

DQ11

ITEP OVERLAY
CZDQOD0

AH-8640D-MC

COPYRIGHT 75-78

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA

This microfiche card contains a grid of frames. The leftmost column consists of 12 frames, each containing a small diagram or schematic. The remaining frames in the grid contain text, likely data tables or lists, arranged in a structured format. The text is too small to be legible in this image.

IDENTIFICATION

PRODUCT CODE: AC-8639D-MC
PRODUCT NAME: CZDQODO DQ11 ITEP OVRLY
PROGRAM DATE: OCTOBER 1978
MAINTAINER: DIAGNOSTICS
AUTHORS: R A JONES
 JOHN EGOLF
 FAY BASHAW

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.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

(COPYRIGHT (C) 1975, 1978, BY DIGITAL EQUIPMENT CORPORATION

1.0 ABSTRACT.

THIS PROGRAM IS DESIGNED AS A MAINTENANCE AID FOR FIELD SERVICE PERSONEL. IT WILL VERIFY THE PROPER OPERATION OF A COMPLETE COMMUNICATION LINK FROM ONE PDP-11 SYSTEM TO ANOTHER OR TO A COMMUNICATION TEST CENTER.

THIS PROGRAM MUST BE USED IN CONJUNCTION WITH THE INTERPROCESSOR TEST PROGRAM(DZITP) ON A PDP-11 SYSTEM WITH A DL-11 INTERFACE.

2.0 REQUIREMENTS.

2.1 EQUIPMENT

- A. PDP-11 SYSTEM WITH 4K OF CORE.
- B. A DQ11 COMMUNICATION INTERFACE.

2.2 STORAGE.

4K OF CORE

3.0 LOADING PROCEDURE

THIS PROGRAM IS IN ABSOLUTE FORMAT.
THE ABS LOADER MUST BE USED TO LOAD THE PROGRAM.

4.0 OPERATING PROCEDURES.

- A. TWO METHODS OF ENTERING PARAMETERS ARE PROVIDED
 - 1. LOAD ADDRESS 200 AND START TO ENTER PARAMS FROM CONSOLE TTY, PROCEED TO SECTION B.
 - 2. LOAD ADDRESS 200 AND SET SWITCH REGISTER BIT 15 BEFORE STARTING TO ENTER PARAMS FROM CONSOLE SWITCHES, PROCEED TO SECTION C.
- *THE PROGRAM MAY BE RESTARTED AT LOC 204 (ONCE PARAMETERS HAVE ALREADY BEEN SELECTED)

B. CONSOLE DIALOGUE PARAMETER INPUT (CURRENT VALUES FOR PARAMETERS ARE FOUND IN OVERLAY)

- 1. THE PROGRAM WILL TYPEOUT THE NAME OF THE VARIABLE OVERLAY.
 - A. IF YOU WISH TO SETUP JUST THE INDICATED OVERLAY, TYPE A CARAGE RETURN
 - B. IF YOU WISH TO SETUP A DN11, TYPE IN DN.
 - C. IF YOU WISH TO SETUP A DM11BB, TYPE IN DMB.

IF DN OR DMB WAS TYPED IN STEP 1 ABOVE THEN THE BUS ADDRESS, VECTOR ETC. REFERED TO IN STEPS 2 THRU 7, PERTAIN TO THE DN11 OR DMBB.

- 2. THE PROGRAM WILL TYPE THE DEFAULT BUS ADDRESS OF THE INTERFACE UNDER TEST.
 - A. TYPE A CAR. RETURN TO USE DEFAULT BUS ADDRESS
 - B. TYPEIN ACTUAL BUS ADDRESS
- 3. THE PROGRAM WILL TYPE OUT THE DEFAULT VECTOR ADDRESS
 - A. TYPE A CAR. RETURN TO USE DEFAULT ADDRESS
 - B. TYPEIN ACTUAL VECTOR ADDRESS
- 4. THE PROGRAM WILL TYPE OUT THE DEFAULT INTERFACE PRIORITY
NOTE: 200=PRIO 4, 240=PRIO 5, 300=PRIO 6, ETC.

