TEST CONNECTOR(S) ON THAT UNIT, AND THEN PROCEED TO TEST THAT UNIT. IF THE OPERATOR ANSWERS N (NO) TO THE ABOVE QUESTION, THE PROGRAM WILL PERFORM TESTING ON ALL UNITS WITHOUT ALLOWING MANUAL INTERVENTION BETWEEN UNITS. IN THIS CASE,

4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082

ALL TEST CONNECTOR(S) ON ALL UNITS SHOULD BE INSTALLED PRIOR TO RUNNING THE PROGRAM.

2. SHOULD SWITCH PACK AND AX3-15 PRINTOUT BE ALLOWED (L) N ?

IF THE OPERATOR ANSWERS YES, THE PROGRAM WILL ALLOW THE PRINTOUT OF SWITCH PACKS 1-3 AND MODEM INTERFACE REG AX3-15 ON ANY PASS IN WHICH THE CORRESPONDING TESTS ARE RUN. THE DEFAULT IS NO, WHICH ONLY ALLOWS THE PRINTOUT ON THE FIRST PASS AFTER LOADING, IF THE TESTS ARE RUN.

3. SHOULD SWITCH PACK TESTS BE ALLOWED (L) N ?

IF THE OPERATOR ANSWERS YES, THE PROGRAM WILL ALLOW THE READING AND COMPARISON OF SWITCH PACKS 1-3 TO VALUES ENTERED INTO THE HARDWARE P-TABLE FOR THIS UNIT, IF THE CORRESPONDING TEST IS RUN. IF ALLOWED, SWITCH PACK ERRORS WILL BE REPORTED. THE DEFAULT IS NO, AND THE TESTS ARE NOT RUN.

4. MSG TIMER VALUE (0-177777), 0 = LONGEST TIME-OUT : (0) 0 ?

THIS VALUE CONTROLS THE DURATION OF THE RECEIVER MESSAGE TIME-OUT IN A NUMBER OF TESTS WHICH SEND AND RECEIVE MESSAGES ON THE VARIOUS MODEM INTERFACES WITH EXTERNAL LOOPBACK. THE SMALLER THE VALUE, THE LONGER THE TIME-OUT (UP TO SEVERAL SECONDS). THE DEFAULT IS 0.

6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE

THE FULL CAPABILITY OF THE HARDWARE DIALOGUE IS REVEALED BY THE FOLLOWING DISCUSSION OF WHAT HAPPENS INTERNALLY.

AS SOON AS THE QUESTION '# UNITS?' IS ANSWERED (WITH THE NUMBER N, SAY) SPACE IN CORE IS ALLOCATED FOR N P-TABLES. ALL OF THE P-TABLES ARE OF THE SAME FORMAT, AND THERE IS A ONE-TO ONE CORRESPONDENCE BETWEEN THE HARDWARE PARAMETER QUESTIONS AND THE SLOTS IN THE P-TABLE FORMAT.

ON THE FIRST TRIP THRU THE QUESTIONS, ALL OF THE SLOTS IN ALL OF THE P-TABLES ARE FILLED. IF THE OPERATOR TYPES IN LESS THAN N EXPLICIT VALUES IN RESPONSE TO A PARTICULAR QUESTION, THESE VALUES ARE PLACED IN THE P-TABLES (ONE VALUE GOING INTO THE PROPER SLOT OF EACH P-TABLE BEGINNING WITH THE FIRST P-TABLE) UNTIL THE STRING OF VALUES IS EXHAUSTED. THE LAST VALUE IN THE STRING BECOMES THE NEW DEFAULT AND IS USED TO FILL THAT SLOT IN THE REMAINING P-TABLES.

ON SUBSEQUENT TRIPS THRU THE QUESTIONS, THE SAME PROCESS IS CARRIED OUT, EXCEPT THAT THE EARLIEST P-TABLE NOT TO HAVE RECEIVED AN EXPLICIT VALUE IN ANY OF ITS SLOTS NOW ASSUMES THE ROLE THAT TABLE NUMBER ONE PLAYED IN THE FIRST TRIP.

4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138

THE SERIES OF QUESTIONS IS REISSUED UNTIL AT LEAST ONE QUESTION HAS RECEIVED AN EXPLICIT VALUE FROM THE OPERATOR. IN GIVING A STRING OF VALUES, COMMAS WITHOUT INTERVENING VALUES MAY BE USED TO INDICATE A REPETITION OF THE LAST NAMED VALUE.

A STRING OF VALUES MAY BE GIVEN AS A RANGE (6-10 FOR EXAMPLE). IF THE VALUES REPRESENT PURE NUMERICAL DATA, THIS SAMPLE RANGE TRANSLATES TO THE STRING 6,7,8,9,10 (AN INCREMENT OF 1). IF THE VALUES ARE ADDRESSES, THE SAMPLE RANGE TRANSLATES TO THE STRING 6,8,10 (AN INCREMENT OF 2).

NOW LET US SEE HOW WE COULD USE THESE CAPABILITIES TO CONSTRUCT A SET OF P-TABLES. ASSUME THAT WE HAVE 16 UNITS, AND THAT THERE ARE THREE HARDWARE PARAMETERS FOR EACH (THREE SLOTS IN THE P-TABLE, THREE HARDWARE QUESTIONS IN THE DIALOGUE). LET THE DESIRED VALUE FOR THE FIRST PARAMETER BE THE NUMBER 75 FOR ALL 16 TABLES. LET THE DESIRED VALUE FOR THE SECOND PARAMETER BE EQUAL TO THE UNIT NUMBER (0,1,2,...,15) EXCEPT FOR UNIT 12, WHICH SHOULD RECEIVE THE VALUE 11. LET THE DESIRED VALUE FOR THE THIRD PARAMETER BE THE NUMBER 76 FOR THE FIRST 7 UNITS AND THE NUMBER 77 FOR THE LAST 9 UNITS.

THE FOLLOWING DIALOGUE WOULD ACCOMPLISH THIS GOAL:

UNITS (D) ? 16

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

UNIT 7
<QUESTION 1> ?
<QUESTION 2> ? 7-11,,13-15
<QUESTION 3> ? 77

THE FIRST TIME THE SERIES IS ASKED, SLOT ONE RECEIVES A 75 IN ALL 16 TABLES. SLOT TWO RECEIVES THE VALUES 0,1,2,...,6 IN TABLES 0 THRU 6 AND A CONSTANT 6 IN TABLES 7 THRU 15. SLOT THREE RECEIVES A CONSTANT 76 IN ALL 16 TABLES.

THE SECOND TIME THRU THE SERIES, TABLES 7 THRU THE END ARE GOING TO BE AFFECTED (NOTE THAT THIS PIECE OF INFORMATION IS PRINTED OUT FOR THE OPERATOR IN THE FORM 'UNIT XX' AT THE BEGINNING OF EACH SERIES). QUESTION 1 IS RESPONDED TO BY A <CR>, SO SLOT ONE STAYS AT CONSTANT 75 IN TABLES 7 THRU 15, SINCE NO NEW EXPLICIT VALUES ARE TYPED IN. SLOT TWO GETS THE VALUES 7,8,9,10,11 IN TABLES 7 THRU 11, AND GETS AN 11 IN SLOT 12, AND GETS THE VALUES 13,14,15 IN TABLES 13 THRU 15. SLOT THREE GETS THE VALUE 77 IN TABLES 7 THRU 15.

4139
4140
4141
4142
4143
4144
4145
4146

THE DIALOGUE IS TERMINATED WHEN THE SOFTWARE RECOGNIZES THAT
16 EXPLICIT VALUES HAVE BEEN GIVEN FOR AT LEAST ONE QUESTION
(NAMELY QUESTION 2).

4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203

7.0 DEVICE INFORMATION TABLES

* MAINTENANCE REGISTER - BSEL1

RUN = BIT7
MCLR = BIT6
STEPLU = BIT4
LULoop = BIT3
ROMO = BIT2
ROMI = BIT1
STEPMP = BIT0

* OBUS REG 10 - TRANSMITTER BUFFER

TX7 = BIT7
TX6 = BIT6
TX5 = BIT5
TX4 = BIT4
TX3 = BIT3
TX2 = BIT2
TX1 = BIT1
TX0 = BIT0

* OBUS REG 11

OC = BIT7
GOAH = BIT3
ABORT = BIT2
EOM = BIT1
SOM = BIT0

* OBUS REG 12

IC = BIT7
BPOLL = BIT6
LULP = BIT5

* OBUS REG 13

POLL = BIT7
DTR = BIT6
SELFR = BIT5
HDX = BIT4
MAINT1 = BIT3
MAINT2 = BIT2
SELSBY = BIT1

```
4204
4205
4206      :*****
4207      * OBUS REG 14
4208      :*****
4209      TXEN   = BIT6
4210      DISSI  = BIT5
4211      RDAX   = BIT4
4212      WAX    = BIT3
4213      ENAX   = BIT2
4214      AX2    = BIT1
4215      AX1    = BIT0
4216
4217      :*****
4218      * OBUS REG 17
4219      :*****
4220      CRC2   = BIT7
4221      CRC1   = BIT6
4222      IDLE   = BIT5
4223      SECA   = BIT4
4224      STRIP  = BIT3
4225      RDALL  = BIT2
4226      IERR   = BIT1
4227      DDCMP  = BIT0
4228
4229      :*****
4230      * IBUS REG 10 - RECEIVER BUFFER
4231      :*****
4232      RX7    = BIT7
4233      RX6    = BIT6
4234      RX5    = BIT5
4235      RX4    = BIT4
4236      RX3    = BIT3
4237      RX2    = BIT2
4238      RX1    = BIT1
4239      RX0    = BIT0
4240
4241      :*****
4242      * IBUS REG 11
4243      :*****
4244      OC     = BIT7
4245      OACT   = BIT6
4246      SW3    = BIT5
4247      ORDY   = BIT4
4248      SW2    = BIT3
4249      SW1    = BIT2
4250      SW0    = BIT1
4251      UNRR   = BIT0
4252
4253      :*****
4254      * IBUS REG 12
4255      :*****
4256      IC     = BIT7
4257      IACT   = BIT6
4258      LULP   = BIT5
4259      IRDY   = BIT4
4259      OVR    = BIT3
```

4260 RAB - BIT2
4261 EBLK - BI 1
4262 BCC - BIT0

4263
4264 :*****
4265 * IBUS REG 13

4266 :*****
4267 RING = BIT7
4268 DTR = BIT6
4269 RTS = BIT5
4270 HDX = BIT4
4271 MODR = BIT3
4272 CS = BIT2
4273 STBY = BIT1
4274 CARR = BIT0

4275
4276 :*****
4277 * IBUS REG 14

4278 :*****
4279 READY = BIT7
4280 TXEN - BIT6
4281 DISSI = BIT5
4282 RDAX = BIT4
4283 WAX = BIT3
4284 ENAX = BIT2
4285 AX2 = BIT1
4286 AX1 = BIT0

4287
4288 :*****
4289 * IBUS REG 17

4290 :*****
4291 SGR = BIT7
4292 SIG0 = BIT6
4293 TXDATA - BIT5
4294 OCOR = BIT4
4295 ICIR = BIT3
4296 TESTMD - BIT2
4297 MCLK = BIT1
4298 DDCMP = BIT0

4299
4300 :*****
4301 * AX0-15 - USYRT REG 0 (READ ONLY)

4302 :*****
4303 RX7 - BIT7
4304 RX6 = BIT6
4305 RX5 - BIT5
4306 RX4 = BIT4
4307 RX3 = BIT3
4308 RX2 = BIT2
4309 RX1 = BIT1
4310 RX0 = BIT0

4311
4312 :*****
4313 * AX0-16 - USYRT REG 1 (READ ONLY)

4314 :*****
4315 RERR = BIT7

4316 ASBC2 = BIT6
4317 ASBC1 = BIT5
4318 ASBC0 = BIT4
4319 ROR = BIT3
4320 RABT = BIT2
4321 REOM = BIT1
4322 RSOM = BIT0

4323
4324 ;*****
4325 * AX1-15 - USYRT REG 2

4326 ;*****
4327 TX7 = BIT7
4328 TX6 = BIT6
4329 TX5 = BIT5
4330 TX4 = BIT4
4331 TX3 = BIT3
4332 TX2 = BIT2
4333 TX1 = BIT1
4334 TX0 = BIT0

4335
4336 ;*****
4337 * AX1-16 - USYRT REG 3

4338 ;*****
4339 TERR = BIT7
4340 TXGA = BIT3
4341 TXAB = BIT2
4342 TEOM = BIT1
4343 TSOM = BIT0

4344
4345 ;*****
4346 * AX2-15 - USYRT REG 4

4347 ;*****
4348 SYN7 = BIT7
4349 SYN6 = BIT6
4350 SYN5 = BIT5
4351 SYN4 = BIT4
4352 SYN3 = BIT3
4353 SYN2 = BIT2
4354 SYN1 = BIT1
4355 SYN0 = BIT0
4356 SYNCH = 226

4357
4358 ;*****
4359 * AX2-16 - USYRT REG 5

4360 ;*****
4361 APA = BIT7
4362 DDC = BIT6
4363 STR = BIT5
4364 SEC = BIT4
4365 IDL = BIT3
4366 CRCTY2 = BIT2
4367 CRCTY1 = BIT1
4368 CRCTY0 = BIT0

4369
4370 ;*****
4371 * AX3-15 - USYRT REG 6

4372
4373
' 4
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395

I422 = BIT7
XYZ = BIT6
V35 = BIT4
INTGRL = BIT3
OP = BIT1
TEST = BIT0
AX315U = I422!XYZ!V35!INTGRL.OP

* AX3-16 - USYRT REG 7

TXLEN2 = BIT7
TXLEN1 = BIT6
TXLENO = BIT5
RXLEN2 = BIT2
RXLEN1 = BIT1
RXLENO = BIT0

4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452

8.0 TEST DESCRIPTIONS

```
*****  
TEST 1 - BIT STUFFING TEST  
*  
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS  
* INITIATED IN BIT MODE . TWO LEADING FLAGS ARE SENT,  
* FOLLOWED BY ALL SIXTEEN CHARS IN DATA PATTERN S. THIS PATTERN  
* CONSISTS OF CHARACTERS WHICH REQUIRE NO BIT STUFFING AND CHARACTERS  
* WHICH REQUIRE BIT STUFFING INDIVIDUALLY AND IN COMBINATION WITH  
* ADJACENT CHARACTERS. ALL 16 CHARACTERS ARE READ AND COMPARED  
* BY THE RECEIVER.  
* PATTERN S = 000,017,036,074,170,360,037,076,174,370,077,176,374,  
* 177,376,377  
*****
```

```
*****  
TEST 2 - RCV OVERRUN ERROR SET AND CLEAR TEST  
*  
* IN THIS TEST, A RCV OVERRUN ERROR IS FORCED IN EACH OF 2 SUBTESTS.  
* IN THE FIRST, A MESSAGE IS INITIATED, 64 001 CHARS ARE SENT, AND THE  
* RECEIVER IS NOT SERVICED IN RESPONSE TO THE USYRT RCV FLAG, WHICH CAUSES RCV  
* OVERRUN TO SET. THEN, A CHECK IS MADE TO INSURE THAT OVRR IS NOT  
* CLEARED BY THE LINE UNIT READING THE USYRT STATUS.  
* THEN, IC IS SET TO CLEAR THE ERROR, AND THIS IS VERIFIED.  
*  
* IN THE SECOND SUBTEST, RCV OVRUN IS FORCED AGAIN, AND A MASTER CLEAR  
* IS ISSUED TO CLEAR THE ERROR, AND THIS IS VERIFIED.  
*****
```

```
*****  
TEST 3 - ABORT SEQUENCE TEST  
*  
* SET BIT MODE, CRC, AND ENABLE THE DEVICE FOR  
* TRANSMIT AND RECEIVE. SEND 2 FLAGS AND 4 DATA CHARS (001).  
* AS THE FIRST DATA CHAR IS BEING TRANSMITTED,  
* SET THE ABORT BIT (REG 11).  
* ON THE RECEIVER SIDE, CHECK FOR RECEPTION OF THE FIRST DATA CHAR  
* AND THEN THE SETTING OF RAB AND REOM A CHAR TIME LATER.  
* ALSO, CHECK FOR IACT = 0. THEN, CHECK THAT RAB  
* IS CLEARED BY READING THE USYRT STATUS, TRANSMITTING A NEW MSG,  
* RECEIVING THE FIRST CHAR (003) AND CHECKING FOR RAB CLEARED.
```

4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508

*
* REPEAT THE ABOVE SEQUENCE, SET IC, AND CHECK THAT
* THIS CLEARS RAB.
*
* REPEAT THE ABOVE SEQUENCE, ISSUE MASTER CLEAR, CHECK THAT THIS
* CLEARS RAB.
*
;*****

;*****
TEST 4 - ABORT AND IDLE FLAGS TEST
*
* TRANSMIT THE SAME ABORT SEQUENCE AS IN THE PREVIOUS TEST, BUT
* WITH THE IDLE BIT SET. CHECK THAT FLAGS ARE SENT AND RECEIVED
* (NOT ABORT CHARACTERS) BY VERIFYING THAT RAB DOES
* NOT SET, AND THAT THE MESSAGE TERMINATES WITH EBLK = 1.
;*****

;*****
TEST 5 - TRANSMITTER UNDERRUN ERROR, IDLE ABORT CHARS, BIT MODE
*
* A MESSAGE IS INITIATED IN BIT MODE, 4 001 CHARS ARE SENT, AND THE TRANSMITTER
* IS NOT SERVICED IN RESPONSE TO THE LAST TX FLAG, WHICH CAUSES TX
* UNDERRUN ERROR TO SET. ON THE RECEIVER SIDE, CHECK THAT THE DATA
* CHAR IS RECEIVED, AND THAT 8 CYCLES LATER THE RAB BIT SETS, AND
* THE DEVICE IDLES ABORT CHARACTERS.
;*****

;*****
TEST 6 - RECEIVER DISABLE TEST
*
* TRANSMIT AND RECEIVE ARE ENABLED IN BIT MODE, AND 2 FLAGS
* ARE SENT, FOLLOWED BY 5 252 DATA CHARS. AFTER THE SECOND DATA CHAR HAS BEGUN
* TO BE RECEIVED, IC IS SET.
* THEN, THE PROGRAM CHECKS THAT A USYRT RCV FLAG IS NOT GENERATED, AND
* THE RECEIVER DATA PATH STOPS OPERATING IN THE MIDDLE OF THE CHAR.
;*****

;*****
TEST 7 - ASSEMBLED BIT COUNT TEST

4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564

*
* THE FOLLOWING SEQUENCE IS PERFORMED 8 TIMES, EACH TIME USING A
* DIFFERENT TX CHAR LENGTH (FROM 2 TO 8 BITS) AND A RCV CHAR LENGTH = 8
* BITS :
* A MESSAGE IS INITIATED IN BIT MODE, NO CRC.
* 2 FLAGS ARE SENT, FOLLOWED BY 3 000 DATA CHARACTERS AND A
* TERMINATING FLAG. AFTER THE RECEIVER HAS RECEIVED THE MESSAGE, AX0-16
* IS READ TO RETRIEVE THE ASSEMBLED BIT COUNT. THIS COUNT IS CHECKED TO INSURE
* THAT IT IS CORRECT FOR THE TX CHAR LENGTH USED IN THAT TRANSMISSION.
;*****

;*****
TEST 8 - SECONDARY STATION ADDRESS BIT TEST

*
* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
* BIT MODE, AND THE SECA BIT (REG 17) IS SET.
* 2 FLAGS ARE SENT, FOLLOWED BY 252, 000, AND A TERMINATING FLAG.
* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
* RECEIVED.
*
* NEXT, THE SECONDARY STATION ADDRESS BITS IN AX2-15 ARE LOADED
* WITH THE FIRST WORD OF DATA PATTERN T . 2 FLAGS ARE SENT,
* FOLLOWED BY THE FIRST WORD OF DATA PATTERN T, A 000 CHAR,
* AND A TERMINATING FLAG.
* THEN, THE RCV'D DATA IS CHECKED TO MAKE SURE THAT THE SEC STATION
* ADDRESS IS RCV'D AS THE FIRST DATA CHAR, FOLLOWED BY 000.

*
* THEN, THE SUBTEST IS REPEATED FOR EACH OF THE REMAINING WORDS OF
* DATA PATTERN T.
* PATTERN T = 000,125,252,176,177
;*****

;*****
TEST 9 - RDALL (ALL PARTIES ADDRESS) BIT TEST

*
* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
* BIT MODE, AND THE SECA BIT IS SET.
* 2 FLAGS ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
* RECEIVED.
* NEXT, THE RDALL BIT IN REG 17 IS SET TO 1. 2 FLAGS
* ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE REC'D DATA IS CHECKED TO MAKE SURE THAT 377
* IS REC'D AS THE FIRST DATA CHAR, FOLLOWED BY 125.
;*****

4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596
4597
4598
4599
4600
4601
4602
4603
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620

```
*****  
TEST 10 - INSERT ERROR (IERR) BIT TEST - CHAR MODE, NO CRC  
*  
* THE LINE UNIT IS PLACED IN DDCMP MODE WITH NO ERROR DETECTION, AND 2  
* SYNCHS, A 000 CHAR, A 377 CHAR, AND 2 SYNCHS ARE LOADED INTO THE  
* TRANSMITTER SILO. THEN, THE LU IS CLOCKED UNTIL THE 2ND BIT OF THE 000  
* CHAR IS ABOUT TO BE SENT AND THE IERR BIT IS SET FOR A CLOCK TIME AND  
* THEN CLEARED. IN THE SAME WAY, IERR IS SET PRIOR TO THE SENDING OF THE 4TH  
* AND 5TH BITS OF THE 000 CHAR. IT IS ALSO SET FOR THE SENDING OF THE FIRST  
* 4 BITS OF THE 377 CHAR. THE PROGRAM READS THE FIRST RCV'D CHAR FROM AX0  
* AND CHECKS IT TO BE 032, AND READS THE 2ND CHAR AND CHECKS IT TO BE 377.  
* THEN, A MASTER CLEAR IS DONE TO IDLE THE DEVICE.  
*****  
  
*****  
TEST 11 - SWITCH PACK PRINTOUT AND TEST  
*  
* - READ AND PRINT SWITCH PACK 1 :  
* THE PROGRAM READS REG 11 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,  
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO  
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE  
* SWITCHES ARE IN BITS 1,2,3,5.  
*  
* - READ AND PRINT SWITCH PACK 2 :  
* THE PROGRAM READS REG 15 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,  
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO  
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE  
* SWITCHES ARE IN BITS 0-7.  
*  
* - READ AND PRINT SWITCH PACK 3 :  
* THE PROGRAM READS REG 16 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,  
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO  
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE  
* SWITCHES ARE IN BITS 0-7.  
*****  
  
*****  
TEST 12 - REG AX3-15 PRINTOUT  
*  
* IN THIS TEST, REG AX3-15 IS READ AND THE CONTENTS PRINTED OUT IF DESIRED BY  
* THE OPERATOR, AS INDICATED IN THE SOFTWARE P-TABLE. THE DEFAULT IS TO NOT  
* PRINT THE REG.  
*****
```

4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676

```
*****  
TEST 13 - CRC GENERATION TEST  
*  
* - CRC-16, CHAR MODE:  
* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE WITH CRC-16 SELECTED -  
* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU  
* TO CLOCK THE DATA. AT THE END OF THE MESSAGE THE  
* PROGRAM CHECKS FOR BCC = 1 (IN REG 12) INDICATING NO ERROR.  
*  
* - CRC-CCITT - 1'S PRESET:  
* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT  
* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.  
*  
* - CRC-CCITT - 0'S PRESET:  
* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT  
* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.  
*****
```

```
*****  
TEST 14 - CRC ERROR DETECTION TEST  
*  
* - CRC-16, CHAR MODE :  
* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE, WITH CRC-16 SELECTED -  
* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU  
* TO CLOCK THE DATA. JUST BEFORE THE FIRST BIT OF THE LAST 000 CHAR IS SENT,  
* THE IERR BIT IS SET IN REG 17 TO CAUSE A 1 TO BE SENT, INTRODUCING A DATA  
* ERROR. AT THE END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC 0, INDICATING  
* AN ERROR.  
*  
* - CRC-CCITT - 1'S PRESET :  
* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT THE  
* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC 1, INDICATING AN ERROR.  
*  
* - CRC-CCITT - 0'S PRESET :  
* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT THE  
* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC 1, INDICATING AN ERROR.  
*****
```

```
*****  
TEST 15 - VRC PARITY GENERATION TEST  
*  
* SUBTEST 1 - TEST OF CORRECT ODD VRC PARITY GENERATION :  
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH ODD VRC AND 7-BIT CHARS SELECTED.  
* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)  
* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.  
* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 1 AND FOR THE
```


4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732

* LAST 4 CHARS IT SHOULD = 0.
*
* SUBTEST 2 - TEST OF CORRECT EVEN VRC PARITY GENERATION :
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH EVEN VRC AND 7-BIT CHARS SELECTED.
* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)
* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.
* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 0 AND FOR THE
* LAST 4 CHARS IT SHOULD = 1.
*
* DATA PATTERN Q = 000,120,125,137,040,052,057,177
;*****

;*****
TEST 16 - VRC ERROR DETECTION TEST
*
* SUBTEST 1 - FORCING OF BCC USING ODD VRC
* THE LINE UNIT IS PLACED IN CHAR MODE WITH ODD VRC AND 7-BIT CHARS SELECTED.
* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
* RECEIVED (INDICATING AN ERROR).
*
* SUBTEST 2 - FORCING OF BCC USING EVEN VRC
* THE LINE UNIT IS PLACED IN CHAR MODE WITH EVEN VRC AND 7-BIT CHARS SELECTED.
* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
* RECEIVED (INDICATING AN ERROR).
*
* DATA PATTERN R = 000,100,120,124,164,172,176,177,000,100,120,124,164,
* 172,176.
;*****

;*****
TEST 17 - INTEGRAL MODEM INTERFACE TEST - CHAR MODE, CRC
*
* THE INTEGRAL MODEM IS SELECTED BY THE PROGRAM IN AX3-15, AND A
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
* ON THE LINE UNIT OR AT THE CABLE. THE MESSAGE CONSISTS OF
* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
* SKIPPED FOR THAT UNIT.
;*****

4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788

```
*****  
TEST 18 - V.35 MODEM INTERFACE TEST - CHAR MODE, CRC  
*  
* THE V.35 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A  
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR  
* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,  
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF  
* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT  
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE  
* SKIPPED FOR THAT UNIT.  
*****
```

```
*****  
TEST 19 - RS 232C AND RS 423 MODEM INTERFACE TEST - CHAR MODE, CRC  
*  
* THE RS232C RS423 (XYZ) MODEM INTERFACE IS SELECTED BY THE PROGRAM IN  
* AX3-15, AND A MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURN-  
* AROUND CONNECTOR ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,  
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS  
* OF 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE  
* P-TABLE FOR THE CURRENT UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS  
* PROVIDED, THE TEST WILL BE SKIPPED FOR THAT UNIT.  
*****
```

```
*****  
TEST 20 - RS 422 MODEM INTERFACE TEST - CHAR MODE, CRC  
*  
* THE RS 422 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A  
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR  
* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,  
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF  
* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT  
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE  
* SKIPPED FOR THAT UNIT.  
*****
```

```
*****  
TEST 21 - HALF-DUPLEX BIT (HALF DUPX) TEST  
*  
* THIS TEST VERIFIES THAT SETTING HALF-DUPLEX BIT IN REG 13 INHIBITS  
* LOADING OF THE USYRT TRANSMITTER FROM THE TRANSMITTER SILO.  
* A MASTER CLEAR IS ISSUED, DDCMP MODE IS ENTERED, AND THE HALF DUPX
```

4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844

* BIT IN REG 13 IS SET. A MESSAGE IS LOADED INTO THE TX SILO
* CONSISTING OF 2 SYNCHS, 000,125,252,377,000, AND 2 MORE SYNCHS.
* THE LINE UNIT IS THEN CLOCKED EXTENSIVELY, AND THE TX SILO IS CHECKED TO
* BE STILL LOADED (NO CHARS SHOULD HAVE BEEN REMOVED) AND THE RECEIVER
* IS MONITORED TO INSURE THAT NO RCV FLAGS ARE GENERATED.

TEST 22 - HALF-DUPLEX RCV DISABLED TEST WITH SILOS DISABLED

* THIS TEST SENDS A MESSAGE IN HDX, CHAR MODE, WITH NO ERROR DETECTION, AND
* THE SILOS DISABLED. THE MSG CONSISTS OF 2 SYNCHS AND 2 000 CHARS.
* THE DATA IS SENT WITH LULOOK SET FOR TTL DATA LOOPBACK. THE PROGRAM CHECKS
* THAT THE RECEIVER NEVER BECOMES ACTIVE, BECAUSE THE RCV CLOCK IS INHIBITED
* WHEN THE HDX BIT IS SET.

TEST 23 - INTERACTION OF MODEM CONTROL BITS

* THIS TEST WILL BE RUN ONLY IF THE P-TABLE FOR THIS UNIT INDICATES THAT
* THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. OTHERWISE, THE TEST WILL
* BE SKIPPED FOR THE UNIT.
* THE FOLLOWING SUBTESTS ARE PERFORMED:
* - A MASTER CLEAR IS DONE AND REG 13 IS READ AND CHECKED FOR INITIALIZED
* STATE, WITH LULOOK SET TO 1. THEN, LULOOK IS CLEARED AND REG 13 IS READ
* AND CHECKED FOR THE PROPER STATE, WITH LULOOK CLEARED.
* REG 13 IS THEN LOADED WITH 0'S, AND READ AND CHECKED FOR THE
* INITIALIZED STATE.
* REG 17 IS THEN READ AND CHECKED FOR INITIALIZED STATE.
* - RUN IS SET IN BSEL1, AND REG 13 IS READ AND CHECKED FOR RING SET.
* - POLL IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGQ SET.
* - BPOLL IS SET IN REG 12, ONLY TO LIGHT THE LED FOR THIS SIGNAL.
* - DTR IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR DTR AND MODR SET.
* - SELFR IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGR SET.
* - HDX IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR HDX SET.
* - MAINT1 IS SET IN REG 3, AND REG 17 IS READ AND CHECKED FOR TEST MODE SET.
* - SELSBY IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR STBY SET.
* - A MASTER CLEAR IS DONE, 2 TSOM'S ARE LOADED INTO THE TX SILO, THE LINE
* UNIT IS CLOCKED UNTIL THE TRANSMITTER IS ACTIVE, AND REG 13 IS READ AND
* CHECKED FOR RTS, CS, CARR SET.

4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900

TEST 24 - DATA TEST - BIT MODE, NO ERR DET

*
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH ERROR DETECTION
* INHIBITED. THE MESSAGE CONSISTS OF 2 FLAGS, PAT A REPEATED 3 TIMES,
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
* 375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
;*****

;*****

TEST 25 - DATA TEST - CHAR MODE, NO ERR DET

*
* A MESSAGE IS INITIATED IN CHAR MODE, WITH ERROR DETECTION
* INHIBITED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
* 375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
;*****

;*****

TEST 26 - DATA TEST - BIT MODE, CRC-CCITT-1

*
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-1 ERROR
* DETECTION. THE MESSAGE CONSISTS OF 2 FLAGS, PAT A REPEATED 3 TIMES,
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
* 375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
;*****

4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956

```
*****  
TEST 27 - DATA TEST - BIT MODE, CRC-CCITT-0  
*  
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-0 ERROR  
* DETECTION. THE MESSAGE CONSISTS OF 2 FLAGS, PAT A REPEATED 3 TIMES,  
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
* TEST WILL NOT BE RUN.  
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177  
* 8-BIT CHARACTERS ARE USED.  
*****
```

```
*****  
TEST 28 - DATA TEST - CHAR MODE, CRC-16  
*  
* A MESSAGE IS INITIATED IN CHAR MODE, WITH CRC-16 ERROR  
* DETECTION. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,  
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
* TEST WILL NOT BE RUN.  
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177  
* 8-BIT CHARACTERS ARE USED.  
*****
```

```
*****  
TEST 29 - DATA TEST - CHAR MODE, ODD VRC  
*  
* A MESSAGE IS INITIATED IN CHAR MODE, WITH ODD VRC ERROR DETECTION  
* SELECTED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,  
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
* TEST WILL NOT BE RUN.  
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177  
* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).  
*****
```

4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012

```
*****  
; TEST 30 - DATA TEST - CHAR MODE, EVEN VRC  
*  
* A MESSAGE IS INITIATED IN CHAR MODE, WITH EVEN VRC ERROR DETECTION  
* SELECTED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,  
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
* TEST WILL NOT BE RUN.  
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177  
* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).  
;*****
```

```
*****  
; TEST 31 - CONTIGUOUS ONES IN SEC. STA. ADRS. MODE, BIT MODE  
*  
* IN THIS TEST, A MESSAGE CONSISTING OF 5 ONES CHARS (377 OCT)  
* IS SENT IN SECONDARY STATION ADDRESS MODE, WITH THE STATION ADRS  
* FOR THIS LINE = 377. THE PROGRAM CHECKS FOR CORRECT RECEPTION OF  
* THE FIRST CHARACTER (STATION ADDRESS) AND THE REMAINING 4  
* ONES CHARACTERS (DATA). THIS TEST EXERCISES THE SECONDARY STATION  
* ADDRESS LOGIC, AND CHECKS THAT THE SEL. STA ADRS. CAN BE BIT-STUFFED  
* AND TRANSMITTED AND RECEIVED CORRECTLY.  
;*****
```

```
*****  
; TEST 32 - DDCMP MESSAGE TEST - CHAR MODE  
*  
* IN THIS TEST, THREE USYRT MESSAGES ARE SENT TO SIMULATE A DDCMP HEADER,  
* DDCMP DATA MESSAGE, AND THE START OF A NEW DDCMP HEADER.  
* FIRST, THE DATA IN PATTERN A IS TRANSMITTED AND RECEIVED  
* AND THEN CRC (CRC-16) IS SENT, FOLLOWED BY THE DATA IN PATTERN A  
* AGAIN AND THE CRC ON THAT DATA, AND FINALLY THE DATA IN 'MSG1' IS  
* SENT WITH ITS CORRESPONDING CRC.  
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
* 375,373,367,357,337,277,177  
* MSG1 - SYNCH,SYNCH,SYNCH,SYNCH,000,125,252,377,SYNCH,SYNCH  
;*****
```

8.1 DATA PATTERNS USED

5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063
5064
5065
5066
5067
5068

***** DATA PATTERN A *****

PATA: .BYTE 125
.BYTE 252
.BYTE 000
.BYTE 377
.BYTE 001
.BYTE 002
.BYTE 004
.BYTE 010
.BYTE 020
.BYTE 040
.BYTE 100
.BYTE 200
.BYTE 376
.BYTE 375
.BYTE 373
.BYTE 367
.BYTE 357
.BYTE 337
.BYTE 277
.BYTE 177

***** DATA PATTERN Q *****

PATQ: .BYTE 000
.BYTE 120
.BYTE 125
.BYTE 137
.BYTE 040
.BYTE 052
.BYTE 057
.BYTE 177

***** DATA PATTERN R *****

PATR: .BYTE 000
.BYTE 100
.BYTE 120
.BYTE 124
.BYTE 164
.BYTE 172
.BYTE 176
.BYTE 177
.BYTE 000
.BYTE 100
.BYTE 120
.BYTE 124
.BYTE 164
.BYTE 172
.BYTE 176

***** DATA PATTERN S *****

PATS: .BYTE 000
.BYTE 017
.BYTE 036

5069	.BYTE	074
5070	.BYTE	170
5071	.BYTE	360
5072	.BYTE	037
5073	.BYTE	076
5074	.BYTE	174
5075	.BYTE	370
5076	.BYTE	077
5077	.BYTE	176
5078	.BYTE	374
5079	.BYTE	177
5080	.BYTE	376
5081	.BYTE	377

***** DATA PATTERN T *****

5083	PATT:	.BYTE	000
5084		.BYTE	125
5085		.BYTE	252
5086		.BYTE	176
5087		.BYTE	177
5088		.BYTE	177
5089			
5090			
5091			

CZDMSA MB203 STATIC DIAG #2
CZDMSA.P11 25-JUN-79 14:01

MACY11 30A(1052) 17-JUL-79 15:33 K 3
PROGRAM DOCUMENT PAGE 7

SEQ 0036

5093

5095
5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146

9.0 ERROR INFORMATION

9.1 ERROR REPORTING

ERRORS ARE REPORTED BY THE PROGRAM AS THEY OCCUR (IF NOT INHIBITED). THE REPORT CONFORMS TO THE DIAGNOSTIC SUPERVISOR ERROR REPORT FORMAT, AND CONSISTS OF A DESCRIPTION OF THE ERROR, THE TEST NUMBER, SUBTEST NUMBER, PC OF THE ERROR CALL, DEVICE ADDRESS, AND BASIC AND EXTENDED ERROR INFORMATION.

THE FOLLOWING EXAMPLE PROVIDES A TYPICAL ERROR REPORT, WHICH DESCRIBES AN 'IRDY NOT SET' ERROR, AND PROVIDES THE PC OF THE ERROR CALL AND THE PC OF THE CALL TO THE SUBROUTINE REPORTING IT, THE FAILING REGISTER NAME, AND DEVICE REGISTER CONTENTS :

CZDMS DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC: 006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

FAILING REG: INBUS/OUTBUS REG 12

LINE UNIT INBUS REGS:
REG10 REG11 REG12 REG13
000 120 000 257
REG14 REG15 REG16 REG17
024 377 377 035

LINE UNIT EXTENDED REGS:
AX0-15 AX0-16 AX1-15 AX1-16
000 000 000 000
AX2-15 AX2-16 AX3-15 AX3-16
000 000 000 000

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

IF EXTENDED ERROR INFORMATION HAD BEEN INHIBITED USING THE IXE FLAG PRIOR TO RUNNING THE TEST, THE ABOVE ERROR WOULD HAVE BEEN REPORTED IN THE FOLLOWING SHORTENED FORM :

CZDMS DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC:006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

FAILING REG: INBUS/OUTBUS REG 12

5148
5149
5150
5151
5152
5153
5154
5155
5156
5157

0

```
5159          .TITLE CZDMSA M8203 STATIC TESTS #2
5168          002000          .-2000
5169
5170
5171
5172
5173
5174
5175          .MCALL SVC
5176 002000          SVC          ; INITIALIZE SUPERVISOR MACROS
5177
5178
5179
5180
5181
5182 002000          BGNMOD LU2MOD
5183
5184
5185          000001          $LSTIN= 1
5186          000001          $LSTTAG= 1
5187          000001          SVCINS= 1          ; LIST INSTRUCTIONS, SHIFTED RIGHT
5188          000001          SVCTST= 1          ; LIST TEST TAGS, SHIFTED RIGHT
5189          000001          SVCSUB= 1          ; LIST SUBTEST TAGS, SHIFTED RIGHT
5190          000001          SVCGBL= 1          ; LIST GLOBAL TAGS, SHIFTED RIGHT
5191          000001          SVCTAG= 1          ; LIST OTHER TAGS, SHIFTED RIGHT
5192
5193          ; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
5194          ; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
5195          ; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
5196          ; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.
5197
5198
```

5200
5201
5202
5203
5204
5205
5206 002000
5207
5209
5210
5211
5212
5213
5215
5216 002000
(4) 002000
(4) 002000 103
(4) 002001 132
(4) 002002 104
(4) 002003 115
(4) 002004 123
(6) 002005 000
(6) 002006 000
(5) 002007 000
(5) 002010
(4) 002010 101
(5) 002011
(4) 002011 060
(5) 002012
(4) 002012 000000
(5) 002014
(4) 002014 000055
(5) 002016
(4) 002016 036336
(5) 002020
(4) 002020 037300
(5) 002022
(4) 002022 002226
(5) 002024
(4) 002024 002256
(5) 002026
(4) 002026 037750
(5) 002030
(4) 002030 000000
(5) 002032
(4) 002032 000000
(5) 002034
(4) 002034 000000
(5) 002036
(4) 002036 000000
(5) 002040
(4) 002040 002124
(5) 002042
(4) 002042 000000
(5) 002044
(4) 002044 000000

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

POINTER BGNSW,BGNSFT,BGNAU,BGNDU

:XX
: IF ANY OPTIONAL POINTERS ARE TO BE USED IN THE 'HEADER', CHANGE
: 'POINTER' TO CONTAIN THE CORRECT ARGUMENTS. IF ALL OPTIONAL
: POINTERS ARE TO BE USED, CHANGE 'POINTER' TO BE 'POINTER ALL'.
:XX

HEADER CZDMS,A,0,45.,0

L\$NAME::
.ASCII /C/
.ASCII /Z/
.ASCII /D/
.ASCII /M/
.ASCII /S/
.BYTE 0
.BYTE 0
.BYTE 0
L\$REV::
.ASCII /A/
L\$DEPO::
.ASCII /O/
L\$UNIT::
.WORD 0
L\$TIML::
.WORD 45.
L\$HPCP::
.WORD L\$HARD
L\$SPCP::
.WORD L\$SOFT
L\$HPTP::
.WORD L\$HW
L\$SPTP::
.WORD L\$SW
L\$LADP::
.WORD L\$LAST
L\$STA::
.WORD 0
L\$CO::
.WORD 0
L\$DTYP::
.WORD 0
L\$APT::
.WORD 0
L\$DTP::
.WORD L\$DISPATCH
L\$EXP1::
.WORD 0
L\$EXP2::
.WORD 0

(5) 002046
(4) 002046 000000
(5) 002050
(4) 002050 003
(3) 002051 000
(5) 002052
(4) 002052 000000
(5) 002054 000000
(5) 002056
(4) 002056 000000
(5) 002060
(4) 002060 003162
(5) 002062
(4) 002062 000000
(5) 002064
(4) 002064 000000
(5) 002066
(4) 002066 000000
(5) 002070
(4) 002070 023144
(5) 002072
(4) 002072 023062
(5) 002074
(4) 002074 000000
(5) 002076
(4) 002076 003170
(5) 002100
(4) 002100 104035
(5) 002102
(4) 002102 000000
(5) 002104
(4) 002104 022026
(5) 002106
(4) 002106 023060
(5) 002110
(4) 002110 023000
(5) 002112
(4) 002112 022020
(5) 002114
(4) 002114 000000
(5) 002116
(4) 002116 000000
(5) 002120
(4) 002120 000000

L\$EXP3::
L\$MREV:: .WORD 0
L\$EF:: .BYTE C\$REVISION
 .BYTE C\$EDIT
L\$SPC:: .WORD 0
L\$DEVP:: .WORD 0
L\$REPP:: .WORD L\$DVTYP
L\$EXP4:: .WORD 0
L\$EXP5:: .WORD 0
L\$AUT:: .WORD 0
L\$DUT:: .WORD L\$AU
L\$LUN:: .WORD L\$DU
L\$DESP:: .WORD 0
L\$LOAD:: .WORD L\$DESC
 EMT E\$LOAD
L\$ETP:: .WORD 0
L\$IICP:: .WORD L\$INIT
L\$CCP:: .WORD L\$CLEAN
L\$ACP:: .WORD L\$AUTO
L\$PRT:: .WORD L\$PROT
L\$TEST:: .WORD 0
L\$DLY:: .WORD 0
L\$HIME:: .WORD 0

5217
5219
5220
5221
5223
5224
5225
5226
5227
5228
5229
5230

:XX
: CHANGE THE 'HEADER' TO CONTAIN THE PROPER ARGUMENTS.
:XX

.EVEN

5282
5283
5284
5285
5286
5287
5288
5289
(3)
(3)
(3)
5290
5291
5292
5293
5294
5295
5296
(3)
5297
5298
5299
5300
5301
5302

.SBTTL SOFTWARE P-TABLE

:/ THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
:/ PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.

BGNSW SFPTBL

.WORD L10001-L\$SW/2

L\$SW::
SFPTBL::

MIFLAG: .WORD C ; -1 IF MAN. INTERVENTION DESIRED, =0 IF NOT
PRNFLG: .WORD 0 ; =1 IF SW PACK AND AX3-15 PRINTOUT ALLOWED ALWAYS
SWIFLG: .WORD 0 ; =1 IF SWITCH PACK VERIFICATION TEST SHOULD BE RUN
TCOUNT: .WORD 0 ; INITIAL MSG TIME-OUT VALUE (0=LONGEST TIME-OUT)

ENDSW

L10001:

5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323

002266

.SBTTL GLOBAL EQUATES SECTION

:/
:/ THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
:/ ARE USED IN MORE THAN ONE TEST.
:/

EQUALS

:
: BIT DIFINITIONS

BIT15== 10000
BIT14== 4000
BIT13== 2000
BIT12== 1000
BIT11== 400
BIT10== 200
BIT09== 100
BIT08== 40
BIT07== 20
BIT06== 10
BIT05== 4
BIT04== 2
BIT03== 1
BIT02== 4
BIT01== 2
BIT00== 1

BIT9== BIT09
BIT8== BIT08
BIT7== BIT07
BIT6== BIT06
BIT5== BIT05
BIT4== BIT04
BIT3== BIT03
BIT2== BIT02
BIT1 = BIT01
BIT0-- BIT00

:
: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

EF.START=- 32. : START COMMAND WAS ISSUED
EF.RESTART=- 31. : RESTART COMMAND WAS ISSUED

(1)
(1)
(1)
(1) 10000
(1) 04000
(1) 02000
(1) 01000
(1) 00400
(1) 00200
(1) 00100
(1) 00040
(1) 00020
(1) 00010
(1) 00004
(1) 00002
(1) 00001
(1)
(1) 00100
(1) 00040
(1) 00020
(1) 00010
(1) 00004
(1) 00002
(1) 00001
(1)
(1)
(1)
(1) 00040
(1) 00037

```
(1) 000036 EF.CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
(1) 000035 EF.NEW== 29. ; A NEW PASS HAS BEEN STARTED
(1) 000034 EF.PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED
(1)
(1)
(1)
(1)
(1)
(1) 000340 PRI07== 340
(1) 000300 PRI06== 300
(1) 000240 PRI05== 240
(1) 000200 PRI04== 200
(1) 000140 PRI03== 140
(1) 000100 PRI02== 100
(1) 000040 PRI01== 40
(1) 000000 PRI00== 0
(1)
(1) ;OPERATOR FLAG BITS
(1)
(1) 000004 EVL== 4
(1) 000010 LOT== 10
(1) 000020 ADR== 20
(1) 000040 IDU== 40
(1) 000100 ISR== 100
(1) 000200 UAM== 200
(1) 000400 BOE== 400
(1) 001000 PNT== 1000
(1) 002000 PRI== 2000
(1) 004000 IXE = 4000
(1) 010000 IBE== 10000
(1) 020000 IER== 20000
(1) 040000 LOE== 40000
(1) 100000 HOE== 100000
```

5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347

```
*****
;* PROGRAM EVENT FLAG DEFINITIONS
*****
```

```
*****
;* MAINTENANCE REGISTER - BSEL1
*****
```

```
RUN = BIT7
MCLR = BIT6
STEPLU = BIT4
LULOOP = BIT3
ROMO = BIT2
ROMI = BIT1
STEPMP = BIT0
```

```
*****
```

```
5348 ;* OBUS REG 10 - TRANSMITTER BUFFER
5349 :*****
5350 000200 TX7 = BIT7
5351 000100 TX6 = BIT6
5352 000040 TX5 = BIT5
5353 000020 TX4 = BIT4
5354 000010 TX3 = BIT3
5355 000004 TX2 = BIT2
5356 000002 TX1 = BIT1
5357 000001 TX0 = BIT0
5358
5359 :*****
5360 ;* OBUS REG 11
5361 :*****
5362 000200 OC = BIT7
5363 000010 GOAH = BIT3
5364 000004 ABORT = BIT2
5365 000002 EOM = BIT1
5366 000001 SOM = BIT0
5367
5368 :*****
5369 ;* OBUS REG 12
5370 :*****
5371 000200 IC = BIT7
5372 000100 BPOLL = BIT6
5373 000040 LULP = BIT5
5374
5375 :*****
5376 ;* OBUS REG 13
5377 :*****
5378 000200 POLL = BIT7
5379 000100 DTR = BIT6
5380 000040 SELFR = BIT5
5381 000020 HDX = BIT4
5382 000010 MAINT1 = BIT3
5383 000004 MAINT2 = BIT2
5384 000002 SELSBY = BIT1
5385
5386 :*****
5387 ;* OBUS REG 14
5388 :*****
5389 000100 TXEN = BIT6
5390 000040 DISSI = BIT5
5391 000020 RDAX = BIT4
5392 000010 WAX = BIT3
5393 000004 ENAX = BIT2
5394 000002 AX2 = BIT1
5395 000001 AX1 = BIT0
5396
5397 :*****
5398 ;* OBUS REG 17
5399 :*****
5400 000200 CRC2 = BIT7
5401 000100 CRC1 = BIT6
5402 000040 IDLE = BIT5
5403 000020 SECA = BIT4
```

5404	000010	STRIP	= BIT3
5405	000004	RDAIL	= BIT2
5406	000002	IERR	= BIT1
5407	000001	DDCMP	= BIT0
5408			
5409		:*****	
5410		:* IBUS REG 10 - RECEIVER BUFFER	
5411		:*****	
5412	000200	RX7	= BIT7
5413	000100	RX6	= BIT6
5414	000040	RX5	= BIT5
5415	000020	RX4	= BIT4
5416	000010	RX3	= BIT3
5417	000004	RX2	= BIT2
5418	000002	RX1	= BIT1
5419	000001	RX0	= BIT0
5420			
5421		:*****	
5422		:* IBUS REG 11	
5423		:*****	
5424	000200	OC	= BIT7
5425	000100	OACT	= BIT6
5426	000040	SW3	= BIT5
5427	000020	ORDY	= BIT4
5428	000010	SW2	= BIT3
5429	000004	SW1	= BIT2
5430	000002	SW0	= BIT1
5431	000001	UNRR	= BIT0
5432			
5433		:*****	
5434		:* IBUS REG 12	
5435		:*****	
5436	000200	IC	= BIT7
5437	000100	IACT	= BIT6
5438	000040	LULP	= BIT5
5439	000020	IRDY	= BIT4
5440	000010	OVRR	= BIT3
5441	000004	RAB	= BIT2
5442	000002	EBLK	= BIT1
5443	000001	BCC	= BIT0
5444			
5445		:*****	
5446		:* IBUS REG 13	
5447		:*****	
5448	000200	RING	= BIT7
5449	000100	DTR	= BIT6
5450	000040	RTS	= BIT5
5451	000020	HDX	= BIT4
5452	000010	MODR	= BIT3
5453	000004	CS	= BIT2
5454	000002	STBY	= BIT1
5455	000001	CARR	= BIT0
5456			
5457		:*****	
5458		:* IBUS REG 14	
5459		:*****	

5460	000200	READY	=	BIT7
5461	000100	TXEN	=	BIT6
5462	000040	DISSI	=	BIT5
5463	000020	RDAX	=	BIT4
5464	000010	WAX	=	BIT3
5465	000004	ENAX	=	BIT2
5466	000002	AX2	=	BIT1
5467	000001	AX1	=	BIT0

::*****
:* IBJS REG 17
::*****

5472	000200	SIGR	=	BIT7
5473	000100	SIGQ	=	BIT6
5474	000040	TXDATA	=	BIT5
5475	000020	OCOR	=	BIT4
5476	000010	ICIR	=	BIT3
5477	000004	TESTMD	=	BIT2
5478	000002	MCLK	=	BIT1
5479	000001	DDCMP	=	BIT0

::*****
:* AX0-15 - USYRT REG 0 (READ ONLY)
::*****

5484	000200	RX7	=	BIT7
5485	000100	RX6	=	BIT6
5486	000040	RX5	=	BIT5
5487	000020	RX4	=	BIT4
5488	000010	RX3	=	BIT3
5489	000004	RX2	=	BIT2
5490	000002	RX1	=	BIT1
5491	000001	RX0	=	BIT0

::*****
:* AX0-16 - USYRT REG 1 (READ ONLY)
::*****

5496	000200	RERR	=	BIT7
5497	000100	ASBC2	=	BIT6
5498	000040	ASBC1	=	BIT5
5499	000020	ASBC0	=	BIT4
5500	000010	ROR	=	BIT3
5501	000004	RABT	=	BIT2
5502	000002	REOM	=	BIT1
5503	000001	RSOM	=	BIT0