- A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
 - B. TYPEIN ACTUAL VALUE
5. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#1
IF REQUIRED BY THE ISR.(SEE SECT. 10.0 IN OVERLAY LISTING FOR PARAMETER DESCRIPTION)
- A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
 - B. TYPEIN ACTUAL VALUE
6. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#2
IF REQUIRED BY THE ISR.
- A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
 - B. ENTER ACTUAL VALUE
7. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#3
IF REQUIRED BY THE OVERLAY.
- A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
THE DN-11 WILL USE PARAM #3 AS THE # TO DIAL.
IF USING A MODEM WITHOUT AUTOMATIC HANDSHAKING,
THE NUMBER MUST TERMINATE WITH A
'END-OF-NUMBER' CHARACTER (:).
 - B. ENTER ACTUAL VALUE.
8. THE PROGRAM WILL RETURN TO STEP B1 IF THIS SETUP
WAS FOR DN11 OR DM11BB.
9. THE PROGRAM WILL REQUEST THAT SWITCH REGISTER BE SET.
- A. SETUP SWITCH REGISTER AS SPECIFIED IN STEP D.
AND TYPE A CAR. RETURN.

NOTE: IF ANY OF THE ABOVE ITEMS 2 THRU 7 WERE CHANGED BY ENTERING
NEW VALUES, THE NEW VALUE BECOMES THE DEFAULT VALUE FOR SUBSEQUENT
RESTARTS OF THE PROGRAM.

MANUAL PARAMETER INPUT FROM SWITCH REGISTER

1. THE PROGRAM HALTS FOR ISR(INTERFACE SERVICE ROUTINE) SPECIFICATION

SWR14=SETUP DM-11B ISR

SWR13=SETUP DN-11 ISR

SWR=000000=SETUP VARIABLE ISR

2. THE FOLLOWING HALTS ARE REPEATED FOR EACH ISR SPECIFIED.

SETUP SEQUENCE IS: DN11,DM11-~~BB~~ THEN VARIABLE OVERLAY. (EACH ENTRY SET SWITCHES THEN HIT CONTINUE.)

A. HALT FOR BUS ADDRESS OF INTERFACE

B. HALT FOR VECTOR ADDRESS OF INTERFACE

C. HALT FOR PRIORITY OF INTERFACE

D. HALT FOR INTERFACE PARAM #1 (SEE SECT. 10.0 IN OVERLAY LISTING FOR PARAMETER DESCRIPTION)

E. HALT FOR INTERFACE PARAM #2 (DN11 AND DMBB PARAMETERS ARE DISCUSSED IN SECT. 10.0 OF THE MONITOR.)

F. GO BACK TO STEP A IF THIS SETUP WAS FOR DN OR DMB.

3. HALT FOR OPERATIONAL SWITCH SETTINGS. (SEE STEP D.)

A. PRESS CONTINUE TO START TESTING

BEFORE ATTEMPTING TO RUN THIS PROGRAM, THE OPERATOR MUST ACCERTAIN THE COMPLETE COMMUNICATION LOOP AND PROCEDURES TO BE USED, INCLUDING THE TYPE OF MODEMS, THE TYPE OF INTERFACE BEING USED AT THE OTHER CPU AND THE MODES OF OPERATION, DATA AND PARAMETERS TO BE USED AT EACH CPU.

THIS WILL REQUIRED VOCAL COMMUNICATION WITH THE OPERATOR AT THE OTHER CPU UNLESS ITS CONFIGURATION AND OPERATION ARE FIXED AS A TEST CENTER.

AFTER DETERMINING THAT THE EQUIPMENTS ARE COMPATIBLE AND AGREEING ON THE MODE AND VARIABLE PARAMETERS TO BE USED, THE SYSTEM WHICH IS TO RECEIVE DATA FIRST SHOULD BE LOADED AND STARTED. IF THE MODEM BEING USED ON THIS SYSTEM HAS AN AUTOMATIC ANSWER FEATURE, IT SHOULD BE ENABLED.

THE SYSTEM WHICH IS TO TRANSMIT FIRST SHOULD THEN BE LOADED AND STARTED AND THE CONNECTION ESTABLISHED EITHER MANUALLY OR AUTOMATICALLY (VIA DN-11).

D. OPERATIONAL SWITCH SETTINGS.

SW15=1 HALT ON ERROR

SW14=1 SINGLE PASS

SW14 HAS NO EFFECT IF SW04=0

SW13=1 INHIBIT ERROR TYPEOUTS

SW12=1 INHIBIT ALL TYPEOUTS EXCEPT ERRORS

IF SW12=0 AND SW04=1 END PASS IS TYPED

AND TRANSMITTED/RECEIVED DATA IS TYPED.