::*****
:* AX1-15 - USYRT REG 2
::*****

5508	000200	TX7	=	BIT7
5509	000100	TX6	=	BIT6
5510	000040	TX5	=	BIT5
5511	000020	TX4	=	BIT4
5512	000010	TX3	=	BIT3
5513	000004	TX2	=	BIT2
5514	000002	TX1	=	BIT1
5515	000001	TX0	=	BIT0


```
5516  
5517  
5518  
5519  
5520 000200  
5521 000010  
5522 000004  
5523 000002  
5524 000001  
5525  
5526  
5527  
5528  
5529 000200  
5530 000100  
5531 000040  
5532 000020  
5533 000010  
5534 000004  
5535 000002  
5536 000001  
5537 000226  
5538  
5539  
5540  
5541  
5542 000200  
5543 000100  
5544 000040  
5545 000020  
5546 000010  
5547 000004  
5548 000002  
5549 000001  
5550  
5551  
5552  
5553  
5554 000200  
5555 000100  
5556 000020  
5557 000010  
5558 000002  
5559 000001  
5560 000332  
5561  
5562  
5563  
5564  
5565 000200  
5566 000100  
5567 000040  
5568 000004  
5569 000002  
5570 000001  
5571
```

```
*****  
* AX1-16 - USYRT REG 3  
*****  
TERR = BIT7  
TXGA = BIT3  
TXAB = BIT2  
TEOM = BIT1  
TSOM = BIT0  
  
*****  
* AX2-15 - USYRT REG 4  
*****  
SYN7 = BIT7  
SYN6 = BIT6  
SYN5 = BIT5  
SYN4 = BIT4  
SYN3 = BIT3  
SYN2 = BIT2  
SYN1 = BIT1  
SYN0 = BIT0  
SYNCH = 226  
  
*****  
* AX2-16 - USYRT REG 5  
*****  
APA = BIT7  
DDC = BIT6  
STR = BIT5  
SEC = BIT4  
IDL = BIT3  
CRCTY2 = BIT2  
CRCTY1 = BIT1  
CRCTY0 = BIT0  
  
*****  
* AX3-15 - USYRT REG 6  
*****  
I422 = BIT7  
XYZ = BIT6  
V35 = BIT4  
INTGRL = BIT3  
OP = BIT1  
TEST = BIT0  
AX315U = I422!XYZ!V35!INTGRL!OP  
  
*****  
* AX3-16 - USYRT REG 7  
*****  
TXLEN2 = BIT7  
TXLEN1 = BIT6  
TXLEN0 = BIT5  
RXLEN2 = BIT2  
RXLEN1 = BIT1  
RXLEN0 = BIT0
```

```
5572
5573
5574
5575
5576
5577
5578
5579      004000
5580      00200C
5581      001000
5582      000400
5583
5584
5585
5586
5587
5588
5589
5590
5591      004000
5592      002000
5593      001000
5594      000400
5595
5596
5597
5598
5599
5600
5601
5602      002266
5603      002270
5604      002272
5605      002274
5606      002276
5607      002300
5608      002302
5609      002304
5610      002306
5611      002310
5612      002312
5613      002314
5614      002316
5615      002320
5616      002322
5617      002324
5618
5619
5620
5621
5622
5623      100000
5624
5625      100000
5626      100000
5627

;*****
;* TX CONTROL BITS DEFINED ON WORD BASIS
;*****
TXGOA    = BIT11
TXABT    = BIT10
TXEOM    = BIT9
TXSOM    = BIT8

;*****
;* RCV CONTROL BITS DEFINED ON WORD BASIS
;*****
RXOVR    = BIT11
RXABT    = BIT10
RXEBL    = BIT9
RXBCC    = BIT8

;*****
;* ADDRESS EQUATES FOR REGISTER STORAGE TABLE (LUREG:)
;*****
LUR10    = LUREG+0      ;LINE UNIT IBUS REG 10
LUR11    = LUREG+2      ;LINE UNIT IBUS REG 11
LUR12    = LUREG+4      ;LINE UNIT IBUS REG 12
LUR13    = LUREG+6      ;LINE UNIT IBUS REG 13
LUR14    = LUREG+10     ;LINE UNIT IBUS REG 14
LUR15    = LUREG+12     ;LINE UNIT IBUS REG 15
LUR16    = LUREG+14     ;LINE UNIT IBUS REG 16
LUR17    = LUREG+16     ;LINE UNIT IBUS REG 17
AX0.15   = LUREG+20     ;USYRT REG 0
AX0.16   = LUREG+22     ;USYRT REG 1
AX1.15   = LUREG+24     ;USYRT REG 2
AX1.16   = LUREG+26     ;USYRT REG 3
AX2.15   = LUREG+30     ;USYRT REG 4
AX2.16   = LUREG+32     ;USYRT REG 5
AX3.15   = LUREG+34     ;USYRT REG 6
AX3.16   = LUREG+36     ;USYRT REG 7

CHPCHK   = BIT15
BCCCHK   = BIT15
CRCCHK   = BIT15
```

5628 100000 TCCHEK BIT15

5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649

021000
122000

000001
000002

```
*****  
;* MICROINSTRUCTION DEFINITIONS  
*****  
MVIOX = 021000 ;MOVE IBUS TO OBUS*  
MVIXO = 122000 ;MOVE IBUS* TO OBUS
```

```
***** ERROR1 BIT FLAG DEFINITIONS *****  
RRDYTO = BIT0  
WRDYTO = BIT1
```

```
5651 .SBTTL GLOBAL DATA SECTION
5652
5653 :////////////////////
5654 :/ THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
5655 :/ IN MORE THAN ONE TEST.
5656 :////////////////////
5657
5658 :*****
5659 :* STORAGE FOR DEVICE REGISTERS
5660 :*****
5661 002266 000020 LUREG: .BLKW 16.
5662
5663 :*****
5664 :* MISCELLANEOUS STORAGE
5665 :*****
5666 002326 000000 SCRACH: .WORD 0 ;GEN'L PURPOSE SCRATCH WORD
5667 002330 000000 LOGDEV: .WORD 0 ;LOGICAL DEVICE NUMBER
5668 002332 000000 PSTACK: .WORD 0 ;CONTAINS BASE LEVEL PROGRAM STACK POINTER
5669 002334 000000 PPRIOR: .WORD 0 ;CPU PRIORITY FOR PRINTOUT
5670 002336 000000 SUBRPC: .WORD 0 ;PC OF SUBR CALL FOR ERROR REPORTS
5671 002340 000000 INTFLG: .WORD 0 ;INTERRUPT RECEIVED FLAGS
5672 ; BIT 0 FOR TX, BIT 1 FOR RCV
5673 002342 000000 ERRFLG: .WORD 0 ;SUBROUTINE ERROR FLAG
5674 002344 000000 TIMFLG: .WORD 0 ;EVENT TIME-OUT FLAG
5675 002346 000000 RETADR: .WORD 0 ;SUBR ERROR RETURN ADDRESS
5676 002350 000000 REDBYT: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM LU REG
5677 002352 000000 WRIBYT: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO LU REG
5678 002354 000000 RAX15: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 15
5679 002356 000000 RAX16: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 16
5680 002360 000000 WAX15: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 15
5681 002362 000000 WAX16: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 16
5682 002364 000000 REGNUM: .WORD 0 ;NUMBER (10-17) OF LINE UNIT REG BEING TESTED
5683 002366 000000 AXNUM: .WORD 0 ;NUMBER (0-7) OF EXTENDED REG BYTE BEING TESTED
5684 002370 000000 GOODAT: .WORD 0 ;STORAGE FOR EXPECTED DATA
5685 002372 000000 BADDAT: .WORD 0 ;STORAGE FOR ACTUAL DATA
5686 002374 000000 LOADAT: .WORD 0 ;CONTAINS TEST DATA LOADED INTO REG
5687 002376 000000 FRSTIM: .WORD 0 ;FLAG=0 IF PROGRAM JUST LOADED
5688 002400 000000 FRSPAS: .WORD 0 ;FLAG=0 IF FIRST PASS AFTER LOAD
5689 002402 000000 STARES: .WORD 0 ;FLAG=0 IF FIRST TIME THRU AFTER STA OR RES
5690 002404 000000 SAVE4: .WORD 0 ;SAVE LOC 4 HERE (ERROR TRAP VECTOR)
5691 002406 000000 SAVE6: .WORD 0 ;SAVE LOC 6 HERE (ERROR TRAP VECTOR)
5692 002410 000000 ERROR1: .WORD 0 ;SUBR ERROR BIT FLAGS (DEF'D IN GLOBAL EQUATES)
5693 002412 000000 TXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA TO LOAD INTO TX SILO
5694 002414 000000 RXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA READ FROM RCV SILO
5695 002416 000000 DISILO: .WORD 0 ;CONTAINS CURRENT STATE OF DISSI IN BIT 5
5696 002420 000000 CHPTYP: .WORD 0 ;USYRT CHIP TYPE, =0 FOR SIG, ELSE =1
5697 002422 000000 MODINT: .WORD 0 ;MODEM INTERFACE SELECTION
5698 002424 000000 SAVLEN: .WORD 0 ;SAVED TX AND RCV CHAR LENGTHS
5699 002426 000000 DEVMAP: .WORD 0 ;BIT MAP OF ACTIVE DEVICES
5700 002430 000000 DEVPTR: .WORD 0 ;DEVICE MAP BIT POINTER
5701 002432 000000 UNIT: .WORD 0 ;CONTAINS UNIT NUMBER (1 TO N)
5702 002434 000000 TSTNUM: .WORD 0 ;CONTAINS TEST NUMBER FOR SOME TESTS
5703
5704 :***** CURRENT DEVICE PARAMETERS *****
5705 002436 160170 MPCSR: .WORD 160170 ;POINTER TO MICROPROCESSOR CSR'S
5706 002440 160171 BSEL1: .WORD 160171 ;POINTER TO BSEL1
```

```
5707 002442      BSEL4:
5708 002442 160174 SEL4: .WORD 160174      ;POINTER TO SEL4
5709 002444 160176 SEL6: .WORD 160176      ;POINTER TO SEL6
5710 002446 000300 MPIVEC: .WORD 300      ;MICROPROCESSOR INPUT INTERRUPT VECTOR
5711 002450 000304 MPOVEC: .WORD 304      ;MICROPROCESSOR OUTPUT INTERRUPT VECTOR
5712 002452 000240 MPRIOR: .WORD 240      ;MICROPROCESSOR DEVICE PRIORITY
5713 002454 000000 LUSWI1: .WORD 0        ;LINE UNIT SWITCH PACK #1
5714 002456 000000 LUSWI2: .WORD 0        ;LINE UNIT SWITCH PACK #2
5715 002460 000000 LUSWI3: .WORD 0        ;LINE UNIT SWITCH PACK #3
5716 002462 000000 TSTCON: .WORD 0        ;TEST CONNECTOR INDICATOR
5717 002464 000004 BDRATE: .WORD 4        ;BAUD RATE
5718
5719 ;***** STORAGE FOR DATA READ IN ADDRESS TESTS *****
5720 002466 000 REDDAT: .BYTE 0
5721 002467 000 .BYTE 0
5722 002470 000 .BYTE 0
5723 002471 000 .BYTE 0
5724 002472 000 .BYTE 0
5725 002473 000 .BYTE 0
5726 002474 000 .BYTE 0
5727 002475 000 .BYTE 0
5728
5729 ;:***** GEN'L PURPOSE SCRATCH STORAGE *****
5730 002476 000000 REG0: .WORD 0
5731 002500 000000 REG1: .WORD 0
5732 002502 000000 REG2: .WORD 0
5733 002504 000000 REG3: .WORD 0
5734 002506 000000 REG4: .WORD 0
5735 002510 000000 REG5: .WORD 0
5736 002512 000000 REG6: .WORD 0
5737 002514 000000 REG7: .WORD 0
5738
5739 ;:***** SCRATCH STORAGE FOR MESSAGE REPORTING *****
5740 002516 000000 TMP0: .WORD 0
5741 002520 000000 TMP1: .WORD 0
5742 002522 000000 TMP2: .WORD 0
5743 002524 000000 TMP3: .WORD 0
5744 002526 000000 TMP4: .WORD 0
5745 002530 000000 TMP5: .WORD 0
5746 002532 000000 TMP6: .WORD 0
5747 002534 000000 TMP7: .WORD 0
5748
5749 ;***** INBUS LU REG BIT MASKS FOR UNPREDICTABLE BITS *****
5750 002536 UPBITS:
5751 002536 000 .BYTE 000      ;MASK FOR REG 10
5752 002537 056 .BYTE 056      ;MASK FOR REG 11
5753 002540 000 .BYTE 000      ;MASK FOR REG 12
5754 002541 257 .BYTE 257      ;MASK FOR REG 13
5755 002542 100 .BYTE 100      ;MASK FOR REG 14
5756 002543 377 .BYTE 377      ;MASK FOR REG 15
5757 002544 377 .BYTE 377      ;MASK FOR REG 16
5758 002545 306 .BYTE 306      ;MASK FOR REG 17
5759
5760 002546 200 R14NRW: .BYTE 200      ;REG 14 NON-R/W BITS
5761
5762 ;***** MASKS FOR EXTENDED REGISTER NON-READ/WRITE BITS *****
```

5763	002547		ANBITS:		
5764	002547	377	.BYTE	377	:MASK FOR AX0-15
5765	002550	377	.BYTE	377	:MASK FOR AX0-16
5766	002551	000	.BYTE	000	:MASK FOR AX1-15
5767	002552	360	.BYTE	360	:MASK FOR AX1-16
5768	002553	000	.BYTE	000	:MASK FOR AX2-15
5769	002554	000	.BYTE	000	:MASK FOR AX2-16
5770	002555	004	.BYTE	004	:MASK FOR AX3-15
5771	002556	030	.BYTE	030	:MASK FOR AX3-16

5772					
5773					
5774	002557		:***** DATA PATTERN A *****		
5775	002557	125	PATA:		
5776	002560	252	.BYTE	125	
5777	002561	000	.BYTE	252	
5778	002562	377	.BYTE	000	
5779	002563	001	.BYTE	377	
5780	002564	002	.BYTE	001	
5781	002565	004	.BYTE	002	
5782	002566	010	.BYTE	004	
5783	002567	020	.BYTE	010	
5784	002570	040	.BYTE	020	
5785	002571	100	.BYTE	040	
5786	002572	200	.BYTE	100	
5787	002573	376	.BYTE	200	
5788	002574	375	.BYTE	376	
5789	002575	373	.BYTE	375	
5790	002576	367	.BYTE	373	
5791	002577	357	.BYTE	367	
5792	002600	337	.BYTE	357	
5793	002601	277	.BYTE	337	
5794	002602	177	.BYTE	277	
5795			.BYTE	177	

5796					
5797	002603		:***** DATA PATTERN B *****		
5798	002603	000	PATB:		
5799	002604	000	.BYTE	000	
5800	002605	040	.BYTE	000	
5801	002606	100	.BYTE	040	
5802	002607	220	.BYTE	100	
5803	002610	000	.BYTE	220	
5804	002611	000	.BYTE	000	
5805	002612	051	.BYTE	000	
5806			.BYTE	051	

5807					
5808					
5809	002613	000	:***** DATA PATTERN Q *****		
5810	002614	120	PATQ:		
5811	002615	125	.BYTE	000	
5812	002616	137	.BYTE	120	
5813	002617	040	.BYTE	125	
5814	002620	052	.BYTE	137	
5815	002621	057	.BYTE	040	
5816	002622	177	.BYTE	052	
5817			.BYTE	057	
5818			.BYTE	177	

:***** DATA PATTERN R *****

5819	002623	000	PATR:	.BYTE	000
5820	002624	100		.BYTE	100
5821	002625	120		.BYTE	120
5822	002626	124		.BYTE	124
5823	002627	164		.BYTE	164
5824	002630	172		.BYTE	172
5825	002631	176		.BYTE	176
5826	002632	177		.BYTE	177
5827	002633	000		.BYTE	000
5828	002634	100		.BYTE	100
5829	002635	120		.BYTE	120
5830	002636	124		.BYTE	124
5831	002637	164		.BYTE	164
5832	002640	172		.BYTE	172
5833	002641	176		.BYTE	176

5834
5835 :***** DATA PATTERN S *****

5836	002642	000	PATS:	.BYTE	000
5837	002643	017		.BYTE	017
5838	002644	036		.BYTE	036
5839	002645	074		.BYTE	074
5840	002646	170		.BYTE	170
5841	002647	360		.BYTE	360
5842	002650	037		.BYTE	037
5843	002651	076		.BYTE	076
5844	002652	174		.BYTE	174
5845	002653	370		.BYTE	370
5846	002654	077		.BYTE	077
5847	002655	176		.BYTE	176
5848	002656	374		.BYTE	374
5849	002657	177		.BYTE	177
5850	002660	376		.BYTE	376
5851	002661	377		.BYTE	377

5852
5853 :***** DATA PATTERN T *****

5854	002662	000	PATT:	.BYTE	000
5855	002663	125		.BYTE	125
5856	002664	252		.BYTE	252
5857	002665	176		.BYTE	176
5858	002666	177		.BYTE	177

5859
5860 002667
5861 002670
5862
5863
5864
5865
5866

ENDPAT:
.EVEN

5867 :*** TEST MESSAGES TO BE TRANSMITTED ***
5868

5869	002670	000400	MSG1:	TXSOM
5870	002672	000400		TXSOM
5871	002674	000000		000
5872	002676	000125		125
5873	002700	000252		252
5874	002702	000377		377

5875	002704	000000	000
5876	002706	001000	TXEOM
5877	002710	001000	TXEOM
5878	002712	001000	TXEOM
5879	002714	001000	TXEOM
5880			
5881	002716	000400	MSG2: TXSOM
5882	002720	000400	TXSOM
5883	002722	000000	000
5884	002724	000377	377
5885	002726	001000	TXEOM
5886	002730	001000	TXEOM
5887			
5888	002732	000001	MSG3: 001
5889	002734	000001	001
5890	002736	000001	001
5891	002740	000001	001
5892	002742	002000	TXABT
5893	002744	000400	TXSOM
5894	002746	000400	TXSOM
5895	002750	000003	003
5896	002752	000003	003
5897	002754	000003	003
5898	002756	000003	003
5899	002760	000003	003

5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916

*** RECEIVED DATA BUFFER (64. WORDS) ***
RCVBUF: .BLKW 64.


```
6009 003356 006301 ASL R1
6010 003360 006301 ASL R1
6011 003362 052701 000004 BIS #4,R1 ;SET DESTINATION = BSEL4
6012 003366 052701 021000 BIS #MVIOX,R1 ;SET REST OF MOVE INSTRUCTION
6013 003372 010137 003402 MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
6014 003376 004737 003240 JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION
6015 003402 000000 2$: .WORD 0 ;INSTRUCTION GOES HERE
6016 003404 117737 177032 002350 MOVB @BSEL4,REDBYT ;GET LU REG CONTENTS INTO REDBYT
6017 003412 105037 002351 CLRB REDBYT+1 ;CLR HI BYTE OF STORAGE
6018 003416 012601 MOV (SP)+,R1 ;RESTORE R1
6019 003420 000207 RTS PC ;RETURN
6020
6021
6022
6023
6024
6025
6026 ;*****
6027 ;* WRITLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO
6028 ;* EXECUTE AN INSTRUCTION WHICH LOADS THE BYTE CONTAINED IN WRIBYT
6029 ;* INTO THE LU REG WHOSE NUMBER IS PASSED IN REGNUM,
6030 ;*****
6031 003422 010146 WRITLU: MOV R1,-(SP) ;SAVE R1
6032 003424 013701 002364 MOV REGNUM,R1 ;GET LINE UNIT REG NUMBER
6033 003430 052701 000100 BIS #100,R1 ;SET SOURCE = BSEL4
6034 003434 052701 122000 BIS #MVIXO,R1 ;SET REST OF MOVE INSTRUCTION
6035 003440 010137 003462 MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT
6036 003444 105037 002353 CLRB WRIBYT+1 ;CLR HI BYTE OF STORAGE
6037 003450 113777 002352 176764 MOVB WRIBYT,@BSEL4 ;LOAD BYTE INTO BSEL4
6038 003456 004737 003240 JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION
6039 003462 000000 2$: .WORD 0
6040 003464 012601 MOV (SP)+,R1 ;RESTORE R1
6041 003466 000207 RTS PC ;RETURN
6042
6043
6044
6045
6046
6047
6048 ;*****
6049 ;* GETREG - THIS SUBROUTINE READS THE LINE UNIT REGISTERS 10-17 INTO THE
6050 ;* REGISTER STORAGE TABLE (LUREG:).
6051 ;*****
6052 003470 010146 GETREG: MOV R1,-(SP) ;SAVE R1
6053 003472 013746 002364 MOV REGNUM,-(SP) ;SAVE CURRENT REG NO.
6054 003476 012701 002266 MOV #LUR10,R1 ;INIT POINTER TO REG STORAGE TABLE
6055 003502 012737 000010 002364 MOV #10,REGNUM ;INIT LU REG NO. TO 10
6056 003510 004737 003344 3$: JSR PC,READLU ;READ A LINE UNIT REG
6057 003514 113721 002350 MOVB REDBYT,(R1)+ ;PUT BYTE READ INTO TABLE
6058 003520 105021 CLRB (R1)+ ;CLEAR UPPER BYTE OF TABLE ENTRY
6059 003522 005237 002364 INC REGNUM ;INCREMENT REG NO.
6060 003526 023727 002364 000020 CMP REGNUM,#20 ;SEE IF ALL REGS READ YET
6061 003534 002765 BLT 3$ ;BR IF NOT
6062 003536 012637 002364 MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
6063 003542 012601 MOV (SP)+,R1 ;RESTORE R1
6064 003544 000207 RTS PC ;RETURN
```

6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077 003546
6078 003546 152777 000006 176664
6079 003554 017677 000000 176662
6080 003562 152777 000206 176650
6081 003570 062716 000002
6082 003574 U00207
6083
6084
6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095 003576 010146
6096 003600 013746 002364
6097 003604 042737 000001 002410
6098 003612 012737 000014 002364
6099 003620 113737 002366 002352
6100 003626 006237 002352
6101 003632 152737 000024 002352
6102 003640 053737 002416 002352
6103 003646 004737 003422
6104 003652 005001
6105 003654 004737 003344 6\$:
6106 003660 132737 000200 002350
6107 003666 001006
6108 003670 005201
6109 003672 001370
6110 003674 052737 000001 002410
6111 003702 000424
6112 003704 012737 000015 002364 9\$:
6113 003712 004737 003344
6114 003716 113737 002350 002354
6115 003724 105037 002355
6116 003730 012737 000016 002364
6117 003736 004737 003344
6118 003742 113737 002350 002356
6119 003750 105037 002357
6120 003754 012637 002364 12\$:

```
*****  
* LOOPIN - THIS SUBROUTINE PLACES THE MICROPROCESSOR IN A LOOP ON AN  
* INSTRUCTION, BY MOVING THE INSTRUCTION FROM THE WORD FOLLOWING THE CALL  
* INTO SEL6, AND SETTING RUN AND ROMI IN BSEL1. THE SUBROUTINE RETURNS  
* WITH THE MICROPROCESSOR STUCK IN THE LOOP, AND IF IT IS DESIRED TO  
* TERMINATE THE LOOP, THE PDP-11 PROGRAM MUST CLEAR THE RUN BIT IN  
* BSEL1, OR CALL SUBROUTINE MSTCLR TO DO THIS.  
*****
```

```
LOOPIN:  
BISB #ROMO,ROMI,@BSEL1 ;SET ROMO, ROMI BITS IN BSEL1  
MOV @ (SP),@SEL6 ;PUT MICROINSTRUCTION INTO SEL6  
BISB #RUN,ROMO,ROMI,@BSEL1 ;SET RUN, ROMO, ROMI IN BSEL1  
ADD #2,(SP) ;FIX UP RETURN PC  
RTS PC ;RETURN WITH MICROPROCESSOR STUCK IN SINGLE  
; INSTRUCTION LOOP
```

```
*****  
* READAX - THIS SUBROUTINE READS THE USYRT REG PAIR WHOSE NUMBER (0-3)  
* IS PASSED IN BITS 1,2 OF AXNUM ON ENTRY, AND RETURNS THE BYTES READ IN  
* RAX15 AND RAX16. IF THE LINE UNIT DOES NOT RESPOND WITH READY IN REG 14,  
* RRDYTO BIT IS SET IN ERROR1 ON RETURN.  
*****
```

```
READAX: MOV R1,-(SP) ;SAVE R1  
MOV REGNUM,-(SP) ;STORE CURRENT REG NO.  
BIC #RRDYTO,ERROR1 ;CLEAR ERROR BIT  
MOV #14,REGNUM ;SET LU REG NO. - 14  
MOVB AXNUM,WRIBYT ;SET UP AX REG NO. BITS  
ASR WRIBYT  
BISB #RDAX.ENAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14  
BIS DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT  
JSR PC,WRITLU ;SET RDAX AND ENAX IN REG 14  
CLR R1 ;INIT TIMER  
6$: JSR PC,READLU ;READ REG 14  
BITB #READY,REDBYT ;SEE IF READY BIT SET IN REG 14 YET  
BNE 9$ ;BR IF READY SET  
INC R1 ;INCR TIMER  
BNE 6$ ;BR IF TIMER DIDN'T TIME OUT YET  
BIS #RRDYTO,ERROR1 ;SET ERROR FLAG FOR TIME OUT ON READ RDY  
BR 12$ ;BR TO RETURN  
9$: MOV #15,REGNUM ;SET REG NO. = 15  
JSR PC,READLU ;READ REG 15  
MOVB REDBYT,RAX15 ;STORE REG AX-15  
CLRB RAX15+1 ;CLR HI BYTE OF STORAGE  
MOV #16,REGNUM ;SET REG NO. = 16  
JSR PC,READLU ;READ REG 16  
MOVB REDBYT,RAX16 ;STORE REG AX-16  
CLRB RAX16+1 ;CLR HI BYTE OF STORAGE  
12$: MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
```

```
6121 003760 012601          MOV    (SP)+,R1      ;RESTORE R1
6122 003762 000207          RTS     PC           ;RETURN
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133 003764 010146          *****
6134 003766 013746 002364          * WRITAX - THIS SUBROUTINE WRITES THE USYRT REG PAIR WHOSE NUMBER (0-3) IS
6135 003772 042737 000002 002410          * PASSED IN BITS 1,2 OF AXNUM ON ENTRY, WITH THE DATA FROM WAX15 AND
6136 004000 012737 000014 002364          * WAX16. IF LINE UNIT DOES NOT RESPOND WITH READY IN REG 14, WRDYTO BIT
6137 004006 113737 002366 002352          * IS SET IN ERROR1 ON RETURN.
6138 004014 006237 002352          *****
6139 004020 053737 002416 002352          WRITAX: MOV    R1,-(SP)      ;SAVE R1
6140 004026 004737 003422          MOV    REGNUM,-(SP)     ;SAVE CURRENT REG NO.
6141 004032 012737 000015 002364          BIC    #WRDYTO,ERROR1   ;CLEAR ERROR BIT
6142 004040 105037 002361          MOV    #14,REGNUM      ;SET LU REG NO. = 14
6143 004044 113737 002360 002352          MOV    AXNUM,WRIBYT    ;SET AX REG NO. BITS
6144 004052 004737 003422          ASR    WRIBYT
6145 004056 005237 002364          BIS    DISILO,WRIBYT    ;SET PROPER STATE OF DISSI BIT
6146 004062 105037 002363          JSR    PC,WRITLU       ;SET AX NO. BITS IN REG 14
6147 004066 113737 002362 002352          MOV    #15,REGNUM      ;SET REG NO. = 15
6148 004074 004737 003422          CLRB  WAX15+1          ;CLR HI BYTE OF STORAGE
6149 004100 012737 000014 002364          MOV    WAX15,WRIBYT    ;SET UP BYTE TO WRITE INTO REG 15
6150 004106 113737 002366 002352          JSR    PC,WRITLU       ;WRITE BYTE INTO REG 15
6151 004114 006237 002352          INC    REGNUM          ;SET REG NO. = 16
6152 004120 152737 000014 002352          CLRB  WAX16+1          ;CLR HI BYTE OF STORAGE
6153 004126 053737 002416 002352          MOV    WAX16,WRIBYT    ;SET UP BYTE TO WRITE INTO REG 16
6154 004134 004737 003422          JSR    PC,WRITLU       ;WRITE BYTE INTO REG 16
6155 004140 005001          MOV    #14,REGNUM      ;SET REG NO. = 14
6156 004142 004737 003344 6$:          MOV    AXNUM,WRIBYT    ;SET AX REG NO. BITS
6157 004146 132737 000200 002350          ASR    WRIBYT
6158 004154 001005          BISB  #ENAX!WAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14
6159 004156 005201          BIS   DISILO,WRIBYT    ;SET PROPER STATE OF DISSI BIT
6160 004160 001370          JSR   PC,WRITLU       ;SET ENAX AND WAX IN REG 14
6161 004162 052737 000002 002410          CLR   R1              ;INIT PROGRAM TIMER
6162 004170 012637 002364 9$:          JSR   PC,READLU       ;READ REG 14
6163 004174 012601          BITB  #READY,REDBYT    ;SEE IF READY BIT SET IN REG 14 YET
6164 004176 000207          BNE   9$              ;BR IF READY SET
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174 004200 010146          *****
6175 004202 013746 002366          * GETALL - THIS SUBROUTINE READS THE LINE UNIT REGS 10-17 AND THE EXTENDED
6176 004206 012737 015172 002516          * REGISTERS AX0-AX3 INTO REGISTER STORAGE TABLE (LUREG:).
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
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
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350
6351
6352
6353
6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378
6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507
6508
6509
6510
6511
6512
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526
6527
6528
6529
6530
6531
6532
6533
6534
6535
6536
6537
6538
6539
6540
6541
6542
6543
6544
6545
6546
6547
6548
6549
6550
6551
6552
6553
6554
6555
6556
6557
6558
6559
6560
6561
6562
6563
6564
6565
6566
6567
6568
6569
6570
6571
6572
6573
6574
6575
6576
6577
6578
6579
6580
6581
6582
6583
6584
6585
6586
6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599
6600
6601
6602
6603
6604
6605
6606
6607
6608
6609
6610
6611
6612
6613
6614
6615
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630
6631
6632
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643
6644
6645
6646
6647
6648
6649
6650
6651
6652
6653
6654
6655
6656
6657
6658
6659
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677
6678
6679
6680
6681
6682
6683
6684
6685
6686
6687
6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703
6704
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717
6718
6719
6720
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730
6731
6732
6733
6734
6735
6736
6737
6738
6739
6740
6741
6742
6743
6744
6745
6746
6747
6748
6749
6750
6751
6752
6753
6754
6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774
6775
6776
6777
6778
6779
6780
6781
6782
6783
6784
6785
6786
6787
6788
6789
6790
6791
6792
6793
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839
6840
6841
6842
6843
6844
6845
6846
6847
6848
6849
6850
6851
6852
6853
6854
6855
6856
6857
6858
6859
6860
6861
6862
6863
6864
6865
6866
6867
6868
6869
6870
6871
6872
6873
6874
6875
6876
6877
6878
6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912
6913
6914
6915
6916
6917
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943
6944
6945
6946
6947
6948
6949
6950
6951
6952
6953
6954
6955
6956
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967
6968
6969
6970
6971
6972
6973
6974
6975
6976
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991
6992
6993
6994
6995
6996
6997
6998
6999
7000
```

```

6177 004214 032737 000001 002366      BIT      #BIT0,AXNUM      ;SEE IF LO OR HI BYTE
6178 004222 001403                BEQ      1$              ;BR IF LO BYTE
6179 004224 012737 015175 002516      MOV      #DH6,TMP0      ;SET AX HI BYTE NO.
6180 004232 004737 003470                JSR      PC,GETREG      ;READ AND STORE REGS 10-17
6181 004236 142777 000010 176174      BICB    #LULOO, @BSEL1 ;CLEAR LULOO
6182 004244 012701 002306      MOV      #AX0.15,R1    ;INIT POINTER TO REG STORAGE TABLE
6183 004250 005037 002366      CLR      AXNUM         ;INIT AX REG BYTE NO. TO 0
6184 004254 004737 003576      JSR      PC,READAX     ;READ 2 AX REG BYTES
6185 004260 113721 002354      MOVVB   RAX15,(R1)+    ;PUT LO BYTE READ INTO TABLE
6186 004264 105021                CLRB    (R1)+          ;CLEAR UPPER BYTE OF TABLE ENTRY
6187 004266 113721 002356      MOVVB   RAX16,(R1)+    ;PUT HI BYTE READ INTO TABLE
6188 004272 105021                CLRB    (R1)+          ;CLEAR UPPER BYTE OF TABLE ENTRY
6189 004274 062737 000002 002366      ADD      #2,AXNUM      ;INCR AX REG BYTE NO.
6190 004302 023727 002366 000010      CMP      AXNUM,#10     ;SEE IF ALL REGS READ YET
6191 004310 002761                BLT     3$              ;BR IF NOT
6192 004312 012637 002366      MOV      (SP)+,AXNUM   ;RESTORE CURRENT AX REG BYTE NO.
6193 004316 012601                MOV      (SP)+,R1      ;RESTORE R1
6194 004320 013737 002366 002520      MOV      AXNUM,TMP1
6195 004326 006237 002520      ASR     TMP1           ;GET EXTENDED REG NO. FOR PRINTOUT
6196 004332 000207                RTS      PC              ;RETURN
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209

```

```

*****
* OSIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ORDY (REG 11)
* AND OCOR (REG 17) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET
* AS PASSED IN BIT 0 (ORDY) AND BIT 1 (OCOR) OF THE WORD FOLLOWING THE
* CALL.
* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS IN
* RETADR.
*****

```

```

6210 004334 013746 002364      OSIRDY: MOV      REGNUM,-(SP) ;SAVE LU REG NO.
6211 004340 013746 002336      MOV      SUBRPC,-(SP)
6212 004344 005737 002336      TST     SUBRPC         ;SEE IF THIS IS A NESTED CALL
6213 004350 001006                BNE     1$              ;BR IF YES
6214 004352 016637 000004 002336      MOV      4(SP),SUBRPC
6215 004360 162737 000004 002336      SUB     #4,SUBRPC      ;GET PC OF SUBROUTINE CALL
6216 004366 012737 000011 002364      JSR     #11,REGNUM     ;SET REG NO. TO 11
6217 004374 004737 003344      JSR     PC,READLU     ;READ REG 11
6218 004400 032776 000001 000004      BIT     #BIT0,@4(SP)  ;GET EXPECTED STATE OF ORDY
6219 004406 001413                BEQ     3$              ;BR IF EXPECTED ORDY = 0
6220 004410 132737 000020 002350      BITB   #ORDY,REDBYT   ;SEE IF ORDY = 1
6221 004416 001022                BNE     9$              ;BR IF ORDY = 1
6222 004420 004737 004200      JSR     PC,GETALL     ;GET REGS FOR PRINTOUT
6223
6224 004424                ;REPORT ORDY NOT SET
        ERRDF 7,EM7,ERR4
        (4) 004424 104455
        (5) 004426 000007
        (5) 004430 013304
        (5) 004432 016356
6225 004434 000451
6226 004436 132737 000020 002350      BR      16$           ;TAKE ERROR RETURN
6227 004444 001407                BITB   #ORDY,REDBYT   ;SEE IF ORDY = 0
6228 004446 004737 004200      BEQ     9$              ;BR IF ORDY = 0
        JSR     PC,GETALL ;GET REGS FOR PRINTOUT

```

```

TRAP  CSERDF
.WORD 7
.WORD EM7
.WORD ERR4

```

```
6229 ;REPORT ORDY NOT CLEARED
6230 004452 ERRDF 8,EM8,ERR4
(4) 004452 104455 TRAP C$ERDF
(5) 004454 000010 .WORD 8
(5) 004456 013321 .WORD EM8
(5) 004460 016356 .WORD ERR4
6231 004462 000436 BR 16$ ;TAKE ERROR RETURN
6232 004464 012737 000017 002364 9$: MOV #17,REGNUM ;SET REG NO. = 17
6233 004472 004737 003344 JSR PC,READLU ;READ LU REG 17
6234 004476 132776 000002 000004 BITB #BIT1,24(SP) ;GET EXPECTED STATE OF OCOR
6235 004504 001413 BEQ 12$ ;BR IF EXPECTED OCOR = 0
6236 004506 132737 000020 002350 BITB #OCOR,REDBYT ;SEE IF OCOR = 1
6237 004514 001031 BNE 20$ ;BR IF OCOR = 1
6238 004516 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6239 ;REPORT OCOR NOT SET
6240 004522 ERRDF 9,EM9,ERR4
(4) 004522 104455 TRAP C$ERDF
(5) 004524 000011 .WORD 9
(5) 004526 013342 .WORD EM9
(5) 004530 016356 .WORD ERR4
6241 004532 000412 BR 16$ ;TAKE ERROR RETURN
6242 004534 132737 000020 002350 12$: BITB #OCOR,REDBYT ;SEE IF OCOR = 0
6243 004542 001416 BEQ 20$ ;BR IF OCOR = 0
6244 004544 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6245 ;REPORT OCOR NOT CLEARED
6246 004550 ERRDF 10,EM10,ERR4
(4) 004550 104455 TRAP C$ERDF
(5) 004552 000012 .WORD 10
(5) 004554 013357 .WORD EM10
(5) 004556 016356 .WORD ERR4
6247 004560 016637 000002 002364 16$: MOV 2(SP),REGNUM ;RESTORE LU REG NO.
6248 004566 013706 002332 MOV PSTACK,SP ;RESTORE STACK POINTER TO BASE LEVEL
6249 004572 013746 002346 MOV RETADR,-(SP) ;FIX ERROR RETURN PC
6250 004576 000407 BR 23$
6251 004600 062766 000002 000004 20$: ADD #?,4(SP) ;FIX UP ERROR-FREE RETURN PC
6252 004606 012637 002336 MOV (SP)+,SUBRPC
6253 004612 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6254 004616 000207 23$: RTS PC ;RETURN
6255
6256
6257
6258
6259
6260
6261 ;*****
6262 ;* WAIT50 - THIS SUBROUTINE STALLS FOR AT LEAST 50 MICRO-SEC, AND THEN RETURNS.
6263 ;*****
6263 004620 010146 WAIT50: MOV R1,-(SP) ;SAVE R1
6264 004622 012701 000310 MOV #200.,R1 ;INIT COUNTER
6265 004626 005301 3$: DEC R1 ;DECREMENT COUNTER
6266 004630 001376 BNE 3$ ;BR IF NOT DONE YET
6267 004632 012601 MOV (SP)+,R1 ;RESTORE R1
6268 004634 000207 RTS PC ;RETURN
6269
6270
6271
6272
```

6273
6274
6275
6276
6277 004636 000240
6278 004640 000240
6279 004642 000240
6280 004644 000207
6281
6282
6283
6284
6285
6286
6287
6288
6289

* STALL - THIS SUBROUTINE STALLS FOR ABOUT A MICRO-SEC.

STALL: NOP
NOP
NOP
RTS PC

6290 004646 013746 002364
6291 004652 042737 170000 002412
6292 004660 012737 000011 002364
6293 004666 113737 002413 002352
6294 004674 004737 003422
6295 004700 012737 000010 002364
6296 004706 113737 002412 002352
6297 004714 004737 003422
6298 004720 012637 002364
6299 004724 000207
6300
6301
6302
6303
6304
6305

* LDTXSI - THIS SUBROUTINE LOADS THE TX SILO (REGS 10,11) WITH THE DATA PASSED
* IN BITS 0-11 OF TXWORD.

LDTXSI: MOV REGNUM, -(SP) ;SAVE LU REG NO.
BIC #170000, TXWORD ;CLEAR UNUSED BITS
MOV #11, REGNUM ;SET REG NO. = 11
MOVB TXWORD+1, WRIBYT ;SET DATA TO BE WRITTEN INTO REG 11
JSR PC, WRITLU ;LOAD DATA INTO REG 11
MOV #10, REGNUM ;SET REG NO. = 10
MOVB TXWORD, WRIBYT ;SET DATA TO BE WRITTEN INTO REG 10
JSR PC, WRITLU ;LOAD DATA INTO REG 10
MOV (SP)+, REGNUM ;RESTORE LU REG NO.
RTS PC ;RETURN

6311 004726 010146
6312 004730 017601 000002
6313 004734 001426
6314 004736 100006
6315 004740 042701 100000
6316 004744 005737 002420
6317 004750 001401
6318 004752 005301
6319 004754 152777 000010 175456 2\$:
6320 004762 152777 000020 175450 3\$:
6321 004770 004737 004636
6322 004774 142777 000020 175436
6323 005002 004737 004636
6324 005006 005301
6325 005010 001364
6326 005012 062766 000002 000002 6\$:
6327 005020 012601
6328 005022 000207

* STPLU - THIS SUBROUTINE CLOCKS THE LINE UNIT FOR THE NO. OF CYCLES PASSED
* IN BITS 0-14 OF THE WORD FOLLOWING THE CALL.
* IF BIT 15 = 1, A CHECK IS MADE TO DETERMINE IF THE USYRT CHIP TYPE
* REQUIRES DECREMENTING THE NO. OF CYCLES BY 1.

STPLU: MOV R1, -(SP) ;SAVE R1
MOV @2(SP), R1 ;GET DESIRED NO. OF CYCLES
BEQ 6\$;IF DESIRED CYCLES = 0, RETURN
BPL 2\$;BR IF CHIP TYPE CHECK NOT NECESSARY
BIC #BIT15, R1 ;CLEAR FLAG BIT
TST CHPTYP ;SEE IF SIG USYRT
BEQ 2\$;BR IF YES
DEC R1 ;DECREMENT CYCLE COUNT
BISB #LULOOP, @BSEL1 ;SET LU LOOP BIT
BISB #STEPLU, @BSEL1 ;SET THE STEPLU BIT (CLOCK THE TRANSMITTER)
JSR PC, STALL ;STALL
BICB #STEPLU, @BSEL1 ;CLEAR THE STEPLU BIT (CLOCK THE RECEIVER)
JSR PC, STALL ;STALL
DEC R1 ;DECREMENT CYCLE COUNTER
BNE 3\$;BR IF NOT DONE YET
ADD #2, 2(SP) ;FIX UP RETURN PC
MOV (SP)+, R1 ;RESTORE R1
RTS PC ;RETURN


```
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339 005024 013746 002364
6340 005030 013746 002336
6341 005034 005737 002336
6342 005040 001006
6343 005042 016637 000004 002336
6344 005050 162737 000004 002336
6345 005056 012737 000011 002364
6346 005064 004737 003344
6347 005070 032776 000001 000004
6348 005076 001413
6349 005100 132737 000100 002350
6350 005106 001031
6351 005110 004737 004200
6352
6353 005114
(4) 005114 104455
(5) 005116 000013
(5) 005120 013400
(5) 005122 016356
6354 005124 000412
6355 005126 132737 000100 002350
6356 005134 001416
6357 005136 004737 004200
6358
6359 005142
(4) 005142 104455
(5) 005144 000014
(5) 005146 013415
(5) 005150 016356
6360 005152 016637 000002 002364
6361 005160 013706 002332
6362 005164 013746 002346
6363 005170 000407
6364 005172 062766 000002 000004
6365 005200 012637 002336
6366 005204 012637 002364
6367 005210 000207
6368
6369
6370
6371
6372
6373
6374
6375
6376
```

```

:*****
:* OACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF OACT (REG 11) AND
:* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
:* WORD FOLLOWING THE CALL.
:*****
OACTIV: MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV SUBRPC,-(SP)
TST SUBRPC ;SEE IF THIS IS A NESTED CALL
BNE 1$ ;BR IF YES
MOV 4(SP),SUBRPC ;GET PC OF SUBROUTINE CALL
SUB #4,SUBRPC ;SET REG NO. = 11
1$: MOV #11,REGNUM ;READ REG 11
JSR PC,READLU ;GET EXPECTED STATE OF OACT
BIT #BIT0,@4(SP) ;BR IF EXPECTED OACT = 0
BEQ 3$ ;SEE IF OACT = 1
BITB #OACT,REDBYT ;BR IF OACT = 1
BNE 9$ ;GET REGS FOR PRINTOUT
JSR PC,GETALL
;REPORT OACT NOT SET
ERRDF 11,EM11,ERR4
TRAP C$ERDF
.WORD 11
.WORD EM11
.WORD ERR4
BR 6$ ;TAKE ERROR RETURN
3$: BITB #OACT,REDBYT ;SEE IF OACT = 0
BEQ 9$ ;BR IF OACT = 0
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT OACT NOT CLEARED
ERRDF 12,EM12,ERR4
TRAP C$ERDF
.WORD 12
.WORD EM12
.WORD ERR4
6$: MOV 2(SP),REGNUM ;RESTORE LU REG NO.
MOV PSTACK,SP ;RESTORE PROGRAM STACK TO BASE LEVEL
MOV RETADR,-(SP) ;FIX UP ERROR RETURN PC
BR 12$
9$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
MOV (SP)+,SUBRPC
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
12$: RTS PC ;RETURN
:*****
:* INITRN - THIS SUBROUTINE INITIATES TRANSMISSION OF A MESSAGE, BY DOING A
:* MASTER CLEAR, LOADING AX2-15 AND REG 17 WITH THE DATA PASSED IN THE 2
:* WORDS FOLLOWING THE CALL, LOADING 2 SOM CHARS INTO THE TX SILO, AND
```



```
6433 005540 004737 004334 JSR PC,OSIRDY ;CHK FOR ORDY = 1, OCOR = 0
6434 005544 000001 1 ADD #2,6(SP) ;FIX UP RETURN PC
6435 005546 062766 000002 000006 ADD (SP)+,AXNUM ;RESTORE AX BYTE NO.
6436 005554 012637 002366 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6437 005560 012637 002364 MOV (SP)+,R1 ;RESTORE R1
6438 005564 012601 002336 CLR SUBRPC ;CLEAR SUBR CALL PC
6439 005566 005037 002336 RTS PC ;RETURN
6440 005572 000207
```

6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488

```
*****  
* TXCHAR - THIS SUBROUTINE INITIATES TRANSMISSION OF A CHARACTER, BY LOADING  
* THE TX SILO WITH DATA PASSED IN BITS 0-11 OF THE WORD FOLLOWING THE CALL  
* AND CLOCKS THE LINE UNIT WITH THE NUMBER OF CYCLES PASSED IN BITS 0-14  
* OF THE SECOND WORD FOLLOWING THE CALL. IF BIT 15 = 1, A CHK IS MADE TO  
* DETERMINE IF THE USYRT CHIP TYPE REQUIRES DECREMENTING THE NO. OF CYCLES  
* BY 1. THE PROGRAM CHECKS FOR VALID STATES OF ORDY,  
* OCOR, AND OACT THROUGHOUT THE PROCESS.  
* IF AN ERROR IS DETECTED, A RETURN IS MADE TO THE TEST, AT THE ADDRESS  
* CONTAINED IN RETADR.  
*****
```

```
TXCHAR: MOV R1,-(SP) ;SAVE R1  
MOV R2,-(SP) ;SAVE R2  
MOV 4(SP),SUBRPC ;GET PC OF SUBR CALL  
SUB #4,SUBRPC ;GET DATA TO BE TRANSMITTED  
MOV @4(SP),TXWORD ;LOAD THE TX SILO WITH THE DATA  
JSR PC,LDTXSI ;WAIT FOR DATA TO RIPPLE DOWN SILO  
JSR PC,WAIT50 ;INCR POINTER  
ADD #2,4(SP) ;INIT CYCLE COUNT  
CLR R1 ;GET DESIRED NO. OF CYCLES  
MOV @4(SP),R2 ;SEE IF CHIP TYPE CHK SHOULD BE MADE  
TST R2 ;BR IF NOT  
BPL 9$ ;CLEAR FLAG BIT  
BIC #BIT15,R2 ;SEE IF SIG USYRT  
TST CHPTYP ;BR IF YES  
BEQ 9$ ;DECREMENT NO. OF CYCLES  
DEC R2 ;CHK OACT = 1  
9$: JSR PC,OACTIV ;SEE IF REQUIRED CYCLES DONE YET  
1 ;BR IF YES  
CMP R1,R2 ;CHK ORDY=1, OCOR=1  
BEQ 12$ ;STEP LU ONE CYCLE  
JSR PC,STPLU ;INCR CYCLE COUNT  
1 ;BR IF YES  
INC R1 ;CHK ORDY=1, OCOR=0  
BR 9$ ;FIX UP RETURN PC  
12$: JSR PC,OSIRDY ;CLEAR SUBR CALL PC  
1 ;RESTORE R2  
ADD #2,4(SP) ;RESTORE R1  
CLR SUBRPC  
MOV (SP)+,R2  
MOV (SP)+,R1
```

6489 005744 000207 RTS PC ;RETURN

6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507
6508
6509
6510
6511
6512
6513
6514
6515
6516
6517
(4)
(5)
(5)
(5)
6518
6519
6520
6521
6522
6523
(4)
(5)
(5)
(5)
6524
6525
6526
6527
6528
6529
6530
6531
6532
6533
(4)
(5)
(5)

005744 000207
005746 013746 002364
005752 013746 002336
005756 005737 002336
005762 001006
005764 016637 000004 002336
005772 162737 000004 002336
006000 012737 000012 002364
006006 004737 003344
006012 032776 000002 000004
006020 001413
006022 132737 000020 002350
006030 001022
006032 004737 004200
006036
006036 104455
006040 000021
006042 013436
006044 016356
006046 000451
006050 132737 000020 002350
006056 001407
006060 004737 004200
006064
006064 104455
006066 000022
006070 013453
006072 016356
006074 000436
006076 012737 000017 002364
006104 004737 003344
006110 132776 000001 000004
006116 001413
006120 132737 000010 002350
006126 001031
006130 004737 004200
006134
006134 104455
006136 000023
006140 013474

```
*****  
* ISIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ICIR (REG 17)  
* AND IRDY (REG 12) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET  
* AS PASSED IN BIT 0 (ICIR) AND BIT 1 (IRDY) OF THE WORD FOLLOWING THE  
* CALL.  
* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS  
* IN RETADR.  
*****  
ISIRDY: MOV REGNUM,-(SP) ;SAVE LU REG NO.  
MOV SUBRPC,-(SP)  
TST SUBRPC ;SEE IF THIS IS A NESTED CALL  
BNE 1$ ;BR IF YES  
MOV 4(SP),SUBRPC  
SUB #4,SUBRPC ;GET PC OF SUBR CALL  
1$: MOV #12,REGNUM ;SET REG NO. TO 12  
JSR PC,READLU ;READ REG 12  
BIT #BIT1,@4(SP) ;GET EXPECTED STATE OF IRDY  
BEQ 3$ ;BR IF EXPECTED IRDY = 0  
BITB #IRDY,REDBYT ;SEE IF IRDY = 1  
BNE 9$ ;BR IF IRDY = 1  
JSR PC,GETALL ;GET REGS FOR PRINTOUT  
;REPORT IRDY NOT SET  
ERRDF 17,EM17,ERR4  
  
TRAP C$ERDF  
.WORD 17  
.WORD EM17  
.WORD ERR4  
  
BR 16$ ;TAKE ERROR EXIT  
3$: BITB #IRDY,REDBYT ;SEE IF IRDY = 0  
BEQ 9$ ;BR IF IRDY = 0  
JSR PC,GETALL ;GET REGS FOR PRINTOUT  
;REPORT IRDY NOT CLEARED  
ERRDF 18,EM18,ERR4  
  
TRAP C$ERDF  
.WORD 18  
.WORD EM18  
.WORD ERR4  
  
BR 16$ ;TAKE ERROR RETURN  
9$: MOV #17,REGNUM ;SET REG NO. = 17  
JSR PC,READLU ;READ REG 17  
BITB #BIT0,@4(SP) ;GET EXPECTED STATE OF ICIR  
BEQ 12$ ;BR IF EXPECTED ICIR = 0  
BITB #ICIR,REDBYT ;SEE IF ICIR = 1  
BNE 20$ ;BR IF ICIR = 1  
JSR PC,GETALL ;GET REGS FOR PRINTOUT  
;REPORT ICIR NOT SET  
ERRDF 19,EM19,ERR4  
  
TRAP C$ERDF  
.WORD 19  
.WORD EM19
```

```
(5) 006142 016356 .WORD ERR4
6534 006144 000412
6535 006146 132737 000010 002350 12$: BR 16$ ;TAKE ERROR RETURN
;SEE IF ICIR = 0
6536 006154 001416 BEQ 20$ ;BR IF ICIR = 0
6537 006156 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6538 ;REPORT ICIR NOT CLEARED
6539 006162 FRRDF 20,EM20,ERR4
(4) 006162 104455 TRAP C$ERDF
(5) 006164 000024 .WORD 20
(5) 006166 013511 .WORD EM20
(5) 006170 016356 .WORD ERR4
6540 006172 016637 000002 002364 16$: MOV 2(SP),REGNUM ;RESTORE LU REG NO.
6541 006200 013706 002332 MOV PSTACK,SP ;RESTORE STACK POINTER TO BASE LEVEL
6542 006204 013746 002346 MOV RETADR,-(SP) ;FIX ERROR RETURN PC
6543 006210 000407 BR 23$
6544 006212 062766 000002 000004 20$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
6545 006220 012637 002336 MOV (SP)+,SUBRPC
6546 006224 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6547 006230 000207 23$: RTS PC ;RETURN
6548
6549
6550
6551
6552
6553
6554 ;*****
6555 ;* IACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF IACT (REG 12) AND
6556 ;* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
6557 ;* WORD FOLLOWING THE CALL.
6558 ;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN
6559 ;* RETADR.
6560 ;*****
6560 006232 013746 002364 IACTIV: MOV REGNUM,-(SP) ;SAVE LU REG NO.
6561 006236 013746 002336 MOV SUBRPC,-(SP)
6562 006242 005737 002336 TST SUBRPC ;SEE IF THIS IS A NESTED CALL
6563 006246 001006 BNE 1$ ;BR IF YES
6564 006250 016637 000004 002336 MOV 4(SP),SUBRPC
6565 006256 162737 000004 002336 SUB #4,SUBRPC ;GET PC OF SUBR CALL
6566 006264 012737 000012 002364 1$: MOV #12,REGNUM ;SET REG NO. = 12
6567 006272 004737 003344 JSR PC,READLU ;READ REG 12
6568 006276 032776 000001 000004 BIT #BIT0,@4(SP) ;GET EXPECTED STATE OF IACT
6569 006304 001413 BEQ 3$ ;BR IF EXPECTED IACT = 0
6570 006306 132737 000100 002350 BITB #IACT,REDBYT ;SEE IF IACT = 1
6571 006314 001031 BNE 9$ ;BR IF IACT = 1
6572 006316 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6573 ;REPORT IACT NOT SET
6574 006322 FRRDF 21,EM21,ERR4
(4) 006322 104455 TRAP C$ERDF
(5) 006324 000025 .WORD 21
(5) 006326 013532 .WORD EM21
(5) 006330 016356 .WORD ERR4
6575 006332 000412
6576 006334 132737 000100 002350 3$: BR 6$ ;TAKE ERROR EXIT
6577 006342 001416 BEQ 9$ ;SEE IF IACT = 0
6578 006344 004737 004200 JSR PC,GETALL ;BR IF IACT = 0
6579 ;REPORT IACT NOT CLEARED ;GET REGS FOR PRINTOUT
6580 006350 FRRDF 22,EM22,ERR4
```



```
6625 006566 001031  
6626 006570 004737 004200  
6627  
6628 006574  
  (4) 006574 104455  
  (5) 006576 000037  
  (5) 006600 013647  
  (5) 006602 017546  
6629 006604 000412  
6630 006606 132737 000002 002356 12$: BR 16$ :TAKE ERROR RETURN  
6631 006614 001416 BITB #REOM,RAX16 :SEE IF REOM = 0  
6632 006616 004737 004200 BEQ 20$ :BR IF REOM = 0  
6633 :REPORT JSR PC,GETALL :GET REGS FOR PRINTOUT  
6634 006622 :REPORT REOM NOT SET  
  (4) 006622 104455 ERRDF 31,EM31,ERR6  
  (5) 006624 000036 TRAP C$ERDF  
  (5) 006626 013626 .WORD 31  
  (5) 006630 017546 .WORD EM31  
6635 006632 016637 000002 002366 16$: MOV 2(SP),AXNUM :RESTORE AX BYTE NO.  
6636 006640 013706 002332 MOV PSTACK,SP :RESTORE STACK POINTER TO BASE LEVEL  
6637 006644 013746 002346 MOV RETADR,-(SP) :FIX ERROR RETURN PC  
6638 006650 000407 BR 23$  
6639 006652 062766 000002 000004 20$: ADD #2,4(SP) :FIX UP ERROR-FREE RETURN PC  
6640 006660 012637 002336 MOV (SP)+,SUBRPC  
6641 006664 012637 002366 MOV (SP)+,AXNUM :RESTORE AX BYTE NO.  
6642 006670 000207 23$: RTS PC :RETURN
```

6643
6644
6645
6646
6647
6648

```
6649 :*****  
6650 :* RDRXSI - THIS SUBROUTINE READS THE RCV SILO (REGS 10,12) AND RETURNS THE  
6651 :* SILO ENTRY IN BITS 0-11 OF RXWORD.  
6652 :*****
```

```
6652 006672 013746 002364 RDRXSI: MOV REGNUM,-(SP) :SAVE LU REG NO.  
6653 006676 012737 000012 002364 MOV #12,REGNUM :SET REG NO. = 12  
6654 006704 004737 003344 JSR PC,READLU :READ LU REG 12  
6655 006710 113737 002350 002415 MOV# REDBYT,RXWORD+1 :GET HI BITS OF SILO ENTRY  
6656 006716 042737 170000 002414 BIC #170000,RXWORD :CLEAR UNUSED BITS  
6657 006724 012737 000010 002364 MOV #10,REGNUM :SET REG NO. = 10  
6658 006732 004737 003344 JSR PC,READLU :READ REG 10  
6659 006736 113737 002350 002414 MOV# REDBYT,RXWORD :GET LOW BITS OF SILO ENTRY  
6660 006744 012637 002364 MOV (SP)+,REGNUM :RESTORE LU REG NO.  
6661 006750 000207 RTS PC :RETURN
```

6662
6663
6664
6665
6666
6667

```
6668 :*****  
6669 :* RCV1ST - THIS SUBROUTINE RECEIVES THE FIRST CHAR OF A MESSAGE, AND MONITORS  
6670 :* STATUS OF THE RECEIVER. FIRST, A CHECK IS MADE FOR IACT = 0, IRDY = 0,  
6671 :* ICIR = 1, AND RSOM = 0. THEN, THE LINE UNIT IS CLOCKED USING  
6672 :* STEPLU UNTIL IRDY = 1. THE PROGRAM CHECKS FOR THIS TO OCCUR WITHIN 3  
:* CYCLES AFTER THE NO. OF CYCLES PASSED IN THE WORD FOLLOWING THE CALL.
```

```
6673          ;*      IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
6674          ;*      CONTAINED IN RETADR.
6675          ;*****
6676 006752 010146 RCV1ST: MOV R1,-(SP)      ;SAVE R1
6677 006754 010346      MOV R3,-(SP)      ;SAVE R3
6678 006756 013746 002364      MOV REGNUM,-(SP)    ;SAVE LU REG NO.
6679 006762 016637 000006 002336      MOV 6(SP),SUBRPC
6680 006770 162737 000004 002336      SUB #4,SUBRPC      ;GET PC OF SUBROUTINE CALL
6681 006776 012737 000012 002364      MOV #12,REGNUM    ;SET LU REG NO. = 12
6682 007004 005001      CLR R1          ;INIT CYCLE COUNT TO 0
6683 007006 017603 000006      MOV @6(SP),R3     ;GET CYCLE COUNT LIMIT
6684 007012 062703 000003      ADD #3,R3
6685 007016 005776 000006      TST @6(SP)      ;SEE IF DESIRED CYCLES 0
6686 007022 001414      BEQ 8$          ;BR IF YES
6687 007024 004737 006232      JSR PC,IACTIV    ;CHK FOR IACT = 0
6688 007030 000000      0
6689 007032 004737 005746      JSR PC,ISIRDY    ;CHK FOR ICIR = 1, IRDY = 0
6690 007036 000001      1
6691 007040 004737 006420      JSR PC,RSEOM     ;CHK RSOM = 0, REOM = 0 IN AX0-16
6692 007044 000000      0
6693 007046 004737 004726 6$:      JSR PC,STPLU     ;CLOCK LU FOR 1 CYCLE
6694 007052 000001      1
6695 007054 004737 004620 8$:      JSR PC,WAIT50    ;ALLOW SILO DATA TO RIPPLE
6696 007060 005201      INC R1          ;INCREMENT CYCLE COUNT
6697 007062 004737 003344      JSR PC,READLU   ;READ REG 12
6698 007066 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY = 1 YET
6699 007074 001005      BNE 9$          ;BR IF IRDY = 1
6700 007076 020103      CMP R1,R3      ;SEE IF LIMIT EXCEEDED
6701 007100 002762      BLT 6$          ;BR IF NOT YET
6702 007102 004737 005746      JSR PC,ISIRDY    ;CHK FOR ICIR = 1, IRDY = 1
6703 007106 000003      3
6704 007110 020176 000006 9$:      CMP R1,@6(SP)   ;SEE IF LESS THAN REQUIRED CYCLES
6705 007114 002003      BGE 12$        ;BR IF NOT
6706 007116 004737 005746      JSR PC,ISIRDY    ;CHK FOR ICIR = 1, IRDY = 0
6707 007122 000001      1
6708 007124 004737 006232 12$:     JSR PC,IACTIV    ;CHK FOR IACT = 1
6709 007130 000001      1
6710 007132 004737 005746      JSR PC,ISIRDY    ;CHK FOR ICIR = 1, IRDY = 1
6711 007136 000003      3
6712 007140 062766 000002 000006 ADD #2,6(SP)     ;FIX UP RETURN PC
6713 007146 012637 002364      MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6714 007152 012603      MOV (SP)+,R3     ;RESTORE R3
6715 007154 012601      MOV (SP)+,R1     ;RESTORE R1
6716 007156 005037 002336      CLR SUBRPC      ;CLEAR SUBR CALL PC
6717 007162 000207      RTS PC          ;RETURN
6718
6719
6720
6721
6722
6723          ;*****
6724          ;* STPERR - THIS SUBROUTINE LOADS THE CONTENTS OF THE FIRST WORD FOLLOWING THE
6725          ;* CALL INTO REG 17, AND SETS THE IERR BIT, AND CLOCKS THE LINE UNIT
6726          ;* FOR THE NO. OF CYCLES PASSED IN THE 2ND WORD FOLLOWING THE CALL. THEN,
6727          ;* IT RESTORES REG 17 TO ITS ORIGINAL CONTENTS, CLEARING THE IERR BIT.
6728          ;*****
```



```

6729 007164 013746 002364 STPERR: MOV REGNUM,-(SP) ;SAVE LU REG NO.
6730 007170 012737 000017 002364 MOV #17,REGNUM ;SET LU REG NO. = 17
6731 007176 017637 000002 002352 MOV @2(SP),WRIBYT
6732 007204 152737 000002 002352 BISB #IERR,WRIBYT
6733 007212 004737 003422 JSR PC,WRITLU ;SET IERR BIT IN REG 17
6734 007216 062766 000002 C00002 ADD #2,2(SP) ;INCREMENT SUBR ARGUMENT POINTER
6735 007224 017637 000002 007236 MOV @2(SP),3$ ;GET DESIRED NO. OF CYCLES
6736 007232 004737 004726 JSR PC,STPLU ;CLOCK LU FOR DESIRED NO. OF CYCLES
6737 007236 000000 3$: .WORD 0 ;NO. OF CYCLES GOES HERE
6738 007240 142737 000002 002352 BICB #IERR,WRIBYT
6739 007246 004737 003422 JSR PC,WRITLU ;CLEAR IERR BIT IN REG 17
6740 007252 062766 000002 000002 ADD #2,2(SP) ;FIX UP RETURN PC
6741 007260 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6742 007264 000207 RTS PC ;RETURN
6743
6744
6745
6746
6747
6748
6749

```

```

:*****
:* CKDATA - THIS SUBROUTINE READS THE RCV SILO AND COMPARES THE SILO ENTRY
:* TO BITS 0-11 OF THE FIRST WORD FOLLOWING THE CALL. IF THERE IS A
:* MISMATCH, THE ERROR IS REPORTED AND A RETURN IS MADE TO THE TEST AT THE
:* ADDRESS CONTAINED IN RETADR. IF BIT 15 = 0 IN THE FIRST WORD
:* FOLLOWING THE CALL, THE SUBROUTINE WILL NOT CHECK THE BCC BIT (SILO
:* BIT 8). IF THERE ARE NO ERRORS, THE LINE UNIT IS CLOCKED FOR THE
:* NUMBER OF CYCLES PASSED IN THE SECOND WORD FOLLOWING THE CALL.
:*****

```

```

6750
6751
6752
6753
6754
6755
6756
6757 007266 010146 CKDATA: MOV R1,-(SP) ;SAVE R1
6758 007270 013746 002364 MOV REGNUM,-(SP) ;SAVE LU REG NO.
6759 007274 016637 000004 002336 MOV 4(SP),SUBRPC
6760 007302 162737 000004 002336 SUB #4,SUBRPC ;GET PC OF SUBR CALL
6761 007310 017601 000004 MOV @4(SP),R1 ;GET EXPECTED SILO ENTRY
6762 007314 042701 170000 BIC #170000,R1 ;CLEAR UNUSED BITS FOR COMPARE
6763 007320 004737 006672 JSR PC,RDRXSI ;READ RCV SILO
6764 007324 023727 002424 000347 CMP SAVLEN,#TXLEN2!TXLEN1!TXLEN0!RXLEN2!RXLEN1!RXLENO
6765 007332 001005 BNE 4$ ;BR IF CHAR LENGTH NOT = 7
6766 007334 042701 000200 BIC #BIT7,R1 ;MASK OFF 8TH BIT
6767 007340 042737 000200 002414 BIC #BIT7,RXWORD
6768 007346 120137 002414 4$: CMPB R1,RXWORD ;COMPARE EXPECTED BITS 0-7 TO ACTUAL
6769 007352 001445 BEQ 6$ ;BR IF MATCH
6770 007354 005037 002370 CLR GOODAT
6771 007360 110137 002370 MOVB R1,GOODAT ;GET EXPECTED DATA
6772 007364 005037 002372 CLR BADDAT
6773 007370 113737 002414 002372 MOVB RXWORD,BADDAT ;GET ACTUAL DATA
6774 007376 012737 000011 002364 MOV #11,REGNUM ;SET REG NO. = 11
6775 007404 004737 003344 JSR PC,READLU ;READ REG 11
6776 007410 132737 000001 002350 BITB #UNRR,REDBYT ;SEE IF TX UNDERRUN ERROR
6777 007416 001410 BEQ 5$ ;BR IF NOT, TO REPORT DATA ERROR
6778 007420 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6779
6780 ;REPORT TX UNDERRUN ERROR
ERRDF 54,EM54,ERR4

```

```

(4) 007424 104455
(5) 007426 000066
(5) 007430 014541
(5) 007432 016356

```

```

TRAP CSERDF
.WORD 54
.WORD EM54
.WORD ERR4

```

```

6781 007434 000137 010116          JMP      36$          ;TAKE ERROR EXIT
6782 007440 012737 000010 002364 5$:  MOV      #10,REGNUM ;SET REG NO. 10
6783 007446 004737 004200          JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
6784          ;REPORT RCV'D DATA MISCOMPARE
6785 007452          ERRDF  34,EM34,ERR8
(4) 007452 104455          TRAP    C$ERRDF
(5) 007454 000042          .WORD  34
(5) 007456 013664          .WORD  EM34
(5) 007460 020656          .WORD  ERR8
6786 007462 000137 010116          JMP      36$          ;TAKE ERROR EXIT
6787 007466 000301 002364 6$:  SWAB    R1          ;SET LU REG NO. FOR ERROR REPORTS
6788 007470 012737 000012          MOV      #12,REGNUM ;COMPARE EXPECTED SILO BITS 8-11 TO ACTUAL
6789 007476 120137 002415          CMPB    R1,RXWORD+1 ;BR IF MISMATCH
6790 007502 001002          BNE     7$          ;CONTINUE
6791 007504 000137 010072          JMP      22$
6792 007510 005037 002370 7$:  CLR     GOODAT      ;SET EXPECTED DATA
6793 007514 110137 002370          MOVB    R1,GOODAT
6794 007520 005037 002372          CLR     BADDAT
6795 007524 113737 002415 002372  MOVB    RXWORD+1,BADDAT ;SET ACTUAL DATA
6796 007532 032776 100000 000004  BIT     #BCCCHK,@4(SP) ;SEE IF BCC SHOULD BE IGNORED
6797 007540 001433          BEQ     10$         ;BR IF YES
6798 007542 132701 000001          BITB    #BCC,R1     ;SEE IF EXPECTED BIT - 1
6799 007546 001014          BNE     8$          ;BR IF YES
6800 007550 132737 000001 002415  BITB    #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 0
6801 007556 001424          BEQ     10$         ;BR IF YES
6802 007560 004737 004200          JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
6803          ;REPORT BCC NOT CLEARED
6804          ERRDF  35,EM35,ERR8
(4) 007564 104455          TRAP    C$ERRDF
(5) 007566 000043          .WORD  35
(5) 007570 013712          .WORD  EM35
(5) 007572 020656          .WORD  ERR8
6805 007574 000137 010116          JMP      36$          ;TAKE ERROR EXIT
6806 007600 132737 000001 002415 8$:  BITB    #BCC,RXWORD+1 ;SEE IF ACTUAL BIT = 1
6807 007606 001010          BNE     10$        ;BR IF YES
6808 007610 004737 004200          JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
6809          ;REPORT BCC NOT SET
6810          ERRDF  36,EM36,ERR8
(4) 007614 104455          TRAP    C$ERRDF
(5) 007616 000044          .WORD  36
(5) 007620 013732          .WORD  EM36
(5) 007622 020656          .WORD  ERR8
6811 007624 000137 010116          JMP      36$          ;TAKE ERROR EXIT
6812 007630 000002 002415 10$: BITB    #EBLK,R1     ;SEE IF EXPECTED BIT - 1
6813 007634 132701 000002          BNE     12$        ;BR IF YES
6814 007636 001014          BITB    #EBLK,RXWORD+1 ;SEE IF ACTUAL BIT = 0
6815 007644 001424          BEQ     14$        ;BR IF YES
6816 007646 004737 004200          JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
6817          ;REPORT EBLK NOT CLEARED
6818          ERRDF  37,EM37,ERR8
(4) 007652 104455          TRAP    C$ERRDF
(5) 007654 000045          .WORD  37
(5) 007656 013746          .WORD  EM37
(5) 007660 020656          .WORD  ERR8
6820 007662 000137 010116          JMP      36$          ;TAKE ERROR EXIT
  
```

6821	007666	132737	000002	002415	12\$:	BITB	#EBLK,RXWORD+1	;SEE IF ACTUAL BIT - 1		
6822	007674	001010				BNE	14\$;BR IF YES		
6823	007676	004737	004200			JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
6824					;REPORT	EBLK NOT SET				
6825	007702					ERRDF	38,EM38,ERR8			
(4)	007702	104455							TRAP	C\$ERDF
(5)	007704	000046							.WORD	38
(5)	007706	013767							.WORD	EM38
(5)	007710	020656							.WORD	ERR8
6826	007712	000137	010116			JMP	36\$;TAKE ERROR EXIT		
6827	007713				14\$:					
6828	007716	132701	000004			BITB	#RAB,R1	;SEE IF EXPECTED BIT = 1		
6829	007722	001014				BNE	16\$;BR IF YES		
6830	007724	132737	000004	002415		BITB	#RAB,RXWORD+1	;SEE IF ACTUAL BIT - 0		
6831	007732	001424				BEQ	18\$;BR IF YES		
6832	007734	004737	004200			JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
6833					;REPORT	RAB NOT CLEARED				
6834	007740					ERRDF	39,EM39,ERR8			
(4)	007740	104455							TRAP	C\$ERDF
(5)	007742	000047							.WORD	39
(5)	007744	014004							.WORD	EM39
(5)	007746	020656							.WORD	ERR8
6835	007750	000137	010116			JMP	36\$;TAKE ERROR EXIT		
6836	007754	132737	000004	002415	16\$:	BITB	#RAB,RXWORD+1	;SEE IF ACTUAL BIT = 1		
6837	007762	001010				BNE	18\$;BR IF YES		
6838	007764	004737	004200			JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
6839					;REPORT	RAB NOT SET				
6840	007770					ERRDF	40,EM40,ERR8			
(4)	007770	104455							TRAP	C\$ERDF
(5)	007772	000050							.WORD	40
(5)	007774	014024							.WORD	EM40
(5)	007776	020656							.WORD	ERR8
6841	010000	000137	010116			JMP	36\$;TAKE ERROR EXIT		
6842	010004				18\$:					
6843	010004	132701	000010			BITB	#OVRR,R1	;SEE IF EXPECTED BIT = 1		
6844	010010	001014				BNE	20\$;BR IF YES		
6845	010012	132737	000010	002415		BITB	#OVRR,RXWORD+1	;SEE IF ACTUAL BIT - 0		
6846	010020	001424				BEQ	22\$;BR IF YES		
6847	010022	004737	004200			JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
6848					;REPORT	OVRR NOT CLEARED				
6849	010026					ERRDF	41,EM41,ERR8			
(4)	010026	104455							TRAP	C\$ERDF
(5)	010030	000051							.WORD	41
(5)	010032	014040							.WORD	EM41
(5)	010034	020656							.WORD	ERR8
6850	010036	000137	010116			JMP	36\$;TAKE ERROR EXIT		
6851	010042	132737	000010	002415	20\$:	BITB	#OVRR,RXWORD+1	;SEE IF ACTUAL BIT = 1		
6852	010050	001010				BNE	22\$;BR IF YES		
6853	010052	004737	004200			JSR	PC,GETALL	;GET REGS FOR PRINTOUT		
6854					;REPORT	OVRR NOT SET				
6855	010056					ERRDF	42,EM42,ERR8			
(4)	010056	104455							TRAP	C\$ERDF
(5)	010060	000052							.WORD	42
(5)	010062	014061							.WORD	EM42
(5)	010064	020656							.WORD	ERR8
6856	010066	000137	010116			JMP	36\$;TAKE ERROR EXIT		

```

6857 010072
6858 010072 062766 000002 0000C4 22$: ADD #2,4(SP) ;INCR SUBROUTINE ARGUMENT POINTER
6859 010100 017637 000004 010112 MOV @4(SP),24$ ;GET DESIRED CYCLE COUNT
6860 010106 004737 004726 JSR PC,STPLU ;CLOCK LU FOR DESIRED CYCLES
6861 010112 000000 24$: .WORD 0
6862 010114 000407 BR 38$ ;TAKE ERROR-FREE EXIT
6863 010116 011637 002364 36$: MOV (SP),REGNUM ;RESTORE LU REG NO.
6864 010122 013706 002332 MOV PSTACK,SP ;RESTORE PROGRAM STACK TO BASE LEVEL
6865 010126 013746 002346 MOV RETADR,-(SP) ;FIX UP ERROR RETURN PC
6866 010132 000406 BR 40$
6867 010134 062766 000002 000004 38$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
6868 010142 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
6869 010146 012601 MOV (SP)+,R1 ;RESTORE R1
6870 010150 005037 002336 40$: CLR SUBRPC ;CLEAR SUBROUTINE PC
6871 010154 000207 RTS PC ;RETURN
6872
6873
6874
6875
6876
6877
6878 ;*****
6879 ;* LODATA - THIS SUBROUTINE LOADS THE TRANSMITTER SILO WITH 2 SOM'S, THE DATA
6880 ;* IN PATTERN A REPEATED 3 TIMES (60 CHARS), AND 2 EOM'S. IN ADDITION, THE
6881 ;* DATA CHARS ARE ALSO LOADED INTO THE RECEIVED MSG BUFFER (RCVBUF:), AS
6882 ;* EXPECTED DATA FOR LATER COMPARISON.
6883 ;*****
6883 010156 010146 LODATA: MOV R1,-(SP) ;SAVE R1
6884 010160 010346 MOV R3,-(SP) ;SAVE R3
6885 010162 010446 MOV R4,-(SP) ;SAVE R4
6886 010164 012737 000400 002412 MOV #TXSOM, TXWORD
6887 010172 004737 004646 JSR PC,LDTXSI ;LOAD AN SOM INTO TX SILO
6888 010176 004737 004646 JSR PC,LDTXSI ;LOAD ANOTHER SOM
6889 010202 012701 000003 MOV #3,R1 ;INIT COUNTER
6890 010206 012704 002762 MOV #RCVBUF,R4 ;GET POINTER TO RCV BUF
6891 010212 012703 002557 3$: MOV #PATA,R3 ;GET POINTER TO DATA PATTERN
6892 010216 005037 002412 6$: CLR TXWORD
6893 010222 112337 002412 MOV (R3)+,TXWORD ;GET A DATA CHAR
6894 010226 013724 002412 MOV TXWORD,(R4)+ ;LOAD A DATA CHAR INTO RCV BUF
6895 010232 004737 004646 JSR PC,LDTXSI ;LOAD DATA CHAR INTO TX SILO
6896 010236 020327 002603 CMP R3,#PATB ;SEE IF AT END OF PATTERN A YET
6897 010242 103765 BLO 6$ ;BR IF NOT YET
6898 010244 005301 DEC R1 ;DECREMENT COUNTER
6899 010246 001361 BNE 3$ ;BR IF NOT DONE YET
6900 010250 052764 100400 177776 BIS #CRCCHK!RXBCC,-2(R4) ;SET UP TO CHK BCC - 1 ON LAST DATA CHAR
6901 010256 012737 001000 002412 MOV #TXEOM, TXWORD
6902 010264 004737 004646 JSR PC,LDTXSI ;LOAD AN EOM INTO TX SILO
6903 010270 004737 004646 JSR PC,LDTXSI ;LOAD ANOTHER EOM
6904 010274 012604 MOV (SP)+,R4 ;RESTORE R4
6905 010276 012603 MOV (SP)+,R3 ;RESTORE R3
6906 010300 012601 MOV (SP)+,R1 ;RESTORE R1
6907 010302 000207 RTS PC ;RETURN
6908
6909
6910
6911
6912

```

6913
6914
6915
6916
6917
6918 010304 013746 002366
6919 010310 013746 002364
6920 010314 012737 000004 002366
6921 010322 017637 000004 002360
6922 010330 005037 002362
6923 010334 004737 003764
6924 010340 012737 000017 002364
6925 010346 062766 000002 000004
6926 010354 017637 000004 002352
6927 010362 004737 003422
6928 010366 012737 000006 002366
6929 010374 062766 000002 000004
6930 010402 017637 000004 002360
6931 010410 062766 000002 000004
6932 010416 017637 000004 002362
6933 010424 013737 002362 002424
6934 010432 142777 000010 172000
6935 010440 004737 003764
6936 010444 152777 000010 171766
6937 010452 062766 000002 000004
6938 010460 012637 002364
6939 010464 012637 002366
6940 010470 005037 002336
6941 010474 000207

```
*****  
* SETUP - THIS SUBROUTINE LOADS THE FIRST WORD AFTER THE CALL INTO AX2-15  
* (SYNCH CHAR), LOADS THE SECOND WORD AFTER THE CALL INTO REG 17  
* LOADS THE THIRD WORD INTO AX3-15, AND LOADS THE FOURTH WORD INTO AX3-16.  
*****  
SETUP:  MOV     AXNUM,-(SP)      ;SAVE AX BYTE NO.  
        MOV     REGNUM,-(SP)   ;SAVE LU REG NO.  
        MOV     #4,AXNUM       ;SET AX BYTE NO. FOR AX2  
        MOV     @4(SP),WAX15  
        CLR     WAX16  
        JSR     PC,WR;TAX      ;SET SYNCH CHAR IN AX2-15, CLEAR AX2-16  
        MOV     #17,R;NUM     ;SET LU REG NO. = 17  
        ADD     #2,4(P)        ;INCREMENT ARGUMENT POINTER  
        MOV     @4(SP),WRIBYT  
        JSR     PC,WR;TLU     ;LOAD REG 17  
        MOV     #6,AXNUM      ;SET AX BYTE NO. FOR AX3  
        ADD     #2,4(SP)       ;INCREMENT ARGUMENT POINTER  
        MOV     @4(SP),WAX15  
        ADD     #2,4(SP)       ;INCREMENT ARGUMENT POINTER  
        MOV     @4(SP),WAX16  
        MOV     WAX16,SAVLEN   ;STORE TX AND RCV CHAR LENGTHS  
        BICB   #LULOOP,@BSEL1 ;CLEAR LULOOP  
        JSR     PC,WR;ITAX    ;LOAD AX3-15, AX3-16  
        BICB   #LULOOP,@BSEL1 ;SET LULOOP  
        ADD     #2,4(SP)       ;FIX RETURN PC  
        MOV     (SP)+,REGNUM   ;RESTORE LU REG NO.  
        MOV     (SP)+,AXNUM    ;RESTORE AX BYTE NO.  
        CLR     SUBRPC        ;CLEAR SUBROUTINE PC STORAGE  
        RTS     PC            ;RETURN
```

6942
6943
6944
6945
6946
6947
6948
6949
6950
6951
6952 010476 010146
6953 010500 010246
6954 010502 017601 000004
6955 010506 062766 000002 000004
6956 010514 017602 000004
6957 010520 062766 000002 000004
6958 010526 012137 002412
6959 010532 004737 004646
6960 010536 005302
6961 010540 001372
6962 010542 004737 004620
6963 010546 012602
6964 010550 012601
6965 010552 000207
6966
6967
6968

```
*****  
* LODMSG - THIS SUBROUTINE LOADS THE NO. OF WORDS PASSED IN THE SECOND WORD  
* FOLLOWING THE CALL FROM THE MSG BUFFER WHOSE ADDRESS IS IN THE FIRST  
* WORD FOLLOWING THE CALL, INTO THE TRANSMITTER SILO.  
*****  
LODMSG: MOV     R1,-(SP)      ;SAVE R1  
        MOV     R2,-(SP)     ;SAVE R2  
        MOV     @4(SP),R1    ;GET MSG POINTER INTO R1  
        ADD     #2,4(SP)     ;INCR ARG POINTER  
        MOV     @4(SP),R2    ;GET WORD COUNT INTO R2  
        ADD     #2,4(SP)     ;FIX UP RETURN PC  
6$:    MOV     (R1)+,TXWORD   ;GET NEXT MSG WORD  
        JSR     PC,LDTXSI    ;LOAD A WORD INTO TX SILO  
        DEC     R2           ;DECR COUNT  
        SNE     6$          ;BR IF NOT DONE YET  
        JSR     PC,WAIT50    ;WAIT FOR SILO TO RIPPLE  
        MOV     (SP)+,R2     ;RESTORE R2  
        MOV     (SP)+,R1     ;RESTORE R1  
        RTS     PC          ;RETURN
```

6969
6970
6971
6972
6973
6974
6975
6976
6977 010554 010146
6978 010556 010246
6979 010560 017601 000004
6980 010564 062766 000002 000004
6981 010572 017602 000004
6982 010576 062766 000002 000004
6983 010604 112137 002412
6984 010610 105037 002413
6985 010614 004737 004646
6986 010620 005302
6987 010622 001370
6988 010624 004737 004620
6989 010630 012602
6990 010632 012601
6991 010634 000207
6992
6993
6994
6995
6996
6997
6998
6999
7000
7001
7002 010636 010146
7003 010640 010246
7004 010642 004737 010476
7005 010646 002670
7006 010650 000010
7007 010652 012701 002674
7008 010656 012702 002762
7009 010662 012122
7010 010664 020 27 002706
7011 010670 103774
7012 010672 052762 100400 177776
7013 010700 012722 000160
7014 010704 012722 000034
7015 010710 012602
7016 010712 012601
7017 010714 000207
7018
7019
7020
7021
7022
7023
7024

```
*****  
* LDBYTS - THIS SUBROUTINE LOADS THE NO. OF BYTES PASSED IN THE SECOND WORD  
* FOLLOWING THE CALL FROM THE MSG BUFFER WHOSE ADDRESS IS IN THE FIRST  
* WORD FOLLOWING THE CALL, INTO THE LOW BYTE OF THE TX SILO. FOR EACH  
* BYTE LOADED, A 0 IS LOADED INTO THE HI 4 BITS OF THE TX SILO.  
*****
```

```
LDBYTS: MOV R1,-(SP) ;SAVE R1  
MOV R2,-(SP) ;SAVE R2  
MOV @4(SP),R1 ;GET DATA POINTER INTO R1  
ADD #2,4(SP) ;INCR ARGUMENT POINTER  
MOV @4(SP),R2 ;GET BYTE COUNT INTO R2  
ADD #2,4(SP) ;FIX UP RETURN PC  
6$: MOV (R1)+,TXWORD ;GET NEXT DATA BYTE  
CLRB TXWORD+1 ;CLEAR HI BYTE  
JSR PC,LDTXSI ;LOAD A SILO ENTRY  
DEC R2 ;DECR BYTE COUNT  
BNE 6$ ;BR IF NOT DONE YET  
JSR PC,WAIT50 ;WAIT FOR SILO TO RIPPLE  
MOV (SP)+,R2 ;RESTORE R2  
MOV (SP)+,R1 ;RESTORE R1  
RTS PC ;RETURN
```

```
*****  
* LDMSG1 - THIS SUBROUTINE LOADS THE TRANSMITTER SILO WITH MSG1, AND LOADS  
* THE DATA CHARS INTO THE RCV MSG BUFFER (RCVBUF:), AS EXPECTED DATA  
* FOR LATER COMPARISON.  
*****
```

```
LDMSG1: MOV R1,-(SP) ;SAVE R1  
MOV R2,-(SP) ;SAVE R2  
JSR PC,LODMSG ;LOAD MSG1 INTO TX SILO (WITH ONLY 1 EOM)  
MSG1  
8.  
MOV #MSG1+4,R1 ;GET POINTER TO MSG1  
MOV #RCVBUF,R2 ;GET POINTER TO MSG BUF  
3$: MOV (R1)+,(R2)+ ;LOAD A CHAR INTO MSG BUF  
CMP R1,#MSG1+14. ;SEE IF DID LAST DATA CHAR YET  
BLO 3$ ;BR IF NOT  
BIS #CRCCHK!RXBCC,-2(R2) ;SET EXPECTED BCC  
MOV #160,(R2)+ ;LOAD HI CRC BYTE  
MOV #034,(R2)+ ;LOAD LO CRC BYTE  
MOV (SP)+,R2 ;RESTORE R2  
MOV (SP)+,R1 ;RESTORE R1  
RTS PC ;RETURN
```

```
*****  
* LODSIL - THIS SUBROUTINE REPEATEDLY LOADS THE DATA PASSED IN THE FIRST WORD  
*****
```

```
7025      ;*      FOLLOWING THE CALL INTO THE TX SILO, FOR THE NO. OF TIMES PASSED IN THE
7026      ;*      SECOND WORD FOLLOWING THE CALL.
7027      ;*****
7028 010716 010146 LODSIL: MOV      R1,-(SP)      ;SAVE R1
7029 010720 017637 000002 002412 MOV      @2(SP),TXWORD ;GET DATA
7030 010726 062766 000002 000002 ADD      #2,2(SP)      ;INCR ARGUMENT POINTER
7031 010734 017601 000002      MOV      @2(SP),R1      ;GET COUNT
7032 010740 004737 004646 3$: JSR      PC,LDTXSI      ;LOAD WORD INTO TX SILO
7033 010744 005301      DEC      R1              ;DECR COUNT
7034 010746 001374      BNE     3$              ;BR IF NOT ALL LOADED YET
7035 010750 004737 004620      JSR      PC,WAIT50      ;ALLOW SILO DATA TO RIPPLE
7036 010754 062766 000002 000002 ADD      #2,2(SP)      ;FIX UP RETURN PC
7037 010762 012601      MOV      (SP)+,R1      ;RESTORE R1
7038 010764 000207      RTS      PC            ;RETURN
7039
7040
7041
7042
7043
7044      ;*****
7045      ;* CKLPBK - THIS SUBROUTINE DETERMINES IF THE TEST CALLING IT CAN BE RUN. THE
7046      ;* TEST PASSES THE DESIRED MODEM INTERFACE TYPE IN THE WORD FOLLOWING THE
7047      ;* CALL, AND IF THE PROPER EXTERNAL LOOPBACK HAS BEEN PROVIDED BY THE
7048      ;* OPERATOR FOR THAT INTERFACE, AND IF THE BAUD RATE IS CORRECT, A NORMAL
7049      ;* RETURN IS MADE TO RUN THE TEST. IF NOT, A RETURN IS MADE TO THE TEST,
7050      ;* AT THE ADDRESS IN RETADR (RETADR CONTAINS THE TEST EXIT ADDRESS, SO
7051      ;* THE TEST GETS SKIPPED).
7052      ;* IF BIT 15 IS SET IN THE WORD FOLLOWING THE CALL, THE TEST WILL NOT
7053      ;* BE RUN UNLESS THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. IF THE
7054      ;* TEST IS TO BE RUN, THE SUBROUTINE RETURNS THE MODEM SELECT BITS FOR
7055      ;* AX3-15 IN MODINT. IF NECESSARY, THE SUBROUTINE WILL SET MAINT1, MAINT2,
7056      ;* OR DTR.
7057      ;* IF THE PROGRAM PASSES '0' IN THE WORD FOLLOWING THE CALL, THE SUBROUTINE
7058      ;* WILL ATTEMPT TO RUN WHICHEVER MODEM INTERFACE IS SELECTED BY CABLE,
7059      ;* SWITCH, OR TEST CONNECTOR. IF SUCCESSFUL, THE SELECTED INTERFACE WILL
7060      ;* BE PASSED BACK TO THE TEST IN MODINT.
7061      ;*****
7062 010766 013746 002364 CKLPBK: MOV      REGNUM,-(SP) ;SAVE REG NO.
7063 010772 013746 002366      MOV      AXNUM,-(SP) ;SAVE AX BYTE NO.
7064 010776 010246      MOV      R2,-(SP)      ;SAVE R2
7065 011000 016637 000006 002336 MOV      6(SP),SUBRPC ;GET PC OF SUBR CALL
7066 011006 162737 000004 002336 SUB      #4,SUBRPC
7067 011014 012737 000333 002422 MOV      #I422!XYZ!V35!INTGRL!OP!TEST,MODINT ;INIT MODEM SELECT BITS
7068 011022 032776 100000 000006 BIT      #TCCHEK,@6(SP) ;SEE IF H3254,5 CHECK IS DESIRED
7069 011030 001405      BEQ     1$              ;BR IF NOT
7070 011032 005737 002462      TST     TSTCON         ;SEE IF H3254,5 INSTALLED
7071 011036 001074      BNE     7$              ;BR IF NOT, TO SKIP TEST
7072 011040 000137 012102      JMP     32$            ;BR TO RUN TEST
7073      ;IF NO EXTERNAL LPBK, SKIP TEST
7074 011044 023727 002462 000004 1$: CMP      TSTCON,#4      ;SEE IF NO LPBK
7075 011052 001466      BEQ     7$              ;BR IF NO LPBK, TO SKIP TEST
7076 011054 142777 000010 171356 BICB    #LULOOP,@BSEL1 ;CLEAR LULOOP
7077 011062 012737 000006 002366 MOV      #6,AXNUM      ;SET AX BYTE NO. FOR AX3-15
7078 011070 004737 003576      JSR      PC,READAX     ;READ AX3-15
7079
7080      ;*** SEE IF AN INTERFACE IS REQUESTED ***
```

```
7081
7082 011074 027627 000006 000010      CMP      @6(SP),#INTGRL ;SEE IF INTEGRAL MODEM REQUESTED
7083 011102 001422                      BEQ      4$             ;BR IF INTGRL MODEM REQUESTED
7084 011104 027627 000006 000020      CMP      @6(SP),#V35   ;SEE IF V.35 REQUESTED
7085 011112 001512                      BEQ      10$            ;BR IF V.35 REQUESTED
7086 011114 027627 000006 000200      CMP      @6(SP),#1422  ;SEE IF 422 REQUESTED
7087 011122 001002                      BNE      2$             ;BR IF 422 NOT REQUESTED
7088 011124 000137 011414                      JMP      14$            ;422 REQUESTED
7089 011130 027627 000006 000100 2$:  CMP      @6(SP),#XYZ   ;SEE IF XYZ REQUESTED
7090 011136 001002                      BNE      3$             ;BR IF XYZ NOT REQUESTED
7091 011140 000137 011470                      JMP      17$            ;XYZ REQUESTED
7092 011144 000137 011546 3$:      JMP      21$            ;NONE REQUESTED, FIND AN INTERFACE TO TEST
7093
7094 ;SEE IF INTEGRAL MODEM CAN BE RUN
7095 011150 005737 002462 4$:      TST      TSTCON        ;SEE IF H3254 AND H3255 USED
7096 011154 001050                      BNE      8$             ;BR IF NOT
7097 011156 023727 002464 000004 5$:  CMP      BDRATE,#4     ;SEE IF BAUD RATE > OR = 56K
7098 011164 002405                      BLT      6$             ;BR IF BAUD RATE TOO SLOW FOR INTGRL MODM
7099 011166 042737 000010 002422  BIC      #INTGRL,MODINT ;ASSERT INTEGRAL MODEM
7100 011174 000137 012102                      JMP      32$            ;GO TO RUN TEST
7101 ;PRINT 'BAUD RATE INCORRECT'
7102 011200 6$:
7103 011200 023727 002402 000001      CMP      STARES,#1     ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
7104 011206 001026                      BNE      40$            ;BR IF NOT, TO SKIP PRINTING
7105 011210 PRINTF #FMT25
(7) 011210 012746 013135                      MOV      #FMT25,-(SP)
(6) 011214 012746 000001                      MOV      #1,-(SP)
(3) 011220 010600                      MOV      SP,R0
(4) 011222 104417                      TRAP    C$PNTF
(4) 011224 062706 000004                      ADD     #4,SP
7106 ;PRINT 'TEST XX NOT RUN'
7107 011230 7$:
7108 011230 023727 002402 000001      CMP      STARES,#1     ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
7109 011236 001012                      BNE      40$            ;BR IF NOT, TO SKIP PRINTING
7110 011240 PRINTF #FMT19,TSTNUM
(8) 011240 013746 002434                      MOV      TSTNUM,-(SP)
(7) 011244 012746 013104                      MOV      #FMT19,-(SP)
(6) 011250 012746 000002                      MOV      #2,-(SP)
(3) 011254 010600                      MOV      SP,R0
(4) 011256 104417                      TRAP    C$PNTF
(4) 011260 062706 000006                      ADD     #6,SP
7111 011264 013766 002346 000006 40$:  MOV      RETADR,6(SP)  ;SET TEST EXIT ADDRESS FOR ERRORS
7112 011272 000137 012110                      JMP      33$            ;GO TO SKIP TEST
7113 011276 032737 000010 002354 8$:  BIT      #INTGRL,RAX15 ;SEE IF INTEGRAL MODEM IS SELECTED
7114 011304 001324                      BNE      5$             ;BR IF YES, TO CHECK BAUD RATE
7115 ;PRINT 'MODEM INTERFACE NOT SELECTED'
7116 011306 9$:
7117 011306 023727 002402 000001      CMP      STARES,#1     ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
7118 011314 001363                      BNE      40$            ;BR IF NOT, TO SKIP PRINTING
7119 011316 PRINTF #FMT26
(7) 011316 012746 013165                      MOV      #FMT26,-(SP)
(6) 011322 012746 000001                      MOV      #1,-(SP)
(3) 011326 010600                      MOV      SP,R0
(4) 011330 104417                      TRAP    C$PNTF
(4) 011332 062706 000004                      ADD     #4,SP
7120 011336 000734 BR      7$             ;GO TO PRINT 'TEST NOT RUN'
```



```
7121
7122
7123 011340 032737 000200 002354 :SEE IF V.35 CAN BE RUN
10$: BIT #I422,RAX15 ;SEE IF 422 IS SELECTED
7124 011346 001357 BNE 9$ ;BR IF YES, TO SKIP TEST
7125 011350 005737 002462 TST TSTCON ;SEE IF H3254 AND H3255 USED
7126 011354 001402 BEQ 11$ ;BR IF YES
7127 011356 000137 011374 JMP 12$ ;H3254 AND H3255 NOT USED
7128 011362 042737 000020 002422 11$: BIC #V35,MODINT ;ASSERT V.35
7129 011370 000137 011644 JMP 27$ ;GO SET DTR AND RUN THE TEST
7130 011374 032737 000020 002354 12$: BIT #V35,RAX15 ;SEE IF V.35 IS SELECTED
7131 011402 001402 BEQ 13$ ;BR IF NOT
7132 011404 000137 011362 JMP 11$ ;GO ASSERT V.35 AND RUN TEST
7133 011410 000137 011306 13$: JMP 9$ ;WRONG INTRFCE, GO SKIP TEST
7134
7135
7136 011414 005737 002462 :SEE IF 422 CAN BE RUN
14$: TST TSTCON ;SEE IF H3254 AND H3255 USED
7137 011420 001015 BNE 16$ ;BR IF NOT
7138 011422 012737 000013 002364 MOV #13,REGNUM ;SET REG. NO. = 13
7139 011430 112737 000004 002352 MOVB #MAINT2,WRIBYT
7140 011436 004737 003422 JSR PC,WRITLU ;SET MAINT2 FOR MANUFACTURING TEST OF RS422
7141 011442 042737 000200 002422 15$: BIC #I422,MODINT ;ASSERT 422
7142 011450 000137 011644 JMP 27$ ;GO TO RUN TEST
7143 011454 032737 000200 002354 16$: BIT #I422,RAX15 ;SEE IF 422 IS SELECTED
7144 011462 001367 BNE 15$ ;IF YES, GO ASSERT 422 AND RUN TEST
7145 011464 000137 011306 JMP 9$ ;WRONG INTRFCE, GO SKIP TEST
7146
7147
7148 011470 032737 000200 002354 :SEE IF XYZ CAN BE RUN
17$: BIT #I422,RAX15 ;SEE IF 422 IS SELECTED
7149 011476 001402 BEQ 18$ ;BR IF NOT
7150 011500 000137 011306 JMP 9$ ;WRONG INTRFCE, GO SKIP TEST
7151 011504 032737 000100 002354 18$: BIT #XYZ,RAX15 ;SEE IF XYZ IS SELECTED
7152 011512 001002 BNE 19$ ;BR IF YES
7153 011514 000137 011306 JMP 9$ ;WRONG INTRFCE, GO SKIP TEST
7154 011520 023727 002464 000004 19$: CMP BDRATE,#4 ;SEE IF BAUD RATE < OR = 56K
7155 011526 003402 BLE 20$ ;BR IF YES
7156 011530 000137 011200 JMP 6$ ;BAUD RATE TOO FAST FOR XYZ
7157 011534 042737 000100 002422 20$: BIC #XYZ,MODINT ;ASSERT XYZ
7158 011542 000137 011644 JMP 27$ ;GO TO RUN TEST
7159
7160 ;*** NO INTERFACE REQUESTED - FIND ONE TO TEST ***
7161
7162 011546 032737 000010 002354 21$: BIT #INTGRL,RAX15 ;SEE IF INTEGRAL MODEM SELECTED
7163 011554 001402 BEQ 22$ ;BR IF NOT
7164 011556 000137 011156 JMP 5$ ;SEE IF INTEGRAL MODEM CAN BE RUN
7165 011562 032737 000020 002354 22$: BIT #V35,RAX15 ;SEE IF V.35 SELECTED
7166 011570 001402 BEQ 23$ ;BR IF NOT
7167 011572 000137 011340 JMP 10$ ;GO SEE IF V.35 CAN BE RUN
7168 011576 032737 000200 002354 23$: BIT #I422,RAX15 ;SEE IF 422 SELECTED
7169 011604 001402 BEQ 24$ ;BR IF NOT
7170 011606 000137 011442 JMP 15$ ;GO ASSERT AND RUN 422
7171 011612 005737 002462 24$: TST TSTCON ;SEE IF H3254 AND H3255 USED
7172 011616 001002 BNE 25$ ;BR IF NOT
7173 011620 000137 011362 JMP 11$ ;GO ASSERT AND RUN V.35 BY DEFAULT
7174 011624 032737 000100 002354 25$: BIT #XYZ,RAX15 ;SEE IF XYZ SELECTED
7175 011632 001402 BEQ 26$ ;BR IF NOT
7176 011634 000137 011520 JMP 19$ ;GO SEE IF XYZ CAN BE RUN
```

```

7177 011640 000137 011306      26$:  JMP      9$              ;CO SKIP TEST
7178
7179                               ;*** SET MAINT1 OR MAINT2 IF NEEDED, AND SET DTR ***
7180
7181                               ;SET MAINT1 IF LOCAL LOOPBACK DESIRED
7182 011644 012737 000013 002364 27$:  MOV      #13,REGNUM      ;SET REG NO. = 13
7183 011652 023727 002462 000002      CMP      TSTCON,#2        ;SEE IF MODEM LOCAL LPBK DESIRED
7184 011660 001032                BNE      29$              ;BR IF NOT, TO CHK REMOTE LPBK
7185 011662 112737 000010 002352      MOVB     #MAINT1,WRIBYT
7186 011670 004737 003422                JSR      PC,WRITLU        ;SET MAINT1 IN REG 13
7187 011674 012737 000017 002364      MOV      #17,REGNUM      ;SET REG NO. = 17
7188 011702 012702 000000                MOV      #0,R2           ;INIT TIMER
7189 011706 004737 003344      28$:  JSR      PC,READLU      ;READ REG 17
7190 011712 132737 000004 002350      BITB     #TESTMD,REDBYT  ;SEE IF TEST MODE BIT SET IN REG 17 YET
7191 011720 001050                BNE      31$              ;BR IF YES, TO SET DTR
7192 011722 005202                INC      R2              ;INCREMENT TIMER
7193 011724 001370                BNE      28$             ;BR IF NO TIME-OUT YET
7194 011726 004737 004200      JSR      PC,GETALL       ;GET REGS FOR PRINTOUT
7195                               ;REPORT TEST MODE NOT SET BY MAINT1
7196 011732      ERRDF  52,EM52,ERR4
(4) 011732 104455
(5) 011734 000064          TRAP  C$ERDF
(5) 011736 014451          .WORD 52
(5) 011740 016356          .WORD EM52
7197 011742 000137 011230      JMP      7$              ;BR TO SKIP TEST
7198                               ;SET MAINT2 IF REMOTE LOOPBACK DESIRED
7199 011746 023727 002462 000003 29$:  CMP      TSTCON,#3        ;SEE IF MODEM REMOTE LPBK DESIRED
7200 011754 001032                BNE      31$              ;BR IF NOT, TO SET DTR
7201 011756 112737 000004 002352      MOVB     #MAINT2,WRIBYT
7202 011764 004737 003422                JSR      PC,WRITLU        ;SET MAINT2 IN REG 13
7203 011770 012737 000017 002364      MOV      #17,REGNUM      ;SET REG NO. = 17
7204 011776 012702 000000                MOV      #0,R2           ;INIT TIMER
7205 012002 004737 003344      30$:  JSR      PC,READLU      ;READ REG 17
7206 012006 132737 000004 002350      BITB     #TESTMD,REDBYT  ;SEE IF TEST MODE BIT SET IN REG 17 YET
7207 012014 001012                BNE      31$              ;BR IF YES, TO SET DTR
7208 012016 005202                INC      R2              ;INCREMENT TIMER
7209 012020 001370                BNE      30$             ;BR IF NO TIME-OUT YET
7210 012022 004737 004200      JSR      PC,GETALL       ;GET REGS FOR PRINTOUT
7211                               ;REPORT TEST MODE NOT SET BY MAINT2
7212 012026      ERRDF  53,EM53,ERR4
(4) 012026 104455          TRAP  C$ERDF
(5) 012030 000065          .WORD 53
(5) 012032 014505          .WORD EM53
(5) 012034 016356          .WORD ERR4
7213 012036 000137 011230      JMP      7$              ;BR TO SKIP TEST
7214                               ;SET DTR
7215 012042 012737 000013 002364 31$:  MOV      #13,REGNUM      ;SET REG NO. = 13
7216 012050 004737 003344                JSR      PC,READLU      ;READ REG 13
7217 012054 013737 002350 002352      MOV      REDBYT,WRIBYT
7218 012062 042737 000243 002352      BIC      #243,WRIBYT     ;CLEAR UNNEEDED BITS
7219 012070 152737 000100 002352      BISB     #DTR,WRIBYT     ;SET DTR BIT TO BE WRITTEN
7220 012076 004737 003422                JSR      PC,WRITLU      ;LOAD REG 13
7221
7222
7223 012102 062766 000002 000006 32$:  ADD      #2,6(SP)        ;INCREMENT RETURN ADRS
7224

```

```

7225                                     :*** BRANCH HERE TO SKIP TEST ***
7226 012110 012602                       33$:  MOV      (SP)+,R2          ;RESTORE R2
7227 012112 012637 002366                 MOV      (SP)+,AXNUM        ;RESTORE AX BYTE NO.
7228 012116 012637 002364                 MOV      (SP)+,REGNUM      ;RESTORE LU REG NO.
7229 012122 152777 000010 170310         BISB     #LULOOP,@BSEL1    ;SET LULOOP
7230 012130 005037 002336                 CLR      SUBRPC           ;CLEAR SUBROUTINE CALL PC
7231 012134 000207                       RTS      PC                ;RETURN
7232
7233
7234
7235
7236
7237                                     :*****
7238                                     :*  CHKABT - THIS SUBROUTINE READS AX0-16 AND CHECKS FOR RAB, REOM SET. IF
7239                                     :*  EITHER IS NOT SET, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
7240                                     :*  CONTAINED IN RETADR.
7241                                     :*****
7242 012136 013746 002366                 (CHKABT: MOV     AXNUM,-(SP)  ;SAVE AX BYTE NO.
7243 012142 016637 000002 002336          MOV     2(SP),SUBRPC
7244 012150 162737 000004 002336          SUB     #4,SUBRPC         ;GET PC OF SUBROUTINE CALL
7245 012156 005037 002366                 CLR     AXNUM             ;SET AX0 ADDRESS
7246 012162 004737 003576                 JSR    PC,READAX         ;READ REG AX0
7247 012166 032737 000004 002356          BIT     #RABT,RAX16      ;CHK FOR RAB SET
7248 012174 001007                       BNE     6$               ;BR IF RAB SET
7249 012176 004737 004200                 JSR    PC,GETALL        ;GET REGS FOR PRINTOUT
7250                                     ;REPORT RAB NOT SET
7251 012202                                     ERRDF  40,EM40,ERR6
7251 (4) 012202 104455                                     TRAP  C$ERDF
7251 (5) 012204 000050                                     .WORD 40
7251 (5) 012206 014024                                     .WORD EM40
7251 (5) 012210 017546                                     .WORD ERR6
7252 012212 000412
7253 012214 032737 000002 002356 6$:  BR      8$
7254 012222 001015                       BIT     #REOM,RAX16     ;CHK FOR REOM SET
7255 012224 004737 004200                       BNE     9$             ;BR IF REOM SET
7256                                     JSR    PC,GETALL        ;GET REGS FOR PRINTOUT
7257                                     ;REPORT REOM NOT SET
7257 (4) 012230                                     ERRDF  31,EM31,ERR6
7257 (5) 012232 000037                                     TRAP  C$ERDF
7257 (5) 012234 013647                                     .WORD 31
7257 (5) 012236 017546                                     .WORD EM31
7258 012240 011637 002366                 8$:  MOV     (SP),AXNUM    ;RESTORE AX BYTE NO.
7259 012244 013706 002332                 MOV     PSTACK,SP      ;RESTORE STACK POINTER TO BASE LEVEL
7260 012250 013746 002346                 MOV     RETADR,-(SP)   ;FIX ERROR RETURN PC
7261 012254 000402                       BR      16$
7262 012256 012637 002366                 9$:  MOV     (SP)+,AXNUM  ;RESTORE AX BYTE NO.
7263 012262 000207                       16$:  RTS      PC        ;RETURN
7264
7265
7266
7267
7268
  