SW11=1 USE PREVIOUSLY SPECIFIED DATA

SW10=1 DATA SELECT (WITH SW09)

SW09=1 DATA SELECT (WITH SW10)

00=1 GET DATA FROM OPERATOR

01=1 TEST MESSAGE #1 (\$A QUICK BROWN FOX)

10=1 TEST MESSAGE #2 (\$B NUMERICS)

11=1 TEST MESSAGE #3 (\$C COMTEST/QUICK BROWN FOX/NUMERICS)

SW08=1 TRANSMIT RECEIVED DATA (INTERNAL LOOPBACK MODE)

SW07=1 DO NOT TEST RECEIVED DATA

SW06=1 MONITOR TRANSMITTED DATA ON CONSOLE TTY.*

SW05=1 MONITOR RECEIVED DATA ON CONSOLE TTY.*

* IN MANY CASES, NOT ALL DATA WILL APPEAR ON THE CONSOLE

TTY. THIS IS ESPECIALLY TRUE WHEN THE COMM INTERFACE IS

RUNNING AT A FASTER BAUD THAN THE CONSOLE, BUT EVEN AT EQUAL

OR SLOWER BAUDS, ALL CHARACTERS MAY NOT APPEAR ON THE CONSOLE.

SW04=1 RETURN TO MONITOR FOR END PASS

WHEN SW04=0 PROGRAM LOOPS IN THE OVERLAY NEVER RETURNING TO THE MONITOR.

SW03 1 INTERNAL LOOPBACK MODE

SW02=1 EXTERNAL LOOPBACK MODE

SW01=1 ONE-WAY-IN MODE

SW00 1 ONE-WAY-OUT MODE

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC. 176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(IE) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

1. <CR> IF NO CHANGES ARE TO BE MADE
2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE ;LAST DIGIT FOLLOWED BY <CR>.
3. ^U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING ^G (CNTL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

IF OPERATOR SPECIFIED DATA WAS INDICATED, THE PROGRAM WILL TYPE A REQUEST FOR THE DATA. DATA MAY BE ENTERED AS ASCII CHARACTERS OR OCTAL CODE. TYPE IN THE DATA TERMINATED WITH A CR. OCTAL CODE MAY BE ENTERED BY TYPING AN ^ (UP ARROW) FOLLOWED BY THE OCTAL CODE (IN THE RANGE 000 TO 377) SEPERATED BY SPACES AND TERMINATED BY ^ (UP ARROW).
I.E. ABCD^ 000 123 377^ EFG (CAR.RETURN)

A TYPICAL SWITCH SETTING FOR HALF-DUPLEX=003150 THIS SETTING USES INTERNAL LOOPBACK MODE, LOOPS IN OVERLAY, MONITORS TRANSMITTED AND RECEIVED DATA ON THE CONSOLE TTY, AND TESTS RECEIVED DATA USING TEST MESSAGE #3.

A TYPICAL SWITCH SETTING FOR FULL-DUPLEX=003144 THIS SETTING IS THE SAME AS ABOVE EXCEPT IT USES THE EXTERNAL LOOPBACK MODE.

ALL STANDARD MESSAGES (TEST MESSAGES 1-3) ARE PRECEDED BY 2 FILL CHARACTERS (177), AND ARE FOLLOWED BY A CR (015), LF (012), RECEIVE TERMINATING CHARACTER (001), 4 FILLS (177), AND A TRANSMIT TERMINATING CHARACTER (000). DURING TRANSMISSION, WHEN A 000 CHARACTER IS SEEN THE TRANSMISSION IS STOPPED. DURING RECEPTION, WHEN A 001 CHARACTER IS RECEIVED, THE RECEIVER IS SHUT OFF. IF THE MESSAGE WAS INPUTED BY THE OPERATER, THE TERMINATING CHARACTERS ARE ADDED.

TEST MODES

INTERNAL LOOPBACK MODE

1. THE OVERLAY WAITS TO RECEIVE A MESSAGE (TERMINATED BY <001>)
2. VERIFIES THE DATA AGAINST THE DATA SELECTED BY SW09 AND SW10 (SW7=0)
3. TRANSMIT THE DATA SELECTED BY SW09 AND SW10 (SW8 0) OR TRANSMIT THE RECEIVED DATA (SW8=1)
4. RETURNS TO MONITOR FOR 'END PASS' (SW4=1) OR GO TO STEP 1. (SW4=0)

EXTERNAL LOOPBACK MODE

1. THE OVERLAY SETS REQUEST TO SEND
2. WAIT FOR CLEAR TO SEND
3. TRANSMITS THE SELECTED DATA
4. RESETS REQUEST TO SEND
5. WAIT FOR MESSAGE TO BE RECEIVED
6. VERIFIES THE DATA (SW07=0)
7. RETURNS TO MONITOR FOR 'END PASS'. (SW04=1) OR GO TO STEP 1 (SW04=0)

ONE-WAY-IN MODE

1. THE OVERLAY WAITS FOR MESSAGE TO BE RECEIVED.
2. VERIFIES THE DATA (SW07 0)
3. RETURNS TO MONITOR FOR 'END PASS' (SW04=1) OR GO TO STEP 1 (SW04=0)

ONE-WAY-OUT MODE

1. THE OVERLAY SETS REQUEST TO SEND
2. WAITS FOR CLEAR TO SEND
3. TRANSMITS SELECTED DATA
4. RETURNS TO MONITOR FOR 'END PASS'. (SW04=1) OR GO TO STEP 1 (SW04=0)

- E. THE OVERLAY IS THEN ENTERED AND A CONNECTION ESTABLISHED EITHER MANUALLY OR AUTOMATICALLY.

IF ONE-WAY-IN OR INTERNAL LOOPBACK MODES ARE SELECTED.
THE OVERLAY WILL SET DATA TERMINAL READY AND WAIT FOR DATA.

IF ONE-WAY-OUT OR EXTERNAL LOOPBACK MODES WERE SELECTED.
THE OVERLAY WILL SET DATA TERMINAL READY AND REQUEST TO SEND.
THE OVERLAY WILL THEN WAIT FOR CLEAR TO SEND BEFORE ATTEMPTING TO TRANSMIT DATA.

THE PROGRAM WILL PRINTOUT A 'WAITING FOR CLEAR TO SEND'
MESSAGE AND THE CONTENTS OF THE XMIT CSR EVERY 60 SECS.
UNTIL CLEAR TO SEND IS ASSERTED.

F. IF SW04=0 THE OVERLAY WILL CONTINUE TO TRANSMIT/RECEIVE DATA.

IF SW04=1 THE OVERLAY WILL RETURN TO THE MONITOR AND TYPE 'END PASS'.

IF BOTH SW04=1 AND SW14=1, THE PROGRAM WILL REQUEST NEW INTERFACE PARAMS AFTER ONE PASS OF THE SELECTED TEST MODE.

TEST EXECUTION MAY BE INTERRUPTED BY TYPING THE FOLLOWING CHARACTERS ON THE CONSOLE TTY.

LINE FEED = RESTART PROGRAM AT LOCATION 200.

QUESTION MARK = PRINTOUT FIRST 8 WORDS OF INPUT BUFFER.(ASCII)

THEN TYPE EITHER:

*WXXXXXX TO PRINTOUT THE 8 WORDS AT LOC XXXXXX.

*BXXXXXX TO PRINTOUT THE 16 BYTES AFTER LOC XXXXXX.

*C TO CONTINUE

PROGRAM MUST BE RESTARTED AT 200 AFTER PRINTING.
CARRIAGE RETURN - RESTART AT REQUEST FOR NEW OPERATIONAL SWITCHES.

5.0 PROGRAM AND/OR OPERATOR ACTION

IF THE OPERATOR WISHES TO MANUALLY EXAMINE THE TRANSMIT OR RECEIVE BUFFERS, DO THE FOLLOWING; TO FIND THE STARTING ADDRESS OF THE RECEIVE BUFFER, LOAD ADDRESS 11020 AND EXAMINE. TO FIND THE STARTING ADDRESS OF THE TRANSMIT BUFFER, LOAD ADDRESS 11022 AND EXAMINE.

5.1 NORMAL HALTS SEE SECTION 4.

6.0 ERRORS

6.1 ERROR REPORTING

THE ONLY ERROR REPORT FROM THE CONTROL PROGRAM OCCURS IF THE INTERFACE SPECIFIED IS NOT LOADED.