```

```
7270 .SBTTL GLOBAL ERROR REPORT SECTION
7271
7272 :////////////////////
7273 :/ THE GLOBAL ERROR REPORT SECTION CONTAINS ERROR MESSAGES
7274 :/ THAT ARE USED IN MORE THAN ONE TEST.
7275 :////////////////////
7276
7277
7278 012264 052045 047445 022466 FMT1: .ASCIZ /%T%06%N/
012272 000116
7279 012274 047045 040445 040506 FMT2: .ASCIZ /%N%AFAILING REG: /
012302 046111 047111 020107
012310 042522 035107 000040
7280 012316 040445 054105 042520 FMT3: .ASCIZ /%AEXPECTED: %03%S5%AACTUAL: %03%N/
012324 052103 042105 020072
012332 047445 022463 032523
012340 040445 041501 052524
012346 046101 020072 047445
012354 022463 000116
7281 012360 047045 052045 047045 FMT4: .ASCIZ /%N%T%N%T%N/
012366 052045 047045 000
7282 012373 045 031517 051445 FMT5: .ASCIZ /%03%S5%03%S5%03%S5%03%N/
012400 022465 031517 051445
012406 022465 031517 051445
012414 022465 031517 047045
012422 000
7283 012423 045 032123 047445 FMT6: .ASCIZ /%S4%03%S5%03%S5%03%S5%03%N/
012430 022463 032523 047445
012436 022463 032523 047445
012444 022463 032523 047445
012452 022463 000116
7284 012456 052045 047445 022462 FMT7: .ASCIZ /%T%02%N/
012464 000116
7285 012466 040445 054105 042524 FMT8: .ASCIZ /%AEXTENDED REG AX%01%A-%T%N/
012474 042116 042105 051040
012502 043505 040440 022530
012510 030517 040445 022455
012516 022524 000116
7286 012522 052045 047045 000 FMT9: .ASCIZ /%T%N/
7287 012527 045 050101 020103 FMT10: .ASCIZ /%APC OF SUBR CALL: %06%N/
012534 043117 051440 041125
012542 020122 040503 046114
012550 020072 047445 022466
012556 000116
7288 012560 040445 042522 020107 FMT11: .ASCIZ /%AREG %02%A LOADED WITH: %03%N/
012566 047445 022462 020101
012574 047514 042101 042105
012602 053440 052111 035110
012610 022440 031517 047045
012616 000
7289 012617 045 046501 031070 FMT12: .ASCIZ /%AM8203 SW PACK #1 (REG 11) %03%N/
012624 031460 051440 020127
012632 040520 045503 021440
012640 020061 051050 043505
012646 030440 024461 036440
012654 022440 031517 047045
```

7290	012662	000			
	012663	045	046501	031070	FMT13: .ASCIZ /%AM8203 SW PACK #2 (REG 15) = %03%N/
	012670	031460	051440	020127	
	012676	040520	045503	021440	
	012704	020062	051050	043505	
	012712	030440	024465	036440	
	012720	022440	031517	047045	
	012726	000			
7291	012727	045	046501	031070	FMT14: .ASCIZ /%AM8203 SW PACK #3 (REG 16) = %03%N/
	012734	031460	051440	020127	
	012742	040520	045503	021440	
	012750	020063	051050	043505	
	012756	030440	024466	036440	
	012764	022440	031517	047045	
	012772	000			
7292	012773	045	046501	042117	FMT15: .ASCIZ /%MODEM INTERFACE REG (AX3-15) %03%N/
	013000	046505	044440	052116	
	013006	051105	040506	042503	
	013014	051040	043505	024040	
	013022	054101	026463	032461	
	013030	020051	020075	047445	
	013036	022463	000116		
7293	013042	047045	040445	047506	FMT18: .ASCIZ /%N%AFOR DEVICE AT ADRS %06%A ,%N/
	013050	020122	042504	044526	
	013056	042503	040440	020124	
	013064	042101	051522	020040	
	013072	047445	022466	020101	
	013100	022454	000116		
7294	013104	047045	040445	042524	FMT19: .ASCIZ /%N%ATEST %D2%A NOT RUN%N/
	013112	052123	022440	031104	
	013120	040445	047040	052117	
	013126	051040	047125	047045	
	013134	000			
7295	013135	045	022516	041101	FMT25: .ASCIZ /%N%ABAUD RATE INCORRECT/
	013142	052501	020104	040522	
	013150	042524	044440	041516	
	013156	051117	042522	052103	
	013164	000			
7296	013165	045	022516	046501	FMT26: .ASCIZ /%N%MODEM INTERFACE NOT SELECTED/
	013172	042117	046505	044440	
	013200	052116	051105	040506	
	013206	042503	047040	052117	
	013214	051440	046105	041505	
	013222	042524	000104		
7297					
7298					
7299					
7300	013226	042522	020107	047516	EM2: .ASCIZ /REG NOT INITIALIZED BY MST CLR/
	013234	020124	047111	052111	
	013242	040511	044514	042532	
	013250	020104	054502	046440	
	013256	052123	041440	051114	
	013264	000			
7301	013265	122	043505	046440	EM3: .ASCIZ /REG MISCOMPARE/
	013272	051511	047503	050115	
	013300	051101	000105		

7302	013304	051117	054504	047040	EM7:	.ASCIZ	/ORDY NOT SET/
	013312	052117	051440	052105			
	013320	000					
7303	013321	117	042122	020131	EM8:	.ASCIZ	/ORDY NOT CLEARED/
	013326	047516	020124	046103			
	013334	040505	042522	000104			
7304	013342	041517	051117	047040	EM9:	.ASCIZ	/OCOR NOT SET/
	013350	052117	051440	052105			
	013356	000					
7305	013357	117	047503	020122	EM10:	.ASCIZ	/OCOR NOT CLEARED/
	013364	047516	020124	046103			
	013372	040505	042522	000104			
7306	013400	040517	052103	047040	EM11:	.ASCIZ	/OACT NOT SET/
	013406	052117	051440	052105			
	013414	000					
7307	013415	117	041501	020124	EM12:	.ASCIZ	/OACT NOT CLEARED/
	013422	047516	020124	046103			
	013430	040505	042522	000104			
7308	013436	051111	054504	047040	EM17:	.ASCIZ	/IRDY NOT SET/
	013444	052117	051440	052105			
	013452	000					
7309	013453	111	042122	020131	EM18:	.ASCIZ	/IRDY NOT CLEARED/
	013460	047516	020124	046103			
	013466	040505	042522	000104			
7310	013474	041511	051111	047040	EM19:	.ASCIZ	/ICIR NOT SET/
	013502	052117	051440	052105			
	013510	000					
7311	013511	111	044503	020122	EM20:	.ASCIZ	/ICIR NOT CLEARED/
	013516	047516	020124	046103			
	013524	040505	042522	000104			
7312	013532	040511	052103	047040	EM21:	.ASCIZ	/IACT NOT SET/
	013540	052117	051440	052105			
	013546	000					
7313	013547	111	041501	020124	EM22:	.ASCIZ	/IACT NOT CLEARED/
	013554	047516	020124	046103			
	013562	040505	042522	000104			
7314	013570	051522	046517	047040	EM28:	.ASCIZ	/RSOM NOT CLEARED/
	013576	052117	041440	042514			
	013604	051101	042105	000			
7315	013611	122	047523	020115	EM29:	.ASCIZ	/RSOM NOT SET/
	013616	047516	020124	042523			
	013624	000124					
7316	013626	042522	046517	047040	EM30:	.ASCIZ	/REOM NOT CLEARED/
	013634	052117	041440	042514			
	013642	051101	042105	000			
7317	013647	122	047505	020115	EM31:	.ASCIZ	/REOM NOT SET/
	013654	047516	020124	042523			
	013662	000124					
7318	013664	041522	023526	020104	EM34:	.ASCIZ	/RCV'D DATA MISCOMPARE/
	013672	040504	040524	046440			
	013700	051511	047503	050115			
	013706	051101	000105				
7319	013712	041502	020103	047516	EM35:	.ASCIZ	/BCC NOT CLEARED/
	013720	020124	046103	040505			
	013726	042522	000104				
7320	013732	041502	020103	047516	EM36:	.ASCIZ	/BCC NOT SET/

7321	013740	020124	042523	000124		
	013746	041105	045514	047040	EM37:	.ASCIZ /EBLK NOT CLEARED/
	013754	052117	041440	042514		
	013762	051101	042105	000		
7322	013767	105	046102	020113	EM38:	.ASCIZ /EBLK NOT SET/
	013774	047516	020124	042523		
	014002	000124				
7323	014004	040522	020102	047516	EM39:	.ASCIZ /RAB NOT CLEARED/
	014012	020124	046103	040505		
	014020	042522	000104			
7324	014024	040522	020102	047516	EM40:	.ASCIZ /RAB NOT SET/
	014032	020124	042523	000124		
7325	014040	053117	051122	047040	EM41:	.ASCIZ /OVRR NOT CLEARED/
	014046	052117	041440	042514		
	014054	051101	042105	000		
7326	014061	117	051126	020122	EM42:	.ASCIZ /OVRR NOT SET/
	014066	047516	020124	042523		
	014074	000124				
7327	014076	053523	050040	041501	EM43:	.ASCIZ /SW PACK #1 INCORRECT/
	014104	020113	030443	044440		
	014112	041516	051117	042522		
	014120	052103	000			
7328	014123	123	020127	040520	EM44:	.ASCIZ /SW PACK #2 INCORRECT/
	014130	045503	021440	020062		
	014136	047111	047503	051122		
	014144	041505	000124			
7329	014150	053523	050040	041501	EM45:	.ASCIZ /SW PACK #3 INCORRECT/
	014156	020113	031443	044440		
	014164	041516	051117	042522		
	014172	052103	000			
7330	014175	122	053103	051440	EM46:	.ASCIZ /RCV SILO NOT CLEARED BY IC/
	014202	046111	020117	047516		
	014210	020124	046103	040505		
	014216	042522	020104	054502		
	014224	044440	000103			
7331	014230	051501	042523	041115	EM47:	.ASCIZ /ASSEMB BIT COUNT INCORRECT/
	014236	041040	052111	041440		
	014244	052517	052116	044440		
	014252	041516	051117	042522		
	014260	052103	000			
7332	014263	117	042104	053040	EM48:	.ASCIZ /ODD VRC PARITY BIT NOT SET/
	014270	041522	050040	051101		
	014276	052111	020131	044502		
	014304	020124	047516	020124		
	014312	042523	000124			
7333	014316	042117	020104	051126	EM49:	.ASCIZ /ODD VRC PARITY BIT NOT CLEARED/
	014324	020103	040520	044522		
	014332	054524	041040	052111		
	014340	047040	052117	041440		
	014346	042514	051101	042105		
	014354	000				
7334	014355	105	042526	020116	EM50:	.ASCIZ /EVEN VRC PARITY BIT NOT SET/
	014362	051126	020103	040520		
	014370	044522	054524	041040		
	014376	052111	047040	052117		
	014404	051440	052105	000		

7335	014411	105	042526	020116	EM51:	.ASCIZ	/EVEN VRC PARITY BIT NOT CLEARED/
	014416	051126	020103	040520			
	014424	044522	054524	041040			
	014432	052111	047040	052117			
	014440	041440	042514	051101			
	014446	042105	000				
7336	014451	124	051505	020124	EM52:	.ASCIZ	/TEST MODE NOT SET BY MAINT1/
	014456	047515	042504	047040			
	014464	052117	051440	052105			
	014472	041040	020131	040515			
	014500	047111	030524	000			
7337	014505	124	051505	020124	EM53:	.ASCIZ	/TEST MODE NOT SET BY MAINT2/
	014512	047515	042504	047040			
	014520	052117	051440	052105			
	014526	041040	020131	040515			
	014534	047111	031124	000			
7338	014541	124	020130	047125	EM54:	.ASCIZ	/TX UNDERRUN ERROR/
	014546	042504	051122	047125			
	014554	042440	051122	051117			
	014562	000					
7339	014563	104	051124	047040	EM55:	.ASCIZ	/DTR NOT SET/
	014570	052117	051440	052105			
	014576	000					
7340	014577	122	047111	020107	EM56:	.ASCIZ	/RING NOT SET/
	014604	047516	020124	042523			
	014612	000124					
7341	014614	047515	051104	047040	EM57:	.ASCIZ	/MODR NOT SET/
	014622	052117	051440	052105			
	014630	000					
7342	014631	110	054104	047040	EM58:	.ASCIZ	/HDX NOT SET/
	014636	052117	051440	052105			
	014644	000					
7343	014645	123	041124	020131	EM59:	.ASCIZ	/STBY NOT SET/
	014652	047516	020124	042523			
	014660	000124					
7344	014662	052122	020123	047516	EM60:	.ASCIZ	/RTS NOT SET/
	014670	020124	042523	000124			
7345	014676	051503	047040	052117	EM61:	.ASCIZ	/CS NOT SET/
	014704	051440	052105	000			
7346	014711	103	051101	020122	EM62:	.ASCIZ	/CARR NOT SET/
	014716	047516	020124	042523			
	014724	000124					
7347	014726	044523	050507	047040	EM63:	.ASCIZ	/SIGQ NOT SET/
	014734	052117	051440	052105			
	014742	000					
7348	014743	123	043511	020122	EM64:	.ASCIZ	/SIGR NOT SET/
	014750	047516	020124	042523			
	014756	000124					
7349	014760	052122	020123	047516	EM65:	.ASCIZ	/RTS NOT CLEARED/
	014766	020124	046103	040505			
	014774	042522	000104				
7350	015000	040503	051122	047040	EM66:	.ASCIZ	/CARR NOT CLEARED/
	015006	052117	041440	042514			
	015014	051101	042105	000			
7351							
7352							


```

7353
7354 015021 111 041116 051525 DH1: .ASCIZ &INBUS/OUTBUS REG &
      015026 047457 052125 052502
      015034 020123 042522 020107
      015042 000
7355 015043 114 047111 020105 DH2: .ASCIZ /LINE UNIT INBUS REGS :/
      015050 047125 052111 044440
      015056 041116 051525 051040
      015064 043505 020123 000072
7356 015072 042522 030507 020060 DH3: .ASCIZ /REG10 REG11 REG12 REG13/
      015100 020040 042522 030507
      015106 020061 020040 042522
      015114 030507 020062 020040
      015122 042522 030507 000063
7357 015130 020040 020040 042522 DH4: .ASCIZ / REG14 REG15 REG16 REG17/
      015136 030507 020064 020040
      015144 042522 030507 020065
      015152 020040 042522 030507
      015160 020066 020040 042522
      015166 030507 000067
7358 015172 032461 000 DH5: .ASCIZ /15/
7359 015175 061 000066 DH6: .ASCIZ /16/
7360 015200 044514 042516 052440 DH7: .ASCIZ /LINE UNIT EXTENDED REGS :/
      015206 044516 020124 054105
      015214 042524 042116 042105
      015222 051040 043505 020123
      015230 000072
7361 015232 054101 026460 032461 DH8: .ASCIZ /AX0-15 AX0-16 AX1-15 AX1-16/
      015240 020040 054101 026460
      015246 033061 020040 054101
      015254 026461 032461 020040
      015262 054101 026461 033061
      015270 000
7362 015271 040 020040 040440 DH9: .ASCIZ / AX2-15 AX2-16 AX3-15 AX3-16/
      015276 031130 030455 020065
      015304 040440 031130 030455
      015312 020066 040440 031530
      015320 030455 020065 040440
      015326 031530 030455 000066
  
```

```

7363
7364 .EVEN
7365
7366
7367
7368
7369
  
```

```

7370 015334 BGNMSG ERR1
      (3) 015334
7371 015334 PRINTB #FMT1,#ADDRES,MPCSR
      (9) 015334 013746 002436
      (8) 015340 012746 036452
      (7) 015344 012746 012264
      (6) 015350 012746 000003
      (3) 015354 010600
      (4) 015356 104414
      (4) 015360 062706 000010
  
```

ERR1::

```

MOV MPCSR, -(SP)
MOV #ADDRES, -(SP)
MOV #FMT1, -(SP)
MOV #3, -(SP)
MOV SP, R0
TRAP C$PNTB
ADD #10, SP
  
```

Line	Address	Offset	Offset	Label	Code	Comment
7372	015364			ENDMSG		
(3)	015364					
(3)	015364	104423				
7373						
7374						
7375						
7376	015366			BGNMSG ERR2		
(3)	015366					
7377	015366			PRINTB #FMT1,#ADDRES,MPCSR		
(9)	015366	013746	002436			
(8)	015372	012746	036452			
(7)	015376	012746	012264			
(6)	015402	012746	000003			
(3)	015406	010600				
(4)	015410	104414				
(4)	015412	062706	000010			
7378	015416			PRINTB #FMT2		
(7)	015416	012746	012274			
(6)	015422	012746	000001			
(3)	015426	010600				
(4)	015430	104414				
(4)	015432	062706	000004			
7379	015436			PRINTB #FMT7,#DH1,REGNUM		
(9)	015436	013746	002364			
(8)	015442	012746	015021			
(7)	015446	012746	012456			
(6)	015452	012746	000003			
(3)	015456	010600				
(4)	015460	104414				
(4)	015462	062706	000010			
7380	015466			PRINTB #FMT3,GOODAT,BADDAT		
(9)	015466	013746	002372			
(8)	015472	013746	002370			
(7)	015476	012746	012316			
(6)	015502	012746	000003			
(3)	015506	010600				
(4)	015510	104414				
(4)	015512	062706	000010			
7381	015516			PRINTX #FMT4,#DH2,#DH3		
(9)	015516	012746	015072			
(8)	015522	012746	015043			
(7)	015526	012746	012360			
(6)	015532	012746	000003			
(3)	015536	010600				
(4)	015540	104415				
(4)	015542	062706	000010			
7382	015546			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13		
(11)	015546	013746	002274			
(10)	015552	013746	002272			
(9)	015556	013746	002270			
(8)	015562	013746	002266			
(7)	015566	012746	012373			
(6)	015572	012746	000005			
(3)	015576	010600				
(4)	015600	104415				
(4)	015602	062706	000014			

L10002: TRAP C\$MSG

ERR2::

MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #10,SP

MOV #FMT2,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #4,SP

MOV REGNUM,-(SP)
MOV #DH1,-(SP)
MOV #FMT7,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #10,SP

MOV BADDAT,-(SP)
MOV GOODAT,-(SP)
MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,RO
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,RO
TRAP C\$PNTX
ADD #14,SP

7383	015606			PRINTX #FMT9,#DH4			
(8)	015606	012746	015130			MOV	#DH4,-(SP)
(7)	015612	012746	012522			MOV	#FMT9,-(SP)
(6)	015616	012746	000002			MOV	#2,-(SP)
(3)	015622	010600				MOV	SP,R0
(4)	015624	104415				TRAP	C\$PNTX
(4)	015626	062706	000006			ADD	#6,SP
7384	015632			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17			
(11)	015632	013746	002304			MOV	LUR17,-(SP)
(10)	015636	013746	002302			MOV	LUR16,-(SP)
(9)	015642	013746	002300			MOV	LUR15,-(SP)
(8)	015646	013746	002276			MOV	LUR14,-(SP)
(7)	015652	012746	012423			MOV	#FMT6,-(SP)
(6)	015656	012746	000005			MOV	#5,-(SP)
(3)	015662	010600				MOV	SP,R0
(4)	015664	104415				TRAP	C\$PNTX
(4)	015666	062706	000014			ADD	#14,SP
7385	015672			ENDMSG			
(3)	015672						
(3)	015672	104423				L10003:	TRAP C\$MSG
7386							
7387							
7388							
7389							
7390							
7391	015674			BGNMSG ERR3			
(3)	015674					ERR3::	
7392	015674			PRINTB #FMT1,#ADDRES,MPCSR			
(9)	015674	013746	002436			MOV	MPCSR,-(SP)
(8)	015700	012746	036452			MOV	#ADDRES,-(SP)
(7)	015704	012746	012264			MOV	#FMT1,-(SP)
(6)	015710	012746	003003			MOV	#3,-(SP)
(3)	015714	010600				MOV	SP,R0
(4)	015716	104414				TRAP	C\$PNTB
(4)	015720	062706	000010			ADD	#10,SP
7393	015724			PRINTB #FMT2			
(7)	015724	012746	012274			MOV	#FMT2,-(SP)
(6)	015730	012746	000001			MOV	#1,-(SP)
(3)	015734	010600				MOV	SP,R0
(4)	015736	104414				TRAP	C\$PNTB
(4)	015740	062706	000004			ADD	#4,SP
7394	015744			PRINTB #FMT8,TMP1,TMP0			
(9)	015744	013746	002516			MOV	TMP0,-(SP)
(8)	015750	013746	002520			MOV	TMP1,-(SP)
(7)	015754	012746	012466			MOV	#FMT8,-(SP)
(6)	015760	012746	000003			MOV	#3,-(SP)
(3)	015764	010600				MOV	SP,R0
(4)	015766	104414				TRAP	C\$PNTB
(4)	015770	062706	000010			ADD	#10,SP
7395	015774			PRINTB #FMT3,GOODAT,BADDAT			
(9)	015774	013746	002372			MOV	BADDAT,-(SP)
(8)	016000	013746	002370			MOV	GOODAT,-(SP)
(7)	016004	012746	012316			MOV	#FMT3,-(SP)
(6)	016010	012746	000003			MOV	#3,-(SP)
(3)	016014	010600				MOV	SP,R0
(4)	016016	104414				TRAP	C\$PNTB

(7)	016274	012746	012522				MOV	#FMT9,-(SP)
(6)	016300	012746	000002				MOV	#2,-(SP)
(3)	016304	010600					MOV	SP,R0
(4)	016306	104415					TRAP	C\$PNTX
(4)	016310	062706	000006				ADD	#6,SP
7403	016314			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16			
(11)	016314	013746	002324				MOV	AX3.16,-(SP)
(10)	016320	013746	002322				MOV	AX3.15,-(SP)
(9)	016324	013746	002320				MOV	AX2.16,-(SP)
(8)	016330	013746	002316				MOV	AX2.15,-(SP)
(7)	016334	012746	012423				MOV	#FMT6,-(SP)
(6)	016340	012746	000005				MOV	#5,-(SP)
(3)	016344	010600					MOV	SP,R0
(4)	016346	104415					TRAP	C\$PNTX
(4)	016350	062706	000014				ADD	#14,SP
7404	016354			ENDMSG				
(3)	016354							
(3)	016354	104423				L10004:	TRAP	C\$MSG
7405								
7406								
7407								
7408								
7409								
7410	016356			BGNMSG	ERR4			
(3)	016356					ERR4::		
7411	016356			PRINTB	#FMT10,SUBRPC			
(8)	016356	013746	002336				MOV	SUBRPC,-(SP)
(7)	016362	012746	012527				MOV	#FMT10,-(SP)
(6)	016366	012746	000002				MOV	#2,-(SP)
(3)	016372	010600					MOV	SP,R0
(4)	016374	104414					TRAP	C\$PNTB
(4)	016376	062706	000006				ADD	#6,SP
7412	016402			PRINTB	#FMT1,#ADDRES,MPCSR			
(9)	016402	013746	002436				MOV	MPCSR,-(SP)
(8)	016406	012746	036452				MOV	#ADDRES,-(SP)
(7)	016412	012746	012264				MOV	#FMT1,-(SP)
(6)	016416	012746	000003				MOV	#3,-(SP)
(3)	016422	010600					MOV	SP,R0
(4)	016424	104414					TRAP	C\$PNTB
(4)	016426	062706	000010				ADD	#10,SP
7413	016432			PRINTB	#FMT2			
(7)	016432	012746	012274				MOV	#FMT2,-(SP)
(6)	016436	012746	000001				MOV	#1,-(SP)
(3)	016442	010600					MOV	SP,R0
(4)	016444	104414					TRAP	C\$PNTB
(4)	016446	062706	000004				ADD	#4,SP
7414	016452			PRINTB	#FMT7,#DH1,REGNUM			
(9)	016452	013746	002364				MOV	REGNUM,-(SP)
(8)	016456	012746	015021				MOV	#DH1,-(SP)
(7)	016462	012746	012456				MOV	#FMT7,-(SP)
(6)	016466	012746	000003				MOV	#3,-(SP)
(3)	016472	010600					MOV	SP,R0
(4)	016474	104414					TRAP	C\$PNTB
(4)	016476	062706	000010				ADD	#10,SP
7415	016502			PRINTX	#FMT4,#DH2,#DH3			
(9)	016502	012746	015072				MOV	#DH3,-(SP)

(4)	016764	104415			TRAP	(SPNTX
(4)	016766	062706	000006		ADD	#6,SP
7422	016772			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
(11)	016772	013746	002324		MOV	AX3.16,-(SP)
(10)	016776	013746	002322		MOV	AX3.15,-(SP)
(9)	017002	013746	002320		MOV	AX2.16,-(SP)
(8)	017006	013746	002316		MOV	AX2.15,-(SP)
(7)	017012	012746	012423		MOV	#FMT6,-(SP)
(6)	017016	012746	000005		MOV	#5,-(SP)
(3)	017022	010600			MOV	SP,RO
(4)	017024	104415			TRAP	(SPNTX
(4)	017026	062706	000014		ADD	#14,SP
7423	017032			ENDMSG		
(3)	017032					
(3)	017032	104423			L10005:	TRAP
7424						C\$MSG
7425						
7426						
7427						
7428						
7429	017034			BGNMSG	ERR5	
(3)	017034					
7430	017034			PRINTB	#FMT1,#ADDRES,MPCSR	
(9)	017034	013746	002436			
(8)	017040	012746	036452		MOV	MPCSR,-(SP)
(7)	017044	012746	012264		MOV	#ADDRES,-(SP)
(6)	017050	012746	000003		MOV	#FMT1,-(SP)
(3)	017054	010600			MOV	#3,-(SP)
(4)	017056	104414			MOV	SP,RO
(4)	017060	062706	000010		TRAP	(SPNTB
7431	017064			PRINTB	#FMT11,REGNUM,LOADAT	
(9)	017064	013746	002374		ADD	#10,SP
(8)	017070	013746	002364		MOV	LOADAT,-(SP)
(7)	017074	012746	012560		MOV	REGNUM,-(SP)
(6)	017100	012746	000003		MOV	#FMT11,-(SP)
(3)	017104	010600			MOV	#3,-(SP)
(4)	017106	104414			MOV	SP,RO
(4)	017110	062706	000010		TRAP	(SPNTB
7432	017114			PRINTB	#FMT2	
(7)	017114	012746	012274		ADD	#10,SP
(6)	017120	012746	000001		MOV	#FMT2,-(SP)
(3)	017124	010600			MOV	#1,-(SP)
(4)	017126	104414			MOV	SP,RO
(4)	017130	062706	000004		TRAP	(SPNTB
7433	017134			PRINTB	#FMT8,TMP1,TMP0	
(9)	017134	013746	002516		ADD	#4,SP
(8)	017140	013746	002520		MOV	TMP0,-(SP)
(7)	017144	012746	012466		MOV	TMP1,-(SP)
(6)	017150	012746	000003		MOV	#FMT8,-(SP)
(3)	017154	010600			MOV	#3,-(SP)
(4)	017156	104414			MOV	SP,RO
(4)	017160	062706	000010		TRAP	(SPNTB
7434	017164			PRINTB	#FMT3,GOODAT,BADDAT	
(9)	017164	013746	002372		ADD	#10,SP
(8)	017170	013746	002370		MOV	BADDAT,-(SP)
(7)	017174	012746	012316		MOV	GOODAT,-(SP)
					MC/	#FMT3,-(SP)

(6) 017200 012746 000003
(3) 017204 010600
(4) 017206 104414
(4) 017210 062706 000010
7435 017214
(9) 017214 012746 015072
(8) 017220 012746 015043
(7) 017224 012746 012360
(6) 017230 012746 000003
(3) 017234 010600
(4) 017236 104415
(4) 017240 062706 000010
7436 017244
(11) 017244 013746 002274
(10) 017250 013746 002272
(9) 017254 013746 002270
(8) 017260 013746 002266
(7) 017264 012746 012373
(6) 017270 012746 000005
(3) 017274 010600
(4) 017276 104415
(4) 017300 062706 000014
7437 017304
(8) 017304 012746 015130
(7) 017310 012746 012522
(6) 017314 012746 000002
(3) 017320 010600
(4) 017322 104415
(4) 017324 062706 000006
7438 017330
(11) 017330 013746 002304
(10) 017334 013746 002302
(9) 017340 013746 002300
(8) 017344 013746 002276
(7) 017350 012746 012423
(6) 017354 012746 000005
(3) 017360 010600
(4) 017362 104415
(4) 017364 062706 000014
7439 017370
(9) 017370 012746 015232
(8) 017374 012746 015200
(7) 017400 012746 012360
(6) 017404 012746 000003
(3) 017410 010600
(4) 017412 104415
(4) 017414 062706 000010
7440 017420
(11) 017420 013746 002314
(10) 017424 013746 002312
(9) 017430 013746 002310
(8) 017434 013746 002306
(7) 017440 012746 012373
(6) 017444 012746 000005
(3) 017450 010600
(4) 017452 104415

PRINTX #FMT4,#DH2,#DH3

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX

7441	(4)	017454	062706	000014			ADD	#14,SP
		017460			PRINTX	#FMT9,#DH9		
	(8)	017460	012746	015271			MOV	#DH9,-(SP)
	(7)	017464	012746	012522			MOV	#FMT9,-(SP)
	(6)	017470	012746	000002			MOV	#2,-(SP)
	(3)	017474	010600				MOV	SP,R0
	(4)	017476	104415				TRAP	C\$PNTX
7442	(4)	017500	062706	000006			ADD	#6,SP
		017504			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16		
	(11)	017504	013746	002324			MOV	AX3.16,-(SP)
	(10)	017510	013746	002322			MOV	AX3.15,-(SP)
	(9)	017514	013746	002320			MOV	AX2.16,-(SP)
	(8)	017520	013746	002316			MOV	AX2.15,-(SP)
	(7)	017524	012746	012423			MOV	#FMT6,-(SP)
	(6)	017530	012746	000005			MOV	#5,-(SP)
	(3)	017534	010600				MOV	SP,R0
	(4)	017536	104415				TRAP	C\$PNTX
	(4)	017540	062706	000014			ADD	#14,SP
7443		017544			ENDMSG			
	(3)	017544						
	(3)	017544	104423				L10006:	TRAP C\$MSG
7444								
7445								
7446								
7447								
7448								
7449		017546			BGNMSG	ERR6		
	(3)	017546						
7450		017546			PRINTB	#FMT10,SUBRPC		ERR6::
	(8)	017546	013746	002336			MOV	SUBRPC,-(SP)
	(7)	017552	012746	012527			MOV	#FMT10,-(SP)
	(6)	017556	012746	000002			MOV	#2,-(SP)
	(3)	017562	010600				MOV	SP,R0
	(4)	017564	104414				TRAP	C\$PNTB
	(4)	017566	062706	000006			ADD	#6,SP
7451		017572			PRINTB	#FMT1,#ADDRES,MPCSR		
	(9)	017572	013746	002436			MOV	MPCSR,-(SP)
	(8)	017576	012746	036452			MOV	#ADDRES,-(SP)
	(7)	017602	012746	012264			MOV	#FMT1,-(SP)
	(6)	017606	012746	000003			MOV	#3,-(SP)
	(3)	017612	010600				MOV	SP,R0
	(4)	017614	104414				TRAP	C\$PNTB
	(4)	017616	062706	000010			ADD	#10,SP
7452		017622			PRINTB	#FMT2		
	(7)	017622	012746	012274			MOV	#FMT2,-(SP)
	(6)	017626	012746	000001			MOV	#1,-(SP)
	(3)	017632	010600				MOV	SP,R0
	(4)	017634	104414				TRAP	C\$PNTB
	(4)	017636	062706	000004			ADD	#4,SP
7453		017642			PRINTB	#FMT8,TMP1,TMPC		
	(9)	017642	013746	002516			MOV	TMPO,-(SP)
	(8)	017646	013746	002520			MOV	TMP1,-(SP)
	(7)	017652	012746	012466			MOV	#FMT8,-(SP)
	(6)	017656	012746	000003			MOV	#3,-(SP)
	(3)	017662	010600				MOV	SP,R0
	(4)	017664	104414				TRAP	C\$PNTB

(7) 020634 012746 012423
(6) 020640 012746 000005
(3) 020644 010600
(4) 020646 104415
(4) 020650 062706 000014
7480 020654
(3) 020654
(3) 020654 104423
7481
7482
7483
7484
7485
7486 020656
(3) 020656
7487 020656
(8) 020656 013746 002336
(7) 020662 012746 012527
(6) 020666 012746 000002
(3) 020672 010600
(4) 020674 104414
(4) 020676 062706 000006
7488 020702
(9) 020702 013746 002436
(8) 020706 012746 036452
(7) 020712 012746 012264
(6) 020716 012746 000003
(3) 020722 010600
(4) 020724 104414
(4) 020726 062706 000010
7489 020732
(7) 020732 012746 012274
(6) 020736 012746 000001
(3) 020742 010600
(4) 020744 104414
(4) 020746 062706 000004
7490 020752
(9) 020752 013746 002364
(8) 020756 012746 015021
(7) 020762 012746 012456
(6) 020766 012746 000003
(3) 020772 010600
(4) 020774 104414
(4) 020776 062706 000010
7491 021002
(9) 021002 013746 002372
(8) 021006 013746 002370
(7) 021012 012746 012316
(6) 021016 012746 000003
(3) 021022 010600
(4) 021024 104414
(4) 021026 062706 000010
7492 021032
(9) 021032 012746 015072
(8) 021036 012746 015043
(7) 021042 012746 012360

ENDMSG

BGNMSG ERR8

PRINTB #FMT10,SUBRPC

PRINTB #FMT1,#ADDRES,MPCSR

PRINTB #FMT2

PRINTB #FMT7,#DH1,REGNUM

PRINTB #FMT3,GOODAT,BADDAT

PRINTX #FMT4,#DH2,#DH3

MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

L10010:

TRAP C\$MSG

ERR8::

MOV SUBRPC,-(SP)
MOV #FMT10,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP

MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #FMT2,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #4,SP

MOV REGNUM,-(SP)
MOV #DH1,-(SP)
MOV #FMT7,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV BADDAT,-(SP)
MOV GOODAT,-(SP)
MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)

(6) 021046 012746 000003
(3) 021052 010600
(4) 021054 104415
(4) 021056 062706 000010
7493 021062
(11) 021062 013746 002274
(10) 021066 013746 002272
(9) 021072 013746 002270
(8) 021076 013746 002266
(7) 021102 012746 012373
(6) 021106 012746 000005
(3) 021112 010600
(4) 021114 104415
(4) 021116 062706 000014
7494 021122
(8) 021122 012746 015130
(7) 021126 012746 012522
(6) 021132 012746 000002
(3) 021136 010600
(4) 021140 104415
(4) 021142 062706 000006
7495 021146
(11) 021146 013746 002304
(10) 021152 013746 002302
(9) 021156 013746 002300
(8) 021162 013746 002276
(7) 021166 012746 012423
(6) 021172 012746 000005
(3) 021176 010600
(4) 021200 104415
(4) 021202 062706 000014
7496 021206
(9) 021206 012746 015232
(8) 021212 012746 015200
(7) 021216 012746 012360
(6) 021222 012746 000003
(3) 021226 010600
(4) 021230 104415
(4) 021232 062706 000010
7497 021236
(11) 021236 013746 002314
(10) 021242 013746 002312
(9) 021246 013746 002310
(8) 021252 013746 002306
(7) 021256 012746 012373
(6) 021262 012746 000005
(3) 021266 010600
(4) 021270 104415
(4) 021272 062706 000014
7498 021276
(8) 021276 012746 015271
(7) 021302 012746 012522
(6) 021306 012746 000002
(3) 021312 010600
(4) 021314 104415
(4) 021316 062706 000006

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

PRINTX #FMT9,#DH9

MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #L17,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH9,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

7499	021322			PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
(11)	021322	013746	002324		MOV AX3.16,-(SP)
(10)	021326	013746	002322		MOV AX3.15,-(SP)
(9)	021332	013746	002320		MOV AX2.16,-(SP)
(8)	021336	013746	002316		MOV AX2.15,-(SP)
(7)	021342	012746	012423		MOV #FMT6,-(SP)
(6)	021346	012746	000005		MOV #5,-(SP)
(3)	021352	010600			MOV SP,R0
(4)	021354	104415			TRAP C\$PNTX
(4)	021356	062706	000014		ADD #14,SP
7500	021362			ENDMSG	
(3)	021362				L10011: TRAP C\$MSG
(3)	021362	104423			
7501					
7502					
7503					
7504					
7505					
7506	021364			BGNMSG ERR10	
(3)	021364				ERR10::
7507	021364			PRINTB #FMT1,#ADDRES,MPCSR	
(9)	021364	013746	002436		MOV MPCSR,-(SP)
(8)	021370	012746	036452		MOV #ADDRES,-(SP)
(7)	021374	012746	012264		MOV #FMT1,-(SP)
(6)	021400	012746	000003		MOV #3,-(SP)
(3)	021404	010600			MOV SP,R0
(4)	021406	104414			TRAP C\$PNTB
(4)	021410	062706	000010		ADD #10,SP
7508	021414			PRINTB #FMT2	
(7)	021414	012746	012274		MOV #FMT2,-(SP)
(6)	021420	012746	000001		MOV #1,-(SP)
(3)	021424	010600			MOV SP,R0
(4)	021426	104414			TRAP C\$PNTB
(4)	021430	062706	000004		ADD #4,SP
7509	021434			PRINTB #FMT8,TMP1,TMP0	
(9)	021434	013746	002516		MOV TMP0,-(SP)
(8)	021440	013746	002520		MOV TMP1,-(SP)
(7)	021444	012746	012466		MOV #FMT8,-(SP)
(6)	021450	012746	000003		MOV #3,-(SP)
(3)	021454	010600			MOV SP,R0
(4)	021456	104414			TRAP C\$PNTB
(4)	021460	062706	000010		ADD #10,SP
7510	021464			PRINTX #FMT4,#DH2,#DH3	
(9)	021464	012746	015072		MOV #DH3,-(SP)
(8)	021470	012746	015043		MOV #DH2,-(SP)
(7)	021474	012746	012360		MOV #FMT4,-(SP)
(6)	021500	012746	000003		MOV #3,-(SP)
(3)	021504	010600			MOV SP,R0
(4)	021506	104415			TRAP C\$PNTX
(4)	021510	062706	000010		ADD #10,SP
7511	021514			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13	
(11)	021514	013746	002274		MOV LUR13,-(SP)
(10)	021520	013746	002272		MOV LUR12,-(SP)
(9)	021524	013746	002270		MOV LUR11,-(SP)
(8)	021530	013746	002266		MOV LUR10,-(SP)
(7)	021534	012746	012373		MOV #FMT5,-(SP)

(6) 021540 012746 000005
(3) 021544 010600
(4) 021546 104415
(4) 021550 062706 000014
7512 021554
(8) 021554 012746 015130
(7) 021560 012746 012522
(6) 021564 012746 000002
(3) 021570 010600
(4) 021572 104415
(4) 021574 062706 000006
7513 021600
(11) 021600 013746 002304
(10) 021604 013746 002302
(9) 021610 013746 002300
(8) 021614 013746 002276
(7) 021620 012746 012423
(6) 021624 012746 000005
(3) 021630 010600
(4) 021632 104415
(4) 021634 062706 000014
7514 021640
(9) 021640 012746 015232
(8) 021644 012746 015200
(7) 021650 012746 012360
(6) 021654 012746 000003
(3) 021660 010600
(4) 021662 104415
(4) 021664 062706 000010
7515 021670
(11) 021670 013746 002314
(10) 021674 013746 002312
(9) 021700 013746 002310
(8) 021704 013746 002306
(7) 021710 012746 012373
(6) 021714 012746 000005
(3) 021720 010600
(4) 021722 104415
(4) 021724 062706 000014
7516 021730
(8) 021730 012746 015271
(7) 021734 012746 012522
(6) 021740 012746 000002
(3) 021744 010600
(4) 021746 104415
(4) 021750 062706 000006
7517 021754
(11) 021754 013746 002324
(10) 021760 013746 002322
(9) 021764 013746 002320
(8) 021770 013746 002316
(7) 021774 012746 012423
(6) 022000 012746 000005
(3) 022004 010600
(4) 022006 104415
(4) 022010 062706 000014

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

PRINTX #FMT9,#DH9

PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16

MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH9,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV AX3.16,-(SP)
MOV AX3.15,-(SP)
MOV AX2.16,-(SP)
MOV AX2.15,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

CZDMSA M8203 STATIC TESTS #2
CZDMSA.P11 25-JUN-79 14:01

MACY11 30A(1052) 17-JUL-79 15:33 ^{C 9} PAGE 9-68
GLOBAL ERROR REPORT SECTION

SEQ 0106

7518 022014
(3) 022014
(3) 022014 104423
7519
7520
7521
7522
7523

ENDMSG

L10012: TRAP CSMSG

7525
7526
7527
7528
7529
7530
7531
7532
7533 022016
(3) 022016
7534
7535 022016
(3) 022016
(3) 022016 104425
7536
7537
7538
7539
7540
7541
7542

.SBTTL REPORT CODING SECTION

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

BGNRPT

LSRPT::

ENDRPT

L10013:

TRAP CSRPT

.EVEN


```
7562 .SBTTL INITIALIZE SECTION
7563
7564 :////////////////////////////////////////////////////////////////////
7565 :// THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
7566 :// AT THE BEGINNING OF THE TEST SEQUENCE ON THE NEXT UNIT.
7567 :////////////////////////////////////////////////////////////////////
7568
7569 022026 BGNINIT
7570 (3) 022026 LSINIT::
7571 022026 010637 002332 MOV SP,PSTACK ;SAVE BASE-LEVEL STACK POINTER
7572 022032 005037 002336 CLR SUBRPC ;CLEAR SUBR CALL PC
7573 022036 005037 002416 CLR DISILO ;CLEAR CURRENT STATE OF DISSI
7574 022042 005037 002420 CLR CHPTYP ;CLEAR USYRT CHIP TYPE INDICATOR
7575 022046 005037 002410 CLR ERROR1 ;CLEAR ERROR FLAGS
7576 022052 005037 002424 CLR SAVLEN ;CLEAR CHAR LENGTH FROM SETUP
7577 022056 005737 002376 TST FRSTIM ;SEE IF FIRST TIME THROUGH AFTER LOAD
7578 022062 001007 BR 6$ ;BR IF NOT
7579 022064 013737 000004 002404 MOV @#4,SAVE4 ;SAVE ERROR TRAP VECTOR
7580 022072 013737 000006 002406 MOV @#6,SAVE6
7581 022100 000406 BR 9$
7582 022102 013737 002404 000004 6$: MOV SAVE4,@#4 ;RESTORE ERROR TRAP VECTOR
7583 022110 013737 002406 000006 MOV SAVE6,@#6
7584 022116 012737 000001 002376 9$: MOV #1,FRSTIM ;MARK FLAG FOR NEXT TIME THROUGH
7585 ;SEE IF PROGRAM JUST STARTED, BR IF YES
7586 022124 READEF #EF.START
7587 (3) 022124 012700 000040 MOV #EF.START,R0
7588 (3) 022130 104447 TRAP CSREFG
7589 022132 BCOMPLETE STARST
7590 (2) 022132 103415 BCS STARST
7591 ;SEE IF PROGRAM JUST RESTARTED, BR IF YES
7592 022134 READEF #EF.RESTART
7593 (3) 022134 012700 000037 MOV #EF.RESTART,R0
7594 (3) 022140 104447 TRAP CSREFG
7595 022142 BCOMPLETE STARST
7596 (2) 022142 103411 BCS STARST
7597 ;SEE IF THIS IS A NEW PASS, BR IF YES
7598 022144 READEF #EF.NEW
7599 (3) 022144 012700 000035 MOV #EF.NEW,R0
7600 (3) 022150 104447 TRAP CSREFG
7601 022152 BCOMPLETE NEWST
7602 (2) 022152 103411 BCS NEWST
7603 ;SEE IF PROGRAM WAS JUST CONTINUED
7604 022154 READEF #EF.CONTINUE
7605 (3) 022154 012700 000036 MOV #EF.CONTINUE,R0
7606 (3) 022160 104447 TRAP CSREFG
7607 022162 BCOMPLETE ENDIT
7608 (2) 022162 103556 BCS ENDIT
7609 022164 000416 BR GETPRM
7610 022166 STARST:
7611 022166 005037 002402 CLR STARES ;CLEAR FLAG TO SHOW JUST HAD STA OR RES
7612 7600 ;CLEAR DEVICE MAP
7613 7601 022172 005037 002426 CLR DEVMAP
7614 7602 022176 NEWST:
7615 7603 022176 012737 177777 002330 MOV #-1,LOGDEV ;RESET LOGICAL DEVICE TO -1
7616 7604 022204 005237 002400 INC FRSPAS ;INCREMENT NO. OF PASSES AFTER LOAD
```

```
7605 022210 005237 002402          INC      STARES          ;INCREMENT NO. OF PASSES SINCE STA OR RES
7606 022214 012737 000001 002430  MOV      #BIT0,DEVPTTR  ;INIT DEVICE MAP BIT POINTER
7607                                     ; GET UNIBUS ADDRESS, VECTOR, PRIORITY LEVEL, SWITCH PACKS, TEST
7608                                     ; CONNECTOR INFORMATION FOR THIS LOGICAL DEVICE
7609 022222          GETPRM:
7610 022222 005237 002330          INC      LOGDEV          ;INCREMENT LOGICAL DEVICE NUMBER
7611 022226 023737 002330 002012  CMP      LOGDEV,L$UNIT  ;SEE IF MAXIMUM UNIT NO. EXCEEDED
7612 022234 002360          BGE      NEWST          ;BR IF YES
7613 022236          GPHARD LOGDEV,R1  ;GET P-TABLE POINTER INTO R1
(3) 022236 013700 002330          MOV      LOGDEV,RO
(3) 022242 104442          TRAP    ($GPHRD
(3) 022244 010001          MOV      RO,R1
7614 022246          BCOMPLETE      10$  ;BR IF DEVICE AVAILABLE
(2) 022246 103403          BCS      10$
7615 022250 006337 002430          ASL      DEVPTTR        ;SHIFT DEVICE MAP BIT POINTER
7616 022254 000762          BR       GETPRM         ;SKIP THIS DEVICE
7617 022256 053737 002430 002426 10$:  BIS      DEVPTTR,DEVMAP ;SET BIT FOR THIS DEVICE IN DEVICE MAP
7618 022264 006337 002430          ASL      DEVPTTR        ;SHIFT DEVICE MAP BIT POINTER
7619 022270 062701 000002          ADD      #2,R1          ;INCREMENT R1 PAST MICROPROCESSOR TYPE
7620 022274 011137 002436          MOV      (R1),MPCSR     ;STORE POINTER TO MICROPROCESSOR CSR'S
7621 022300 011137 002440          MOV      (R1),BSEL1
7622 022304 005237 002440          INC      BSEL1          ;GET POINTER TO BSEL1 (MAINTENANCE REGISTER)
7623 022310 011137 002442          MOV      (R1),SEL4
7624 022314 062737 000004 002442          ADD      #4,SEL4        ;GET POINTER TO SEL4
7625 022322 012137 002444          MOV      (R1)+,SEL6
7626 022326 062737 000006 002444          ADD      #6,SEL6        ;STORE POINTER TO SEL6
7627 022334 011137 002446          MOV      (R1),MPIVEC    ;GET MICROPROCESSOR INPUT INTRPT VECTOR
7628 022340 012137 002450          MOV      (R1)+,MPOVEC
7629 022344 062737 000004 002450          ADD      #4,MPOVEC      ;GET MICROPROCESSOR OUTPUT INTRPT VECTOR
7630 022352 012137 002452          MOV      (R1)+,MPRIOR   ;G T MICROPROCESSOR DEVICE PRIORITY
7631 022356 062701 000002          ADD      #2,R1          ;INCREMENT R1 PAST LU TYPE
7632 022362 012137 002454          MOV      (R1)+,LUSWI1   ;GET LU SWITCH PACK #1
7633 022366 012137 002456          MOV      (R1)+,LUSWI2   ;GET LU SWITCH PACK #2
7634 022372 012137 002460          MOV      (R1)+,LUSWI3   ;GET LU SWITCH PACK #3
7635 022376 012137 002462          MOV      (R1)+,TSTCON   ;GET TEST CONNECTOR INDICATOR
7636 022402 011137 002464          MOV      (R1),BDRATE    ;GET BAUD RATE
7637                                     ;SEE IF MANUAL INTERVENTION DESIRED BETWEEN UNITS FOR INSTALLATION OR REMOVAL
7638                                     ; OF TEST CONNECTORS, BR IF NOT
7639 022406 005737 002256          TST      MIFLAG
7640 022412 001442          BEQ     22$
7641                                     ;SEE IF MANUAL INTERVENTION ALLOWED BY SUPERVISOR
7642 022414          MANUAL
(3) 022414 104450          TRAP    ($MANI
7643                                     ;BR IF ALLOWED
7644 022416          BCOMPLETE      18$
(2) 022416 103412          BCS      18$
7645                                     ;PRINT MSG THAT OPERATOR INTERVENTION IS NOT ALLOWED
7646 022420          PRINTF #FMT16
(7) 022420 012746 022522          MOV      #FMT16,-(SP)
(6) 022424 012746 000001          MOV      #1,-(SP)
(3) 022430 010600          MOV      SP,RO
(4) 022432 104417          TRAP    ($PNTF
(4) 022434 062706 000004          ADD      #4,SP
7647 022440          16$:  BREAK          ;HANG UNTIL ^C TYPED
(3) 022440 104422          TRAP    ($BRK
7648 022442 000776          BR       16$
```

```
7649 022444
7650
7651 022444
(8) 022444 013746 002436
(7) 022450 012746 022644
(6) 022454 012746 000002
(3) 022460 010600
(4) 022462 104417
(4) 022464 062706 000006
7652 022470 005037 002502
7653 022474
7654
7655 022474
(3) 022474 104443
(3) 022476 000404
(4) 022500 002502
(5) 022502 000120
(5) 022504 022733
(5) 022506 000001
(3) 022510
7656 022510 023727 002502 000001
7657 022516 001366
7658 022520
7659 022520
7660 022520
(3) 022520
(3) 022520 104411
7661
7662 022522 047045 040445 040515
022530 052516 046101 044440
022536 052116 051105 042526
022544 052116 047511 020116
022552 051511 047040 052117
022560 040440 046114 053517
022566 042105 022441 116
7663 022573 045 052101 050131
022600 020105 047503 052116
022606 047522 026514 020103
022614 057050 024503 036040
022622 051103 020076 047524
022630 050040 047522 042503
022636 042105 022472 000116
7664 022644 047045 040445 047111
022652 052123 046101 020114
022660 042524 052123 041440
022666 047117 042516 052103
022674 051117 051450 020051
022702 047117 052440 044516
022710 020124 052101 040440
022716 051104 020123 020072
022724 022440 033117 047045
022732 000
7665 022733 124 C50131 020105
022740 054474 036076 051103
022746 020076 044127 047105
022754 051040 040505 054504
```

18\$:
;TYPE 'INSTALL TEST CONNECTOR(S) ON UNIT AT ADRS XXXXXX'
PRINTF #FMT17,MPCSR

MOV MPCSR, -(SP)
MOV #FMT17, -(SP)
MOV #2, -(SP)
MOV SP, R0
TRAP C\$PNTF
ADD #6, SP

CLR REG2

20\$:
;ASK OPERATOR TO 'TYPE <Y> <CR> WHEN READY TO PROCEED'
GMANIL TYPEY,REG2,1,NO

TRAP C\$GMAN
BR 10000\$
.WORD REG2
.WORD T\$CODE
.WORD TYPEY
.WORD 1

10000\$:

22\$:
ENDIT:
ENDINIT

L10015:
TRAP C\$INIT

FMT16: .ASCII /%N%AMANUAL INTERVENTION IS NOT ALLOWED!%N/

.ASCIZ /%ATYPE CONTROL-C (^C) <CR> TO PROCEED:%N/

FMT17: .ASCIZ /%N%AINSTALL TEST CONNECTOR(S) ON UNIT AT ADRS : %06%N/

TYPEY: .ASCIZ /TYPE <Y><CR> WHEN READY TO PROCEED /

022762	052040	020117	051120
022770	041517	042505	020104
022776	000		
	02300C		

.EVEN

7666
7667
7668
7669
7670
7671

7673
7674
7675
7676
7677
7678
7679
7680
(3)
7681
7682
(3)
(3)
7683
7684
7685
7686
7687
7688
7689
(3)
(3)
7690
7691
7692
(3)
(3)
7693
7694
7695
7696
7697

```
.SBTTL AUTO DROP UNIT SECTION
://////
:/ THE AUTO DROP CODING DETERMINES WHETHER OR NOT THE DEVICE WHOSE P-TABLE
:/ WAS JUST OBTAINED IS READY FOR TESTING, AND IT IS DROPPED IF NOT READY.
://////

          BGNAUTO
          L$AUTO::
;ESTABLISH PRIORITY = 7
          SETPRI #PRI07
          MOV #PRI07,R0
          TRAP C$SPRI
;SET UP NON-EXISTENT MEMORY ERROR TRAP VECTOR
          MOV #6$,@#4
          MOV #PRI07,@#6
          TST @MPCSR
          BR 9$
;ADDRESS SELO
;TAKE THIS BRANCH IF DEVICE RESPONDS
;COME HERE IF DEVICE CSR IS NON-EXISTENT
6$: ADD #4,SP
          DODU LOGDEV
;DROP THIS UNIT FROM TESTING
          MOV LOGDEV,R0
          TRAP C$DODU
9$: MOV SAVE4,@#4
          MOV SAVE6,@#6
          ENDAUTO
          L10016:
          TRAP C$AUTO
```


7699
7700
7701
7702
7703
7704
7705
7706 023060
(3) 023060
7707
7708
7709 023060
(3) 023060
(3) 023060 104412
7710
7711
7712
7713
7714

.SBTTL CLEANUP CODING SECTION

://
:// THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
:// AT THE END OF THE TEST SEQUENCE ON A PARTICULAR UNIT.
://

BGNCLN

L\$CLEAN::

ENDCLN

L10017: TRAP C\$CLEAN

```
7716 .SBITL DROP UNIT SECTION
7717
7718 :////////////////////////////////////////////////////////////////////
7719 :// THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
7720 :// TO NO LONGER BE TESTED.
7721 :////////////////////////////////////////////////////////////////////
7722
7723 023062          BGNDU
7724 (3) 023062          LSDU::
7725 ;ISSUE UNIBUS RESET TO CLEAN UP
7726 (3) 023062 104433          BRESET
7727 ;PRINT 'UNIT XX DROPPED'
7728 (8) 023064 013746 002330          TRAP    C$RESET
7729 (7) 023070 012746 023112          MOV     LOGDEV,-(SP)
7730 (6) 023074 012746 000002          MOV     #FMT27,-(SP)
7731 (3) 023100 010600          MOV     #2,-(SP)
7732 (4) 023102 104417          MOV     SP,R0
7733 (4) 023104 062706 000006          TRAP   C$PNTF
7734 023110          ENDDU          ADD     #6,SP
7735 (3) 023110 104453          L10020:
7736 (3) 023110          TRAP    C$DU
7737 7729 023112 047045 040445 047125 FMT27: .ASCIZ /%N%AUNIT %D2%A DROPPED%N/
7738 023120 052111 022440 031104
7739 023126 040445 042040 047522
7740 023134 050120 042105 047045
7741 023142 000
7742 7731 023144          .EVEN
7743
7744
7745
7746
```

7738
7739
7740
7741
7742
7743
7744
7745
7746
(3)
7747
(3)
(3)
7748
7749
7750
7751
7752
7753

023144
023144
023144
023144 104452

.SBTTL ADD UNIT SECTION

:/ THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
:/ TO BE (A) TESTED FOR THE FIRST TIME, OR (B) RESUMED IN TESTING. IF
:/ 'EF.AUNIT' IS SET, THE UNIT WILL BE TESTED AS A NEW UNIT.