IF DATA IS RECEIVED AND SWITCH 7 (NO DATA COMPARE) IS RESET, THE DATA WILL BE COMPARED AGAINST THE PRESELECTED DATA AFTER A LINE FEED CHARACTER IS RECEIVED. IF THERE IS A MISMATCH, THE FOLLOWING ERROR REPORT IS PRINTED:

RECEIVED DATA=RRRRRR
DATA SHOULD BE TTTTTT
DATA COMPARE ERROR; BAD DATA-BBB GOOD DATA GGG

WHERE RRRRRR IS THE RECEIVE BUFFER (UP TO 512 CHARACTERS)
TTTTTT IS THE TRANSMIT BUFFER (UP TO 512 CHARACTERS)
BBB IS THE BAD DATA CHARACTER
GGG IS THE GOOD DATA CHARACTER

IF THE INTERFACE DETECTS A DATA ERROR, THE FOLLOWING
WILL BE PRINTED BEFORE THE DATA IS COMPARED:

THERE WAS A RECEIVER ERROR. RECEIVER DATA REGISTER =XXXXXX

WHERE XXXXXX IS THE CONTENTS OF THE RECEIVER DATA REGISTER
THE LOW BYTE IS THE DATA, AND THE HIGH BYTE IS THE ERROR BITS.

IF A RECEIVE TERMINATING CHARACTER<001> IS NOT DETECTED
WITHIN 512 CHARACTERS A 'BUFFER FULL' PRINTOUT WILL OCCUR.

7.0 RESTRICTIONS

THE OPERATION OF THIS PROGRAM REQUIRES COORDINATION BETWEEN
THE OPERATOR AND THE OPERATOR OF ANOTHER PDP-11 SYSTEM
UNLESS ONE OF THE SYSTEMS IS ALWAYS OPERATING IN A FIXED
MODE. THE FOLLOWING TABLE LISTS THE VALID COMBINATIONS:

CPU #1	CPU #2
ONE-WAY-OUT	ONE-WAY-IN
ONE-WAY-IN	ONE-WAY-OUT
EXTERNAL-LOOPBACK	INTERNAL-LOOPBACK
INTERNAL-LOOPBACK	EXTERNAL-LOOPBACK
EXTERNAL-LOOPBACK	EXTERNAL-LOOPBACK (FULL DUPLEX)

WHEN THE COMMUNICATION LINK INVOLVES MODEMS THE FOLLOWING
RESTRICTION APPLY:

IF RUNNING IN FULL DUPLEX MODE BOTH SYSTEMS
MUST BE IN EXTERNAL LOOP BACK MODE.

BOTH SYSTEMS SHOULD BE RUNNING IDENTICAL ROUTINES.

EXAMPLE:
SWITCHES 14,13,7,4 SHOULD BE THE SAME
ON BOTH CPU S

IF PROGRAM IS WAITING IN A SCAN ROUTINE AND TYPES OUT
A 'WAITING MESSAGE',IF AN INCOMING MESSAGE STARTS DURING
THE TYPE OUT, IT WILL BE LOST BECAUSE THE TYPEOUT PRIORITY
IS AT LEVEL 7. THIS WILL RESULT IN OVERRUN OR SILO OVER-
RUN ERRORS, DEPENDING ON THE DEVICE.TO AVOID THIS SITUATION
RUN WITH SWITCH 13 UP. IF OVERRUN DOES OCCURE DURING A
TYPEOUT THE PROGRAM SHOULD BE RESTARTED.

IF USING AN ASYNCHRONOUS DEVICE, MODEMS AND THE
MAYNARD TEST STATION AND INITIALIZE DOES NOT CLEAR THE
CONNECTION (EXAMPLE THE DJ11) IF THE PROGRAM IS RESTARTED
IN THE MIDDLE OF A MESSAGE AT LOC 204 OR BY HITTING CR
AN IMMEDIATE ERROR MESSAGE FROM MAYNARD WILL BE RE-

CEIVED. THIS IS BECAUSE THE TEST STATION IS STILL LOOKING FOR THE REST OF THE INTERRUPTED MESSAGE. TO AVOID THIS ERROR, RESTART PROGRAM ONLY AT THE END OF THE MESSAGE CURRENTLY BEING TRANSMITTED.

8.0 MISCELLANEOUS

ITEP WAS CHECKED OUT USING THE FOLLOWING BELL TELEPHONE MODEMS.
201A (HALF-DUPLEX SYNCHRONOUS 2000 BAUD)
202C (HALF-DUPLEX ASYNCHRONOUS 1200 BAUD)
103A (FULL-DUPLEX ASYNCHRONOUS 110 BAUD)

9.0 PROGRAM DESCRIPTION

9.1 THE DQ11 INTERFACE SERVICE PARAMS ARE SETUP, AS SPECIFIED BY THE OPERATOR, BY THE ITEP CONTROL PROGRAM.

TIME: PROVIDES A MEANS OF MEASURING ELAPSED TIME. IT IS INCREMENTED EVERY SECOND BY A CLOCK INTERRUPT ROUTINE IN ITEP.

9.2 WHEN THE OVERLAY IS FIRST ENTERED BY ITEP AT LOCATION START:, THE CONTENTS OF THE SWITCH REGISTER ARE STORED IN REGISTER 0. THE MODE AND DATA SELECTIONS ARE FIXED AT THIS TIME AND CANNOT BE ALTERED WITHOUT RETURNING TO THE CONTROL PROGRAM. THE INTERRUPT VECTORS AND VARIABLES ARE THEN SETUP. THE SELECTED ROUTINE DETERMINED BY THE MODE IS THEN ENTERED

9.3 THE OVERLAY THEN LOOPS IN ROUTINES: \$OWI ,IF 'ONE WAY IN' MODE WAS SELECTED. \$OWO,IF 'ONE WAY OUT' MODE WAS SELECTED. \$ILB, IF 'INTERNAL LOOP BACK' MODE WAS SELECTED. \$XLB,IF 'EXTERNAL LOOP BACK' WAS SELECTED.

9.31 \$OWI: IN THIS ROUTINE THE RECEIVER IS INITIALIZED AND PROGRAM LOOPS WAITING FOR THE RECEIVER TO FINISH. IF NOTHING IS RECEIVED FOR 60 SECS A 'WAITING' MESSAGE IS TYPED. WHEN THE RECEIVER IS DONE, THE PROGRAM CHECKS DATA IF SWITCHES PERMIT, AND TYPES END PASS DEPENDING ON SWITCH SETTINGS.

9.32 \$OWO: THE TRANSMITTER IS INITIALIZED AND PROGRAM LOOPS WAITING FOR TRANSMITTER TO FINISH, A 'WAITING' MESSAGE IS TYPED EVERY 60 SECS IF THERE IS NO ACTION. WHEN THE TRANSMITTER IS DONE, THE PROGRAM EITHER LOOPS BACK TO \$OWO OR TYPES END PASS DEPENDING ON SWITCH SETTINGS.

9.33 \$ILB: THE RECEIVER IS INITIALIZED AND PROGRAM LOOPS WAITING FOR RECEIVER TO FINISH, A 'WAITING' MESSAGE IS TYPED EVERY 60 SEC IF NO ACTION. WHEN RECEIVER IS DONE PROGRAM CHECKS DATA IF SWITCH SETTINGS PERMIT, AND END PASS IS TYPED IF SWITCH SETTINGS PERMIT. THEN THE TRANSMITTER IS INITIALIZED, A 'WAITING' MESSAGE IS TYPED EVERY 60 SEC IF NO ACTION. WHEN TRANSMITTER IS DONE PROGRAM RETURNS TO START OF ROUTINE. (\$ILB)

9.34 \$XLB: IF IN HALF DUPLEX THE TRANSMITTER IS INITIALIZED, A 'WAITING MESSAGE IS TYPED EVERY 60 SEC IF THERE IS NO ACTION

WHEN THE TRANSMITTER IS DONE THE RECEIVER IS INITIALIZED
,A 'WAITING' MESSAGE IS TYPED EVERY 60 SEC IF THERE IS NO ACTION.
WHEN THE RECEIVER IS DONE,DATA IS CHECKED IF SWITCH SETTINGS
PERMIT AND END PASS IS TYPED IF SWITCHES ALLOW.THE PROGRAM NOW
REPEATS CYCLE STARTING AT \$XLB.
IF IN FULL DUPLEX THE RECEIVER AND TRANSMITTER ARE INITIALIZED
, A 'WAITING' MESSAGE IS TYPED EVERY 60 SEC IF THERE IS NO
ACTION. WHEN BOTH THE RECEIVER AND TRANSMITTER ARE DONE,DATA IS
CHECKED, END PASS IS TYPED AND PROGRAM LOOPS TO \$XLB DEPENDING
ON THE SWITCH SETTINGS.