BGNAU

ENDAU

LSAU::

L10021:

TRAP CSAU

.SBTTL HARDWARE TESTS

7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807

023146
(3) 023146
023146 012737 023270 002346
023154 004737 005212
023160 000000
023162 000300
023164 004737 010554
023170 002642
023172 000020
023174 012737 001000 002412
023202 004737 004646
023206 004737 004646
023212 004737 004726
023216 000300
023220 012701 002642
023224 112137 023234 6\$:
023230 004737 007266 8\$:
023234 000000
023236 000000
023240 020127 002661
023244 103767
023246 111137 023264
023252 052737 001000 023264
023260 004737 007266
023264 000000 12\$:
023266 000000
023270 004737 003276 24\$:
023274
(3) 023274
(3) 023274 104401

```
*****
.SBTTL TEST 1 - BIT STUFFING TEST
*
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
* INITIATED IN BIT MODE . TWO LEADING FLAGS ARE SENT,
* FOLLOWED BY ALL SIXTEEN CHARS IN DATA PATTERN S. THIS PATTERN
* CONSISTS OF CHARACTERS WHICH REQUIRE NO BIT STUFFING AND CHARACTERS
* WHICH REQUIRE BIT STUFFING INDIVIDUALLY AND IN COMBINATION WITH
* ADJACENT CHARACTERS. ALL 16 CHARACTERS ARE READ AND COMPARED
* BY THE RECEIVER.
* PATTERN S = 000,017,036,074,170,360,037,076,174,370,077,176,374,
* 177,376,377
*****
BGNTST
```

```
T1::
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
CRC2!CRC1 ;BIT MODE, NO ERR DETECTION
JSR PC,LDBYTS ;LOAD PAT S INTO TX SILO
PATS
16.
MOV #TXEOM,TXWORD ;LOAD 2 EOM'S INTO TX SILO
JSR PC,LDTXSI
JSR PC,LDTXSI
JSR PC,STPLU ;CLK MORE THAN ENTIRE MSG
192.
MOV #PATS,R1 ;INIT PAT S POINTER
6$: MOVB (R1)+,8$
JSR PC,CKDATA ;CHK A RCV'D CHAR
8$: .WORD 0
0
CMP R1,#PATS+15. ;SEE IF 15 CHARS CHECKED YET
BLO 6$ ;BR IF NOT YET
MOVB (R1),12$
BIS #RXEBL,12$ ;GET SET TO CHK EBLK = 1
JSR PC,CKDATA ;CHK LAST CHAR AND EBLK - 1
12$: .WORD 0
0
24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10022: TRAP C$ETST
```

```
*****
.SBTTL TEST 2 - RCV OVERRUN ERROR SET AND CLEAR TEST
```

7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819 023276
(3) 023276
7820 023276 012737 023644 002346
7821
7822
7823
7824 023304 004737 005212
7825 023310 000226
7826 023312 000311
7827 023314 004737 010716
7828 023320 000001
7829 023322 000100
7830 023324 004737 006752
7831 023330 000030
7832 023332 004737 004726
7833 023336 000730
7834 023340 004737 010716
7835 023344 000001
7836 023346 000074
7837 023350 004737 004726
7838 023354 000100
7839 023356 012701 000100
7840 023362 004737 007266
7841 023366 000001
7842 023370 000000
7843 023372 005301
7844 023374 001372
7845 023376 004737 007266
7846 023402 004001
7847 023404 000010
7848 023406 004737 007266
7849 023412 004001
7850 023414 000010
7851 023416 012737 000012 002364
7852 023424 012737 000200 002352
7853 023432 004737 003422
7854 023436 004737 006672
7855 023442 132737 000010 002415
7856 023450 001407
7857 023452 004737 004200
7858
7859 023456
(4) 023456 104455
(5) 023460 000051
(5) 023462 014040

```

:
: * IN THIS TEST, A RCV OVERRUN ERROR IS FORCED IN EACH OF 2 SUBTESTS.
: * IN THE FIRST, A MESSAGE IS INITIATED, 64 001 CHARS ARE SENT, AND THE
: * RECEIVER IS NOT SERVICED IN RESPONSE TO THE USYRT RCV FLAG, WHICH CAUSES RCV
: * OVERRUN TO SET. THEN, A CHECK IS MADE TO INSURE THAT OVRR IS NOT
: * CLEARED BY THE LINE UNIT READING THE USYRT STATUS.
: * THEN, IC IS SET TO CLEAR THE ERROR, AND THIS IS VERIFIED.
:
: * IN THE SECOND SUBTEST, RCV OVRUN IS FORCED AGAIN, AND A MASTER CLEAR
: * IS ISSUED TO CLEAR THE ERROR, AND THIS IS VERIFIED.
:
:*****

```

BGN1ST

```

:*****
:
: T2::
:-----
: MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
:-----
: CAUSE OVRR, SET IC TO CLEAR IT
:-----
:-----
: JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
: SYNCH
: CRC2!CRC1!STRIP!DDCMP ;DDCMP, NO ERR DET
: JSR PC,LODSIL ;LOAD 64 001 CHARS INTO TX SILO
: 001
: 64.
: JSR PC,RCV1ST ;CLOCK UNTIL FIRST DATA CHAR RCV'D
: 24.
: JSR PC,STPLU ;CLOCK UNTIL 59 MORE RCV'D
: 472.
: JSR PC,LODSIL ;LOAD 60 MORE INTO TX SILO
: 001
: 60.
: JSR PC,STPLU ;CLK 8 MORE TIMES TO FORCE UNDERRUN
: 64.
: MOV #64,R1 ;READ AND CHK 64 CHARS FROM RCV SILO
: 6$: JSR PC,CKDATA
: 001
: 0
: DEC R1
: BNE 6$
: JSR PC,CKDATA ;READ CHAR, CHK OVRR - 1
: 4001
: 8.
: JSR PC,CKDATA ;READ CHAR, CHK OVRR STILL = 1
: 4001
: 8.
: MOV #12,REGNUM ;SET REG NO. = 12
: MOV #IC,WRIBYT
: JSR PC,WRITLU ;SET IC TO CLEAR RCVR
: JSR PC,RDRXSI ;READ RCV SILO
: BITB #OVRR,RXWORD+1 ;CHK FOR OVRR CLEARED
: BEQ 8$ ;BR IF OVRR CLEARED
: JSR PC,GETALL ;GET REGS FOR PRINTOUT
:
: REPORT OVRR NOT CLEARED
: ERRDF 41,EM41,ERR7

```

```

TRAP C$ERDF
.WORD 41
.WORD EM41

```

```
(5) 023464 020224  
7860 023466 000466  
7861 023470  
7862  
7863  
7864  
7865 023470 004737 005212  
7866 023474 000226  
7867 023476 000311  
7868 023500 004737 010716  
7869 023504 000001  
7870 023506 000100  
7871 023510 004737 006752  
7872 023514 000030  
7873 023516 004737 004726  
7874 023522 000730  
7875 023524 004737 010716  
7876 023530 000001  
7877 023532 000074  
7878 023534 004737 004726  
7879 023540 000100  
7880 023542 012701 000100  
7881 023546 004737 007266  
7882 023552 000001  
7883 023554 000000  
7884 023556 005301  
7885 023560 001372  
7886 023562 004737 007266  
7887 023566 004001  
7888 023570 000010  
7889 023572 004737 007266  
7890 023576 004001  
7891 023600 000010  
7892 023602 012737 000012 002364  
7893 023610 004737 003276  
7894 023614 004737 006672  
7895 023620 132737 000010 002415  
7896 023626 001406  
7897 023630 004737 004200  
7898  
7899 023634  
(4) 023634 104455  
(5) 023636 000051  
(5) 023640 014040  
(5) 023642 020224  
7900 023644 004737 003276  
7901 023650  
(3) 023650  
(3) 023650 104401  
7902  
7903  
7904  
7905  
7906  
7907  
7908
```

BR 24\$
8\$:-----
: CAUSE OVRR, SET MST CLR TO CLEAR IT
:-----
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
SYNCH
CRC2!CRC1!STRIP!DDCMP ;DDCMP, NO ERR DET
JSR PC,LODSIL ;LOAD 64 001 CHARS INTO TX SILO
001
64.
JSR PC,RCV1ST ;CLOCK UNTIL FIRST DATA CHAR RCV'D
24.
JSR PC,STPLU ;CLOCK UNTIL 59 MORE RCV'D
472.
JSR PC,LODSIL ;LOAD 60 MORE INTO TX SILO
001
60.
JSR PC,STPLU ;CLK 8 MORE TIMES TO FORCE UNDERRUN
64.
MOV #64,R1 ;READ AND CHK 64 CHARS FROM RCV SILO
9\$: JSR PC,CKDATA
001
0
DEC R1
BNE 9\$
JSR PC,CKDATA ;READ CHAR, CHK OVRR = 1
4001
8.
JSR PC,CKDATA ;READ CHAR, CHK OVRR STILL = 1
4001
8.
MOV #12,REGNUM ;SET REG NO. = 12
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,RDRXSI ;READ RCV SILO
BITB #OVRR,RXWORD+1 ;CHK FOR OVRR CLEARED
BEQ 24\$;BR IF OVRR CLEARED
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT OVRR NOT CLEARED
ERRDF 41,EM41,ERR7
TRAP C\$ERDF
.WORD 41
.WORD EM41
.WORD ERR7
24\$: JSR PC,MSTCLR ;ISSUE CLEAN UP MST CLR
ENDTST
L10023:
TRAP C\$ETST
:*****
:SBTTL TEST 3 - ABORT SEQUENCE TEST

```

7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
  
```

```

: *
: * SET BIT MODE, CRC, AND ENABLE THE DEVICE FOR
: * TRANSMIT AND RECEIVE. SEND 2 FLAGS AND 4 DATA CHARS (001).
: * AS THE FIRST DATA CHAR IS BEING TRANSMITTED,
: * SET THE ABORT BIT (REG 11).
: * ON THE RECEIVER SIDE, CHECK FOR RECEPTION OF THE FIRST DATA CHAR
: * AND THEN THE SETTING OF RAB AND REOM A CHAR TIME LATER.
: * ALSO, CHECK FOR IACT = 0. THEN, CHECK THAT RAB
: * IS CLEARED BY READING THE USYRT STATUS, TRANSMITTING A NEW MSG,
: * RECEIVING THE FIRST CHAR (003) AND CHECKING FOR RAB CLEARED.
: *
: * REPEAT THE ABOVE SEQUENCE, SET IC, AND CHECK THAT
: * THIS CLEARS RAB.
: *
: * REPEAT THE ABOVE SEQUENCE, ISSUE MASTER CLEAR, CHECK THAT THIS
: * CLEARS RAB.
: *
: *****
  
```

 BGN*ST

```

7927 023652
(3) 023652
7928 023652 012737 024202 002346
7929
7930
7931
7932 023660 004737 005212
7933 023664 000000
7934 023666 000000
7935 023670 004737 010476
7936 023674 002732
7937 023676 000014
7938 023700 004737 006752
7939 023704 000060
7940 023706 004737 007266
7941 023712 000001
7942 023714 000010
7943 023716 004737 007266
7944 023722 003001
7945 023724 000000
7946 023726 004737 006232
7947 023732 000000
7948 023734 004737 006752
7949 023740 000060
7950 023742 004737 007266
7951 023746 000003
7952 023750 000000
7953
7954
7955
7956 023752 004737 005212
7957 023756 000000
7958 023760 000000
7959 023762 004737 010476
7960 023766 002732
7961 023770 000014
7962 023772 004737 006752
7963 023776 000060
  
```

```

:-----T3:-----
: MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
:-----
: CAUSE ABORT, START NEW MSG TO CLEAR IT
:-----
: JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
: 000
: 000
: JSR PC,LODMSG ;BIT MODE, CRC
: MSG3 ;LOAD MSG INTO TX SILO
: 12.
: JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
: 48.
: JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
: 001
: 8.
: JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
: RXABT!RXEBL!001
: 0
: JSR PC,IACTIV ;CHK FOR IACT = 0
: 0
: JSR PC,RCV1ST ;CLK AND RCV NEW MSG
: 48.
: JSR PC,CKDATA ;CHK CHAR = 003
: 003
: 0
:-----
: CAUSE ABORT, SET IC TO CLEAR IT
:-----
: JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
: 000
: 000
: JSR PC,LODMSG ;BIT MODE, CRC
: MSG3 ;LOAD MSG INTO TX SILO
: 12.
: JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
: 48.
  
```

```
7964 024000 004737 007266 JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
7965 024004 000001 001
7966 024006 000010 8.
7967 024010 004737 007266 JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
7968 024014 003001 RXABT!RXEBL!001
7969 024016 000000 0
7970 024020 012737 000012 002364 MOV #12,REGNUM ;SET REG NO. = 12
7971 024026 012737 000200 002352 MOV #IC,WRIBYT
7972 024034 004737 003422 JSR PC,WRITLU ;SET IC TO CLEAR RCVR
7973 024040 004737 006672 JSR PC,RDRXSI ;READ RCV SILO
7974 024044 132737 000004 002415 BITB #RAB,RXWORD+1 ;CHK FOR RAB CLEARED
7975 024052 001407 BEQ 8$ ;BR IF RAB CLEARED
7976 024054 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
7977 :REPORT RAB NOT CLEARED
7978 024060 ERRDF 39,EM39,ERR7
(4) 024060 104455 TRAP C$ERDF
(5) 024062 000047 .WORD 39
(5) 024064 014004 .WORD EM39
(5) 024066 020224 .WORD ERR7
7979 024070 000444 BR 24$
7980 024072 8$:
7981 -----
7982 :CAUSE ABORT, ISSUE MASTER CLEAR TO CLEAR IT
7983 -----
7984 024072 004737 005212 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
7985 024076 000000 000
7986 024100 000000 000 ;BIT MODE, CRC
7987 024102 004737 010476 JSR PC,LODMSG ;LOAD MSG INTO TX SILO
7988 024106 002732 MSG3
7989 024110 000014 12.
7990 024112 004737 006752 JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
7991 024116 000060 48.
7992 024120 004737 007266 JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
7993 024124 000001 001
7994 024126 000010 8.
7995 024130 004737 007266 JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
7996 024134 003001 RXABT!RXEBL!001
7997 024136 000000 0
7998 024140 012737 000012 002364 MOV #12,REGNUM ;SET REG NO. = 12
7999 024146 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
8000 024152 004737 006672 JSR PC,RDRXSI ;READ RCV SILO
8001 024156 132737 000004 002415 BITB #RAB,RXWORD+1 ;CHK FOR RAB CLEARED
8002 024164 001406 BEQ 24$ ;BR IF RAB CLEARED
8003 024166 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
8004 :REPORT RAB NOT CLEARED
8005 024172 ERRDF 39,EM39,ERR7
(4) 024172 104455 TRAP C$ERDF
(5) 024174 000047 .WORD 39
(5) 024176 014004 .WORD EM39
(5) 024200 020224 .WORD ERR7
8006 024202 004737 003276 24$: JSR PC,MSTCLR ;ISSUE MST CLR TO CLEAN UP
8007 024206 ENDTST
(3) 024206 L10024:
(3) 024206 104401 TRAP C$ETST
8008
8009
```


8010
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020

:SBTTL TEST 4 - ABORT AND IDLE FLAGS TEST
:*****
: * TRANSMIT THE SAME ABORT SEQUENCE AS IN THE PREVIOUS TEST, BUT
: * WITH THE IDLE BIT SET. CHECK THAT FLAGS ARE SENT AND RECEIVED
: * (NOT ABORT CHARACTERS) BY VERIFYING THAT RAB DOES
: * NOT SET, AND THAT THE MESSAGE TERMINATES WITH EBLK = 1.
:*****

8021 024210
(3) 024210
8022 024210 012737 024264 002346
8023 024216 004737 005212
8024 024222 000000
8025 024224 000040
8026 024226 004737 010476
8027 024232 002732
8028 024234 000005
8029 024236 004737 006752
8030 024242 000060
8031 024244 004737 007266
8032 024250 000001
8033 024252 000010
8034 024254 004737 007266
8035 024260 001001
8036 024262 000000
8037 024264 004737 003276
8038 024270
(3) 024270
(3) 024270 104401

BGNTST
T4::
MOV #24\$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
IDLE ;BIT MODE, NO ERROR DET
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG3
5
JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
48.
JSR PC,CKDATA ;CHK CHR = 001, CLK FLAG CHAR
001
8.
JSR PC,CKDATA ;CHK RAB = 0, EBLK = 1
RXEBL!001
0
24\$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR
ENDTST
L10025: TRAP CSETST

8039
8040
8041
8042
8043
8044

:SBTTL TEST 5 - TRANSMITTER UNDERRUN ERROR, IDLE ABORT CHARS, BIT MODE
:*****
: * A MESSAGE IS INITIATED IN BIT MODE, 4 001 CHARS ARE SENT, AND THE TRANSMITTER
: * IS NOT SERVICED IN RESPONSE TO THE LAST TX FLAG, WHICH CAUSES TX
: * UNDERRUN ERROR TO SET. ON THE RECEIVER SIDE, CHECK THAT THE DATA
: * CHAR IS RECEIVED, AND THAT 8 CYCLES LATER THE RAB BIT SETS, AND
: * THE DEVICE IDLES ABORT CHARACTERS.
:*****

8053 024272
(3) 024272
8054 024272 012737 024354 002346
8055 024300 004737 005212
8056 024304 000000
8057 024306 000000
8058 024310 012737 000100 002416
8059 024316 004737 010716
8060 024322 000001
8061 024324 000004

BGNTST
T5::
MOV #24\$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
000
MOV #TXEN,DISILO ;SET TX ENB TO KEEP RTS HIGH
JSR PC,LODSIL ;LOAD 4 001 CHARS INTO TX SILO
001
4

8062 024326 004737 006752
8063 024332 000060
8064 024334 004737 007266
8065 024340 000001
8066 024342 000011
8067 024344 004737 007266
8068 024350 003001
8069 024352 000000
8070 024354 004737 003276
8071 024360
(3) 024360
(3) 024360 104401

JSR PC,RCV1ST ;CLK AND RCV FIRST CHAR
48.
JSR PC,CKDATA ;CHK DATA = 001, CLOCK ABORT CHAR
001
9.
JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
RXABT.RXEBL!001
C
248: JSR PC,MSTCLR ;ISSUE MASTER CLEAR
END*ST
L10026.
TRAP C\$ETST

8072
8073
8074
8075
8076
8077

:SBTTL TEST 6 - RECEIVER DISABLE TEST
:
: * TRANSMIT AND RECEIVE ARE ENABLED IN BIT MODE, AND 2 FLAGS
: * ARE SENT, FOLLOWED BY 5 252 DATA CHARS. AFTER THE SECOND DATA CHAR HAS BEGUN
: * TO BE RECEIVED, IC IS SET.
: * THEN, THE PROGRAM CHECKS THAT A USYRT RCV FLAG IS NOT GENERATED, AND
: * THE RECEIVER DATA PATH STOPS OPERATING IN THE MIDDLE OF THE CHAR.
:*****
BGN*ST

8086 024362
(3) 024362
8087 024362 012737 024550 002346
8088 024370 004737 005212
8089 024374 000000
8090 024376 000000
8091 024400 004737 010716
8092 024404 000252
8093 024406 000005
8094 024410 004737 006752
8095 024414 000060
8096 024416 004737 004726
8097 024422 000004
8098 024424 012737 000012 002364
8099 024432 012737 000200 002352
8100 024440 004737 003422
8101 024444 004737 006232
8102 024450 000000
8103 024452 004737 005746
8104 024456 000001
8105 024460 005037 002370
8106 024464 005037 002372
8107 024470 004737 006672
8108 024474 105737 002414
8109 024500 001404
8110 024502 012737 000010 002364
8111 024510 000406
8112 024512 105737 002415
8113 024516 001414
8114 024520 012737 000012 002364

T6:;
MOV #24\$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
000 ;BIT MODE, CRC
JSR PC,LODSIL ;LOAD 5 252 CHARS
252
5
JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
48.
JSR PC,STPLU ;CLK TO MIDDLE OF 2ND CHAR
4
MOV #12,REGNUM ;SET REG NO. = 12
MOV #IC,WRIBYT
JSR PC,WRITLU ;SET IC IN REG 12
JSR PC,IACTIV ;CHK IACT = 0
0
JSR PC,ISIRDY ;CHK ICIR = 1, IRDY = 0
1
CLR GOODAT ;SET EXPECTED DATA = 0
CLR BADDAT
JSR PC,RDRXSI ;READ RCV SILO
TSTB RXWORD ;SEE IF SILO BITS 0-7 = 000
BEQ 9\$;BR IF YES
MOV #10,REGNUM ;SET REG NO. = 10
BR 12\$
TSTB RXWORD+1 ;SEE IF SILO BITS 8-11 = 000
BEQ 24\$;BR IF YES
MOV #12,REGNUM ;SET REG NO. = 12

9\$:

```

8115 024526 113737 002414 002372 128:  MOVB  RXWORD,BADDAT ;GET ACTUAL DATA
8116 024534 004737 004200          JSR   PC,GETALL  ;GET REGS FOR PRINTOUT
8117                                ;REPORT RCV SILO NOT CLEARED BY IC
8118 024540                                ERRDF  46,EM46,ERR2
      (4) 024540 104455                                TRAP  C$ERDF
      (5) 024542 000056                                .WORD 46
      (5) 024544 014175                                .WORD EM46
      (5) 024546 015366                                .WORD ERR2
8119 024550 004737 003276 248:  JSR   PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
8120 024554 004737 003276  ENDTST
      (3) 024554                                L10027:
      (3) 024554 104401                                TRAP  C$ETST

```

8121
8122
8123
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150
8151
8152
8153
8154
8155
8156
8157
8158
8159
8160
8161
8162
8163

```

:*****
:SBTTL      TEST 7 - ASSEMBLED BIT COUNT TEST
:
: THE FOLLOWING SEQUENCE IS PERFORMED 8 TIMES, EACH TIME USING A
: DIFFERENT TX CHAR LENGTH (FROM 2 TO 8 BITS) AND A RCV CHAR LENGTH - 8
: BITS :
: A MESSAGE IS INITIATED IN BIT MODE, NO CRC.
: 2 FLAGS ARE SENT, FOLLOWED BY 3 000 DATA CHARACTERS AND A
: TERMINATING FLAG. AFTER THE RECEIVER HAS RECEIVED THE MESSAGE, AX0-16
: IS READ TO RETRIEVE THE ASSEMBLED BIT COUNT. THIS COUNT IS CHECKED TO INSURE
: THAT IT IS CORRECT FOR THE TX CHAR LENGTH USED IN THAT TRANSMISSION.
:*****

```

```

BGNTST
      (3) 024556
8139 024556 012737 025314 002346          MOV   #248,RETADR ;SET TEST EXIT ADRS FOR ERRORS
8140 024564 004737 005212          JSR   PC,INITRN  ;FIND OUT WHICH USYRT CHIP
8141 024570 000000          0
8142 024572 000000          0
8143 024574 012701 000100          MOV   #TXLEN1,R1 ;SET INITIAL TX LENGTH TO 2 BITS
8144 024600 004737 003276 68:  JSR   PC,MSTCLR  ;ISSUE MASTER CLEAR
8145 024604 004737 010304          JSR   PC,SETUP   ;PROGRAM THE USYRT
8146 024610 000000          000
8147 024612 000300          CRC2!CRC1
8148 024614 000000          000
8149 024616 000000          000
8150 024620 012737 000014 002364          MOV   #14,REGNUM ;SET REG NO. - 14
8151 024626 012737 000140 002352          MOV   #TXEN!DISSI,WRIBYT
8152 024634 004737 003422          JSR   PC,WRITLU  ;SET TXEN AND DISSI IN REG 14
8153 024640 012737 000140 002416          MOV   #TXEN!DISSI,DISILO ;SET DISABLE SILO FLAG
8154 024646 012737 000012 002364          MOV   #12,REGNUM ;SET LU REG NO. - 12
8155 024654 112737 000040 002352          MOVB  #LULP,WRIBYT
8156 024662 004737 003422          JSR   PC,WRITLU  ;SET LULP IN REG 12
8157 024666 012737 000002 002366          MOV   #2,AXNUM  ;SET AX BYTE NO. = 2
8158 024674 105037 002360          CLRB  WAX15
8159 024700 112737 000001 002362          MOVB  #TSOM,WAX16
8160 024706 004737 003764          JSR   PC,WRITAX  ;LOAD SOM CHAR
8161 024712 005004          CLR   R4         ;INIT COUNTER
8162 024714 012737 000011 002364          MOV   #11,REGNUM ;SET REG NO. - 11
8163 024722 004737 004726 78:  JSR   PC,STPLU   ;CLOCK LU FOR A CYCLE

```

8164	024726	000001			1				
8165	024730	004737	003344		JSR	PC,READLU		:READ REG 11	
8166	024734	132737	000100	002350	BITB	#OACT,REDBYT		:SEE IF OACT SET YET	
8167	024742	001014			BNE	10\$:BR IF OACT SET	
8168	024744	005204			INC	R4		:INCR COUNTER	
8169	024746	020427	000004		CMP	R4,#4		:SEE IF COUNT TOO BIG	
8170	024752	002763			BLT	7\$:BR IF NOT	
8171	024754	004737	004200		JSR	PC,GETALL		:GET REGS FOR PRINTOUT	
8172					:REPORT	OACT NOT SET			
8173	024760				ERRDF	11,EM11,ERR7			
(4)	024760	104455						TRAP	C\$ERDF
(5)	024762	000013						.WORD	11
(5)	024764	013400						.WORD	EM11
(5)	024766	020224						.WORD	ERR7
8174	024770	000137	025314		JMP	24\$			
8175	024774	004737	003764	10\$:	JSR	PC,WRITAX		:LOAD ANOTHER SOM CHAR	
8176	025000	004737	004726		JSR	PC,STPLU		:CLK FIRST FLAG	
8177	025004	000010			8.				
8178	025006	105037	002362		CLRB	WAX16			
8179	025012	004737	003764		JSR	PC,WRITAX		:LOAD FIRST 000 CHAR	
8180	025016	004737	004726		JSR	PC,STPLU		:CLK SECOND FLAG	
8181	025022	000010			8.				
8182	025024	004737	003764		JSR	PC,WRITAX		:LOAD SECOND 000 CHAR	
8183	025030	004737	004726		JSR	PC,STPLU		:CLK FIRST 000 CHAR	
8184	025034	000010			8.				
8185	025036	012737	000006	002366	MOV	#6,AXNUM		:SET AX BYTE NO. FOR AX 3	
8186	025044	010137	002362		MOV	R1,WAX16		:GET TX CHAR LENGTH	
8187	025050	004737	003764		JSR	PC,WRITAX		:SET TX CHAR LENGTH	
8188	025054	012737	000002	002366	MOV	#2,AXNUM		:SET AX BYTE NO. = 2	
8189	025062	105037	002362		CLRB	WAX16			
8190	025066	005737	002420		TST	CHPTYP		:SEE IF SIG USYRT	
8191	025072	001403			BEO	5\$:BR IF YES	
8192	025074	112737	000002	002362	MOVB	#TEOM,WAX16		:SET TEOM WITH LAST DATA CHAR	
8193	025102	004737	003764	5\$:	JSR	PC,WRITAX		:LOAD 3RD 000 CHAR	
8194	025106	004737	004726		JSR	PC,STPLU		:CLK 2ND 000 CHAR	
8195	025112	000010			8.				
8196	025114	005737	002420		TST	CHPTYP		:SEE IF SIG USYRT	
8197	025120	001005			BNE	16\$:BR IF NOT	
8198	025122	112737	000002	002362	MOVB	#TEOM,WAX16			
8199	025130	004737	003764		JSR	PC,WRITAX		:LOAD AN EOM CHAR	
8200	025134	012737	000001	002366	MOV	#1,AXNUM		:SET AX BYTE NO. - 1	
8201	025142	005003			CLR	R3			
8202	025144	004737	003576	12\$:	JSR	PC,READAX		:READ AX0	
8203	025150	132737	000002	002356	BITB	#REOM,RAX16		:CHK FOR REOM = 1	
8204	025156	001016			BNE	14\$:BR IF YES	
8205	025160	004737	004726		JSR	PC,STPLU		:CLOCK LU FOR A CYCLE	
8206	025164	000001			1				
8207	025166	005203			INC	R3		:INCR COUNT	
8208	025170	020327	000023		CMP	R3,#19.		:SEE IF COUNT TOO BIG	
8209	025174	002763			BLT	12\$:BR IF NOT	
8210	025176	004737	004200		JSR	PC,GETALL		:GET REGS FOR PRINTOUT	
8211					:REPORT	REOM NOT SET			
8212	025202				ERRDF	31,EM31,ERR10			
(4)	025202	104455						TRAP	C\$ERDF
(5)	025204	000037						.WORD	31
(5)	025206	013647						.WORD	EM31

```

(5) 025210 021364
8213 025212 000440
8214 025214 013702 002356
8215 025220 042702 000217
8216 025224 006102
8217 025226 120201
8218 025230 001421
8219 025232 010137 002370
8220 025236 006237 002370
8221 025242 152737 000002 002370
8222 025250 013737 002356 002372
8223 025256 004737 004200
8224
8225 025262
(4) 025262 104455
(5) 025264 000057
(5) 025266 014230
(5) 025270 015674
8226 025272 000410
8227 025274 005701
8228 025276 001406
8229 025300 062701 000040
8230 025304 042701 000400
8231 025310 000137 024600
8232 025314 005037 002416
8233 025320 004737 003276
8234 025324
(3) 025324
(3) 025324 104401
8235
8236
8237
8238
8239
8240
8241
8242
8243
8244
8245
8246
8247
8248
8249
8250
8251
8252
8253
8254
8255
8256
8257
8258
8259
8260 025326
(3) 025326

```

```

14$: BR 24$
MOV RAX16,R2 ;GET AX0-16 CONTENTS
BIC #217,R2 ;MASK OFF ALL BUT ASSEMB BIT COUNT
ROL R2
CMPB R2,R1 ;CHK FOR CORRECT ASSEMB BIT COUNT
BEQ 9$ ;BR IF MATCH
MOV R1,GOODAT ;SET EXPECTED DATA
ASR GOODAT
BISB #REOM,GOODAT
MOV RAX16,BADAT ;SET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT ASSEMB BIT COUNT INCORRECT
ERRDF 47,EM47,ERR3

9$: BR 24$
TST R1 ;SEE IF ALL DONE YET
BEQ 24$ ;BR IF YES
ADD #TXLEN0,R1 ;INCR TX LENGTH
BIC #400,R1 ;MASK OFF OVERFLOW IF 8 BITS
JMP 6$ ;PROCEED

24$: CLR DISILO
JSR PC,MSTCLR ;ISSUE MASTER CLR TO CLEAN UP

L10030: TRAP C$ERDF
TRAP C$SETST

```

```

:*****
:SBTTL TEST 8 - SECONDARY STATION ADDRESS BIT TEST
:
:* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
:* BIT MODE, AND THE SECA BIT (REG 17) IS SET.
:* 2 FLAGS ARE SENT, FOLLOWED BY 252, 000, AND A TERMINATING FLAG.
:* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
:* RECEIVED.
:
:* NEXT, THE SECONDARY STATION ADDRESS BITS IN AX2-15 ARE LOADED
:* WITH THE FIRST WORD OF DATA PATTERN T. 2 FLAGS ARE SENT,
:* FOLLOWED BY THE FIRST WORD OF DATA PATTERN T, A 000 CHAR,
:* AND A TERMINATING FLAG.
:* THEN, THE RCV'D DATA IS CHECKED TO MAKE SURE THAT THE SEC STATION
:* ADDRESS IS RCV'D AS THE FIRST DATA CHAR, FOLLOWED BY 000.
:
:* THEN, THE SUBTEST IS REPEATED FOR EACH OF THE REMAINING WORDS OF
:* DATA PATTERN T.
:* PATTERN T = 000,125,252,176,177
:*****
BGNTST

```

T8::

```
8261  
8262  
8263  
8264 025326  
      (3) 025326  
      (3) 025326 104402  
8265 025330 012737 025426 002346  
8266 025336 004737 005212  
8267 025342 000000  
8268 025344 000020  
8269 025346 004737 010716  
8270 025352 000252  
8271 025354 000001  
8272 025356 004737 010716  
8273 025362 000000  
8274 025364 000001  
8275 025366 004737 010716  
8276 025372 001000  
8277 025374 000002  
8278 025376 004737 004726  
8279 025402 000060  
8280 025404 004737 006232  
8281 025410 000000  
8282 025412 004737 004726  
8283 025416 000010  
8284 025420 004737 006232  
8285 025424 000000  
8286 025426  
8287 025426  
      (3) 025426  
      (3) 025426 104403  
8288  
8289  
8290  
8291 025430 012701 002662  
8292 025434  
8293 025434  
      (3) 025434  
      (3) 025434 104402  
8294 025436 012737 025546 002346  
8295 025444 111137 025464  
8296 025450 111137 025474  
8297 025454 111137 025532  
8298 025460 004737 005212  
8299 025464 000000  
8300 025466 000020  
8301 025470 004737 010716  
8302 025474 000000  
8303 025476 000001  
8304 025500 004737 010716  
8305 025504 000000  
8306 025506 000001  
8307 025510 004737 010716  
8308 025514 001000  
8309 025516 000002  
8310 025520 004737 006752
```

```
-----  
: SEND MSG WITH INVALID SEC STA ADRS  
-----  
      BGNSUB  
T8.1: TRAP C$BSUB  
      MOV #3$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS  
      JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S  
      000 ;SEC ADRS = 000  
      SECA ;BIT MODE, CRC, SEC ADRS MODE  
      JSR PC,LODSIL ;LOAD 252 INTO TX SILO  
      252  
      1  
      JSR PC,LODSIL ;LOAD 000 DATA INTO TX SILO  
      000  
      1  
      JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO  
      TXEOM  
      2  
      JSR PC,STPLU ;TRANSMIT THE MSG  
      48.  
      JSR PC,IACTIV ;CHK IACT = 0  
      0  
      JSR PC,STPLU ;CLOCK 8 MORE CYCLES  
      8.  
      JSR PC,IACTIV ;CHK IACT = 0  
      0  
3$: ENDSUB  
L10032: TRAP C$ESUB  
-----  
: SEND MSG'S WITH VALID SEC ADRS'S FROM PAT T  
-----  
A11: MOV #PATT,R1 ;INIT DATA PATTERN POINTER  
      BGNSUB  
T8.2: TRAP C$BSUB  
5$: MOV #24$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS  
      MOV (R1),5$ ;SET SEC ADRS  
      MOV (R1),6$ ;SET FIRST DATA CHAR  
      MOV (R1),9$ ;SET EXPECTED DATA CHAR  
      JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S  
      .WORD 0  
      SECA ;BIT MODE, CRC, SEC ADRS MODE  
      JSR PC,LODSIL ;LOAD 1ST DATA CHAR INTO TX SILO  
6$: .WORD 0  
      1  
      JSR PC,LODSIL ;LOAD A 000 CHAR INTO TX SILO  
      000  
      1  
      JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO  
      TXEOM  
      2  
      JSR PC,RCV1ST ;CLOCK AND RCV FIRST DATA CHAR
```

```

8311 025524 000060          48.
8312 025526 004737 007266  JSR    PC,CKDATA      ;CHK FOR CORRECT RCV'D SEC STA ADRS
8313 025532 000000          9$:   .WORD    0
8314 025534 000011          9.
8315 025536 004737 007266  JSR    PC,CKDATA      ;READ AND CHK 000 CHAR, EBLK=1, BCC=0
8316 025542 101000          CRCCHK.RXEBL!000
8317 025544 000000          0
8318 025546          24$:
8319 025546          ENDSUB
(3) 025546          L10033:
(3) 025546 104403          TRAP    C$ESUB
8320 025550 005201          INC    R1              ;INCR PATTERN POINTER
8321 025552 020127 002667  CMP    R1,#ENDPAT     ;SEE IF ALL DONE YET
8322 025556 103726          BLO   A11             ;BR IF NO
8323 025560 004737 003276  JSR    PC,MSTCLR      ;ISSUE MASTER CLEAR
8324 025564          ENDTST
(3) 025564          L10031:
(3) 025564 104401          TRAP    C$ETST

```

```

8325
8326
8327
8328
8329
8330

```

```

:*****
:SBTTL      TEST 9 - RDALL (ALL PARTIES ADDRESS) BIT TEST
:*
:* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
:* BIT MODE, AND THE SECA BIT IS SET.
:* 2 FLAGS ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
:* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
:* RECEIVED.
:* NEXT, THE RDALL BIT IN REG 17 IS SET TO 1. 2 FLAGS
:* ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
:* THEN, THE REC'D DATA IS CHECKED TO MAKE SURE THAT 377
:* IS REC'D AS THE FIRST DATA CHAR, FOLLOWED BY 125.
:*****

```

```

8331
8332
8333
8334
8335
8336
8337
8338
8339
8340
8341
8342
8343 025566
(3) 025566
8344
8345
8346
8347 025566
(3) 025566
(3) 025566 104402
8348 025570 012737 025666 002346
8349 025576 004737 005212
8350 025602 000000
8351 025604 000020
8352 025606 004737 010716
8353 025612 000377
8354 025614 000001
8355 025616 004737 010716
8356 025622 000125
8357 025624 000001
8358 025626 004737 010716
8359 025632 001000

```

```

BGNTST
-----
: SET SEC ADR = 000, SEND ADR - 377, WITH RDALL - 0
-----
          BGNSUB
          T9.1:
          TRAP    C$BSUB
MOV    #3$,RETADR      ;SET SUBTEST EXIT ADRS FOR ERRORS
JSR    PC,INITRN      ;MST CLR, LOAD 2 SOM'S
000          ;SEC ADRS = 000
SECA          ;BIT MODE, CRC, SEC ADRS MODE
JSR    PC,LODSIL      ;LOAD 377 INTO TX SILO
377
1
JSR    PC,LODSIL      ;LOAD 125 DATA INTO TX SILO
125
1
JSR    PC,LODSIL      ;LOAD 2 EOM'S INTO TX SILO
TXEOM

```

```
8360 025634 000002          2
8361 025636 004737 004726   JSR    PC,STPLU      ;TRANSMIT THE MSG
8362 025642 000060          48.
8363 025644 004737 006232   JSR    PC,IACTIV     ;CHK IACT = 0
8364 025650 000000          0
8365 025652 004737 004726   JSR    PC,STPLU      ;CLOCK 8 MORE CYCLES
8366 025656 000010          8.
8367 025660 004737 006232   JSR    PC,IACTIV     ;CHK IACT = 0
8368 025664 000000          0
8369 025666
8370 025666          3$:
(3) 025666
(3) 025666 104403
8371
8372
8373
8374 025670          -----
(3) 02567C          : SET SEC ADR = 000, SEND ADR = 377, WITH RDALL - 1
(3) 025670 104402          -----
8375 025672 012737 025766 002346   MOV    #24$,RETADR   ;SET SUBTES, EXIT ADRS FOR ERRORS
8376 025700 004737 005212          JSR    PC,INITRN     ;MST CLR, LOAD 2 SOM'S
8377 025704 000000          000          ;SEC ADRS = 000
8378 025706 000024          SECA!RDALL          ;BIT MODE, CRC, SEC ADRS MODE, RDALL
8379 025710 004737 010716   JSR    PC,LODSIL     ;LOAD 1ST DATA CHAR INTO TX SILO
8380 025714 000377          377
8381 025716 000001          1
8382 025720 004737 010716   JSR    PC,LODSIL     ;LOAD A 125 CHAR INTO TX SILO
8383 025724 000125          125
8384 025726 000001          1
8385 025730 004737 010716   JSR    PC,LODSIL     ;LOAD 2 EOM'S INTO TX SILO
8386 025734 001000          TXEOM
8387 025736 000002          2
8388 025740 004737 006752   JSR    PC,RCV1ST     ;CLOCK AND RCV FIRST DATA CHAR
8389 025744 000060          48.
8390 025746 004737 007266   JSR    PC,CKDATA     ;CHK FOR 377 CHAR RCV'D
8391 025752 000377          377
8392 025754 000010          8.
8393 025756 004737 007266   JSR    PC,CKDATA     ;READ AND CHK 125 CHAR, EBLK=1, BCC=0
8394 025762 101125          CRCCHK!RXE!125
8395 025764 000000          0
8396 025766          24$:
8397 025766          ENDSUB
(3) 025766
(3) 025766 104403
8398 025770          ENDTST
(3) 025770
(3) 025770 104401
8399
8400
8401
8402
8403
8404
8405
8406
8407
```

L10035: TRAP C\$ESUB

T9.2: TRAP C\$BSUB

L10036: TRAP C\$ESUB

L10034: TRAP C\$ETST

:SBTTL TEST 10 - INSERT ERROR (IERR) BIT TEST - CHAR MODE, NO CRC
:*
:* THE LINE UNIT IS PLACED IN DDCMP MODE WITH NO ERROR DETECTION, AND 2

8408
8409
8410
8411
8412
8413
8414
8415
8416
8417
(3)
8418
8419
8420
8421
8422
8423
8424
8425
8426
8427
8428
8429
8430
8431
8432
8433
8434
8435
8436
8437
8438
8439
8440
8441
8442
8443
8444
8445
8446
8447
8448
8449
(3)
(3)
8450
8451
8452
8453
8454
8455
8456
8457
8458
8459
8460

025772
025772
025772 012737 026120 002346
026000 004737 005212
026004 000226
026006 000011
026010 004737 010476
026014 002722
026016 000004
026020 004737 004726
026024 100021
026026 004737 007164
026032 000011
026034 000001
026036 004737 004726
026042 000001
026044 004737 007164
026050 000011
026052 000002
026054 004737 004726
026060 000003
026062 004737 007164
026066 000011
026070 000004
026072 004737 006752
026076 000014
026100 004737 007266
026104 000032
026106 000010
026110 004737 007266
026114 000377
026116 000000
026120 004737 003276
026124
(3) 026124
(3) 026124 104401

:* SYNCHS, A 000 CHAR, A 377 CHAR, AND 2 SYNCHS ARE LOADED INTO THE
:* TRANSMITTER SILO. THEN, THE LU IS CLOCKED UNTIL THE 2ND BIT OF THE 000
:* CHAR IS ABOUT TO BE SENT AND THE IERR BIT IS SET FOR A CLOCK TIME AND
:* THEN CLEARED. IN THE SAME WAY, IERR IS SET PRIOR TO THE SENDING OF THE 4TH
:* AND 5TH BITS OF THE 000 CHAR. IT IS ALSO SET FOR THE SENDING OF THE FIRST
:* 4 BITS OF THE 377 CHAR. THE PROGRAM READS THE FIRST RCV'D CHAR FROM AX0
:* AND CHECKS IT TO BE 032, AND READS THE 2ND CHAR AND CHECKS IT TO BE 377.
:* THEN, A MASTER CLEAR IS DONE TO IDLE THE DEVICE.
:*****

BGNTST

T10::
MOV #15\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,INITRN ;LOAD 2 SOM'S, CLOCK THEM INTO USYRT
SYNCH
STRIP!DDCMP
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG2+4
4
JSR PC,STPLU ;CLOCK LU UNTIL 2ND BIT OF 000 CHAR
CHPCHK!17.
JSR PC,STPERR ;SET IERR 1 CYCLE
STRIP!DDCMP
1
JSR PC,STPLU ;CLOCK LU UNTIL 4TH BIT OF 000 CHAR
1
JSR PC,STPERR ;SET IERR FOR 2 CYCLES
STRIP!DDCMP
2
JSR PC,STPLU ;CLOCK LU UNTIL 1ST BIT OF 377 CHAR
3
JSR PC,STPERR ;SET IERR FOR 4 CYCLES
STRIP!DDCMP
4
JSR PC,RCV1ST ;CLOCK AND RCV 1ST CHAR
12.
JSR PC,CKDATA ;READ AND COMPARE 1ST CHAR TO 032
032
8.
JSR PC,CKDATA ;READ AND COMPARE 2ND CHAR TO 377
377
0
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP

15\$:
ENDTST

L10037:
TRAP C\$ETST

:*****
:SBTTL TEST 11 - SWITCH PACK PRINTOUT AND TEST
:*
:* - READ AND PRINT SWITCH PACK #1 :
:* THE PROGRAM READS REG 11 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
:* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO

8461
8462
8463
8464
8465
8466
8467
8468
8469
8470
8471
8472
8473
8474
8475
8476 026126
(3) 026126
8477
8478
8479
8480 026126
(3) 026126
(3) 026126 104402
8481 026130 004737 003276
8482 026134 012737 000011 002364
8483 026142 004737 003344
8484 026146 142737 000321 002350
8485 026154 023727 002400 000001
8486 026162 001403
8487 026164 005737 002260
8488 026170 001424
8489 026172
8490
8491 026172
(8) 026172 013746 002436
(7) 026176 012746 013042
(6) 026202 012746 000002
(3) 026206 010600
(4) 026210 104417
(4) 026212 062706 000006
8492
8493 026216
(8) 026216 013746 002350
(7) 026222 012746 012617
(6) 026226 012746 000002
(3) 026232 010600
(4) 026234 104417
(4) 026236 062706 000006
8494 026242 005737 002262
8495 026246 001420
8496 026250 123737 002350 002454
8497 026256 001414
8498 026260 013737 002454 002370
8499 026266 013737 002350 002372
8500 026274 004737 004200
8501

;* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
;* SWITCHES ARE IN BITS 1,2,3,5.
;*
;* - READ AND PRINT SWITCH PACK #2 :
;* THE PROGRAM READS REG 15 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
;* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
;* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
;* SWITCHES ARE IN BITS 0-7.
;*
;* - READ AND PRINT SWITCH PACK #3 :
;* THE PROGRAM READS REG 16 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
;* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
;* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
;* SWITCHES ARE IN BITS 0-7.
;*****

BGNTST
T11::

: READ AND PRINT SWITCH PACK #1. IF DESIRED

BGNSUB
T11.1: TRAP C\$BSUB
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #11,REGNUM ;SET LU REG NO. = 11
JSR PC,READLU ;READ LU REG 11
BICB #321,REDBYT ;MASK OFF NON-SWITCH BITS
CMP FRSPAS,#1 ;SEE IF IN FIRST PASS AFTER LOAD
BEQ 3\$;BR IF YES
TST PRNFLG ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
BEQ 4\$;BR IF NOT

3\$:
:PRINT DEVICE ADDRESS
PRINTF #FMT18,MPCSR
MOV MPCSR,-(SP)
MOV #FMT18,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTF
ADD #6,SP

:PRINT SWITCH PACK #1
PRINTF #FMT12,REDBYT
MOV REDBYT,-(SP)
MOV #FMT12,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTF
ADD #6,SP

4\$: TST SWIFLG ;SEE IF TEST IS ALLOWED
BEQ 6\$;BR IF NOT
CMPB REDBYT,LUSWI1 ;COMPARE SWITCHES TO EXPECTED
BEQ 6\$;BR IF MATCH
MOV LUSWI1,GOODAT ;SET EXPECTED DATA
MOV REDBYT,BADDAT ;SET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT

/:REPORT SWITCH PACK #1 INCORRECT

```
8502 026300 ERRDF 43,EM43,ERR2
(4) 026300 104455
(5) 026302 000053 TRAP C$ERDF
(5) 026304 014076 .WORD 43
(5) 026306 015366 .WORD EM43
8503 026310 6$: .WORD ERR2
8504 026310 ENDSUB
(3) 026310
(3) 026310 104403 L10041: TRAP C$ESUB
8505 -----
8506 : READ AND PRINT SWITCH PACK #2, IF DESIRED
8507 -----
8508 026312 BGNSUB
(3) 026312
(3) 026312 104402 T11.2: TRAP C$BSUB
8509 026314 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
8510 026320 012737 000015 002364 MOV #15,REGNUM ;SET LU REG NO. = 15
8511 026326 004737 003344 JSR PC,READLU ;READ LU REG 15
8512 026332 023727 002400 000001 CMP FRSPAS,#1 ;SEE IF IN FIRST PASS AFTER LOAD
8513 026340 001403 BEQ 3$ ;BR IF YES
8514 026342 005737 002260 TST PRNFLG ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
8515 026346 001412 BEQ 4$ ;BR IF NOT
8516 026350
8517 026350 3$: PRINTF #FMT13,REDBYT
(8) 026350 013746 002350 MOV REDBYT,-(SP)
(7) 026354 012746 012663 MOV #FMT13,-(SP)
(6) 026360 012746 000002 MOV #2,-(SP)
(3) 026364 010600 MOV SP,R0
(4) 026366 104417 TRAP C$PNTF
(4) 026370 062706 000006 ADD #6,SP
8518 026374 005737 002262 4$: TST SWIFLG ;SEE IF TEST IS ALLOWED
8519 026400 001420 BEQ 6$ ;BR IF NOT
8520 026402 123737 002350 002456 CMPB REDBYT,LUSWI2 ;COMPARE SWITCHES TO EXPECTED
8521 026410 001414 BEQ 6$ ;BR IF MATCH
8522 026412 013737 002456 002370 MOV LUSWI2,GOODAT ;SET EXPECTED DATA
8523 026420 013737 002350 002372 MOV REDBYT,BADDAT ;SET ACTUAL DATA
8524 026426 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
8525 ;REPORT SWITCH PACK #2 INCORRECT
8526 026432 ERRDF 44,EM44,ERR2
(4) 026432 104455 TRAP C$ERDF
(5) 026434 000054 .WORD 44
(5) 026436 014123 .WORD EM44
(5) 026440 015366 .WORD ERR2
8527 026442 6$: ENDSUB
8528 026442
(3) 026442
(3) 026442 104403 L10042: TRAP C$ESUB
8529 -----
8530 : READ AND PRINT SWITCH PACK #3, IF DESIRED
8531 -----
8532 026444 BGNSUB
(3) 026444
(3) 026444 104402 T11.3: TRAP C$BSUB
8533 026446 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
8534 026452 012737 000016 002364 MOV #16,REGNUM ;SET LU REG NO. = 16
8535 026460 004737 003344 JSR PC,READLU ;READ LU REG 16
```


8577 026642
(8) 026642 013746 002436
(7) 026646 012746 013042
(6) 026652 012746 000002
(3) 026656 010600
(4) 026660 104417
(4) 026662 062706 000006

PRINTF #FMT18,MPCSR

MOV MPCSR,-(SP)
MOV #FMT18,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTF
ADD #6,SP

8578
8579 026666
(8) 026666 013746 002354
(7) 026672 012746 012773
(6) 026676 012746 000002
(3) 026702 010600
(4) 026704 104417
(4) 026706 062706 000006

:PRINT AX3-15
PRINTF #FMT15,RAX15

MOV RAX15,-(SP)
MOV #FMT15,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTF
ADD #6,SP

8580 026712
8581 026712
(3) 026712
(3) 026712 104401

4\$:
ENDTST

L10044:
TRAP C\$ETST

8582
8583
8584
8585
8586

8587
8588
8589
8590
8591
8592
8593
8594
8595
8596
8597
8598
8599
8600
8601
8602
8603

:*****
:SBTTL TEST 13 - CRC GENERATION TEST
:*****
:* - CRC-16, CHAR MODE:
:* THE FOLLOWING MESSAGE IS SENT IN DDUMP MODE WITH CRC-16 SELECTED -
:* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU
:* TO CLOCK THE DATA. AT THE END OF THE MESSAGE THE
:* PROGRAM CHECKS FOR BCC - 1 (IN REG 12) INDICATING NO ERROR.
:*
:* - CRC-CCITT - 1'S PRESET:
:* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT
:* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.
:*
:* - CRC-CCITT - 0'S PRESET:
:* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT
:* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC - 0, INDICATING NO ERROR.
:*****

8604 026714
(3) 026714

BGNTST

T13::

8605
8606
8607
8608 026714 012737 027244 002346
8609 026722 004737 003276
8610 026726 004737 010304
8611 026732 000226
8612 026734 000011
8613 026736 000000
8614 026740 000000
8615 026742 004737 010476
8616 026746 002670
8617 026750 000011

: CRC 16, CHAR MODE

MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM THE USYRT
SYNCH
STRIP!DDCMP
000
000
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG1
9.