- 9.4 THE RETURN TO MONITOR ROUTINE FOR END PASS AT EOP:
LOCKS OUT INTERRUPTS AND SAVES THE TRANSMITTER INTERRUPT ENABLE
BIT AND ALL GENERAL REGISTERS. IT THEN RETURNS TO THE MONITOR
TO TYPE 'END PASS'. THE MONITOR CHECKS SW14 IF UP IT RETURNS
TO ENTER:, OTHERWISE IT RESTARTS THE PROGRAM.
- 9.5 ENTER: IS ENTERED FROM THE MONITOR AFTER TYPEING 'END PASS',
IT RESTORES THE GENERAL REGISTERS AND THE TRANSMITTER CSR
AS SAVED IN EOP. THE DELAY FLAG IS SET AND PROGRAM RETURNS TO
THE SCAN ROUTINE(OWO,OWI,ILB,XLB) WHERE IT CAME FROM.
- 9.6 THE INITIALIZE TRANSMIT SUBROUTINE AT STARTX:
SETS UP THE INTERFACE AND POINTERS NECESSARY TO
INITIATE A TRANSMIT OPERATION.
AFTER SETTING 'DATA TERMINAL READY' AND 'REQUEST TO SEND' A CHECK
IS MADE ON PARAM2 TO DETERMINE IF HALF DUPLEX OPERATION
WAS SELECTED BY THE OPERATOR. IF IT WAS, THE
SUBROUTINE WAITS FOR CLEAR TO SEND.
A 'WAITING FOR CLEAR TO SEND' PRINTOUT OCCURS
EVERY 30 SECONDS UNTIL CLEAR TO SEND IS ASSERTED.
- 9.7 THE INITIALIZE RECEIVED SUBROUTINE AT STARTR:
SETS UP THE INTERFACE AND POINTERS NECESSARY TO
RECEIVE A MESSAGE.
- 9.8 THE TRANSMIT INTERRUPT SERVICE ROUTINE,
AT XISR:, IS ENTERED VIA TRANSMIT INTERRUPTS
FROM THE INTERFACE.
A TEST IS MADE TO SEE IF THE LAST CHARACTER
TRANSMITTED WAS A NULL (ALL ZEROS) CHARACTER.
IF IT WAS; THE TRANSMIT LOGIC IN THE INTERFACE
IS RESET AND THE TRANSMIT COMPLETE FLAG IS SET.
AT XISR1: THE NEXT CHARACTER IS TRANSMITTED
AND PRINTED ON THE TTY IF THE MONITOR TRANSMIT
SWITCH IS SET.
- 9.9 THE RECEIVE INTERRUPT SERVICE ROUTINE
,AT RISR:, IS ENTERED VIA RECEIVER INTERRUPTS
FROM THE INTERFACE.
THE RECEIVED CHARACTER IS STORED IN
THE INPUT BUFFER AND PRINTED ON THE TTY IF
THE MONITOR RECEIVER SWITCH IS SET.
IF THE INPUT BUFFER IS FULL, A 'BUFFER FULL'
PRINTOUT WILL OCCUR. THIS INDICATES THAT A
LINE FEED CHARACTER WAS NOT RECOGNIZED

IN THE RECEIVED DATA (WITHIN 1000 CHARACTERS).
IF THE RECEIVED CHARACTER IS A LINE FEED,
THE RECEIVED LOGIC IS RESET AND THE
RECEIVE COMPLETE FLAG IS SET.
IF A 'RECEIVE ERROR' IS DETECTED AT RISR:, THE
CSR AND DBR WILL BE SAVED AND PRINTED OUT
AFTER THE COMPLETE MESSAGE HAS BEEN RECEIVED.

- 9.10 THE DATA TEST SUBROUTINE AT TESTD: IS
ENTERED AFTER A COMPLETE MESSAGE HAS BEEN
RECEIVED.
IF A 'RECEIVE ERROR' HAD BEEN DETECTED,
THE CONTENTS OF THE 'RECEIVE BUFFER' AT THE
TIME THE ERROR OCCURRED WILL BE PRINTED.
THE DATA IS COMPARED UNTIL A 'ALL ZEROS'
CHARACTER IS RECOGNIZED. 'FILL' (ALL ONES)
CHARACTERS ARE IGNORED. IF A MISMATCH
IS DETECTED, THE COMPLETE CONTENTS OF THE
INPUT BUFFER AND GOOD DATA IS PRINTED.