8618	026752	004737	004726	JSR	PC,STPLU	;CLOCK THE MSG
8619	026756	000136		94.		
8620	026760	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
8621	026764	000000		000		
8622	026766	000000		0		
8623	026770	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
8624	026774	000125		125		
8625	026776	000000		0		
8626	027000	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
8627	027004	000252		252		
8628	027006	000000		0		
8629	027010	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
8630	027014	000377		377		
8631	027016	000000		0		
8632	027020	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC - 1
8633	027024	100400		CRCCHK!400		
8634	027026	000000		0		

: CRC-CCITT-1'S PRESET, BIT MODE

8639	027030	004737	003276	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR
8640	027034	004737	010304	JSR	PC,SETUP	;PROGRAM THE USYRT
8641	027040	000000		000		
8642	027042	000000		000		
8643	027044	000000		000		
8644	027046	000000		000		
8645	027050	004737	010476	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
8646	027054	002670		MSG1		
8647	027056	000011		9.		
8648	027060	004737	004726	JSR	PC,STPLU	;CLOCK THE MSG
8649	027064	000146		102.		
8650	027066	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
8651	027072	000000		000		
8652	027074	000000		0		
8653	027076	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
8654	027102	000125		125		
8655	027104	000000		0		
8656	027106	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
8657	027112	000252		252		
8658	027114	000000		0		
8659	027116	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
8660	027122	000377		377		
8661	027124	000000		0		
8662	027126	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC 0
8663	027132	101000		CRCCHK.1000		
8664	027134	000000		0		

: CRC-CCITT-0'S PRESET, BIT MODE

8669						
8670	027136	004737	003276	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR
8671	027142	004737	010304	JSR	PC,SETUP	;PROGRAM THE USYRT
8672	027146	000000		000		
8673	027150	000100		CRC1		

8674	027152	000000		000		
8675	027154	000000		000		
8676	027156	004737	010476	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
8677	027162	002670		MSG1		
8678	027164	000011		9.		
8679	027166	004737	004726	JSR	PC,STPLU	;CLOCK THE MSG
8680	027172	000146		102.		
8681	027174	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
8682	027200	000000		000		
8683	027202	000000		0		
8684	027204	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
8685	027210	000125		125		
8686	027212	000000		0		
8687	027214	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
8688	027220	000252		252		
8689	027222	000000		0		
8690	027224	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
8691	027230	000377		377		
8692	027232	000000		0		
8693	027234	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 0
8694	027240	101000		CRCCHK!1000		
8695	027242	000000		0		
8696						
8697	027244	004737	003276	24\$: JSR	PC,MSTCLR	;ISSUE MASTER CLEAR TO CLEAN UP
8698	027250			ENDTST		
(3)	027250					L10045:
(3)	027250	104401				TRAP C\$ETST

8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726

```

:*****
:SBTTL      TEST 14 - CRC ERROR DETECTION TEST
:*
:* - CRC-16, CHAR MODE :
:* THE FOLLOWING MESSAGE IS SENT IN DDMP MODE, WITH CRC-16 SELECTED -
:* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU
:* TO CLOCK THE DATA. JUST BEFORE THE FIRST BIT OF THE LAST 000 CHAR IS SENT,
:* THE IERR BIT IS SET IN REG 17 TO CAUSE A 1 TO BE SENT, INTRODUCING A DATA
:* ERROR. AT THE END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 0, INDICATING
:* AN ERROR.
:*
:* - CRC-CCITT - 1'S PRESET :
:* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT THE
:* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
:*
:* - CRC-CCITT - 0'S PRESET :
:* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT THE
:* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
:*****
:BGNTST
:-----
: CRC 16, CHAR MODE
:-----

```

T14::

8727	027252	012737	027654	002346	MOV #24\$,RETADR	;SET TEST EXIT ADRS FOR ERRORS
8728	027260	004737	005212		JSR PC,INITRN	;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
8729	027264	000226			SYNCH	
8730	027266	000011			STRIP!DDCMP	
8731	027270	004737	005574		JSR PC,TXCHAR	;LOAD 000 CHAR, TX 1ST SYNCH
8732	027274	000000			000	
8733	027276	100010			CHPCHK!8.	
8734	027300	004737	010476		JSR PC,LODMSG	;LOAD MSG INTO TX SILO
8735	027304	002676			MSG1+6	
8736	027306	000006			6	
8737	027310	004737	004726		JSR PC,STPLU	;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
8738	027314	000010			8.	
8739	027316	004737	007164		JSR PC,STPERR	;MAKE 1ST BIT = 1 INSTEAD OF 0
8740	027322	000011			STRIP.DDCMP	
8741	027324	000001			1	
8742	027326	004737	004726		JSR PC,STPLU	;CLOCK REST OF MESSAGE
8743	027332	000122			82.	
8744	027334	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
8745	027340	000001			001	
8746	027342	000000			0	
8747	027344	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 125
8748	027350	000125			125	
8749	027352	000000			0	
8750	027354	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 252
8751	027360	000252			252	
8752	027362	000000			0	
8753	027364	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 377
8754	027370	000377			377	
8755	027372	000000			0	
8756	027374	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 0
8757	027400	100000			CRCCHK!000	
8758	027402	000000			0	
8759						

 : CRC-CCITT-1'S PRESET, BIT MODE

8763	027404	004737	005212		JSR PC,INITRN	;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
8764	027410	000000			000	
8765	027412	000000			000	
8766	027414	004737	005574		JSR PC,TXCHAR	;LOAD 000 CHAR, TX 1ST FLAG
8767	027420	000000			000	
8768	027422	100010			CHPCHK!8.	
8769	027424	004737	010476		JSR PC,LODMSG	;LOAD MSG INTO TX SILO
8770	027430	002676			MSG1+6	
8771	027432	000006			6	
8772	027434	004737	004726		JSR PC,STPLU	;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
8773	027440	000010			8.	
8774	027442	004737	007164		JSR PC,STPERR	;MAKE 1ST BIT = 1 INSTEAD OF 0
8775	027446	000000			000	
8776	027450	000001			1	
8777	027452	004737	004726		JSR PC,STPLU	;CLOCK REST OF MESSAGE
8778	027456	000122			82.	
8779	027460	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
8780	027464	000001			001	
8781	027466	000000			0	
8782	027470	004737	007266		JSR PC,CKDATA	;READ AND COMPARE CHAR TO 125

8783	027474	000125		125		
8784	027476	000000		0		
8785	027500	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
8786	027504	000252		252		
8787	027506	000000		0		
8788	027510	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
8789	027514	000377		377		
8790	027516	000000		0		
8791	027520	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 1
8792	027524	101400		CRCCHK:1400		
8793	027526	000000		0		
8794						
8795						
8796						
8797						

: CRC-CCITT-0'S PRESET, BIT MODE
:-----

8798	027530	004737	005212	JSR	PC,INITRN	;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
8799	027534	000000		000		
8800	027536	000100		CRC1		
8801	027540	004737	005574	JSR	PC,TXCHAR	;LOAD 000 CHAR, TX 1ST FLAG
8802	027544	000000		000		
8803	027546	100010		CHPCHK.8.		
8804	027550	004737	010476	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
8805	027554	002676		MSG1+6		
8806	027556	000006		6		
8807	027560	004737	004726	JSR	PC,STPLU	;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
8808	027564	000010		8.		
8809	027566	004737	007164	JSR	PC,STPERR	;MAKE 1ST BIT = 1 INSTEAD OF 0
8810	027572	000100		CRC1		
8811	027574	000001		1		
8812	027576	004737	004726	JSR	PC,STPLU	;CLOCK REST OF MESSAGE
8813	027602	000122		82.		
8814	027604	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
8815	027610	000001		001		
8816	027612	000000		0		
8817	027614	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
8818	027620	000125		125		
8819	027622	000000		0		
8820	027624	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
8821	027630	000252		252		
8822	027632	000000		0		
8823	027634	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
8824	027640	000377		377		
8825	027642	000000		0		
8826	027644	004737	007266	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC - 1
8827	027650	101400		CRCCHK.1400		
8828	027652	000000		0		

8829						
8830	027654	004737	003276	24\$: JSR	PC,MSTCLR	;ISSUE MASTER CLEAR TO CLEAN UP
8831	027660			ENDTST		
(3)	027660					
(3)	027660	104401				L10046: TRAP CSETST
8832						
8833						
8834						
8835						
8836						

```

8837
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847
8848
8849
8850
8851
8852
8853
8854
8855
8856 027662
(3) 027662
8857
8858
8859
8860 027662 012737 000006 002366
8861 027670 012737 000017 002364
8862 027676
(3) 027676
(3) 027676 104402
8863 027700 012737 030102 002346
8864 027706 004737 005212
8865 027712 000026
8866 027714 000111
8867 027716 004737 010554
8868 027722 002613
8869 027724 000010
8870 027726 004737 010716
8871 027732 001000
8872 027734 000002
8873 027736 005037 002360
8874 027742 012737 000347 002362
8875 027750 004737 003764
8876 027754 004737 004726
8877 027760 000010
8878 027762 004737 004726
8879 027766 000010
8880 027770 005001
8881 027772 004737 004726 2$:
8882 027776 000010
8883 030000 004737 003344
8884 030004 005201
8885 030006 020127 000004
8886 030012 003014
8887 030014 132737 000040 002350
8888 030022 001024
8889 030024 004737 004200

```

```

:*****
:SBTTL TEST 15 - VRC PARITY GENERATION TEST
:
:* SUBTEST 1 - TEST OF CORRECT ODD VRC PARITY GENERATION :
:* THE LINE UNIT IS PLACED IN CHAR MODE, WITH ODD VRC AND 7-BIT CHARS SELECTED.
:* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)
:* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.
:* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 1 AND FOR THE
:* LAST 4 CHARS IT SHOULD = 0.
:*
:* SUBTEST 2 - TEST OF CORRECT EVEN VRC PARITY GENERATION :
:* THE LINE UNIT IS PLACED IN CHAR MODE, WITH EVEN VRC AND 7-BIT CHARS SELECTED.
:* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)
:* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.
:* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 0 AND FOR THE
:* LAST 4 CHARS IT SHOULD = 1.
:*
:* DATA PATTERN Q - 000,120,125,137,040,052,057,177
:*****
BGNTST

```

T15::

: TEST ODD VRC GENERATION

```

MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3
MOV #17,REGNUM ;SET REG NO. = 17
BGNSUB
T15.1:
TRAP CSBSUB
MOV #8$,RETADR ;SET SUBTEST EXIT ADDRESS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
026
CRC1.STRIP.DDCMP ;CHAR MODE, ODD VRC
JSR PC,LDBYTS ;LOAD DATA INTO TX SILO
PATQ
8.
JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO
TXEOM
2
CLR WAX15
MOV #TXLEN2!TXLEN1.TXLENO.RXLEN2!RXLEN1!RXLENO,WAX16
JSR PC,WRITAX ;SET TX AND RCV LENGTHS = 7
JSR PC,STPLU ;CLOCK FIRST SYNCH
8.
JSR PC,STPLU ;CLOCK 2ND SYNCH
8.
CLR R1 ;INIT CHAR COUNT
2$: JSR PC,STPLU ;CLOCK A CHAR
8.
JSR PC,READLU ;READ REG 17
INC R1 ;INCR CHAR COUNT
CMP R1,#4 ;SEE IF 4 CHARS CLKD YET
BGT 4$ ;BR IF YES
BITB #TXDATA,REDBYT ;SEE IF PARITY BIT IS SET
BNE 6$ ;BR IF YES
JSR PC,GETALL ;GET REGS FOR PRINTOUT

```

```
8890 ;REPORT ODD VRC PARITY BIT NOT SET
8891 030030 ERRDF 48,EM48,ERR7
(4) 030030 104455 TRAP C$ERDF
(5) 030032 000060 .WORD 48
(5) 030034 014263 .WORD EM48
(5) 030036 020224 .WORD ERR7
8892 030040 ESCAPE SUB
(3) 030040 104410 TRAP C$ESCAPE
(3) 030042 000040 .WORD L10050-.
8893 030044 132737 000040 002350 4$: BITB #TXDATA,REDBYT ;SEE IF PARITY BIT IS CLEARED
8894 030052 001410 BEQ 6$ ;BR IF YES
8895 030054 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
8896 ;REPORT ODD VRC PARITY BIT NOT CLEARED
8897 030060 ERRDF 49,EM49,ERR7
(4) 030060 104455 TRAP C$ERDF
(5) 030062 000061 .WORD 49
(5) 030064 014316 .WORD EM49
(5) 030066 020224 .WORD ERR7
8898 030070 ESCAPE SUB
(3) 030070 104410 TRAP C$ESCAPE
(3) 030072 000010 .WORD L10050-.
8899 030074 020127 000010 6$: CMP R1,#8. ;SEE IF ALL CHARS TESTED YET
8900 030100 002734 BLT 2$ ;BR IF NOT
8901 030102 8$:
8902 030102 ENDSUB
(3) 030102 L10050:
(3) 030102 104403 TRAP C$ESUB
8903 -----
8904 : TEST EVEN VRC GENERATION
8905 -----
8906 030104 012737 000006 002366 MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3
8907 030112 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
8908 030120 BGNSUB
(3) 030120 T15.2:
(3) 030120 104402 TRAP C$BSUB
8909 030122 012737 030324 002346 MOV #18$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS
8910 030130 004737 005212 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
8911 030134 000026 026
8912 030136 000211 CRC2!STRIP!DDCMP ;CHAR MODE, EVEN VRC
8913 030140 004737 010554 JSR PC,LDBYTS ;LOAD DATA INTO TX SILO
8914 030144 002613 PATQ
8915 030146 000010 8.
8916 030150 004737 010716 JSR PC,LDSIL ;LOAD 2 EOM'S INTO TX SILO
8917 030154 001000 TXEQM
8918 030156 000002 2
8919 030160 005037 002360 CLR WAX15
8920 030164 012737 000347 002362 MOV #TXLEN2!TXLEN1!TXLENO.RXLEN2.RXLEN1!RXLENO,WAX16
8921 030172 004737 003764 JSR PC,WRITAX ;SET TX AND RCV LENGTHS = ?
8922 030176 004737 004726 JSR PC,STPLU ;CLOCK FIRST SYNCH
8923 030202 000010 8.
8924 030204 004737 004726 JSR PC,STPLU ;CLOCK 2ND SYNCH
8925 030210 000010 8.
8926 030212 005001 CLR R1 ;INIT CHAR COUNT
8927 030214 004737 004726 12$: JSR PC,STPLU ;CLOCK A CHAR
8928 030220 000010 8.
8929 030222 004737 003344 JSR PC,READLU ;READ REG 17
```

8930	030226	005201			INC	R1		:INCR CHAR COUNT		
8931	030230	020127	000004		CMP	R1,#4		:SEE IF 4 CHARS CLKD YET		
8932	030234	003014			BGT	14\$:BR IF YES		
8933	030236	132737	000040	002350	BITB	#TXDATA,REDBYT		:SEE IF PARITY BIT IS CLEARED		
8934	030244	001424			BEQ	16\$:BR IF YES		
8935	030246	004737	004200		JSR	PC,GETALL		:GET REGS FOR PRINTOUT		
8936					:REPORT	EVEN VRC PARITY BIT NOT		CLEARED		
8937	030252				ERRDF	51,EM51,ERR7				
(4)	030252	104455							TRAP	C\$ERDF
(5)	030254	000063							.WORD	51
(5)	030256	014411							.WORD	EM51
(5)	030260	020224							.WORD	ERR7
8938	030262				ESCAPE	SUB				
(3)	030262	104410							TRAP	C\$ESCAPE
(3)	030264	000040							.WORD	L10051-
8939	030266	132737	000040	002350	14\$:	BITB	#TXDATA,REDBYT	:SEE IF PARITY BIT IS SET		
8940	030274	001010			BNE	16\$:BR IF YES		
8941	030276	004737	004200		JSR	PC,GETALL		:GET REGS FOR PRINTOUT		
8942					:REPORT	EVEN VRC PARITY BIT NOT		SET		
8943	030302				ERRDF	50,EM50,ERR7				
(4)	030302	104455							TRAP	C\$ERDF
(5)	030304	000062							.WORD	50
(5)	030306	014355							.WORD	EM50
(5)	030310	020224							.WORD	ERR7
8944	030312				ESCAPE	SUB				
(3)	030312	104410							TRAP	C\$ESCAPE
(3)	030314	000010							.WORD	L10051-
8945	030316	020127	000010		16\$:	CMP	R1,#8.	:SEE IF ALL CHARS TESTED YET		
8946	030322	002734			BLT	12\$:BR IF NOT		
8947	030324				18\$:					
8948	030324				ENDSUB					
(3)	030324									L10051:
(3)	030324	104403							TRAP	C\$ESUB
8949	030326	004737	003276		JSR	PC,MSTCLR		:ISSUE MASTER CLEAR TO CLEAN UP		
8950	030332				ENDTST					
(3)	030332									L10047:
(3)	030332	104401							TRAP	C\$ETST

8951
8952
8953
8954
8955
8956
8957
8958
8959
8960
8961
8962
8963
8964
8965
8966
8967
8968
8969

```
*****  
:SBTTL TEST 16 - VRC ERROR DETECTION TEST  
:  
:* SUBTEST 1 - FORCING OF BCC USING ODD VRC  
:* THE LINE UNIT IS PLACED IN CHAR MODE WITH ODD VRC AND 7-BIT CHARS SELECTED.  
:* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER  
:* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM  
:* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING  
:* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE  
:* RECEIVED (INDICATING AN ERROR).  
:  
:* SUBTEST 2 - FORCING OF BCC USING EVEN VRC  
:* THE LINE UNIT IS PLACED IN CHAR MODE WITH EVEN VRC AND 7-BIT CHARS SELECTED.  
:* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
```

8970
8971
8972
8973
8974
8975
8976
8977
8978 030334
(3) 030334
8979
8980
8981
8982 030334 012737 000006 002366
8983 030342 012737 000012 002364
8984 030350
(3) 030350
(3) 030350 104402
8985 030352 012737 030570 002346
8986 030360 004737 005212
8987 030364 000026
8988 030366 000111
8989 030370 004737 010716
8990 030374 000400
8991 030376 000001
8992 030400 004737 010554
8993 030404 002623
8994 030406 000017
8995 030410 004737 010716
8996 030414 001000
8997 030416 000002
8998 030420 005037 002360
8999 030424 012737 000347 002362
9000 030432 013737 002362 002424
9001 030440 004737 003764
9002 030444 004737 004726
9003 030450 000130
9004 030452 012701 000007
9005 030456 004737 007164
9006 030462 000111
9007 030464 000001
9008 030466 004737 004726
9009 030472 000007
9010 030474 005301
9011 030476 001367
9012 030500 004737 004726
9013 030504 000020
9014 030506 012701 000010
9015 030512 012703 002623
9016 030516 112337 030526
9017 030522 004737 007266
9018 030526 100000
9019 030530 000000
9020 030532 005301
9021 030534 001370
9022 030536 012701 000007

: * 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
: * CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
: * NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
: * RECEIVED (INDICATING AN ERROR).

: * DATA PATTERN R = 000,100,120,124,164,172,176,177,000,100,120,124,164,
: * 172,176.

: *****
: BGNTST

T16::

:-----
: TEST ODD VRC ERROR DETECTION
:-----

```
MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3
MOV #12,REGNUM ;SET REG NO.
BGNSUB

T16.1: TRAP CSBSUB
MOV #10$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
026
CRC1!STRIP!DDCMP ;CHAR MODE, ODD VRC
JSR PC,LODSIL ;LOAD A THIRD SOM INTO TX SILO
TXSOM
1
JSR PC,LDBYTS ;LOAD DATA INTO TX BUFFER
PATR
15.
JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO
TXEQM
2
CLR WAX15
MOV #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,wAX16 ;STORE LENGTH 7
MOV WAX16,SAVLEN ;SET TX AND RCV LENGTHS = 7
JSR PC,WRITAX ;CLOCK 1ST 8 CHARS, WITH NO ERRORS
JSR PC,STPLU
88.
MOV #7,R1 ;INIT COUNTER FOR LAST 7 CHARS
3$: JSR PC,STPERR ;ASSERT IERR BIT FOR 1 TIME
CRC1!STRIP!DDCMP
1
JSR PC,STPLU ;CLOCK REST OF CHAR
7
DEC R1 ;DECR COUNTER
BNE 3$ ;BR IF NOT DONE TRANSMITTING YET
JSR PC,STPLU ;CLOCK 2 TERMINATING SYNCHS
16.
MOV #8.,R1 ;INIT COUNTER FOR ERROR-FREE CHARS
MOV #PATR,R3 ;INIT DATA PATTERN POINTER
5$: MOVB (R3)+,6$ ;GFT AN EXPECTED DATA CHAR
JSR PC,CKDATA ;GO CHECK CHAR, CHK BCC=0
6$: BCCCHK!000
0
DEC R1 ;DECR COUNTER
BNE 5$ ;BR IF NOT DONE YET
MOV #7,R1 ;INIT COUNTER FOR ERROR CHARS
```

```

9023 030542 112337 030560      8$:  MOVB   (R3)+,9$      ;GET EXPECTED DATA CHAR
9024 030546 052737 000001 030560  BIS    #BIT0,9$      ;EXPECT ERROR BIT 0 SET
9025 030554 004737 007266      JSR    PC,CKDATA     ;CHECK DATA, CHK BCC=1
9026 030560 100400      9$:  BCCCHK!RXBCC!000
9027 030562 000000      0
9028 030564 005301      DEC    R1             ;DECR COUNTER
9029 030566 001365      BNE    8$             ;BR IF NOT DONE YET
9030 030570      10$:  ENDSUB
9031 030570
(3) 030570
(3) 030570 104403                                     L10053:
9032                                     TRAP      C$ESUB
9033
-----
: TEST EVEN VRC ERROR DETECTION
-----
9034
9035 030572 012737 000006 002366  MOV    #6,AXNUM      ;SET AX BYTE NO. FOR AX3
9036 030600 012737 000012 002366  MOV    #12,REGNUM   ;SET REG NO.
9037 030606  BGNSUB
(3) 030606
(3) 030606 104402                                     T16.2:
9038 030610 012737 031026 002346  MOV    #30$,RETADR  ;SET SUBTEST EXIT ADRS FOR ERRORS
9039 030616 004737 005212      JSR    PC,INITRN    ;MST CLR, LOAD 2 SOM'S
9040 030622 000026      026
9041 030624 000211      CRC2!STRIP!DDCMP   ;CHAR MODE, EVEN VRC
9042 030626 004737 010716      JSR    PC,LODSIL   ;LOAD A THIRD SOM INTO TX SILO
9043 030632 000400      TXSOM
9044 030634 000001      1
9045 030636 004737 010554      JSR    PC,LDBYTS   ;LOAD DATA INTO TX BUFFER
9046 030642 002623      PATR
9047 030644 000017      15.
9048 030646 004737 010716      JSR    PC,LODSIL   ;LOAD 2 EOM'S INTO TX SILO
9049 030652 001000      TXEOM
9050 030654 003002      2
9051 030656 005037 002360      CLR    WAX15
9052 030662 012737 000347 002362  MOV    #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,WAX16
9053 030670 013737 002362 002424  MOV    WAX16,SAVLEN ;STORE LENGTH 7
9054 030676 004737 003764      JSR    PC,WRITAX   ;SET TX AND RCV LENGTHS = 7
9055 030702 004737 004726      JSR    PC,STPLU    ;CLOCK 1ST 8 CHARS, WITH NO ERRORS
9056 030706 000130      88.
9057 030710 012701 000007      MOV    #7,R1       ;INIT COUNTER FOR LAST 7 CHARS
9058 030714 004737 007164      23$: JSR    PC,STPERR   ;ASSERT IERR BIT FOR 1 TIME
9059 030720 000211      CRC2!STRIP.DDCMP
9060 030722 000001      1
9061 030724 004737 004726      JSR    PC,STPIU    ;CLOCK REST OF CHAR
9062 030730 000007      7
9063 030732 005301      DEC    R1           ;DECR COUNTER
9064 030734 001367      BNE    23$         ;BR IF NOT DONE TRANSMITTING YET
9065 030736 004737 004726      JSR    PC,STPLU    ;CLOCK 2 TFRMINATING SYNCHS
9066 030742 000020      16.
9067 030744 012701 000010      MOV    #8,R1       ;INIT COUNTER FOR ERROR-FREE CHARS
9068 030750 012703 002623      MOV    #PATR,P3    ;INIT DATA PATTERN POINTER
9069 030754 112337 030764      25$: MOVB   (R3)+,26$   ;GET EXPECTED DATA CHAR
9070 030760 004737 007266      JSR    PC,CKDATA   ;CHK DATA, CHECK BCC=0
9071 030764 100000      26$: BCCCHK!000
9072 030766 000000      0
9073 030770 005301      DEC    R1           ;DECR COUNTER
9074 030772 001370      BNE    25$         ;BR IF NOT DONE YET

```

```

9075 030774 012701 000007      MOV      #7,R1          ;INIT COUNTER FOR ERROR CHARS
9076 031000 112337 031016      MOVB     (R3)+,29$     ;GET EXPECTED DATA CHAR
9077 031004 052737 000001      BIS      #BIT0,29$    ;SET EXPECTED ERROR BIT 0
9078 031012 004737 007266      JSR      PC,CKDATA    ;CHK DATA, CHK BCC=1
9079 031016 100400      28$:    BCCCHK!RXBCC!000    ;
9080 031020 000000      29$:    0
9081 031022 005301      DEC      R1           ;DECR COUNTER
9082 031024 001365      BNE      28$         ;BR IF NOT DONE YET
9083 031026      30$:
9084 031026      ENDSUB
(3) 031026
(3) 031026 104403      L10054:
9085 031030 004737 003276      JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR TO CLEAN UP TRAP C$ESUB
9086 031034      ENDTST
(3) 031034      L10052:
(3) 031034 104401      TRAP    C$ETST
9087
9088
9089
9090
9091
9092
9093
9094
9095
9096
9097
9098
9099
9100
9101
9102
9103
9104
9105
9106
9107
9108
9109
9110
9111
9112
9113
9114
9115
9116
9117
9118
9119
9120
9121
9122
9123
9124
9125

```

```

:*****
:SBTTL      TEST 17 - INTEGRAL MODEM INTERFACE TEST - CHAR MODE, CRC
:*
:* THE INTEGRAL MODEM IS SELECTED BY THE PROGRAM IN AX3-15, AND A
:* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
:* ON THE LINE UNIT OR AT THE CABLE. THE MESSAGE CONSISTS OF
:* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
:* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
:* SKIPPED FOR THAT UNIT.
:*****
BGNTST

```

```

9102 031036      T17::
(3) 031036
9103 031036 012737 000021 002434      MOV      #17,,TSTNUM ;SET TEST NO.
9104 031044 012737 031316 002346      MOV      #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
9105 031052 004737 003276      JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR
9106 031056 004737 010766      JSR      PC,CKLPBK   ;CHECK LOOPBACK -
9107 031062 000010      INTGRL              ;SEE IF TEST SHOULD BE RUN
9108 031064 012737 000323 031102      MOV      #1422!XYZ!V35!OP.TEST,6$ ;SET UP TO SELECT INTEGRAL MODEM
9109 031072 004737 010304      JSR      PC,SETUP    ;PROGRAM THE USYRT
9110 031076 000226      SYNCH
9111 031100 000011      STRIP!DDCMP
9112 031102 000000      6$:    .WORD    0
9113 031104 000000      000
9114 031106 004737 010636      JSR      PC,LDMSG1   ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
9115 031112 142777 000010 151320      BICB     #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
9116 031120 012737 000012 002364      MOV      #12,REGNUM  ;SET LU REG NO. = 12
9117 031126 012703 002762      MOV      #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9118 031132 013702 002264      9$:    MOV      TCOUNT,R2 ;INIT TIMER
9119 031136 004737 003344      10$:   JSR      PC,READLU   ;READ REG 12
9120 031142 132737 000020 002350      BITB     #IRDY,REDBYT ;SEE IF IRDY IS SET YET
9121 031150 001011      BNE      12$        ;BR IF YES
9122 031152 005202      INC      R2         ;INCREMENT TIMER
9123 031154 001370      BNE      10$        ;BR IF NO TIME-OUT YET
9124 031156 004737 004200      JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
9125      ;REPORT IRDY NOT SET

```

```
9126 031162 ERRDF 17,FM17,ERR7
(4) 031162 104455
(5) 031164 000021 TRAP C$ERDF
(5) 031166 013436 .WORD 17
(5) 031170 020224 .WORD EM17
9127 031172 000451 .WORD ERR7
9128 031174 012337 031204 12$: BR 24$ ;ESCAPE TO END OF TEST
9129 031200 004737 007266 12$: MOV (R3)+,16$
9130 031204 000000 16$: JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
9131 031206 000000 0
9132 031210 020327 003000 16$: CMP R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
9133 031214 103746 BLO 9$ ;BR IF NOT YET
9134 031216 004737 004620 JSR PC,WAIT50 ;STALL FOR 50 MICRO-SEC
9135 031222 004737 005024 JSR PC,OACTIV ;CHECK OACT = 0
9136 031226 000000 0
9137 031230 004737 006232 JSR PC,IACTIV ;CHECK IACT = 0
9138 031234 000000 0
9139 031236 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. 13
9140 031244 004737 003344 JSR PC,READLU ;READ REG 13
9141 031250 042737 000232 002350 BIC #RING!HDX!MODR!STBY,REDBYT ;CLR UNUSED BITS
9142 031256 023727 002350 000000 CMP REDBYT,#0 ;CHECK REG 13 FOR 000 (MODEM SIGNALS SHOULD BE CLEARED)
9143 031264 001414 BEQ 24$ ;BR IF CLEARED
9144 031266 012737 000000 002370 MOV #0,GOODAT ;SET EXPECTED DATA - 0
9145 031274 013737 002350 002372 MOV REDBYT,BADDAT ;SET ACTUAL DATA
9146 031302 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9147 ;REPORT REG MISCOMPARE
9148 031306 ERRDF 3,EM3,ERR3
(4) 031306 104455 TRAP C$ERDF
(5) 031310 000003 .WORD 3
(5) 031312 013265 .WORD EM3
(5) 031314 015674 .WORD ERR3
9149 031316 24$:
9150 031316 ENDTST
(3) 031316 L10055:
(3) 031316 104401 TRAP C$ETST
9151
9152
9153
9154
9155
9156
9157 ;*****
9158 .SBTTL TEST 18 - V.35 MODEM INTERFACE TEST - CHAR MODE, CRC
9159 ;*
9160 ;* THE V.35 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A
9161 ;* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
9162 ;* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
9163 ;* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF
9164 ;* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
9165 ;* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
9166 ;* SKIPPED FOR THAT UNIT.
9167 ;*****
9167 031320 BGINST
(3) 031320 T18::
9168 031320 012737 000022 002434 MOV #18,,TSTNUM ;SET TEST NO.
9169 031326 012737 031570 002346 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
9170 031334 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
```



```
9171 031340 004737 010766 JSR PC,CKLPBK ;CHECK LOOPBACK -
9172 031344 000020 V35 ; SEE IF TEST SHOULD BE RUN
9173 031346 012737 000313 031364 MOV #I422!XYZ!INTGRL!OP!TEST,6$ ;SET UP TO SELECT V35
9174 031354 004737 010304 JSR PC,SETUP ;PROGRAM THE USYRT
9175 031360 000226 SYNCH
9176 031362 000011 STRIP.DDCMP
9177 031364 000000 6$: .WORD 0
9178 031366 000000 000
9179 031370 142777 000010 151042 BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
9180 031376 012737 000013 002364 MOV #13,REGNUM ;SET LU REG NO. - 13
9181 031404 004737 003344 JSR PC,READLU ;READ REG 13
9182 031410 132737 000001 002350 BITB #CARR,REDBYT ;CHECK FOR CARRIER FALSELY SET
9183 031416 001415 BEQ 8$ ;BR IF NOT SET
9184 031420 012737 000000 002370 MOV #000,GOODAT ;SET EXPECTED DATA
9185 031426 013737 002350 002372 MOV REDBYT,BAD^AT ;SET ACTUAL DATA
9186 031434 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9187 ;REPORT CARRIER NOT CLEARED
9188 031440 ERRDF 66,EM66,ERR7
(4) 031440 104455 TRAP C$ERDF
(5) 031442 000102 .WORD 66
(5) 031444 015000 .WORD EM66
(5) 031446 020224 .WORD ERR7
9189 031450 000447 BR 24$
9190 031452 152777 000010 150760 8$: BISB #LULOOP,@BSEL1 ;SET LULOOP AGAIN
9191 031460 004737 010636 JSR PC,LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
9192 031464 142777 000010 150746 BICB #LULOOP,@BSEL1 ;CLEAR LULCOOP, CLOCK MSG
9193 031472 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
9194 031500 012703 002762 MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9195 031504 013702 002264 9$: MOV TCOUNT,R2 ;INIT TIMFR
9196 031510 004737 003344 10$: JSR PC,READLU ;READ REG 12
9197 031514 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
9198 031522 001011 BNE 12$ ;BR IF YES
9199 031524 005202 INC R2 ;INCREMENT TIMER
9200 031526 001370 BNE 10$ ;BR IF NO TIME-OUT YET
9201 031530 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9202 ;REPORT IRDY NOT SET
9203 031534 ERRDF 17,EM17,ERR7
(4) 031534 104455 TRAP C$ERDF
(5) 031536 000021 .WORD 17
(5) 031540 013436 .WORD EM17
(5) 031542 020224 .WORD ERR7
9204 031544 000411 BR 24$ ;ESCAPE TO END OF TEST
9205 031546 012337 031556 12$: MOV (R3)+,16$
9206 031552 004737 007266 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
9207 031556 000000 16$: 0
9208 031560 000000 0
9209 031562 020327 003000 CMP R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
9210 031566 103746 BLO 9$ ;BR IF NOT YET
9211 031570 24$:
9212 031570 ENDTST
(3) 031570
(3) 031570 104401 L10056: TRAP C$ETST
9213
9214
9215
9216
```

9217
9218
9219
9220
9221
9222
9223
9224
9225
9226
9227
9228
9229
9230
9231
9232
9233
9234
9235
9236
9237
9238
9239
9240
9241
9242
9243
9244
9245
9246
9247
9248
9249
9250
9251
9252
9253
9254
9255
9256
9257
9258
9259
9260
9261
9262
9263
9264
9265

```
*****  
:SBTTL TEST 19 - RS 232C AND RS 423 MODEM INTERFACE TEST - CHAR MODE, CRC  
:  
:* THE RS232C & RS423 (XYZ) MODEM INTERFACE IS SELECTED BY THE PROGRAM IN  
:* AX3-15, AND A MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURN-  
:* AROUND CONNECTOR ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,  
:* OR A MODEM TEST MODE. THE MESSAGE CONSISTS  
:* OF 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE  
:* P-TABLE FOR THE CURRENT UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS  
:* PROVIDED, THE TEST WILL BE SKIPPED FOR THAT UNIT.  
:*****
```

```
BGNTST  
T19::  
MOV #19, TSTNUM ;SET TEST NO.  
MOV #24$, RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC, MSTCLR ;ISSUE MASTER CLEAR  
JSR PC, CKLPBK ;CHECK LOOPBACK -  
XYZ ; SEE IF TEST SHOULD BE RUN  
MOV #1422!V35. INTGRL.OP!TEST, 6$ ;SET UP TO SELECT XYZ  
JSR PC, SETUP ;PROGRAM THE USYRT  
SYNCH  
STRIP!DDCMP  
6$: .WORD 0  
000  
JSR PC, LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF  
BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG  
MOV #12, REGNUM ;SET LU REG NO. - 12  
MOV #RCVBUF, R3 ;GET POINTER TO RCV MSG BUF  
9$: MOV TCOUNT, R2 ;INIT TIMER  
10$: JSR PC, READLU ;READ REG 12  
BITB #IRDY, REDBYT ;SEE IF IRDY IS SET YET  
BNE 12$ ;BR IF YES  
INC R2 ;INCREMENT TIMER  
BNE 10$ ;BR IF NO TIME-OUT YET  
JSR PC, GETALL ;GET REGS FOR PRINTOUT  
:REPORT IRDY NOT SET  
ERRDF 17, EM17, ERR7  
TRAP C$ERDF  
.WORD 17  
.WORD EM17  
.WORD ERR7  
BR 24$ ;ESCAPE TO END OF TEST  
12$: MOV (R3)+, 16$  
JSR PC, CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED  
16$: 0  
0  
CMP R3, #RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET  
BLO 9$ ;BR IF NOT YET  
24$:  
ENDTST  
L10057:  
TRAP C$E'TS'
```

9266
9267
9268
9269
9270
9271
9272
9273
9274
9275
9276
9277
9278
9279 031754
(3) 031754
9280 031754 012737 000024 002434
9281 031762 012737 032134 002346
9282 031770 004737 003276
9283 031774 004737 010766
9284 032000 000200
9285 032002 012737 000133 032020
9286 032010 004737 010304
9287 032014 000226
9288 032016 000011
9289 032020 000000
9290 032022 000000
9291 032024 004737 010636
9292 032030 142777 000010 150402
9293 032036 012737 000012 002364
9294 032044 012703 002762
9295 032050 013702 002264
9296 032054 004737 003344
9297 032060 132737 000020 002350
9298 032066 001011
9299 032070 005202
9300 032072 001370
9301 032074 004737 004200
9302
9303 032100
(4) 032100 104455
(5) 032102 000021
(5) 032104 013436
(5) 032106 020224
9304 032110 000411
9305 032112 012337 032122
9306 032116 004737 007266
9307 032122 000000
9308 032124 000000
9309 032126 020327 003000
9310 032132 103746
9311 032134
9312 032134
(3) 032134
(3) 032134 104401
9313
9314

```
*****  
:SBTTL TEST 20 - RS 422 MODEM INTERFACE TEST - CHAR MODE, CRC  
:  
:* THE RS 422 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A  
:* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR  
:* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,  
:* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF  
:* 2 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT  
:* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE  
:* SKIPPED FOR THAT UNIT.  
:*****
```

```
BGNTST  
T20::  
MOV #20, TSTNUM ;SET TEST NO.  
MOV #24$, RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC, MSTCLR ;ISSUE MASTER CLEAR  
JSR PC, CKLPBK ;CHECK LOOPBACK -  
I422 ; SEE IF TEST SHOULD BE RUN  
MOV #XYZ.V35.INTGRL!OP!TEST, 6$ ;SET UP TO SELECT 422  
JSR PC, SETUP ;PROGRAM THE USYRT  
SYNCH  
STRIP!DDCMP  
6$: .WORD 0  
000  
JSR PC, LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF  
BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG  
MOV #12, REGNUM ;SET LU REG NO. = 12  
MOV #RCVBUF, R3 ;GET POINTER TO RCV MSG BUF  
9$: MOV TCOUNT, R2 ;INIT TIMER  
10$: JSR PC, READLU ;READ REG 12  
BITB #IRDY, REDBYT ;SEE IF IRDY IS SET YET  
BNE 12$ ;BR IF YES  
INC R2 ;INCREMENT TIMER  
BNE 10$ ;BR IF NO TIME-OUT YET  
JSR PC, GETALL ;GET REGS FOR PRINTOUT  
;REPORT IRDY NOT SET  
ERRDF 17, EM17, ERR7  
TRAP C$ERDF  
.WORD 17  
.WORD EM17  
.WORD ERR7  
BR 24$ ;ESCAPE TO END OF TEST  
12$: MOV (R3)+, 16$  
JSR PC, CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED  
16$: 0  
0  
CMP R3, #RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET  
BLO 9$ ;BR IF NOT YET  
24$:  
ENDTST  
L10060:  
TRAP C$ETST
```

9315
9316
9317
9318
9319
9320
9321
9322
9323
9324
9325
9326
9327
9328
9329

```
*****
:SBITL      TEST 21 - HALF-DUPLEX BIT (HALF DUPX) TEST
:
:* THIS TEST VERIFIES THAT SETTING HALF-DUPLEX BIT IN REG 13 INHIBITS
:* LOADING OF THE USYRT TRANSMITTE FROM THE TRANSMITTER SILO.
:* A MASTER CLEAR IS ISSUED, DDCMP MODE IS ENTERED, AND THE HALF DUPX
:* BIT IN REG 13 IS SET. A MESSAGE IS LOADED INTO THE TX SILO
:* CONSISTING OF 2 SYNCHS, 000,125,252,377,000, AND 2 MORE SYNCHS.
:* THE LINE UNIT IS THEN CLOCKED EXTENSIVELY, AND THE TX SILO IS CHECKED TO
:* BE STILL LOADED (NO CHARS SHOULD HAVE BEEN REMOVED) AND THE RECEIVER
:* IS MONITORED TO INSURE THAT NO RCV FLAGS ARE GENERATED.
*****
BGNTST
```

9330 032136
(3) 032136
9331 032136 012737 032226 002346
9332 032144 012737 000013 002364
9333 032152 004737 005212
9334 032156 000226
9335 032160 000011
9336 032162 112737 000020 002352
9337 032170 004737 003422
9338 032174 004737 010476
9339 032200 002674
9340 032202 000007
9341 032204 004737 004726
9342 032210 000136
9343 032212 004737 004334
9344 032216 000003
9345 032220 004737 005746
9346 032224 000001
9347 032226 004737 003276
9348 032232
(3) 032232
(3) 032232 104401

```
T21::
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
MOV #13,REGNUM ;SET REG NO. = 13
JSR PC,INTRN ;MST CLR, LOAD 2 SOM'S
SYNCH
STRIP!DDCMP
MOVB #HDX,WRIBYT
JSR PC,WRITLU ;SET HDX BIT IN REG 13
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG1+4
7
JSR PC,STPLU ;CLK MORE THAN ENTIRE MSG
94.
JSR PC,OSIRDY ;CHK ORDY - 1, OCOR = 1
3
JSR PC,ISIRDY ;CHK ICIR - 1, IRDY - 0
1
24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10061: TRAP C$ETST
```

9349
9350
9351
9352
9353
9354
9355
9356
9357
9358
9359
9360
9361
9362

```
*****
:SBITL      TEST 22 - HALF-DUPLEX RCV DISABLED TEST WITH SILOS DISABLED
:
:* THIS TEST SENDS A MESSAGE IN HDX, CHAR MODE, WITH NO ERROR DETECTION, AND
:* THE SILOS DISABLED. THE MSG CONSISTS OF 2 SYNCHS AND 2 000 CHARS.
:* THE DATA IS SENT WITH LULOOP SET FOR TTL DATA LOOPBACK. THE PROGRAM CHECKS
:* THAT THE RECEIVER NEVER BECOMES ACTIVE, BECAUSE THE RCV CLOCK IS INHIBITED
:* WHEN THE HDX BIT IS SET.
*****
BGNTST
```

9363 032234
(3) 032234
9364 032234 012737 032432 002346
9365 032242 004737 003276
9366 032246 004737 010304

```
T22::
MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM USYRT FOR CHAR MODE, NO CRC
```

```
9367 032252 000226 SYNCH
9368 032254 000301 CRC2!CRC1!DDCMP
9369 032256 000000 000
9370 032260 000000 000
9371 032262 012737 000013 002364 MOV #13,REGNUM ;SET LU REG NO. = 13
9372 032270 112737 000020 002352 MOVB #HDX,WRIBYT ;SET HDX BIT IN REG 13
9373 032276 004737 003422 JSR PC,WRITLU
9374 032302 012737 000014 002364 MOV #14,REGNUM ;SET LU REG NO. = 14
9375 032310 112737 000140 002352 MOVB #TXEN!DISSI,WRIBYT ;DISABLE SILOS
9376 032316 004737 003422 JSR PC,WRITLU
9377 032322 012737 000140 002416 MOV #TXEN!DISSI,DISILO
9378 032330 012737 000002 002366 MOV #2,AXNUM ;SET AX BYTE NO FOR AX1
9379 032336 112737 000000 002360 MOVB #000,WAX15 ;SET *SOM IN USYRT
9380 032344 112737 000001 002362 MOVB #TSOM,WAX16
9381 032352 004737 003764 JSR PC,WRITAX
9382 032356 004737 004726 JSR PC,STPLU ;CLOCK FIRST SYNCH OUT
9383 032362 000013 11.
9384 032364 112737 000000 002360 MOVB #000,WAX15 ;LOAD FIRST 000 DATA CHAR INTO USYRT
9385 032372 112737 000000 002362 MOVB #000,WAX16
9386 032400 004737 003764 JSR PC,WRITAX
9387 032404 004737 004726 JSR PC,STPLU ;CLOCK SECOND SYNCH
9388 032410 000010 8.
9389 032412 004737 003764 JSR PC,WRITAX ;LOAD SECOND 000 CHAR
9390 032416 004737 004726 JSR PC,STPLU ;CLOCK FIRST 000 CHAR OUT
9391 032422 000013 11.
9392 032424 004737 006232 JSR PC,IACTIV ;CHK FOR IACT = 0 (RECEIVER NOT ACTIVE)
9393 032430 000000 0
9394 032432 005037 002416 24$: CLR DISILO ;CLEAR D:ENABLE SILO FLAG
9395 032436 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
9396 032442
(3) 032442
(3) 032442 104401 L10062: TRAP C$ETST
9397
9398
9399
9400
9401
9402
9403
9404
9405
9406
9407
9408
9409
9410
9411
9412
9413
9414
9415
9416
9417
9418
9419
9420
```

```
*****
SBTTL TEST 23 - INTERACTION OF MODEM CONTROL BITS
*
* THIS TEST WILL BE RUN ONLY IF THE P-TABLE FOR THIS UNIT INDICATES THAT THE
* H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. OTHERWISE, THE TEST WILL BE
* SKIPPED FOR THE UNIT.
* THE FOLLOWING SUBTESTS ARE PERFORMED:
* - A MASTER CLEAR IS DONE AND REG 13 IS READ AND CHECKED FOR INITIALIZED
* STATE, WITH LULOOK SET TO 1. THEN, LULOOK IS CLEARED AND REG 13 IS READ
* AND CHECKED FOR THE PROPER STATE, WITH LULOOK CLEARED.
* REG 13 IS THEN LOADED WITH 0'S, AND READ AND CHECKED FOR THE INITIALIZED
* STATE.
* REG 17 IS THEN READ AND CHECKED FOR INITIALIZED STATE.
* - RUN IS SET IN BSEL1, AND REG 13 IS READ AND CHECKED FOR RING SET.
* - POLL IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGQ SET.
* - BPOLL IS SET IN REG 12, ONLY TO LIGHT THE LED FOR THIS SIGNAL.
* - DTR IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR DTR AND MODR SET.
* - SELFR IS SET IN REG 17, AND REG 17 IS READ AND CHECKED FOR SIGR SET.
* - HDX IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR HDX SET.
```

```
9421          :* - MAINT1 IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR TEST MODE SET.
9422          :* - SELSBY IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR STBY SET.
9423          :* - A MASTER CLEAR IS DONE, 2 TSOM'S ARE LOADED INTO THE TX SILO, THE LINE
9424          :*   UNIT IS CLOCKED UNTIL THE TRANSMITTER IS ACTIVE, AND REG 13 IS READ AND
9425          :*   CHECKED FOR RTS, CS, CARR SET.
9426          :*****
9427 032444      BGNTST
          (3) 032444
9428 032444 012737 000027 002434      MOV #23,,TSTNUM ;SET TEST NO.
9429 032452 012737 033730 002346      MOV #A12,RETADR ;SET TEST EXIT ADRS FOR ERRORS
9430 032460 004737 010766      JSR PC,CKLPBK ;SEE IF H3254,5 INSTALLED - SKIP TEST IF NOT
9431 032464 100000      TCCHEK
9432
9433          :-----
9434          : DO MASTER CLEAR, CHK REGS 13,17 FOR INITIALIZED STATES
9435          :-----
9436 032466      BGNSUB
          (3) 032466
          (3) 032466 104402
9437 032470 004737 003276      JSR PC,MSTCLR ;ISSUE MASTER CLEAR
9438 032474 012737 000013 002364      MOV #13,REGNUM ;SET REG NO. - 13
9439 032502 004737 003344      JSR PC,READLU ;READ REG 13
9440 032506 023727 002350 000210      CMP REDBYT,#RING!MODR ;CHECK REG 13 FOR INIT'D STATE
9441 032514 001416      BEQ 6$ ;BR IF REG 13 INIT'D
9442 032516 012737 000210 002370      MOV #RING!MODR,GOODAT ;SET EXPECTED DATA
9443 032524 013737 002350 002372      MOV REDBYT,BADDAT ;SET ACTUAL DATA
9444 032532 004737 004200      JSR PC,GETALL ;GET REGS FOR PRINTOUT
9445          :REPORT REG MISCMPARE
9446 032536      ERRDF 3,EM3,ERR2
          (4) 032536 104455
          (5) 032540 000003
          (5) 032542 013265
          (5) 032544 015366
          TRAP CSERDF
          .WORD 3
          .WORD EM3
          .WORD ERR2
9447 032546      ESCAPE SUB
          (3) 032546 104410
          (3) 032550 000170
          TRAP C$ESCAPE
          .WORD L10064-.
9448 032552 142777 000010 147660 6$: BICB #LULOOP,#BSEL1 ;CLEAR LULOOP
9449 032560 004737 003344      JSR PC,READLU ;READ REG 13
9450 032564 023727 002350 000000      CMP REDBYT,#0 ;CHECK FOR INITIALIZED STATE
9451 032572 001416      BEQ 8$ ;BR IF OK
9452 032574 012737 000000 002370      MOV #0,GOODAT ;GET EXPECTED DATA
9453 032602 013737 002350 002372      MOV REDBYT,BADDAT ;GET ACTUAL DATA
9454 032610 004737 004200      JSR PC,GETALL ;GET REGS FOR PRINTOUT
9455          :REPORT REG NOT INITIALIZED BY MASTER CLEAR
9456 032614      ERRDF 2,EM2,ERR2
          (4) 032614 104455
          (5) 032616 000002
          (5) 032620 013226
          (5) 032622 015366
          TRAP CSERDF
          .WORD 2
          .WORD EM2
          .WORD ERR2
9457 032624      ESCAPE SUB
          (3) 032624 104410
          (3) 032626 000112
          TRAP C$ESCAPE
          .WORD L10064-.
9458 032630 005037 002352      8$: CLR WRIBYT ;SET DATA = 0 TO BE WRITTEN
9459 032634 004737 003422      JSR PC,WRITLU ;LOAD 0'S INTO REG 13
9460 032640 004737 003344      JSR PC,READLU ;READ REG 13
9461 032644 023727 002350 000000      CMP REDBYT,#000 ;CHECK FOR REG 13 CLEARED
```

```
9462 032652 001407          BEQ      9$          ;BR IF CLEARED
9463 032654 012737 000000 002370      MOV      #000,GOODAT ;SET EXPECTED DATA
9464 032662 013737 002350 002372      MOV      RELBYT,BADDAT ;SET ACTUAL DATA
9465 032670 000720          BR       3$          ;GO PRINT ERROR
9466 032672 012737 000017 002364 9$:      MOV      #17,REGNUM  ;SET REG NO. = 17
9467 032700 004737 003344          JSR      PC,READLU   ;READ REG 17
9468 032704 042737 000002 002350      BIC      #MCLK,REDBYT ;IGNORE MCLK BIT
9469 032712 123727 002350 000051      CMPB    REDBYT,#TXDATA!ICIR!DDCMP ;CHK REG 17 FOR INIT'D STATE
9470 032720 001407          BEQ      10$         ;BR IF REG 17 INITIALIZED
9471 032722 012737 000051 002370      MOV      #TXDATA!ICIR!DDCMP,GOODAT ;SET EXPECTED DATA
9472 032730 013737 002350 002372      MOV      REDBYT,BADDAT ;SET ACTUAL DATA
9473 032736 000675          BR       3$          ;GO REPORT ERROR
9474 032740          10$:
9475 032740          ENDSUB
(3) 032740
(3) 032740 104403          L10064:
TRAP      C$ESUB
9476
9477
9478 -----
9479 : SET RUN IN BSEL1, CHECK FOR RING SET IN REG 13
9480 -----
9480 032742          BGNSUB
(3) 032742
(3) 032742 104402          T23.2:
TRAP      C$BSUB
9481 032744 004737 003276          JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR
9482 032750 105077 147464          CLRB    @BSEL1      ;CLEAR LLOOP
9483 032754 112777 000200 147456      MOVB    #RUN,@BSEL1 ;SET RUN BIT IN BSEL1
9484 032762 112777 000010 147450      MOVB    #LLOOP,@BSEL1 ;CLEAR RUN, SET LLOOP
9485 032770 012737 000013 002364      MOV      #13,REGNUM ;SET REG NO. = 13
9486 032776 004737 003344          JSR      PC,READLU   ;READ REG 13
9487 033002 132737 000200 002350      BITB    #RING,REDBYT ;SEE IF RING = 1
9488 033010 001010          BNE     9$          ;BR IF RING = 1
9489 033012 004737 004200          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
9490 :REPORT RING NOT SET
ERRDF    56,EM56,ERR7
9491 033016          TRAP      C$ERDF
(4) 033016 104455          .WORD    56
(5) 033020 000070          .WORD    EM56
(5) 033022 014577          .WORD    ERR7
(5) 033024 020224
9492 033026          ESCAPE  SUB
(3) 033026 104410          TRAP      C$ESCAPE
(3) 033030 000002          .WORD    L10065-
9493 033032          9$:
9494 033032          ENDSUB
(3) 033032
(3) 033032 104403          L10065:
TRAP      C$ESUB
9495
9496 -----
9497 : SET POLL IN REG 13, CHK FOR SIGQ SET IN REG 17
9498 -----
9499          BGNSUB
(3) 033034
(3) 033034 104402          T23.3:
TRAP      C$BSUB
9500 033036 004737 003276          JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR
9501 033042 112737 000200 002352      MOVB    #POLL,WRIBYT ;SET REG NO. = 13
9502 033050 012737 000013 002364      MOV      #13,REGNUM ;SET POLL IN REG 13
9503 033056 004737 003422          JSR      PC,WRITLU
```

```

9504 033062 012737 000017 002364      MOV      #17,REGNUM      ;SET REG NO. = 17
9505 033070 004737 003344          JSR      PC,READLU      ;READ REG 17
9506 033074 132737 000100 002350      BITB    #SIGQ,REDBYT    ;SEE IF SIGQ = 1
9507 033102 001006          BNE     6$              ;BR IF SIGQ = 1
9508 033104 004737 004200          JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
9509                                     ;REPORT SIGQ NOT SET
9510 033110          ERRDF  63,EM63,ERR7
      (4) 033110 104455
      (5) 033112 000077          TRAP    C$ERDF
      (5) 033114 014726          .WORD  63
      (5) 033116 020224          .WORD  EM63
9511 033120          .WORD  ERR7
9512 033120      6$:      ENDSUB
      (3) 033120
      (3) 033120 104403          L10066: TRAP    C$ESUB
9513
9514 -----
9515 ; SET BPOLL IN REG 12, TO LIGHT LED ONLY
9516 -----
9517 033122          BGNSUB
      (3) 033122
      (3) 033122 104402          T23.4: TRAP    C$BSUB
9518 033124 004737 003276          JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
9519 033130 012737 000012 002364      MOV     #12,REGNUM      ;SET LU REG NO. = 12
9520 033136 112737 000100 002352      MOVB   #BPOLL,WRIBYT   ;SET BPOLL IN LU REG 12
9521 033144 004737 003422          JSR      PC,WRITLU
9522 033150          ENDSUB
      (3) 033150
      (3) 033150 104403          L10067: TRAP    C$ESUB
9523
9524 -----
9525 ; SET DTR IN REG 13, CHECK FOR DTR AND MODR SET IN REG 13
9526 -----
9527 033152          BGNSUB
      (3) 033152
      (3) 033152 104402          T23.5: TRAP    C$BSUB
9528 033154 004737 003276          JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR
9529 033160 012737 000013 002364      MOV     #13,REGNUM      ;SET REG NO. = 13
9530 033166 112737 000100 002352      MOVB   #DTR,WRIBYT
9531 033174 004737 003422          JSR      PC,WRITLU      ;SET DTR IN REG 13
9532 033200 004737 003344          JSR      PC,READLU      ;READ REG 13
9533 033204 132737 000100 002350      BITB   #DTR,REDBYT     ;SEE IF DTR = 1
9534 033212 001010          BNE     6$              ;BR IF DTR = 1
9535 033214 004737 004200          JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
9536                                     ;REPORT DTR NOT SET
9537 033220          ERRDF  55,EM55,ERR7
      (4) 033220 104455          TRAP    C$ERDF
      (5) 033222 000067          .WORD  55
      (5) 033224 014563          .WORD  EM55
      (5) 033226 020224          .WORD  ERR7
9538 033230          ESCAPE SUB
      (3) 033230 104410          TRAP    C$ESCAPE
      (3) 033232 000026          .WORD  L10070-
9539 033234 132737 000010 002350      6$:  BITB   #MODR,REDBYT  ;SEE IF MODR = 1
9540 033242 001006          BNE     12$             ;BR IF MODR = 1
9541 033244 004737 004200          JSR      PC,GETALL      ;GET REGS FOR PRINTOUT
  
```



```
9542 ;REPORT MODR NOT SET
9543 033250 ERRDF 57,EM57,ERR7
(4) 033250 104455
(5) 033252 000071 TRAP C$ERDF
(5) 033254 014614 .WORD 57
(5) 033256 020224 .WORD EM57
9544 033260 12$: .WORD ERR7
9545 033260
(3) 033260
(3) 033260 104403 L10070: TRAP C$ESUB
9546
9547 -----
9548 ; SET SELFR IN REG 13, CHK FOR SIGR SET IN REG 17
9549 -----
9550 033262 BGNSUB
(3) 033262
(3) 033262 104402 T23.6: TRAP C$BSUB
9551 033264 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
9552 033270 112737 000040 002352 MOVB #SEFR,WRIBYT
9553 033276 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
9554 033304 004737 003422 JSR PC,WRITLU ;SET SELFR IN REG 13
9555 033310 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
9556 033316 004737 003344 JSR PC,READLU ;READ REG 17
9557 033322 132737 000200 002350 BITB #SIGR,REDBYT ;SEE IF SIGR = 1
9558 033330 001006 BNE 6$ ;BR IF SIGR = 1
9559 033332 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9560 ;REPORT SIGR NOT SET
9561 033336 ERRDF 64,EM64,ERR7
(4) 033336 104455 TRAP C$ERDF
(5) 033340 000100 .WORD 64
(5) 033342 014743 .WORD EM64
(5) 033344 020224 .WORD ERR7
9562 033346 6$:
9563 033346 ENDSUB
(3) 033346
(3) 033346 104403 L10071: TRAP C$ESUB
9564
9565 -----
9566 ; SET HDX IN REG 13, CHK FOR HDX SET IN REG 13
9567 -----
9568 033350 BGNSUB
(3) 033350
(3) 033350 104402 T23.7: TRAP C$BSUB
9569 033352 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
9570 033356 112737 000020 002352 MOVB #HDX,WRIBYT
9571 033364 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
9572 033372 004737 003422 JSR PC,WRITLU ;SET HDX IN REG 13
9573 033376 004737 003344 JSR PC,READLU ;READ REG 13
9574 033402 132737 000020 002350 BITB #HDX,REDBYT ;SEE IF HDX = 1
9575 033410 001006 BNE 6$ ;BR IF HDX = 1
9576 033412 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9577 ;REPORT HDX NOT SET
9578 033416 ERRDF 58,EM58,ERR7
(4) 033416 104455 TRAP C$ERDF
(5) 033420 000072 .WORD 58
(5) 033422 014631 .WORD EM58
```

```
(5) 033424 020224                                     .WORD  ERR7
9579 033426                                     6$:
9580 033426                                     ENDSUB
(3) 033426                                     L10072:
(3) 033426 104403                                     TRAP  C$ESUB
9581
9582 -----
9583 : SET MAINT1 IN REG 13, CHK FOR TEST MODE SET IN REG 17
9584 :-----
9585 033430                                     BGNSUB
(3) 033430                                     T23.8:
(3) 033430 104402                                     TRAP  C$BSUB
9586 033432 004737 003276 002352      JSR  PC,MSTCLR      ;ISSUE MASTER CLEAR
9587 033436 112737 000010 002352      MOVB #MAINT1,WRIBYT
9588 033444 012737 000013 002364      MOV  #13,REGNUM    ;SET REG NO. = 13
9589 033452 004737 003422 002364      JSR  PC,WRITLU     ;SET MAINT1 IN REG 13
9590 033456 012737 000017 002364      MOV  #17,REGNUM    ;SET REG NO. = 17
9591 033464 004737 003344 002364      JSR  PC,READLU     ;READ REG 17
9592 033470 132737 000004 002350      BITB #TESTMD,REDBYT ;SEE IF TESTMD = 1
9593 033476 001006 002350      BNE  6$            ;BR IF TESTMD = 1
9594 033500 004737 004200 002350      JSR  PC,GETALL     ;GET REGS FOR PRINTOUT
9595 :REPORT TEST MODE NOT SET BY MAINT1
9596 033504 004737 004200 002350      ERRDF 52,EM52,ERR7
(4) 033504 104455                                     TRAP  C$ERDF
(5) 033506 000064                                     .WORD  52
(5) 033510 014451                                     .WORD  EM52
(5) 033512 020224                                     .WORD  ERR7
9597 033514                                     6$:
9598 033514                                     ENDSUB
(3) 033514                                     L10073:
(3) 033514 104403                                     TRAP  C$ESUB
9599
9600 -----
9601 : SET SELSBY IN REG 13, CHK FOR STBY SET IN REG 13
9602 :-----
9603 033516                                     BGNSUB
(3) 033516                                     T23.9:
(3) 033516 104402                                     TRAP  C$BSUB
9604 033520 004737 003276 002352      JSR  PC,MSTCLR     ;ISSUE MASTER CLEAR
9605 033524 112737 000002 002352      MOVB #SELSBY,WRIBYT
9606 033532 012737 000013 002364      MOV  #13,REGNUM    ;SET REG NO. = 13
9607 033540 004737 003422 002364      JSR  PC,WRITLU     ;SET SELSBY IN REG 13
9608 033544 004737 003344 002364      JSR  PC,READLU     ;READ REG 13
9609 033550 132737 000002 002350      BITB #STBY,REDBYT  ;SEE IF STBY = 1
9610 033556 001006 002350      BNE  6$            ;BR IF STBY = 1
9611 033560 004737 004200 002350      JSR  PC,GETALL     ;GET REGS FOR PRINTOUT
9612 :REPORT STBY NOT SET
9613 033564 004737 004200 002350      ERRDF 59,EM59,ERR7
(4) 033564 104455                                     TRAP  C$ERDF
(5) 033566 000073                                     .WORD  59
(5) 033570 014645                                     .WORD  EM59
(5) 033572 020224                                     .WORD  FRR7
9614 033574                                     6$:
9615 033574                                     ENDSUB
(3) 033574                                     L10074:
(3) 033574 104403                                     TRAP  C$ESUB
```

```
9616  
9617  
9618  
9619  
9620 033576  
  (3) 033576  
  (3) 033576 104402  
9621 033600 004737 005212  
9622 033604 000000  
9623 033606 000000  
9624 033610 012737 000013 002364  
9625 033616 004737 003344  
9626 033622 132737 000040 002350  
9627 033630 001010  
9628 033632 004737 004200  
9629  
9630 033636  
  (4) 033636 104455  
  (5) 033640 000074  
  (5) 033642 014662  
  (5) 033644 020224  
9631 033646  
  (3) 033646 104410  
  (3) 033650 000056  
9632 033652 132737 000004 002350 6$:  
9633 033660 001010  
9634 033662 004737 004200  
9635  
9636 033666  
  (4) 033666 104455  
  (5) 033670 000075  
  (5) 033672 014676  
  (5) 033674 020224  
9637 033676  
  (3) 033676 104410  
  (3) 033700 000026  
9638 033702 132737 000001 002350 9$:  
9639 033710 001006  
9640 033712 004737 004200  
9641  
9642 033716  
  (4) 033716 104455  
  (5) 033720 000076  
  (5) 033722 014711  
  (5) 033724 020224  
9643 033726  
9644 033726  
  (3) 033726  
  (3) 033726 104403  
9645  
9646 033730  
9647 033730 004737 003276  
9648 033734  
  (3) 033734  
  (3) 033734 104401  
9649
```

: DO MASTER CLEAR, LOAD 2 TSOM'S INTO TX SILO, CLOCK TRANSMITTER UNTIL ACTIVE,
: CHECK FOR RTS, CS, CARR SET IN REG 13

BGNSUB

T23.10: TRAP C\$BSUB

:MST CLR, LOAD SOM'S, CLK TRANSMITTER

000
000

MOV #13,REGNUM ;SET REG NO. = 13
JSR PC,READLU ;READ REG 13
BITB #RTS,REDBYT ;SEE IF RTS = 1
BNE 6\$;BR IF RTS = 1
JSR PC,GETALL ;GET REGS FOR PRINTOUT

:REPORT RTS NOT SET
ERRDF 60,EM60,ERR7

TRAP C\$ERDF
.WORD 60
.WORD EM60
.WORD ERR7

ESCAPE SUB

TRAP C\$ESCAPE
.WORD L10075-

6\$: BITB #CS,REDBYT ;SEE IF CS = 1
BNE 9\$;BR IF CS = 1
JSR PC,GETALL ;GET REGS FOR PRINTOUT

:REPORT CS NOT SET
ERRDF 61,EM61,ERR7

TRAP C\$ERDF
.WORD 61
.WORD EM61
.WORD ERR7

ESCAPE SUB

TRAP C\$ESCAPE
.WORD L10075-

9\$: BITB #CARR,REDBYT ;SEE IF CARR = 1
BNE 12\$;BR IF CARR = 1
JSR PC,GETALL ;GET REGS FOR PRINTOUT

:REPORT CARR NOT SET
ERRDF 62,EM62,ERR7

TRAP C\$ERDF
.WORD 62
.WORD EM62
.WORD ERR7

12\$:

ENDSUB

L10075: TRAP C\$ESUB

A12: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP

ENDTST

L10063: TRAP C\$ETST

9650
9651
9652
9653
9654
9655
9656
9657
9658
9659
9660
9661
9662
9663
9664
9665
9666
9667
9668
9669
9670
9671
9672
9673
9674
9675
9676
9677
9678
9679
9680
9681
9682
9683
9684
9685
9686
9687
9688
9689
9690
9691
9692
9693
9694
9695
9696
9697
9698
9699
9700

033736
(3) 033736
033736 012737 000030 002434
033744 012737 034124 002346
033752 004737 003276
033756 004737 010766
033762 000000
033764 013737 002422 034002
033772 004737 010304
033776 000000
034000 000300
034002 000000
034004 000000
034006 004737 010156
034012 012737 001177 003150
034020 142777 000010 146412
034026 012737 000012 002364
034034 012703 002762
034040 013702 002264
034044 004737 003344
034050 132737 000020 002350
034056 001011
034060 005202
034062 001370
034064 004737 004200
034070
(4) 034070 104455
(5) 034072 000021
(5) 034074 013436
(5) 034076 020224
034100 000411
034102 012337 034112
034106 004737 007266
034112 000000
034114 000000
034116 020327 003152
034122 103746

```
*****  
:SBTTL TEST 24 - DATA TEST - BIT MODE, NO ERR DET  
:*****  
: A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH ERROR DETECTION  
: INHIBITED. THE MESSAGE CONSISTS OF 2 FLAGS, PAT A REPEATED 3 TIMES,  
: AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
: THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
: SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
: TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
: TEST WILL NOT BE RUN.  
: PATTERN A - 125,252,000,377,001,002,004,010,020,040,100,200,376,  
: 375,373,367,357,337,277,177  
: 8-BIT CHARACTERS ARE USED.  
:*****  
BGNTST  
T24::  
MOV #24, TSTNUM ;SET TEST NO.  
MOV #24$, RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC, MSTCLR ;ISSUE MASTER CLEAR  
JSR PC, CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION  
0  
MOV MODINT, 6$ ;SET MODEM SELECTION  
JSR PC, SETUP ;PROGRAM THE USYRT  
000  
CRC2!CRC1 ;BIT MODE, NO ERR DET  
6$: .WORD 0 ;MODEM SELECTION GOES HERE  
000  
JSR PC, LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF  
MOV #RXEBL!177, RCVBUF+118. ; SET LAST DATA CHAR IN BUFFER  
BICB #LULOOB, @BSEL1 ;CLEAR LULOOB, CLOCK MSG  
MOV #12, REGNUM ;SET LU REG NO. - 12  
MOV #RCVBUF, R3 ;GET POINTER TO RCV MSG BUF  
9$: MOV TCOUNT, R2 ;INIT TIMER  
10$: JSR PC, READLU ;READ REG 12  
BITB #IRDY, REDBYT ;SEE IF IRDY IS SET YET  
BNE 12$ ;BR IF YES  
INC R2 ;INCREMENT TIMER  
BNE 10$ ;BR IF NO TIME-OUT YET  
JSR PC, GETALL ;GET REGS FOR PRINTOUT  
:REPORT IRDY NOT SET  
ERRDF 17, EM17, ERR7  
TRAP C$ERDF  
.WORD 17  
.WORD EM17  
.WORD ERR7  
BR 24$ ;ESCAPE TO END OF TEST  
12$: MOV (R3)+, 16$  
JSR PC, CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED  
16$: 0  
0  
CMP R3, #RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET  
BLO 9$ ;BR IF NOT YET
```

9701 034124
9702 034124
(3) 034124
(3) 034124 104401

24\$:
ENDTST

L10076:
TRAP C\$ETST

9703
9704
9705
9706
9707
9708
9709
9710
9711
9712
9713
9714
9715
9716
9717
9718
9719
9720
9721

```
*****  
:SBTTL TEST 25 - DATA TEST - CHAR MODE, NO ERR DET  
:  
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH ERROR DETECTION  
:* INHIBITED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,  
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
:* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO  
:* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
:* TEST WILL NOT BE RUN.  
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
:* 375,373,367,357,337,277,177  
:* 8-BIT CHARACTERS ARE USED.  
:*****
```

9722 034126
(3) 034126
9723 034126 012737 000031 002434
9724 034134 012737 034314 002346
9725 034142 004737 003276
9726 034146 004737 010766
9727 034152 000000
9728 034154 013737 002422 034172
9729 034162 004737 010304
9730 034166 000226
9731 034170 000311
9732 034172 000000
9733 034174 000000
9734 034176 004737 010156
9735 034202 012737 000177 003150
9736 034210 142777 000010 146222
9737 034216 012737 000012 002364
9738 034224 012703 002762
9739 034230 013702 002264
9740 034234 004737 003344
9741 034240 132737 000020 002350
9742 034246 001011
9743 034250 005202
9744 034252 001370
9745 034254 004737 004200
9746
9747 034260
(4) 034260 104455
(5) 034262 000021
(5) 034264 013436
(5) 034266 020224
9748 034270 000411
9749 034272 012337 034302

BGNTST

T25::

```
MOV #25, TSTNUM ;SET TEST NO.  
MOV #24$, RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC, MSTCLR ;ISSUE MASTER CLEAR  
JSR PC, CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION  
O  
MOV MODINT, 6$ ;SET MODEM SELECTION  
JSR PC, SETUP ;PROGRAM THE USYRT  
SYNCH  
CRC2!CRC1.STRIP!DDCMP ;CHAR MODE, NO ERR DET  
6$: .WORD 0 ;MODEM SELECTION GOES HERE  
O  
JSR PC, LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF  
MOV #177, RCVBUF+118. ;SET LAST DATA CHAR IN BUFFER  
BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG  
MOV #12, REGNUM ;SET LU REG NO. = 12  
MOV #RCVBUF, R3 ;GET POINTER TO RCV MSG BUF  
9$: MOV TCOUNT, R2 ;INIT TIMER  
10$: JSR PC, READIU ;READ REG 12  
BITB #IRDY, REDBYT ;SEF IF IRDY IS SET YET  
BNE 12$ ;BR IF YES  
INC R2 ;INCREMENT TIMER  
BNE 10$ ;BR IF NO TIME-OUT YET  
JSR PC, GETALL ;GET REGS FOR PRINTOUT  
;REPORT IRDY NOT SET  
ERRDF 17, EM17, ERR7
```

TRAP C\$ERDF
.WORD 17
.WORD EM17
.WORD ERR7

12\$: BR 24\$;ESCAPE TO END OF TEST
MOV (R3)+, 16\$

9750 034276 004737 007266
9751 034302 000000
9752 034304 000000
9753 034306 020327 003152
9754 034312 103746
9755 034314
9756 034314
(3) 034314
(3) 034314 104401

16\$: JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
0
0
CMP R3,#RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET
BLO 9\$;BR IF NOT YET

24\$:
ENDTST

L10077: TRAP C\$ETST

9757
9758
9759
9760
9761
9762
9763
9764
9765
9766
9767
9768
9769
9770
9771
9772
9773
9774
9775

:SBTTL TEST 26 - DATA TEST - BIT MODE, CRC-CCITT-1
:
:* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-1 ERROR
:* DETECTION. THE MESSAGE CONSISTS OF 2 FLAGS, PAT A REPEATED 3 TIMES,
:* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
:* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A - 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* 8-BIT CHARACTERS ARE USED.
:*****

9776 034316
(3) 034316
9777 034316 012737 000032 002434
9778 034324 012737 034504 002346
9779 034332 004737 003276
9780 034336 004737 010766
9781 034342 000000
9782 034344 013737 002422 034362
9783 034352 004737 010304
9784 034356 000000
9785 034360 000000
9786 034362 000000
9787 034364 000000
9788 034366 004737 010156
9789 034372 012737 101177 003150
9790 034400 142777 000010 146032
9791 034406 012737 000012 002364
9792 034414 012703 002762
9793 034420 013702 002264
9794 034424 004737 003344
9795 034430 132737 000020 002350
9796 034436 001011
9797 034440 005202
9798 034442 001370
9799 034444 004737 004200
9800
9801 034450
(4) 034450 104455

BGNTST
T26::
MOV #26.,TSTNUM ;SET TEST NO.
MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
0
MOV MODINT,6\$;SET MODEM SELECTION
JSR PC,SETUP ;PROGRAM THE USYRT
000
000 ;BIT MODE CRC-CCITT-1
6\$: .WORD 0 ;MODEM SELECTION GOES HERE
000
JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
MOV #CRCCHK!RXEBL!177,RCVBUF+118. ;SET LAST DATA CHAR IN BUFFER
BICB #LULOOP,0BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9\$: MOV TCOUNT,R2 ;INIT TIMER
10\$: JSR PC,READLU ;READ REG 12
BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
BNE 12\$;BR IF YES
INC R2 ;INCREMENT TIMER
BNE 10\$;BR IF NO TIME-OUT YET
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT IRDY NOT SET
ERRDF 17,EM17,ERR7

TRAP C\$ERDF


```

9852 034632 001370
9853 034634 004737 004200
9854
9855 034640
(4) 034640 104455
(5) 034642 000021
(5) 034644 013436
(5) 034646 020224
9856 034650 000411
9857 034652 012337 034662
9858 034656 004737 007266
9859 034662 000000
9860 034664 000000
9861 034666 020327 003152
9862 034672 03746
9863 034674
9864 034674
(3) 034674
(3) 034674 04401
9865
9866
9867
9868
9869
9870
9871
9872
9873
9874
9875
9876
9877
9878
9879
9880
9881
9882
9883
9884 034676
(3) 034676
9885 034676 012737 000034 002434
9886 034704 012737 035064 002346
9887 034712 004737 003276
9888 034716 004737 010766
9889 034722 000000
9890 034724 013737 002422 034742
9891 034732 004737 010304
9892 034736 000226
9893 034740 000011
9894 034742 000000
9895 034744 000000
9896 034746 004737 010156
9897 034752 012737 100577 003150
9898 034760 142777 000010 145452
9899 034766 012737 000012 002364
9900 034774 012703 002762

```

```

BNE 10$ ;BR IF NO TIME-OUT YET
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT IRDY NOT SET
ERRDF 17,EM17,ERR7
TRAP C$ERDF
.WORD 17
.WORD EM17
.WORD ERR7
BR 24$ ;ESCAPE TO END OF TEST
MOV (R3)+,16$
JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
16$: 0
0
CMP R3,#RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET
BLO 9$ ;BR IF NOT YET
24$:
ENDTST
L10101:
TRAP C$ETST

```

```

:*****
:SBTTL TEST 28 - DATA TEST - CHAR MODE, CRC-16
:*
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH CRC-16 ERROR
:* DETECTION. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
:* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* 8-BIT CHARACTERS ARE USED.
:*****

```

```

BGNTST
T28::
MOV #28,,TSTNUM ;SET TEST NO.
MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
0
MOV MODINT,6$ ;SET MODEM SELECTION
JSR PC,SETUP ;PROGRAM THE USYRT
6$:
STRIP!DDCMP
.WORD 0 ;MODEM SELECTION GOES HERE
000
JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
MOV #CRCCHK!RXBCC!177,RCVBUF+118. ;SET LAST DATA CHAR IN BUFFER
BICB #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF

```



```
9901 035000 013702 002264 9$: MOV TCOUNT,R2 ;INIT TIMER
9902 035004 004737 003344 10$: JSR PC,READLI ;READ REG 12
9903 035010 132737 000020 002350 BITB #IRDY,REDBY ;SEE IF IRDY IS SET YET
9904 035016 001011 BNE 12$ ;BR IF YES
9905 035020 005202 INC R2 ;INCREMENT TIMER
9906 035022 001370 BNE 10$ ;BR IF NO TIME-OUT YET
9907 035024 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9908 ;REPORT IRDY NOT SET
9909 ERRDF 17,EM17,ERR7
(4) 035030 104455 TRAP C$ERR7
(5) 035032 000021 .WORD 17
(5) 035034 013436 .WORD EM17
(5) 035036 020224 .WORD ERR7
9910 035040 000411 BR 24$ ;ESCAPE TO END OF TEST
9911 035042 012337 035052 12$: MOV (R3)+,16$
9912 035046 004737 007266 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
9913 035052 000000 16$: 0
9914 035054 000000 0
9915 035056 020327 003152 CMP R3,#RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET
9916 035062 103746 BLO 9$ ;BR IF NOT YET
9917 035064 24$:
9918 035064 ENDTST
(3) 035064 L10102:
(3) 035064 104401 TRAP C$ETST
9919
9920
9921
9922
9923
9924
```

```
*****
:SBTTL TEST 29 - DATA TEST - CHAR MODE, ODD VRC
:*
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH ODD VRC ERROR DETECTION
:* SELECTED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
:* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
*****
```

```
9938 035066 BGNTST
(3) 035066 T29::
9939 035066 012737 000035 002434 MOV #29,,TSTNUM ;SET TEST NO.
9940 035074 012737 035250 002346 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
9941 035102 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
9942 035106 004737 010766 JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
9943 035112 000000 J
9944 035114 013737 002422 035132 MOV MODINT,6$ ;SET MODEM SELECTION
9945 035122 004737 010304 JSR PC,SETUP ;PROGRAM THE USYRT
9946 035126 000026 026
9947 035130 000111 CRC1!STRIP.DDCMP
9948 035132 000000 6$: .WORD 0 ;MODEM SELECTION GOES HERE
9949 035134 000347 TXLEN2!TXLEN1.TXLENO!RXLEN2!RXLEN1!RXLENO
```

```
9950 035136 004737 010156 JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
9951 035142 142777 000010 145270 BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG
9952 035150 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
9953 035156 012703 002762 MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9954 035162 013702 002264 9$: MOV TCOUNT,R2 ;INIT TIMER
9955 035166 004737 003344 10$: JSR PC,READLU ;READ REG 12
9956 035172 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
9957 035200 001011 BNE 12$ ;BR IF YES
9958 035202 005202 INC R2 ;INCREMENT TIMER
9959 035204 001370 BNE 10$ ;BR IF NO TIME-OUT YET
9960 035206 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
9961 ;REPORT IRDY NOT SET
9962 ERRDF 17,EM17,ERR7
(4) 035212 104455 TRAP C$ERDF
(5) 035214 000021 .WORD 17
(5) 035216 013436 .WORD EM17
(5) 035220 020224 .WORD ERR7
9963 035222 000412 BR 24$ ;ESCAPE TO END OF TEST
9964 035224 112337 035236 12$: MOVB (R3)+,16$ ;GET AN EXPECTED DATA BYTE
9965 035230 005203 INC R3 ;INCREMENT POINTER
9966 035232 004737 007266 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
9967 035236 100000 16$: BCCCHK
9968 035240 000000 0
9969 035242 020327 003152 CMP R3,#RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET
9970 035246 103745 BLO 9$ ;BR IF NOT YET
9971 035250 24$:
9972 035250 ENDTST
(3) 035250 L10103: TRAP C$ETST
(3) 035250 104401
9973
9974
9975
9976
9977
9978
9979
9980
9981
9982
9983
9984
9985
9986
9987
9988
9989
9990
9991
9992 035252
(3) 035252
9993 035252 012737 000036 002434 MOV #30.,TSTNUM ;SET TEST NO.
9994 035260 012737 035434 002346 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
9995 035266 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
9996 035272 004737 010766 JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
9997 035276 000000 0
9998 035300 013737 002422 035316 MOV MODINT,6$ ;SET MODEM SELECTION
```

```
*****
.SBTTL TEST 30 - DATA TEST - CHAR MODE, EVEN VRC
;*
;* A MESSAGE IS INITIATED IN CHAR MODE, WITH EVEN VRC ERROR DETECTION
;* SELECTED. THE MESSAGE CONSISTS OF 2 SYNCHS, PAT A REPEATED 3 TIMES,
;* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
;* THE TEST WILL BE RUN WITH THE INTEGRAL MODEM SELECTED (EITHER HI OR LO
;* SPEED). IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
;* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
;* TEST WILL NOT BE RUN.
;* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
;* 375,373,367,357,337,277,177
;* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
*****
BGNTST
```

T30::

```
9999 035306 004737 010304 JSR PC,SETUP ;PROGRAM THE USYRT
10000 035312 000026 026
10001 035314 000211 CRC2!STRIP!DDCMP
10002 035316 000000 6$: .WORD 0 ;MODEM SELECTION GOES HERE
10003 035320 000347 TXLEN2!TXLEN1!TXLENO.RXLEN2!RXLEN1!RXLENO
10004 035322 004737 010156 JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
10005 035326 142777 000010 145104 BICB #LULOOB, @BSEL1 ;CLEAR LULOOB, CLOCK MSG
10006 035334 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
10007 035342 012703 002762 MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
10008 035346 013702 002264 9$: MOV TCOUNT,R2 ;INIT TIMER
10009 035352 004737 003344 10$: JSR PC,READLU ;READ REG 12
10010 035356 132737 000020 002350 BITB #IRDY,REDBYT ;SEC IF IRDY IS SET YET
10011 035364 001011 BNE 12$ ;BR IF YES
10012 035366 005202 INC R2 ;INCREMENT TIMER
10013 035370 001370 BNE 10$ ;BR IF NO TIME-OUT YET
10014 035372 004737 004200 JSR PC,GETALL ;GET REGS FOR PRINTOUT
10015 ;REPORT IRDY NOT SET
10016 035376 ERRDF 17,EM17,ERR7
(4) 035376 104455 TRAP C$ERDF
(5) 035400 000021 .WORD 17
(5) 035402 013436 .WORD EM17
(5) 035404 020224 .WORD ERR7
10017 035406 000412 BR 24$ ;ESCAPE TO END OF TEST
10018 035410 112337 035422 12$: MOVB (R3)+,16$ ;GET AN EXPECTED DATA CHAR
10019 035414 005203 INC R3 ;INCREMENT POINTER
10020 035416 004737 007266 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
10021 035422 100000 16$: BCCCHK
10022 035424 000000 0
10023 035426 020327 003152 CMP R3,#RCVBUF+120. ;SEE IF ALL CHARS CHECKED YET
10024 035432 103745 BLO 9$ ;BR IF NOT YET
10025 035434 24$:
10026 035434 ENDTST
(3) 035434
(3) 035434 104401 L10104: TRAP C$ETST
10027
10028
10029
10030
10031
10032
10033 ;*****
10034 .SBTTL TEST 31 - CONTIGUOUS ONES IN SEC. STA. ADRS. MODE, BIT MODE
10035 ;*
10036 ;* IN THIS TEST, A MESSAGE CONSISTING OF 5 ONES CHARS (377 OCT)
10037 ;* IS SENT IN SECONDARY STATION ADDRESS MODE, WITH THE STATION ADRS
10038 ;* FOR THIS LINE = 377. THE PROGRAM CHECKS FOR CORRECT RECEPTION OF
10039 ;* THE FIRST CHARACTER (STATION ADDRESS) AND THE REMAINING 4
10040 ;* ONES CHARACTERS (DATA). THIS TEST EXERCISES THE SECONDARY STATION
10041 ;* ADDRESS LOGIC, AND CHECKS THAT THE SEC. STA. ADRS. CAN BE BIT-STUFFED
10042 ;* AND TRANSMITTED AND RECEIVED CORRECTLY.
10043 ;*****
10043 035436 BGNTST
(3) 035436 T31::
10044 035436 012737 035542 002346 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
10045 035444 004737 005212 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
10046 035450 000377 377
10047 035452 000320 CRC2!CRC1.SECA ;BIT MODE, NO ERROR DET, SEC ADR MODE
```

CZDMSA M8203 STATIC TESTS #2
CZDMSA.P11 25-JUN-79 14:01

MACY11 30A(1052) 17-JUL-79 15:33 PAGE 9-127
TEST 31 - CONTIGUOUS ONES IN SEC. STA. ADRS. MODE, BI; MODE

SEQ 0165

```

10048 035454 004737 010716 JSR PC,LODSIL ;LOAD 5 377-CHARS INTO TX SILO
10049 035460 000377 377
10050 035462 000005 5
10051 035464 004737 010716 JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO
10052 035470 001000 TXEOM
10053 035472 000002 2
10054 035474 004737 004726 JSR PC,STPLU ;CLOCK MORE THAN ENTIRE MSG
10055 035500 000240 160.
10056 035502 004737 007266 JSR PC,CKDATA ;RCV SEC ADRS = 377
10057 035506 000377 377
10058 035510 000000 0
10059 035512 012701 000003 MOV #3,R1 ;RCV 3 MORE 377 CHARS
10060 035516 004737 007266 6$: JSR PC,CKDATA
10061 035522 000377 377
10062 035524 000000 0
10063 035526 005301 DEC R1
10064 035530 001372 BNE 6$
10065 035532 004737 007266 JSR PC,CKDATA ;RCV LAST 377 CHAR, CHK EBLK = 1
10066 035536 001377 1377
10067 035540 000000 0
10068 035542 004737 003276 24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
10069 035546 ENDTST
(3) 035546 L10105: TRAP C$ETST
(3) 035546 104401
10070
10071
10072
10073
10074
10075
10076
10077
10078
10079
10080
10081
10082
10083
10084
10085
10086
10087
10088

```

```

:*****
:SBTTL TEST 32 - DDCMP MESSAGE TEST - CHAR MODE
:*
:* IN THIS TEST, THREE USYRT MESSAGES ARE SENT TO SIMULATE A DDCMP HEADER,
:* DDCMP DATA MESSAGE, AND THE START OF A NEW DDCMP HEADER.
:* FIRST, THE DATA IN PATTERN A IS TRANSMITTED AND RECEIVED
:* AND THEN CRC (CRC-16) IS SENT, FOLLOWED BY THE DATA IN PATTERN A
:* AGAIN AND THE CRC ON THAT DATA, AND FINALLY THE DATA IN 'MSG1' IS
:* SENT WITH ITS CORRESPONDING CRC.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* MSG1 = SYNCH,SYNCH,SYNCH,SYNCH,000,125,252,377,000,SYNCH,SYNCH
:*****
BGNTST

```

```

10088 035550
(3) 035550
10089 035550 012737 036326 002346 MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS T32::
10090 -----
10091 : TRANSMIT AND RCV ENTIRE MSG
10092 -----
10093 035556 004737 005212 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
10094 035562 000226 SYNCH
10095 035564 000011 STRIP!DDCMP
10096 035566 004737 010554 JSR PC,LDBYTS ;LOAD 20 WORDS OF PAT A INTO TX SILO
10097 035572 002557 PATA
10098 035574 000024 20.
10099 035576 004737 010716 JSR PC,LODSIL ;LOAD AN EOM INTO TX SILO
10100 035602 001000 TXEOM

```

```

10101 035604 000001          1
10102 035606 004737 010554  JSR      PC,LDBYTS      ;LOAD 20 WORDS OF PAT A INTO TX SILO
10103 035612 002557          PATA
10104 035614 000024          20.
10105 035616 004737 010716  JSR      PC,LODSIL     ;LOAD 1 EOM INTO TX SILO
10106 035622 001000          TXEOM
10107 035624 000001          1
10108 035626 004737 010716  JSR      PC,LODSIL     ;LOAD 3 SOM'S INTO TX SILO
10109 035632 000400          TXSOM
10110 035634 000003          3
10111 035636 004737 010476  JSR      PC,LODMSG     ;LOAD MSG1 INTO TX SILO
10112 035642 002670          MSG1
10113 035644 000013          11.
10114 035646 004737 004726  JSR      PC,STPLU      ;CLOCK HDR MSG AND CRC CHARS
10115 035652 000300          192.
10116 035654 012737 000013 002364  MOV      #13,REGNUM    ;SET REG. NO. = 13
10117 035662 004737 003344          JSR      PC,READLU     ;READ REG 13
10118 035666 032737 000040 002350  BIT      #RTS,REDBYT   ;SEE IF RTS SET
10119 035674 001010          BNE      2$           ;BR IF RTS SET
10120 035676 004737 004200          JSR      PC,GETALL     ;GET REGS FOR PRINTOUT
10121                                     ;REPORT RTS NOT SET
10122                                     ERRDF 60,EM60,ERR7
      (4) 035702 104455
      (5) 035704 000074          TRAP    C$ERDF
      (5) 035706 014662          .WORD  60
      (5) 035710 020224          .WORD  EM60
10123 035712 000137 036326          JMP      24$          ;EXIT TEST
10124 035716 004737 004726  2$: JSR      PC,STPLU     ;CLK DATA MSG AND FIRST CRC CHAR
10125 035722 000250          168.
10126 035724 012703 000040          MOV      #32,R3       ;SET COUNTER FOR CHECKING RTS
10127 035730 004737 004726  4$: JSR      PC,STPLU     ;CLOCK LINE UNIT FOR 1 CYCLE
10128 035734 000001          1
10129 035736 004737 003344          JSR      PC,READLU     ;READ REG 13
10130 035742 032737 000040 002350  BIT      #RTS,REDBYT   ;CHK FOR RTS SET
10131 035750 001007          BNE      5$           ;BR IF RTS SET
10132 035752 004737 004200          JSR      PC,GETALL     ;GET REGS FOR PRINTOUT
10133                                     ;REPORT RTS NOT SET
10134                                     ERRDF 60,EM60,ERR7
      (4) 035756 104455          TRAP    C$ERDF
      (5) 035760 000074          .WORD  60
      (5) 035762 014662          .WORD  EM60
      (5) 035764 020224          .WORD  ERR7
10135 035766 000557          BR       24$
10136 035770 005303          5$: DEC      R3         ;DECR COUNTER
10137 035772 001356          BNE      4$           ;BR IF NOT DONE YET
10138                                     -----
10139                                     ; RFAD AND CHK HEADER AND CRC
10140                                     -----
10141 035774 012701 002557          MOV      #PATA,R1     ;INIT PATTERN A POINTER
10142 036000 112137 036010  7$: MOVB    (R1)+,8$     ;GET AN EXPECTED CHAR
10143 036004 004737 007266          JSR      PC,CKDATA    ;READ AND CHK A CHAR
10144 036010 000000          8$: .WORD  0
10145 036012 000000          0
10146 036014 020127 002601          CMP      R1,#PATB-2   ;SEE IF CHKING NEXT-TO-LAST CHAR YET
10147 036020 103767          BLO      7$           ;BR IF NOT YET
10148 036022 004737 007266          JSR      PC,CKDATA    ;READ AND CHK CHAR, BCC=0
  
```

10149 036026 100277
10150 036030 000000
10151 036032 004737 007266
10152 036036 100577
10153 036040 000000
10154 036042 004737 007266
10155 036046 000156
10156 036050 000000
10157 036052 004737 007266
10158 036056 000236
10159 036060 000000

CRCCHK!277
0
JSR PC,CKDATA ;READ AND CHK LAST CHAR, BCC=1
CRCCHK.RXBCC.177
0
JSR PC,CKDATA ;READ AND CHK HI CRC BYTE
156
0
JSR PC,CKDATA ;READ AND CHK LO CRC BYTE
236
0

: READ AND CHK DATA MSG AND CRC

10163 036062 012701 002557
10164 036066 112137 036076
10165 036072 004737 007266
10166 036076 000000
10167 036100 000000
10168 036102 020127 002601
10169 036106 103767
10170 036110 004737 007266
10171 036114 100277
10172 036116 000000
10173 036120 004737 007266
10174 036124 100577
10175 036126 000000
10176 036130 004737 007266
10177 036134 000156
10178 036136 000000
10179 036140 004737 007266
10180 036144 000236
10181 036146 000000

MOV #PATA,R1 ;INIT PATTERN A POINTER
9\$: MOV (R1)+,12\$;GET AN EXPECTED CHAR
JSR PC,CKDATA ;READ AND CHK A CHAR
12\$: .WORD 0
0
CMP R1,#PATB-2 ;SEE IF CHKING NEXT-TO-LAST CHAR YET
BLO 9\$;BR IF NOT YET
JSR PC,CKDATA ;READ AND CHK CHAR, BCC=0
CRCCHK!277
0
JSR PC,CKDATA ;READ AND CHK LAST CHAR, BCC=1
CRCCHK!RXBCC:177
0
JSR PC,CKDATA ;READ AND CHK HI CRC BYTE
156
0
JSR PC,CKDATA ;READ AND CHK LO CRC BYTE
236
0

: CLOCK 3RD MESSAGE ('MSG1' DATA)

10185 036150 012737 000012 002364
10186 036156 112737 000200 002352
10187 036164 004737 003422
10188 036170 012737 000013 002364
10189 036176 004737 004726
10190 036202 000150
10191 036204 004737 003344
10192 036210 032737 000040 002350
10193 036216 001407
10194 036220 004737 004200
10195

MOV #12,REGNUM ;SET REG NO. = 12
MOV #IC,WRIBYT ;SET IC TO CLEAR RECEIVER FOR NEW MSG
JSR PC,WRITLU
MOV #13,REGNUM ;RESTORE REG NO. TO 13
JSR PC,STPLU ;CLOCK THE REST OF MSG
104.
JSR PC,READLU ;READ REG 13
BIT #RTS,REDBYT ;SEE IF RTS IS CLEARED
BEQ 14\$;BR IF RTS CLEARED
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT RTS NOT CLEARED

10196 036224 104455
(4) 036224 000101
(5) 036226 014760
(5) 036230 020224
10197 036234 000434

ERRDF 65,EM65,ERR7
TRAP C\$ERRDF
.WORD 65
.WORD EM65
.WORD ERR7
BR 24\$

: READ AND CHECK 3RD MESSAGE AND CRC

10198
10199
10200

CZDMSA M8203 STATIC TESTS #2
CZDMSA.P11 25-JUN-79 14:01

M 13
MACY11 30A(1052) 17-JUL-79 15:33 PAGE 9-130
TEST 32 - DDCMP MESSAGE TEST - CHAR MODE

SEQ 0168

```
10201 036236 004737 007266      14$: JSR      PC,CKDATA      ;READ AND CHECK 000 DATA CHAR
10202 036242 000000                000
10203 036244 000000                0
10204 036246 004737 007266      JSR      PC,CKDATA      ;READ AND CHECK 125 DATA CHAR
10205 036252 000125                125
10206 036254 000000                0
10207 036256 004737 007266      JSR      PC,CKDATA      ;READ AND CHECK 252 DATA CHAR
10208 036262 000252                252
10209 036264 000000                0
10210 036266 004737 007266      JSR      PC,CKDATA      ;READ AND CHECK 377 DATA CHAR, AND BCC 0
10211 036272 100377                CRCCHK!377
10212 036274 000000                0
10213 036276 004737 007266      JSR      PC,CKDATA      ;READ AND CHECK 000 DATA CHAR, AND BCC 1
10214 036302 100400                CRCCHK.RXBCC.000
10215 036304 000000                0
10216 036306 004737 007266      JSR      PC,CKDATA      ;READ AND CHK HI CRC BYTE
10217 036312 000160                160
10218 036314 000000                0
10219 036316 004737 007266      JSR      PC,CKDATA      ;READ AND CHK LO CRC BYTE
10220 036322 000034                034
10221 036324 000000                0
10222 036326 004737 003276      24$: JSR      PC,MSTCLR      ;ISSUE MASTER CLEAR TO CLEAN UP
10223 036332                ENDTST
      (3) 036332
      (3) 036332 104401                L10106: TRAP C$ETST
10224
10225
10226
10227
10228
```

```
10230 .SBTTL HARDWARE PARAMETER CODING SECTION
10231
10232
10233
10234 :////////////////////////////////////////////////////////////////////
10235 :/ THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
10236 :/ THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
10237 :/ MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
10238 :/ INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
10239 :/ MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
10240 :/ WITH THE OPERATOR.
10241 :////////////////////////////////////////////////////////////////////
10242
10243 036334 BGNHRD
(3) 036334 000046 .WORD L10107-L$HARD/2
(3) 036336 L$HARD::
10244
10245 036336 GPRMA ADDRES,2,0,160000,177776,YES .WORD T$CODE
(4) 036336 001031 .WORD ADDRES
(4) 036340 036452 .WORD T$LLOLIM
(4) 036342 160000 .WORD T$HILIM
(4) 036344 177776
10246 036346 GPRMA VECTOR,4,0,0,674,YES .WORD T$CODE
(4) 036346 002031 .WORD VECTOR
(4) 036350 036500 .WORD T$LLOLIM
(4) 036352 000000 .WORD T$HILIM
(4) 036354 000674
10247 036356 GPRMD PRIRTY,6,0,7000,4,7,YES .WORD T$CODE
(4) 036356 003032 .WORD PRIRTY
(4) 036360 036531 .WORD 7000
(4) 036362 007000 .WORD T$LLOLIM
(4) 036364 000004 .WORD T$HILIM
(4) 036366 000007
10248 036370 GPRMD SWPAC1,12,0,377,0,056,YES .WORD T$CODE
(4) 036370 005032 .WORD SWPAC1
(4) 036372 036562 .WORD 377
(4) 036374 000377 .WORD T$LLOLIM
(4) 036376 000000 .WORD T$HILIM
(4) 036400 000056
10249 036402 GPRMD SWPAC2,14,0,377,0,377,YES .WORD T$CODE
(4) 036402 006032 .WORD SWPAC2
(4) 036404 036623 .WORD 377
(4) 036406 000377 .WORD T$LLOLIM
(4) 036410 000000 .WORD T$HILIM
(4) 036412 000377
10250 036414 GPRMD SWPAC3,16,0,377,0,377,YES .WORD T$CODE
(4) 036414 007032 .WORD SWPAC3
(4) 036416 036664 .WORD 377
(4) 036420 000377 .WORD T$LLOLIM
(4) 036422 000000 .WORD T$HILIM
(4) 036424 000377
10251 036426 GPRMD LOOPBK,20,0,7,0,4,YES .WORD T$CODE
(4) 036426 010032 .WORD LOOPBK
(4) 036430 036725 .WORD 7
(4) 036432 000007 .WORD T$LLOLIM
(4) 036434 000000
```



```
10275 .SBTTL SOFTWARE PARAMETER CODING SECTION
10276
10277
10278 :////////////////////////////////////////////////////////////////////
10279 :// THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
10280 :// THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
10281 :// MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
10282 :// INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
10283 :// MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
10284 :// WITH THE OPERATOR.
10285 :////////////////////////////////////////////////////////////////////
10286
10287 037276          BGNSFT
      (3) 037276 000016
      (3) 037300
                                .WORD L10110-L$SOFT/2
                                L$SOFT::
10288
10289 037300          GPRML  ISMANI,0,1,YES
      (4) 037300 000130
      (4) 037302 037334
      (4) 037304 000001
                                .WORD  T$CODE
                                .WORD  ISMANI
                                .WORD  1
10290 037306          GPRML  ISPRNT,2,1,YES
      (4) 037306 001130
      (4) 037310 037422
      (4) 037312 000001
                                .WORD  T$CODE
                                .WORD  ISPRNT
                                .WORD  1
10291 037314          GPRML  ISWPAK,4,1,YES
      (4) 037314 002130
      (4) 037316 037505
      (4) 037320 000001
                                .WORD  T$CODE
                                .WORD  ISWPAK
                                .WORD  1
10292 037322          GPRMD  TIMCNT,6,0,177777,0,177777,YES
      (4) 037322 003032
      (4) 037324 037552
      (4) 037326 177777
      (4) 037330 000000
      (4) 037332 177777
                                .WORD  T$CODE
                                .WORD  TIMCNT
                                .WORD  177777
                                .WORD  T$LLOLIM
                                .WORD  T$HILIM
10293
10294 037334          ENDSFT
      (2)
      (3) 037334
                                .EVEN
                                L10110:
10295
10296 037334 051511 046440 047101 ISMANI: .ASCIZ /IS MAN. INTERVEN. DESIRED TO MOUNT TEST CONNECTOR(S) /
      037342 020056 047111 042524
      037350 053122 047105 020056
      037356 042504 044523 042522
      037364 020104 047524 046440
      037372 052517 052116 052040
      037400 051505 020124 047503
      037406 047116 041505 047524
      037414 024122 024523 000040
10297 037422 044123 052517 042114 ISPRNT: .ASCIZ /SHOULD SWITCH PACK AND AX3-15 PRINTOUT BE ALLOWED /
      037430 051440 044527 041524
      037436 020110 040520 045503
      037444 040440 042116 040440
      037452 031530 030455 020065
      037460 051120 047111 047524
      037466 052125 041040 020105
      037474 046101 047514 042527
```

```
10298 037502 020104 000  
037505 123 047510 046125 ISWPAK: .ASCIZ /SHOULD SWITCH PACK TESTS BE ALLOWED /  
037512 020104 053523 052111  
037520 044103 050040 041501  
037526 020113 042524 052123  
037534 020123 042502 040440  
037542 046114 053517 042105  
037550 000040  
10299 037552 051515 020107 044524 TIMCNT: .ASCIZ /MSG TIMER VALUE (0-177777), 0 = LONGEST TIME-OUT : /  
037560 042515 020122 040526  
037566 052514 020105 030050  
037574 030455 033467 033467  
037602 024467 020054 020060  
037610 020075 047514 043516  
037616 051505 020124 044524  
037624 042515 047455 052125  
037632 035040 000040  
  
10300 .EVEN  
10301  
10302  
10303  
10304  
10305  
10306  
10307  
10308  
10309 ;***** PATCH AREA FOR DEBUG *****  
10310 037636 PATCH:  
10311 037736 .-.+100  
10312 037736 000240 NOP  
10313 037740 000240 NOP  
10314 037742 000240 NOP  
10315 ;*****  
10316  
10317  
10318  
10319 037744 ENDMOD  
10320  
10321 037744 LASTAD  
(2)  
(4) 037744 000000 .EVEN  
(4) 037746 000000 .WORD 0  
(3) 037750 L$LAST:: .WORD 0  
10322  
10323 . 000001 .END
```

ABORT - 000004	5364#													
ADDRS 036452	7371	7377	7392	7412	7430	7451	7469	7488	7507	10245	10256#			
ADR - 000020 G	5323#													
ANBITS 002547	5763#													
APA = 000200	5542#													
ASBC0 - 000020	5499#													
ASBC1 = 000040	5498#													
ASBC2 = 000100	5497#													
ASSEMB- 000010	5176													
AXNUM 002366	5683#	6099	6137	6150	6175	6177	6183*	6189*	6190	6192*	6194	6385	6393*	
	6436*	6600	6606*	6635*	6641*	6918	6920*	6928*	6939*	7063	7077*	7227*	7242	
	7245*	7258*	7262*	8157*	8185*	8188*	8200*	8569*	8860*	8906*	8982*	9035*	9378*	
AX0.15= 002306	5610#	6182	7401	7420	7440	7459	7477	7497	7515					
AX0.16= 002310	5611#	7401	7420	7440	7459	7477	7497	7515						
AX1 - 000001	5395#	5467#												
AX1.15= 002312	5612#	7401	7420	7440	7459	7477	7497	7515						
AX1.16= 002314	5613#	7401	7420	7440	7459	7477	7497	7515						
AX2 - 000002	5394#	5466#												
AX2.15= 002316	5614#	7403	7422	7442	7461	7479	7499	7517						
AX2.16= 002320	5615#	7403	7422	7442	7461	7479	7499	7517						
AX3.15= 002322	5616#	7403	7422	7442	7461	7479	7499	7517						
AX3.16= 002324	5617#	7403	7422	7442	7461	7479	7499	7517						
AX315U= 000332	5560#													
A11 025434	8292#	8322												
A12 033730	9429	9646#												
BADDAT 002372	5685#	6772*	6773*	6794*	6795*	7380	7395	7434	7491	8106*	8115*	8222*	8499*	
	8523*	8547*	9145*	9185*	9443*	9453*	9464*	9472*						
BAUDRT 037042	10252	10264#												
BCC = 000001	5443#	6798	6800	6806										
BCCCHK- 100000	5625#	6796	9018	9026	9071	9079	9967	10021						
BDRATE 002464	5717#	7097	7154	7636*										
BIT0 000001 G	5323#	5345	5357	5366	5395	5407	5419	5431	5443	5455	5467	5479	5491	
	5503	5515	5524	5536	5549	5559	5570	5643	6177	6218	6347	6527	6568	
	6608	7606	9024	9077										
BIT00 000001 G	5323#													
BIT01 = 000002 G	5323#													
BIT02 = 000004 G	5323#													
BIT03 - 000010 G	5323#													
BIT04 - 000020 G	5323#													
BIT05 = 000040 G	5323#													
BIT06 = 000100 G	5323#													
BIT07 = 000200 G	5323#													
BIT08 = 000400 G	5323#													
BIT09 - 001000 G	5323#													
BIT1 000002 G	5323#	5344	5356	5365	5384	5394	5406	5418	5430	5442	5454	5466	5478	
	5490	5502	5514	5523	5535	5548	5558	5569	5644	6234	6511	6622		
BIT10 002000 G	5323#	5580	5592											
BIT11 - 004000 G	5323#	5579	5591											
BIT12 = 010000 G	5323#													
BIT13 = 020000 G	5323#													
BIT14 - 040000 G	5323#													
BIT15 = 100000 G	5323#	5623	5625	5626	5628	6315	6469							
BIT2 = 000004 G	5323#	5343	5355	5364	5383	5393	5405	5417	5429	5441	5453	5465	5477	
	5489	5501	5513	5522	5534	5547	5568							
BIT3 = 000010 G	5323#	5342	5354	5363	5382	5392	5404	5416	5428	5440	5452	5464	5476	
	5488	5500	5512	5521	5533	5546	5557							

BIT4 = 000020 G	5323#	5341	5353	5381	5391	5403	5415	5427	5439	5451	5463	5475	5487
	5499	5511	5532	5545	5556								
BIT5 = 000040 G	5323#	5352	5373	5380	5390	5402	5414	5426	5438	5450	5462	5474	5486
	5498	5510	5531	5544	5567								
BIT6 = 000100 G	5323#	5340	5351	5372	5379	5389	5401	5413	5425	5437	5449	5461	5473
	5485	5497	5509	5530	5543	5555	5566						
BIT7 = 000200 G	5323#	5339	5350	5362	5371	5378	5400	5412	5424	5436	5448	5460	5472
	5484	5496	5508	5520	5529	5542	5554	5565	6766	6767			
BIT8 = 000400 G	5323#	5582	5594										
BIT9 = 001000 G	5323#	5581	5593										
BOE = 000400 G	5323#												
BPOLL = 000100	5372#	9520											
BSFL1 002440	5706#	5968*	5970*	5971*	5984*	5985*	5990*	6078*	6080*	6181*	6319*	6320*	6322*
	6412*	6413*	6418*	6431*	6934*	6936*	7076*	7229*	7621*	7622*	8568*	9115*	9179*
	9190*	9192*	9242*	9292*	9448*	9482*	9483*	9484*	9682*	9736*	9790*	9844*	9898*
	9951*	10005*											
BSEL4 002442	5707#	6016	6037*										
CARR = 000001	5455#	9182	9638										
CHKABT 012136	7242#												
CHPCHK= 100000	5623#	8426	8733	8768	8803								
CHPTYP 002420	5696#	6316	6426*	6430*	6470	7574*	8190	8196					
CKDATA 007266	6757#	7789	7796	7840	7845	7848	7881	7886	7889	7940	7943	7950	7964
	7967	7992	7995	8031	8034	8064	8067	8312	8315	8390	8393	8442	8445
	8620	8623	8626	8629	8632	8650	8653	8656	8659	8662	8681	8684	8687
	8690	8693	8744	8747	8750	8753	8756	8779	8782	8785	8788	8791	8814
	8817	8820	8823	8826	9017	9025	9070	9078	9129	9206	9256	9306	9696
	9750	9804	9858	9912	9966	10020	10056	10060	10065	10143	10148	10151	10154
	10157	10165	10170	10173	10176	10179	10201	10204	10207	10210	10213	10216	10219
CKLPBK 010766	7062#	9106	9171	9233	9283	9430	9672	9726	9780	9834	9888	9942	9996
CRCCHK= 100000	5626#	6900	7012	8316	8394	8633	8663	8694	8757	8792	8827	9789	9843
	9897	10149	10152	10171	10174	10211	10214						
CRCTYO= 000001	5549#												
CRCTY1= 000002	5548#												
CRCTY2= 000004	5547#												
CRC1 = 000100	5401#	7778	7826	7867	8147	8673	8800	8810	8866	8988	9006	9368	9677
	9731	9839	9947	10047									
CRC2 = 000200	5400#	7778	7826	7867	8147	8912	9041	9059	9368	9677	9731	10001	10047
CS = 000004	5453#	9632											
C\$AU = 000052	5176#	7747											
C\$AUTO= 000061	5176#	7692											
C\$BRK = 000022	5176#	7647											
C\$BSEG= 000004	5176#												
C\$BSUB= 000002	5176#	8264	8293	8347	8374	8480	8508	8532	8862	8908	8984	9037	9436
	9480	9499	9517	9527	9550	9568	9585	9603	9620				
C\$CEFG= 000045	5176#												
C\$CLCK= 000062	5176#												
C\$CLEA= 000012	5176#	7709											
C\$CLOS= 000035	5176#												
C\$CLP1= 000006	5176#												
C\$CVEC= 000036	5176#												
C\$DCLN= 000044	5176#												
C\$DODU= 000051	5176#	7689											
C\$DRPT= 000024	5176#												
C\$DU = 000053	5176#	7728											
C\$EDIT= 000000	5176#	5216											
C\$ERDF 000055	5176#	6224	6230	6240	6246	6353	6359	6517	6523	6533	6539	6574	6580

FMT6	012423	7283#	7384	7399	7403	7418	7422	7438	7442	7457	7461	7475	7479	7495
		7499	7513	7517										
FMT7	012456	7284#	7379	7414	7471	7490								
FMT8	012466	7285#	7394	7433	7453	7509								
FMT9	012522	7286#	7383	7398	7402	7417	7421	7437	7441	7456	7460	7474	7478	7494
		7498	7512	7516										
FRSPAS	002400	5688#	7604*	8485	8512	8536	8571							
FRSTIM	002376	5687#	7577	7584*										
F\$AU =	000015	5176#	7746	7747										
F\$AUTO=	000020	5176#	7680	7692										
F\$BGN =	000040	5176#	5182	7370	7376	7391	7410	7429	7449	7468	7486	7506	7533	7551
		7569	7680	7706	7723	7746	7774	7800	7819	7901	7927	8007	8021	8038
		8053	8071	8086	8120	8138	8234	8260	8264	8287	8293	8319	8324	8343
		8347	8370	8374	8397	8398	8417	8449	8476	8480	8504	8508	8528	8532
		8552	8553	8566	8581	8604	8698	8723	8831	8856	8862	8892	8898	8902
		8908	8938	8944	8948	8950	8978	8984	9031	9037	9084	9086	9102	9150
		9167	9212	9229	9262	9279	9312	9330	9348	9363	9396	9427	9436	9447
		9457	9475	9480	9492	9494	9499	9512	9517	9522	9527	9538	9545	9550
		9563	9568	9580	9585	9598	9603	9615	9620	9631	9637	9644	9648	9668
		9702	9722	9756	9776	9810	9830	9864	9884	9918	9938	9972	9992	10026
		10043	10069	10088	10223	10243	10287	10319						
F\$CLEA=	000007	5176#	7706	7709										
F\$DU -	000016	5176#	7723	7728										
F\$END =	000041	5176#	5182	7372	7385	7404	7423	7443	7462	7480	7500	7518	7535	7660
		7692	7709	7728	7747	7774	7800	7819	7901	7927	8007	8021	8038	8053
		8071	8086	8120	8138	8234	8260	8264	8287	8293	8319	8324	8343	8347
		8370	8374	8397	8398	8417	8449	8476	8480	8504	8508	8528	8532	8552
		8553	8566	8581	8604	8698	8723	8831	8856	8862	8892	8898	8902	8908
		8938	8944	8948	8950	8978	8984	9031	9037	9084	9086	9102	9150	9167
		9212	9229	9262	9279	9312	9330	9348	9363	9396	9427	9436	9447	9457
		9475	9480	9492	9494	9499	9512	9517	9522	9527	9538	9545	9550	9563
		9568	9580	9585	9598	9603	9615	9620	9631	9637	9644	9648	9668	9702
		9722	9756	9776	9810	9830	9864	9884	9918	9938	9972	9992	10026	10043
		10069	10088	10223	10254	10294	10319							
F\$HARD-	000004	5176#	10243	10254										
F\$HW -	000013	5176#	5261	5275										
F\$INIT	000006	5176#	7569	7660										
F\$JMP =	000050	5176#												
F\$MOD -	000000	5176#	5182	10319										
F\$MSG -	000011	5176#	7370	7372	7376	7385	739	7404	7410	7423	7429	7443	7449	7462
		7468	7480	7486	7500	7506	7518							
F\$PROT=	000021	5176#	7551	7555										
F\$PWR =	000017	5176#												
F\$RPT -	000012	5176#	7533	7535										
F\$SEG =	000003	5176#												
F\$SOFT=	000005	5176#	10287	10294										
F\$SRV -	000010	5176#												
F\$SUB	000002	5176#	8264	8287	8293	8319	8347	8370	8374	8397	8480	8504	8508	8528
		8532	8552	8862	8902	8908	8948	8984	9031	9037	9084	9436	9475	9480
		9494	9499	9512	9517	9522	9527	9545	9550	9563	9568	9580	9585	9598
		9603	9615	9620	9644									
F\$SW	000014	5176#	5289	5296										
F\$TEST-	000001	5176#	7774	7800	7819	7901	7927	8007	8021	8038	8053	8071	8086	8120
		8138	8234	8260	8324	8343	8398	8417	8449	8476	8553	8566	8581	8604
		8698	8723	8831	8856	8950	8978	9086	9102	9150	9167	9212	9229	9262
		9279	9312	9330	9348	9363	9396	9427	9648	9668	9702	9722	9756	9776

L\$PRT	002112	G	5216#		
L\$REPP	002062	G	5216#		
L\$REV	002010	G	5216#		
L\$RPT	022016	G	7533#		
L\$SOFT	037300	G	5216	10287#	
L\$SPC	002056	G	5216#		
L\$SPCP	002020	G	5216#		
L\$SPTP	002024	G	5216#		
L\$STA	002030	G	5216#		
L\$SW	002256	G	5216	5289#	
L\$TEST	002114	G	5216#		
L\$TIML	002014	G	5216#		
L\$UNIT	002012	G	5216#	7611	
L10000	002254		5261	5275#	
L10001	002266		5289	5296#	
L10002	015364		7372#		
L10003	015672		7385#		
L10004	016354		7404#		
L10005	017032		7423#		
L10006	017544		7443#		
L10007	020222		7462#		
L10010	020654		7480#		
L10011	021362		7500#		
L10012	022014		7518#		
L10013	022016		7535#		
L10015	022520		7660#		
L10016	023056		7692#		
L10017	023060		7709#		
L10020	023110		7728#		
L10021	023144		7747#		
L10022	023274		7800#		
L10023	023650		7901#		
L10024	024206		8007#		
L10025	024270		8038#		
L10026	024360		8071#		
L10027	024554		8120#		
L10030	025324		8234#		
L10031	025564		8324#		
L10032	025426		8287#		
L10033	025546		8319#		
L10034	025770		8398#		
L10035	025666		8370#		
L10036	025766		8397#		
L10037	026124		8449#		
L10040	026576		8553#		
L10041	026310		8504#		
L10042	026442		8528#		
L10043	026574		8552#		
L10044	026712		8581#		
L10045	027250		8698#		
L10046	027660		8831#		
L10047	030332		8950#		
L10050	030102		8892	8898	8902#
L10051	030324		8938	8944	8948#
L10052	031034		9086#		
L10053	030570		9031#		

CZDMSA M8203 STA*IC TESTS #2
CZDMSA.P11 25-JUN-79 14:01

MACY11 30A(1052) 17-JUL-79 15:33 PAGE 10-11
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0185

ORDY = 000020	5427#	6220	6226																		
OSIRDY 004334	6210#	6389	6406	6433	6477	6483	9343														
OVRR = 000010	5440#	6843	6845	6851	7855	7895															
OSAPTS= 000000	5176#	5216																			
OSAU = 000001	5176#	5206#	5216																		
OSBGNR- 000000	5176#	5216																			
OSBGNS= 000001	5176#	5206#	5216																		
OSDU = 000001	5176#	5206#	5216																		
OSERRT= 000000	5176#	5216																			
OSGNSW= 000001	5176#	5206#	5216																		
OSPOIN= 000001	5176#	5206#	5216																		
OSSETU= 000000	5176#	5216	10321																		
PATA 002557	5774#	6891	10097	10103	10141	10163															
PATB 002603	5797#	6896	10146	10168																	
PATCH 037636	10310#																				
PATQ 002613	5809#	8868	8914																		
PATR 002623	5819#	8993	9015	9046	9068																
PATS 002642	5836#	7780	7787	7792																	
PATT 002662	5854#	8291																			
PNT = 001000 G	5323#																				
POLL = 000200	5378#	9501																			
PRI = 002000 G	5323#																				
PRIOR 002334	5669#																				
PRIPTY 036531	10247	10258#																			
PRI00 = 000000 G	5323#																				
PRI01 = 000040 G	5323#																				
PRI02 = 000100 G	5323#																				
PRI03 = 000140 G	5323#																				
PRI04 = 000200 G	5323#																				
PRI05 = 000240 G	5323#																				
PRI06 = 000300 G	5323#																				
PRI07 = 000340 G	5323#	7682	7684																		
PRNFLG 002260	5292#	8487	8514	8538	8573																
PSTACK 002332	5668#	6248	6361	6541	6582	6636	6864	7259	7571*												
RAB = 000004	5441#	6828	6830	6836	7974	8001															
RABT = 000004	5501#	7247																			
RAX15 002354	5678#	6114*	6115*	6185	7113	7123	7130	7143	7148	7151	7162	7165	7168								
	7174	8579																			
RAX16 002356	5679#	6118*	6119*	6187	6610	6616	6624	6630	7247	7253	8203	8214	8222								
RCVBUF 002762	5906#	6890	7008	9117	9132	9194	9209	9244	9259	9294	9309	9681*	9684								
	9699	9735*	9738	9753	9789*	9792	9807	9843*	9846	9861	9897*	9900	9915								
	9953	9969	10007	10023																	
RCV1ST 006752	6676#	7830	7871	7938	7948	7962	7990	8029	8062	8094	8310	8388	8440								
RDALL = 000004	5405#	8378																			
RDAX = 000020	5391#	5463#	6101																		
RDRXSI 006672	6652#	6763	7854	7894	7973	8000	8107														
READAX 003576	6095#	6184	6607	7078	7246	8202	8570														
READLU 003344	6004#	6055	6105	6113	6117	6156	6217	6233	6346	6415	6427	6510	6526								
	6567	6654	6658	6697	6775	7189	7205	7216	8165	8483	8511	8535	8883								
	8929	9119	9140	9181	9196	9246	9296	9439	9449	9460	9467	9486	9505								
	9532	9556	9573	9591	9608	9625	9686	9740	9794	9848	9902	9955	10009								
	10117	10129	10191																		
READY = 000200	5460#	6106	6157																		
REDBYT 002350	5676#	6016*	6017*	6056	6106	6114	6118	6157	6220	6226	6236	6242	6349								
	6355	6416	6428	6513	6519	6529	6535	6570	6576	6655	6659	6698	6776								
	7190	7206	7217	8166	8484*	8493	8496	8499	8517	8520	8523	8541	8544								

SAVE4	002404	5690#	7579*	7582	7690														
SAVE6	002406	5691#	7580*	7583	7691														
SAVLEN	002424	5698#	5992*	6764	6933*	7576*	9000*	9053*											
SCRACH	002326	5666#																	
SEC =	000020	5545#																	
SECA =	000020	5403#	8268	8300	8351	8378	10047												
SELFR =	000040	5380#	9552																
SELSBY=	000002	5384#	9605																
SEL4	002442	5708#	7623*	7624*															
SEL6	002444	5709#	5969*	6079*	7625*	7626*													
SETUP	010304	6918#	8145	8610	8640	8671	9109	9174	9236	9286	9366	9675	9729	9783					
		9837	9891	9945	9999														
SFPTBL	002256 G	5289#																	
SIGQ =	000100	5473#	9506																
SIGR =	000200	5472#	9557																
SOM =	000001	5366#																	
STALL	004636	6277#	6321	6323	6414	6419	6432												
STARES	002402	5689#	7103	7108	7117	7599*	7605*												
STARST	022166	7587	7590	7598#															
STBY =	000002	5454#	9141	9609															
STEPLU=	000020	5341#	6320	6322	6413	6418	6431												
STEPMP=	000001	5345#	5970	5971															
STPCLK	003240	5967#	6014	6038															
STPERR	007164	6729#	8427	8432	8437	8739	8774	8809	9005	9058									
STPLU	004726	6311#	6479	6693	6736	6860	7785	7832	7837	7873	7878	8096	8163	8176					
		8180	8183	8194	8205	8278	8282	8361	8365	8425	8430	8435	8618	8648					
		8679	8737	8742	8772	8777	8807	8812	8876	8878	8881	8922	8924	8927					
		9002	9008	9012	9055	9061	9065	9341	9382	9387	9390	10054	10114	10124					
		10127	10189																
STR =	000040	5544#																	
STRIP =	000010	5404#	7826	7867	8421	8428	8433	8438	8612	8730	8740	8866	8912	8988					
		9006	9041	9059	9111	9176	9238	9288	9335	9731	9893	9947	10001	10095					
SUBRPC	002336	5670#	6211	6212	6214*	6215*	6252*	6340	6341	6343*	6344*	6365*	6386*	6387*					
		6439*	6459*	6460*	6486*	6504	6505	6507*	6508*	6545*	6561	6562	6564*	6565*					
		6586*	6601	6602	6604*	6605*	6640*	6679*	6680*	6716*	6759*	6760*	6870*	6940*					
		7065*	7066*	7230*	7243*	7244*	7411	7450	7487	7572*									
SVCGBL=	000000	5176#	5182	5190#	5216	5239	5261	5289	5929	5934	7370	7376	7391	7410					
		7429	7449	7468	7486	7506	7533	7551	7569	7680	7706	7723	7746	10243					
		10287	10321#																
SVCINS=	000001	5176#	5187#	5216	5239	5261	5289	5929	5934	6224	6230	6240	6246	6353					
		6359	6517	6523	6533	6539	6574	6580	6614	6620	6628	6634	6780	6785					
		6804	6810	6819	6825	6834	6840	6849	6855	7105	7110	7119	7196	7212					
		7251	7257	7371	7372	7377	7378	7379	7380	7381	7382	7383	7384	7385					
		7392	7393	7394	7395	7396	7397	7398	7399	7400	7401	7402	7403	7404					
		7411	7412	7413	7414	7415	7416	7417	7418	7419	7420	7421	7422	7423					
		7430	7431	7432	7433	7434	7435	7436	7437	7438	7439	7440	7441	7442					
		7443	7450	7451	7452	7453	7454	7455	7456	7457	7458	7459	7460	7461					
		7462	7469	7470	7471	7472	7473	7474	7475	7476	7477	7478	7479	7480					
		7487	7488	7489	7490	7491	7492	7493	7494	7495	7496	7497	7498	7499					
		7500	7507	7508	7509	7510	7511	7512	7513	7514	7515	7516	7517	7518					
		7535	7586	7587	7589	7590	7592	7593	7595	7596	7613	7614	7642	7644					
		7646	7647	7651	7655	7660	7682	7689	7692	7709	7725	7727	7728	7747					
		7800	7859	7899	7901	7978	8005	8007	8038	8071	8118	8120	8173	8212					
		8225	8234	8264	8287	8293	8319	8324	8347	8370	8374	8397	8398	8449					
		8480	8491	8493	8502	8504	8508	8517	8526	8528	8532	8541	8550	8552					
		8553	8577	8579	8581	8698	8831	8862	8891	8892	8897	8898	8902	8908					

TMP5	002530	5745#													
TMP6	002532	5746#													
TMP7	002534	5747#													
T\$OM =	000001	5524#	8159	9380											
T\$TCON	002462	5716#	7070	7074	7095	7125	7136	7171	7183	7199	7635*				
T\$TNUM	002434	5702#	7110	9103*	9168*	9230*	9280*	9428*	9669*	9723*	9777*	9831*	9885*	9939*	
		9993*													
TXAB =	000004	5522#													
TXABT -	002000	5580#	5892												
TXCHAR	005574	6457#	8731	8766	8801										
TXDATA=	000040	5474#	8887	8893	8933	8939	9469	9471							
TXEN =	000100	5389#	5461#	8058	8151	8153	9375	9377							
TXEOM =	001000	5581#	5876	5877	5878	5879	5885	5886	6901	7782	8276	8308	8359	8386	
		8871	8917	8996	9049	10052	10100	10106							
TXGA =	000010	5521#													
TXGOA =	004000	5579#													
TXLENO=	000040	5567#	6764	8229	8874	8920	8999	9052	9949	10003					
TXLEN1=	000100	5566#	6764	8143	8874	8920	8999	9052	9949	10003					
TXLEN2=	000200	5565#	6764	8874	8920	8999	9052	9949	10003						
TXSOM -	000400	5582#	5869	5870	5881	5882	5893	5894	6395	6886	8990	9043	10109		
TXWORD	002412	5693#	6291*	6293	6296	6395*	6396*	6461*	6886*	6892*	6893*	6894	6901*	6958*	
		6983*	6984*	7029*	7782*										
TX0 -	000001	5357#	5515#												
TX1 =	000002	5356#	5514#												
TX2 =	000004	5355#	5513#												
TX3 =	000010	5354#	5512#												
TX4 =	000020	5353#	5511#												
TX5 =	000040	5352#	5510#												
TX6 -	000100	5351#	5509#												
TX7 =	000200	5350#	5508#												
TYPEY	022733	7655	7665#												
T\$ARGC	000002	5216#	7105#	7110#	7119#	7371#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#	
		7392#	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7411#	
		7412#	7413#	7414#	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7430#	7431#	
		7432#	7433#	7434#	7435#	7436#	7437#	7438#	7439#	7440#	7441#	7442#	7450#	7451#	
		7452#	7453#	7454#	7455#	7456#	7457#	7458#	7459#	7460#	7461#	7469#	7470#	7471#	
		7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7487#	7488#	7489#	7490#	7491#	
		7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7507#	7508#	7509#	7510#	7511#	
		7512#	7513#	7514#	7515#	7516#	7517#	7646#	7651#	7727#	8491#	8493#	8517#	8541#	
		8577#	8579#												
T\$CODE-	003032	7655#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10289#	10290#	10291#	10292#	
T\$ERRN=	000101	5176#	6224#	6230#	6240#	6246#	6353#	6359#	6517#	6523#	6533#	6539#	6574#	6580#	
		6614#	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#	6834#	6840#	6849#	
		6855#	7196#	7212#	7251#	7257#	7859#	7899#	7978#	8005#	8118#	8173#	8212#	8225#	
		8502#	8526#	8550#	8891#	8897#	8937#	8943#	9126#	9148#	9188#	9203#	9253#	9303#	
		9446#	9456#	9491#	9510#	9537#	9543#	9561#	9578#	9596#	9613#	9630#	9636#	9642#	
		9693#	9747#	9801#	9855#	9909#	9962#	10016#	10122#	10134#	10196#				
T\$EXCP=	000000	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10292#					
T\$FLAG=	000040	8892#	8898#	8938#	8944#	9447#	9457#	9492#	9538#	9631#	9637#				
T\$GMAN-	000000	5176#													
T\$HILI=	177777	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10292#					
T\$LAST=	000001	5176#	10321#												
T\$LOLI-	000000	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10292#					
T\$LSYM	010000	5176#	5275	5296	7372	7385	7404	7423	7443	7462	7480	7500	7518	7535	
		7660	7692	7709	7728	7747	7800	7901	8007	8038	8071	8120	8234	8287	
		8319	8324	8370	8397	8398	8449	8504	8528	8552	8553	8581	8698	8831	

	8902	8948	8950	9031	9084	9086	9150	9212	9262	9312	9348	9396	9475
	9494	9512	9522	9545	9563	9580	9598	9615	9644	9648	9702	9756	9810
	9864	9918	9972	10026	10069	10223	10254	10294					
T\$LTNO= 000040	10321#												
T\$NEST= 177777	5176#	5182#	5261#	5275#	5289#	5296#	7370#	7372#	7376#	7385#	7391#	7404#	7410#
	7423#	7429#	7443#	7449#	7462#	7468#	7480#	7486#	7500#	7506#	7518#	7533#	7535#
	7551#	7555#	7569#	7660#	7680#	7692#	7706#	7709#	7723#	7728#	7746#	7747#	7774#
	7800#	7819#	7901#	7927#	8007#	8021#	8038#	8053#	8071#	8086#	8120#	8138#	8234#
	8260#	8264#	8287#	8293#	8319#	8324#	8343#	8347#	8370#	8374#	8397#	8398#	8417#
	8449#	8476#	8480#	8504#	8508#	8528#	8532#	8552#	8553#	8566#	8581#	8604#	8698#
	8723#	8831#	8856#	8862#	8902#	8908#	8948#	8950#	8978#	8984#	9031#	9037#	9084#
	9086#	9102#	9150#	9167#	9212#	9229#	9262#	9279#	9312#	9330#	9348#	9363#	9396#
	9427#	9436#	9475#	9480#	9494#	9499#	9512#	9517#	9522#	9527#	9545#	9550#	9563#
	9568#	9580#	9585#	9598#	9603#	9615#	9620#	9644#	9648#	9668#	9702#	9722#	9756#
	9776#	9810#	9830#	9864#	9884#	9918#	9938#	9972#	9992#	10026#	10043#	10069#	10088#
T\$NSO = 000000	10223#	10243#	10254#	10287#	10294#	10319#							
T\$NS1 = 000005	5182#	10319											
	5261#	5275	5289#	5296	7370#	7372	7376#	7385	7391#	7404	7410#	7423	7429#
	7443	7449#	7462	7468#	7480	7486#	7500	7506#	7518	7533#	7535	7551#	7555
	7569#	7660	7680#	7692	7706#	7709	7723#	7728	7746#	7747	7774#	7800	7819#
	7901	7927#	8007	8021#	8038	8053#	8071	8086#	8120	8138#	8234	8260#	8324
	8343#	8398	8417#	8449	8476#	8553	8566#	8581	8604#	8698	8723#	8831	8856#
	8950	8978#	9086	9102#	9150	9167#	9212	9229#	9262	9279#	9312	9330#	9348
	9363#	9396	9427#	9648	9668#	9702	9722#	9756	9776#	9810	9830#	9864	9884#
T\$NS2 = 000002	9918	9938#	9972	9992#	10026	10043#	10069	10088#	10223	10243#	10254	10287#	10294
	8264#	8287	8293#	8319	8347#	8370	8374#	8397	8480#	8504	8508#	8528	8532#
	8552	8862#	8902	8908#	8948	8984#	9031	9037#	9084	9436#	9475	9480#	9494
	9499#	9512	9517#	9522	9527#	9545	9550#	9563	9568#	9580	9585#	9598	9603#
	9615	9620#	9644										
T\$PTNU= 000000	5176#												
T\$SAVL= 177777	5176#												
T\$SEGL= 177777	5176#												
T\$SUBN= 000000	5176#	7774#	7819#	7927#	8021#	8053#	8086#	8138#	8260#	8264#	8293#	8343#	8347#
	8374#	8417#	8476#	8480#	8508#	8532#	8566#	8604#	8723#	8856#	8862#	8908#	8978#
	8984#	9037#	9102#	9167#	9229#	9279#	9330#	9363#	9427#	9436#	9480#	9499#	9517#
	9527#	9550#	9568#	9585#	9603#	9620#	9668#	9722#	9776#	9830#	9884#	9938#	9992#
T\$TAGL= 177777	10043#	10088#											
T\$TAGN= 010111	5176#												
	5176#	5261#	5289#	7370#	7376#	7391#	7410#	7429#	7449#	7468#	7486#	7506#	7533#
	7551#	7569#	7680#	7706#	7723#	7746#	7774#	7819#	7927#	8021#	8053#	8086#	8138#
	8260#	8264#	8293#	8343#	8347#	8374#	8417#	8476#	8480#	8508#	8532#	8566#	8604#
	8723#	8856#	8862#	8908#	8978#	8984#	9037#	9102#	9167#	9229#	9279#	9330#	9363#
	9427#	9436#	9480#	9499#	9517#	9527#	9550#	9568#	9585#	9603#	9620#	9668#	9722#
	9776#	9830#	9884#	9938#	9992#	10043#	10088#	10243#	10287#				
T\$TEMP= 000000	5239#	5275#	5296#	7372#	7385#	7404#	7423#	7443#	7462#	7480#	7500#	7518#	7535#
	7555#	7655#	7660#	7692#	7709#	7728#	7747#	7800#	7901#	8007#	8038#	8071#	8120#
	8234#	8287#	8319#	8324#	8370#	8397#	8398#	8449#	8504#	8528#	8552#	8553#	8581#
	8698#	8831#	8892#	8898#	8902#	8938#	8944#	8948#	8950#	9031#	9084#	9086#	9150#
	9212#	9262#	9312#	9348#	9396#	9447#	9457#	9475#	9492#	9494#	9512#	9522#	9538#
	9545#	9563#	9580#	9598#	9615#	9631#	9637#	9644#	9648#	9702#	9756#	9810#	9864#
	9918#	9972#	10026#	10069#	10223#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#
T\$TEST 000040	10254#	10289#	10290#	10291#	10292#	10294#	10319#						
	5176#	7774#	7819#	7927#	8021#	8053#	8086#	8138#	8260#	8264	8293	8343#	8347
	8374	8417#	8476#	8480	8508	8532	8566#	8604#	8723#	8856#	8862	8908	8978#
	8984	9037	9102#	9167#	9229#	9279#	9330#	9363#	9427#	9436	9480	9499	9517
	9527	9550	9568	9585	9603	9620	9668#	9722#	9776#	9830#	9884#	9938#	9992#

CZDMSA M8203 STATIC TESTS #2
CZDMSA.P11 25-JUN-79 14:01

MACY11 30A(1052) 17-JUL-79 15:33 PAGE 10-19
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0193

WAX16 002362	5681#	6146*	6147	6397*	6922*	6932*	6933	8159*	8178*	8186*	8189*	8192*	8198*
WRDYTO= 000002	8874*	8920*	8999*	9000	9052*	9053	9380*	9385*					
WRIBYT 002352	5644#	6135	6161										
	5677#	6036*	6037	6099*	6100*	6101*	6102*	6137*	6138*	6139*	6143*	6147*	6150*
	6151*	6152*	6153*	6293*	6296*	6401*	6731*	6732*	6738*	6926*	7139*	7185*	7201*
	7217*	7218*	7219*	7852*	7971*	8099*	8151*	8155*	9336*	9372*	9375*	9458*	9501*
	9520*	9530*	9552*	9570*	9587*	9605*	10186*						
WRITAX 003764	6133#	6398	6923	6935	8160	8175	8179	8182	8187	8193	8199	8875	8921
	9001	9054	9381	9386	9389								
WRITLU 003422	6030#	6103	6140	6144	6148	6154	6294	6297	6402	6733	6739	6927	7140
	7186	7202	7220	7853	7972	8100	8152	8156	9337	9373	9376	9459	9503
	9521	9531	9554	9572	9589	9607	10187						
XYZ = 000100	5555#	5560	7067	7089	7151	7157	7174	9108	9173	9234	9285		
X\$ALWA= 000000	5176#												
X\$FALS= 000040	5176#												
X\$OFFS= 000400	5176#												
X\$TRUE= 000020	5176#												
\$LSTIN= 000001	5185#												
\$LSTTA= 000001	5186#												
. - 037750	5168#	5661#	5861#	5906#	5934#	7666#	7731#	8892	8898	8938	8944	9447	9457
	9492	9538	9631	9637	10267#	10311#							

ENDSFT	915#	5176#	10294												
ENDSRV	932#	5176#													
ENDSUB	952#	5176#	8287	8319	8370	8397	8504	8528	8552	8902	8948	9031	9084	9475	9494
	9512	9522	9545	9563	9580	9598	9615	9644							
ENDSW	974#	5176#	5296												
ENDTST	988#	5176#	7800	7901	8007	8038	8071	8120	8234	8324	8398	8449	8553	8581	8698
	8831	8950	9.96	9150	9212	9262	9312	9348	9396	9648	9702	9756	9810	9864	9918
	9972	10026	10069	10223											
EQUALS	1009#	5176#	5323												
ERRDF	1087#	5176#	6224	6230	6240	6246	6353	6359	6517	6523	6533	6539	6574	6580	6614
	6620	6628	6634	6780	6785	6804	6810	6819	6825	6834	6840	6849	6855	7196	7212
	7251	7257	7859	7899	7978	8005	8118	8173	8212	8225	8502	8526	8550	8891	8897
	8937	8943	9126	9148	9188	9203	9253	9303	9446	9456	9491	9510	9537	9543	9561
	9578	9596	9613	9630	9636	9642	9693	9747	9801	9855	9909	9962	10016	10122	10134
	10196														
ERRHRD	1099#	5176#													
ERROR	1109#	5176#													
ERRSF	1118#	5176#													
ERRSOF	1130#	5176#													
ERRTBL	1140#	5176#													
ESCAPE	1156#	5176#	8892	8898	8938	8944	9447	9457	9492	9538	9631	9637			
EXIT	1186#	5176#													
FEQUAL	1228#	5176#													
GETBYT	1246#	5176#													
GETPRI	1264#	5176#													
GETWOR	1256#	5176#													
GMANIA	1286#	5176#													
GMANID	1299#	5176#													
GMANIL	1315#	5176#	7655												
GPHARD	1328#	5176#	7613												
GPRMA	1340#	5176#	10245	10246											
GPRMD	1372#	5176#	10247	10248	10249	10250	10251	10252	10292						
GPRML	1407#	5176#	7655#	10289	10290	10291									
HEADER	1432#	5176#	5216												
INLOOP	1446#	5176#													
IOSETU	1453#	5176#													
IOSTAR	1466#	5176#													
KT11	1488#	5176#													
LASTAD	1659#	5176#	10321												
MANUAL	1677#	5176#	7642												
MEMORY	1685#	5176#													
MSBYTE	2901#	5176#	5216#												
MSCHEC	3206#	5176#													
MSCNTO	3279#	5176#	7655#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10289#	10290#	10291#	10292#
MSCOUN	3124#	5176#	7105#	7110#	7119#	7371#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#	7392#
	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7411#	7412#	7413#	7414#
	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7430#	7431#	7432#	7433#	7434#	7435#	7436#
	7437#	7438#	7439#	7440#	7441#	7442#	7450#	7451#	7452#	7453#	7454#	7455#	7456#	7457#	7458#
	7459#	7460#	7461#	7469#	7470#	7471#	7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7487#
	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7507#	7508#	7509#
	7510#	7511#	7512#	7513#	7514#	7515#	7516#	7517#	7646#	7651#	7727#	8491#	8493#	8517#	8541#
	8577#	8579#													
MSDATA	2614#	5176#	5216#	5929#	5934#										
MSDECR	3063#	5176#	5275#	5296#	7372#	7385#	7404#	7423#	7443#	7462#	7480#	7500#	7518#	7535#	7555#
	7660#	7692#	7709#	7728#	7747#	7800#	7901#	8007#	8038#	8071#	8120#	8234#	8287#	8319#	8324#
	8370#	8397#	8398#	8449#	8504#	8528#	8552#	8553#	8581#	8698#	8831#	8902#	8948#	8950#	9031#

	9084#	9086#	9150#	9212#	9262#	9312#	9348#	9396#	9475#	9494#	9512#	9522#	9545#	9563#	9580#
	9598#	9615#	9644#	9648#	9702#	9756#	9810#	9864#	9918#	9972#	10026#	10069#	10223#	10254#	10294#
	10319#														
MSDEFA	3263#	5176#	7655#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10289#	10290#	10291#	10292#
MSBENDE	3145#	5176#	5275#	5296#	7372#	7385#	7404#	7423#	7443#	7462#	7480#	7500#	7518#	7535#	7660#
	7692#	7709#	7728#	7747#	7800#	7901#	8007#	8038#	8071#	8120#	8234#	8287#	8319#	8324#	8370#
	8397#	8398#	8449#	8504#	8528#	8552#	8553#	8581#	8698#	8831#	8902#	8948#	8950#	9031#	9084#
	9086#	9150#	9212#	9262#	9312#	9348#	9396#	9475#	9494#	9512#	9522#	9545#	9563#	9580#	9598#
MSERRI	9615#	9644#	9648#	9702#	9756#	9810#	9864#	9918#	9972#	10026#	10069#	10223#	10254#	10294#	10319#
	2365#	5176#	6224#	6230#	6240#	6246#	6353#	6359#	6517#	6523#	6533#	6539#	6574#	6580#	6614#
	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#	6834#	6840#	6849#	6855#	7196#	7212#
	7251#	7257#	7859#	7899#	7978#	8005#	8118#	8173#	8212#	8225#	8502#	8526#	8550#	8891#	8897#
	8937#	8943#	9126#	9148#	9188#	9203#	9253#	9303#	9446#	9456#	9491#	9510#	9537#	9543#	9561#
	9578#	9596#	9613#	9630#	9636#	9642#	9693#	9747#	9801#	9855#	9909#	9962#	10016#	10122#	10134#
	10196#														
MSESCA	2921#	5176#	8892#	8898#	8938#	8944#	9447#	9457#	9492#	9538#	9631#	9637#			
MSESCS	2932#	5176#	8892#	8898#	8938#	8944#	9447#	9457#	9492#	9538#	9631#	9637#			
MSEXCP	3186#	5176#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10292#				
MSEXIT	2943#	5176#													
MSEXSE	2965#	5176#													
MSEX TJ	2954#	5176#													
MSGEN	3087#	5176#	5182#	5216#	5239#	5261#	5275#	5289#	5296#	5929#	5934#	7370#	7372#	7376#	7385#
	7391#	7404#	7410#	7423#	7429#	7443#	7449#	7462#	7468#	7480#	7486#	7500#	7506#	7518#	7533#
	7535#	7551#	7569#	7655#	7660#	7680#	7692#	7706#	7709#	7723#	7728#	7746#	7747#	7774#	7800#
	7819#	7901#	7927#	8007#	8021#	8038#	8053#	8071#	8086#	8120#	8138#	8234#	8260#	8264#	8287#
	8293#	8319#	8324#	8343#	8347#	8370#	8374#	8397#	8398#	8417#	8449#	8476#	8480#	8504#	8508#
	8528#	8532#	8552#	8553#	8566#	8581#	8604#	8698#	8723#	8831#	8856#	8862#	8902#	8908#	8948#
	8950#	8978#	8984#	9031#	9037#	9084#	9086#	9102#	9150#	9167#	9212#	9229#	9262#	9279#	9312#
	9330#	9348#	9363#	9396#	9427#	9436#	9475#	9480#	9494#	9499#	9512#	9517#	9522#	9527#	9545#
	9550#	9563#	9568#	9580#	9585#	9598#	9603#	9615#	9620#	9644#	9648#	9668#	9702#	9722#	9756#
	9776#	9810#	9830#	9864#	9884#	9918#	9938#	9972#	9992#	10026#	10043#	10069#	10088#	10223#	10243#
	10254#	10287#	10294#	10321#											
MSGENR	2764#	5176#	7655#												
MSGETS	3079#	5176#	5275#	5296#	7372#	7385#	7404#	7423#	7443#	7462#	7480#	7500#	7518#	7535#	7555#
	7660#	7692#	7709#	7728#	7747#	7800#	7901#	8007#	8038#	8071#	8120#	8234#	8287#	8319#	8324#
	8370#	8397#	8398#	8449#	8504#	8528#	8552#	8553#	8581#	8698#	8831#	8902#	8948#	8950#	9031#
	9084#	9086#	9150#	9212#	9262#	9312#	9348#	9396#	9475#	9494#	9512#	9522#	9545#	9563#	9580#
	9598#	9615#	9644#	9648#	9702#	9756#	9810#	9864#	9918#	9972#	10026#	10069#	10223#	10254#	10294#
	10319#														
MSGETT	2634#	5176#	8892#	8898#	8938#	8944#	9447#	9457#	9492#	9538#	9631#	9637#			
MSGNGB	2689#	5176#	5182#	5216#	5239#	5261#	5289#	5929#	5934#	7370#	7376#	7391#	7410#	7429#	7449#
	7468#	7486#	7506#	7533#	7551#	7569#	7680#	7706#	7723#	7746#	10243#	10287#	10321#		
MSGNIN	3101#	5176#	5216#	5239#	5261#	5289#	5929#	5934#	6224#	6230#	6240#	6246#	6353#	6359#	6517#
	6523#	6533#	6539#	6574#	6580#	6614#	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#
	6834#	6840#	6849#	6855#	7105#	7110#	7119#	7196#	7212#	7251#	7257#	7371#	7372#	7377#	7378#
	7379#	7380#	7381#	7382#	7383#	7384#	7385#	7392#	7393#	7394#	7395#	7396#	7397#	7398#	7399#
	7400#	7401#	7402#	7403#	7404#	7411#	7412#	7413#	7414#	7415#	7416#	7417#	7418#	7419#	7420#
	7421#	7422#	7423#	7430#	7431#	7432#	7433#	7434#	7435#	7436#	7437#	7438#	7439#	7440#	7441#
	7442#	7443#	7450#	7451#	7452#	7453#	7454#	7455#	7456#	7457#	7458#	7459#	7460#	7461#	7462#
	7469#	7470#	7471#	7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7480#	7487#	7488#	7489#
	7490#	7491#	7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7500#	7507#	7508#	7509#	7510#
	7511#	7512#	7513#	7514#	7515#	7516#	7517#	7518#	7535#	7586#	7587#	7589#	7590#	7592#	7593#
	7595#	7596#	7613#	7614#	7642#	7644#	7646#	7647#	7651#	7655#	7660#	7682#	7689#	7692#	7709#
	7725#	7727#	7728#	7747#	7800#	7859#	7899#	7901#	7978#	8005#	8007#	8038#	8071#	8118#	8120#
	8173#	8212#	8225#	8234#	8264#	8287#	8293#	8319#	8324#	8347#	8370#	8374#	8397#	8398#	8449#
	8480#	8491#	8493#	8502#	8504#	8508#	8517#	8526#	8528#	8532#	8541#	8550#	8552#	8553#	8577#

M\$PRIN	2349#	5176#	7105#	7110#	7119#	7371#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#	7392#
	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7411#	7412#	7413#	7414#
	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7430#	7431#	7432#	7433#	7434#	7435#	7436#
	7437#	7438#	7439#	7440#	7441#	7442#	7450#	7451#	7452#	7453#	7454#	7455#	7456#	7457#	7458#
	7459#	7460#	7461#	7469#	7470#	7471#	7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7487#
	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7507#	7508#	7509#
	7510#	7511#	7512#	7513#	7514#	7515#	7516#	7517#	7646#	7651#	7727#	8491#	8493#	8517#	8541#
	8577#	8579#													
M\$PUSH	2337#	5176#	5182#	5261#	5289#	7370#	7376#	7391#	7410#	7429#	7449#	7468#	7486#	7506#	7533#
	7551#	7569#	7680#	7706#	7723#	7746#	7774#	7819#	7927#	8021#	8053#	8086#	8138#	8260#	8264#
	8293#	8343#	8347#	8374#	8417#	8476#	8480#	8508#	8532#	8566#	8604#	8723#	8856#	8862#	8908#
	8978#	8984#	9037#	9102#	9167#	9229#	9279#	9330#	9363#	9427#	9436#	9480#	9499#	9517#	9527#
	9550#	9568#	9585#	9603#	9620#	9668#	9722#	9776#	9830#	9884#	9938#	9992#	10043#	10088#	10243#
	10287#														
M\$PUT	2819#	5176#	7105#	7110#	7119#	7371#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#	7392#
	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7411#	7412#	7413#	7414#
	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7430#	7431#	7432#	7433#	7434#	7435#	7436#
	7437#	7438#	7439#	7440#	7441#	7442#	7450#	7451#	7452#	7453#	7454#	7455#	7456#	7457#	7458#
	7459#	7460#	7461#	7469#	7470#	7471#	7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7487#
	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7507#	7508#	7509#
	7510#	7511#	7512#	7513#	7514#	7515#	7516#	7517#	7646#	7651#	7727#	8491#	8493#	8517#	8541#
	8577#	8579#													
M\$PUT1	2842#	5176#	7105#	7110#	7119#	7371#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#	7392#
	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7411#	7412#	7413#	7414#
	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7430#	7431#	7432#	7433#	7434#	7435#	7436#
	7437#	7438#	7439#	7440#	7441#	7442#	7450#	7451#	7452#	7453#	7454#	7455#	7456#	7457#	7458#
	7459#	7460#	7461#	7469#	7470#	7471#	7472#	7473#	7474#	7475#	7476#	7477#	7478#	7479#	7487#
	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#	7496#	7497#	7498#	7499#	7507#	7508#	7509#
	7510#	7511#	7512#	7513#	7514#	7515#	7516#	7517#	7646#	7651#	7727#	8491#	8493#	8517#	8541#
	8577#	8579#													
M\$RADI	3151#	5176#	7655#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10289#	10290#	10291#	10292#
M\$RBRO	2787#	5176#													
M\$RNRO	2802#	5176#	7613#												
M\$SETS	3071#	5176#	5182#	5261#	5289#	7370#	7376#	7391#	7410#	7429#	7449#	7468#	7486#	7506#	7533#
	7551#	7569#	7680#	7706#	7723#	7746#	7774#	7819#	7927#	8021#	8053#	8086#	8138#	8260#	8264#
	8293#	8343#	8347#	8374#	8417#	8476#	8480#	8508#	8532#	8566#	8604#	8723#	8856#	8862#	8908#
	8978#	8984#	9037#	9102#	9167#	9229#	9279#	9330#	9363#	9427#	9436#	9480#	9499#	9517#	9527#
	9550#	9568#	9585#	9603#	9620#	9668#	9722#	9776#	9830#	9884#	9938#	9992#	10043#	10088#	10243#
	10287#														
M\$STAR	2468#	5176#													
M\$SVC	2746#	5176#	6224	6230	6240	6246	6353	6359	6517	6523	6533	6539	6574	6580	6614
	6620	6628	6634	6780	6785	6804	6810	6819	6825	6834	6840	6849	6855	7105#	7110#
	7119#	7196	7212	7251	7257	7371#	7372#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#
	7385#	7392#	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7404#	7411#
	7412#	7413#	7414#	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7423#	7430#	7431#	7432#
	7433#	7434#	7435#	7436#	7437#	7438#	7439#	7440#	7441#	7442#	7443#	7450#	7451#	7452#	7453#
	7454#	7455#	7456#	7457#	7458#	7459#	7460#	7461#	7462#	7469#	7470#	7471#	7472#	7473#	7474#
	7475#	7476#	7477#	7478#	7479#	7480#	7487#	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#
	7496#	7497#	7498#	7499#	7500#	7507#	7508#	7509#	7510#	7511#	7512#	7513#	7514#	7515#	7516#
	7517#	7518#	7535#	7586#	7589#	7592#	7595#	7613#	7642#	7646#	7647#	7651#	7655#	7660#	7682#
	7689#	7692#	7709#	7725#	7727#	7728#	7747#	7800#	7859	7899	7901#	7978	8005	8007#	8038#
	8071#	8118	8120#	8173	8212	8225	8234#	8264#	8287#	8293#	8319#	8324#	8347#	8370#	8374#
	8397#	8398#	8449#	8480#	8491#	8493#	8502	8504#	8508#	8517#	8526	8528#	8532#	8541#	8550
	8552#	8553#	8577#	8579#	8581#	8698#	8831#	8862#	8891	8892#	8897	8898#	8902#	8908#	8937
	8938#	8943	8944#	8948#	8950#	8984#	9031#	9037#	9084#	9086#	9126	9148	9150#	9188	9203
	9212#	9253	9262#	9303	9312#	9348#	9396#	9436#	9446	9447#	9456	9457#	9475#	9480#	9491

	9492#	9494#	9499#	9510	9512#	9517#	9522#	9527#	9537	9538#	9543	9545#	9550#	9561	9563#
	9568#	9578	9580#	9585#	9596	9598#	9603#	9613	9615#	9620#	9630	9631#	9636	9637#	9642
	9644#	9648#	9693	9702#	9747	9756#	9801	9810#	9855	9864#	9909	9918#	9962	9972#	10016
MSTLAB	10026#	10069#	10122	10134	10196	10223#									
	2739#	5176#	6224#	6230#	6240#	6246#	6353#	6359#	6517#	6523#	6533#	6539#	6574#	6580#	6614#
	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#	6834#	6840#	6849#	6855#	7105#	7110#
	7119#	7196#	7212#	7251#	7257#	7371#	7372#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#
	7385#	7392#	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7404#	7411#
	7412#	7413#	7414#	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7423#	7430#	7431#	7432#
	7433#	7434#	7435#	7436#	7437#	7438#	7439#	7440#	7441#	7442#	7443#	7450#	7451#	7452#	7453#
	7454#	7455#	7456#	7457#	7458#	7459#	7460#	7461#	7462#	7469#	7470#	7471#	7472#	7473#	7474#
	7475#	7476#	7477#	7478#	7479#	7480#	7487#	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#
	7496#	7497#	7498#	7499#	7500#	7507#	7508#	7509#	7510#	7511#	7512#	7513#	7514#	7515#	7516#
	7517#	7518#	7535#	7586#	7589#	7592#	7595#	7613#	7642#	7646#	7647#	7651#	7655#	7660#	7682#
	7639#	7692#	7709#	7725#	7727#	7728#	7747#	7800#	7859#	7899#	7901#	7978#	8005#	8007#	8038#
	8071#	8118#	8120#	8173#	8212#	8225#	8234#	8264#	8287#	8293#	8319#	8324#	8347#	8370#	8374#
	8397#	8398#	8449#	8480#	8491#	8493#	8502#	8504#	8508#	8517#	8526#	8528#	8532#	8541#	8550#
	8552#	8553#	8577#	8579#	8581#	8698#	8831#	8862#	8891#	8892#	8897#	8898#	8902#	8908#	8937#
	8938#	8943#	8944#	8948#	8950#	8984#	9031#	9037#	9084#	9086#	9126#	9148#	9150#	9188#	9203#
	9212#	9253#	9262#	9303#	9312#	9348#	9396#	9436#	9446#	9447#	9456#	9457#	9475#	9480#	9491#
	9492#	9494#	9499#	9510#	9512#	9517#	9522#	9527#	9537#	9538#	9543#	9545#	9550#	9561#	9563#
	9568#	9578#	9580#	9585#	9596#	9598#	9603#	9613#	9615#	9620#	9630#	9631#	9636#	9637#	9642#
	9644#	9648#	9693#	9702#	9747#	9756#	9801#	9810#	9855#	9864#	9909#	9918#	9962#	9972#	10016#
MSTSTL	10026#	10069#	10122#	10134#	10196#	10223#									
	2728#	5176#	6224#	6230#	6240#	6246#	6353#	6359#	6517#	6523#	6533#	6539#	6574#	6580#	6614#
	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#	6834#	6840#	6849#	6855#	7105#	7110#
	7119#	7196#	7212#	7251#	7257#	7371#	7372#	7377#	7378#	7379#	7380#	7381#	7382#	7383#	7384#
	7385#	7392#	7393#	7394#	7395#	7396#	7397#	7398#	7399#	7400#	7401#	7402#	7403#	7404#	7411#
	7412#	7413#	7414#	7415#	7416#	7417#	7418#	7419#	7420#	7421#	7422#	7423#	7430#	7431#	7432#
	7433#	7434#	7435#	7436#	7437#	7438#	7439#	7440#	7441#	7442#	7443#	7450#	7451#	7452#	7453#
	7454#	7455#	7456#	7457#	7458#	7459#	7460#	7461#	7462#	7469#	7470#	7471#	7472#	7473#	7474#
	7475#	7476#	7477#	7478#	7479#	7480#	7487#	7488#	7489#	7490#	7491#	7492#	7493#	7494#	7495#
	7496#	7497#	7498#	7499#	7500#	7507#	7508#	7509#	7510#	7511#	7512#	7513#	7514#	7515#	7516#
	7517#	7518#	7535#	7586#	7589#	7592#	7595#	7613#	7642#	7646#	7647#	7651#	7655#	7660#	7682#
	7689#	7692#	7709#	7725#	7727#	7728#	7747#	7800#	7859#	7899#	7901#	7978#	8005#	8007#	8038#
	8071#	8118#	8120#	8173#	8212#	8225#	8234#	8264#	8287#	8293#	8319#	8324#	8347#	8370#	8374#
	8397#	8398#	8449#	8480#	8491#	8493#	8502#	8504#	8508#	8517#	8526#	8528#	8532#	8541#	8550#
	8552#	8553#	8577#	8579#	8581#	8698#	8831#	8862#	8891#	8892#	8897#	8898#	8902#	8908#	8937#
	8938#	8943#	8944#	8948#	8950#	8984#	9031#	9037#	9084#	9086#	9126#	9148#	9150#	9188#	9203#
	9212#	9253#	9262#	9303#	9312#	9348#	9396#	9436#	9446#	9447#	9456#	9457#	9475#	9480#	9491#
	9492#	9494#	9499#	9510#	9512#	9517#	9522#	9527#	9537#	9538#	9543#	9545#	9550#	9561#	9563#
	9568#	9578#	9580#	9585#	9596#	9598#	9603#	9613#	9615#	9620#	9630#	9631#	9636#	9637#	9642#
	9644#	9648#	9693#	9702#	9747#	9756#	9801#	9810#	9855#	9864#	9909#	9918#	9962#	9972#	10016#
M\$WORD	10026#	10069#	10122#	10134#	10196#	10223#									
	2888#	5176#	5216#	5239#	6224#	6230#	6240#	6246#	6353#	6359#	6517#	6523#	6533#	6539#	6574#
	6580#	6614#	6620#	6628#	6634#	6780#	6785#	6804#	6810#	6819#	6825#	6834#	6840#	6849#	6855#
	7196#	7212#	7251#	7257#	7655#	7859#	7899#	7978#	8005#	8118#	8173#	8212#	8225#	8502#	8526#
	8550#	8891#	8897#	8937#	8943#	9126#	9148#	9188#	9203#	9253#	9303#	9446#	9456#	9491#	9510#
	9537#	9543#	9561#	9578#	9596#	9613#	9630#	9636#	9642#	9693#	9747#	9801#	9855#	9909#	9962#
	10016#	10122#	10134#	10196#	10245#	10246#	10247#	10248#	10249#	10250#	10251#	10252#	10289#	10290#	10291#
M\$XFER	10292#	10321													
OPEN	2410#	5176#													
POINTE	1694#	5176#													
PRINTB	1702#	5176#	5206												
	1768#	5176#	7371	7377	7378	7379	7380	7392	7393	7394	7395	7411	7412	7413	7414
	7430	7431	7432	7433	7434	7450	7451	7452	7453	7469	7470	7471	7487	7488	7489

	7490	7491	7507	7508	7509										
PRINTF	1811#	5176#	7105	7110	7119	7646	7651	7727	8491	8493	8517	8541	8577	8579	
PRINTS	1854#	5176#													
PRINTX	1897#	5176#	7381	7382	7383	7384	7396	7397	7398	7399	7400	7401	7402	7403	7415
	7416	7417	7418	7419	7420	7421	7422	7435	7436	7437	7438	7439	7440	7441	7442
	7454	7455	7456	7457	7458	7459	7460	7461	7472	7473	7474	7475	7476	7477	7478
	7479	7492	7493	7494	7495	7496	7497	7498	7499	7510	7511	7512	7513	7514	7515
	7516	7517													
READBU	1940#	5176#													
REDEF	1949#	5176#	7586	7589	7592	7595									
RFLAGS	1967#	5176#													
SETPRI	1977#	5176#	7682												
SETVEC	1986#	5176#													
SLASH	1998#	5176#													
STARS	2015#	5176#													
SVC	2036#	5175#	5176												
XFER	2299#	5176#													
XFERF	2310#	5176#													
XFERT	2319#	5176#													

. ABS. 037750 000

? ERRORS DETECTED: 1

SAIL:CZDMSA,CZDMSA/CRF/NL:TOC=CZDMP.MLB,CZDMSA.P11
 RUN-TIME: 153 184 15 SECONDS
 RUN-TIME RATIO: 2054/353=5.8
 CORE USED: 19K (37 PAGES)