DQ11 RESTRICTIONS

THE DQ11 HAS TWO MODES OF OPERATION IN ITEP, NORMAL (WHICH
INTERUPTS EVERY TWO CHARACTERS), AND HI-BAUD (ONE INTERUPT PER
MESSAGE, THE WHOLE MESSAGE IS TRANSMITTED OR RECEIVED ON A
COMPLETE WORD COUNT.). WHEN IN HI-BAUD MODE, DATA CANNOT BE
MONITORED ON THE CONSOLE TTY. IF SW 5=1 (MONITOR RECEIVED DATA)
AN 'R' WILL BE TYPED AFTER THE WHOLE MESSAGE IS RECEIVED.
IF SW 6=1 (MONITOR TRANSMIT DATA) A 'T' WILL BE TYPED AFTER THE
WHOLE MESSAGE IS TRANSMITTED. IN NORMAL MODE, EVEN IF THE CONSOLE
TTY IS FASTER OR THE SAME BAUD AS THE DQ11, NOT EVERY CHARACTER
CAN BE TYPED BECAUSE OF THE TWO CHARACTER PER INTERUPT
OPERATION, IN THIS CASE EVERY OTHER CHARACTER WILL BE TYPED
IF DATA MONITORING IS SELECTED. HI-BAUD MODE (SELECTABLE IN PARAM#2)
SHOULD BE USED FOR BAUDS OF 40,000 OR HIGHER. NORMAL MODE IS
SUFFICIENT FOR BAUDS LOWER THAN 40,000 .

10.0 PARAMETERS FOR THE DQ11

PARAM#1 IS NOT USED (0)

PARAM#2 (LOW BYTE)

BIT 0
BIT 1

FULL DUPLEX (1), DEFAULT= HALF DUPLEX (0)
HI-BAUD (1), DEFAULT= NORMAL (0)

(BITS 0,1 ARE NOT LOADED INTO ANY DQ11 REGISTERS, THEY ARE ONLY SOFTWARE FLAGS)

PARAM#2 (HIGH BYTE) IS LOADED INTO THE SYNC REGISTER.
BITS 8-15 SYNC CHARACTER, DEFAULT= 26 (26)

PARAM#3 IS NOT USED (177777)

599
 600
 601
 602
 603 011000 011000
 604 011000 050504 000040
 605 011004 160010
 606 011006 000300
 607 011010 000240
 608 011012 000000
 609 011014 013000
 610 011016 177777
 611 011020 000000
 612 011022 000000
 613 011024 000000
 614 011026 000000
 615 011030 000000
 616 011032 000000
 617 011034 000000
 618 011036 011106
 619 011040
 620 011040 000
 621 011041
 622 011041 001
 623 011042 000000
 624 011044 177570
 625 011046 177570
 626
 627
 628
 629
 630 000000
 631 100000
 632 040000
 633 020000
 634 020000
 635
 636 011050 000000
 637 011052 000000
 638 011054 000000
 639 011056 000000
 640 011060 000000
 641
 642 011062 000000
 643 011064 000000
 644 011066 000000
 645 011070 000000
 646 011072 000000
 647 011074 000000
 648
 649 011076 177560
 650 011100 177562
 651 011102 177564
 652 011104 177566
 653
 654 000001

```

:*****
:      DQ11 INTERFACE SERVICE PARAMS
:*****
DQ11:  .ASCIZ  /DQ /      ;ISR NAME
BA:    160010             ;BUS ADDRESS
RIV:   300                ;VECTOR ADDRESS
PRIOR: 240                ;PRIORITY
PARAM1: 0                 ;PARAM #1
PARAM2: 013000            ;PARAM #2
PARAM3: 177777            ;PARAM #3
IRDA:  .WORD  0           ;INITIAL READ DATA ADDRESS
IXDA:  .WORD  0           ;INITIAL XMIT DATA ADDRESS
SETTLE: .WORD  0          ;LINE SETTLE DELAY FLAG
:
B2016: .WORD  ?           ;ADDR OF BIN TO OCT TYPE ROUTINE
TIME:  .WORD  0           ;TIMER
:
TX.TERM: .WORD  START    ;ADDR OF START OF PROGRAM
RX.TERM: .BYTE  000      ;TRANSMITTER TERMINATING CHAR.
FLAG:   .BYTE  001      ;RECEIVER TERMINATING CHAR.
SWR:    177570
DISPLAY:177570

:*****
:      CONSTANTS + WORKING STORAGE
:*****
STAT=R0
XFLG=100000             ;XMIT COMPLETE FLAG
RFLG=40000              ;RCV COMPLETE FLAG
DSFLG=20000             ;DATA SET STATUS CHANGE FLAG
BIT13=20000            ;INHIBIT PRINTOUTS

SXCSR:  0                ;SAVED XMIT CSR
SRCSR:  0                ;SAVED RCV CSR
ERCSR:  0                ;RCV CSR SAVED ON ERROR
ERDBR:  0                ;RCV DATA REG SAVED ON ERROR
DSSTAT: 0                ;RCV CSR SAVED ON DS CHANGE

TXWC:0
RXWC:0
XCC:   0                 ;XMIT CHAR COUNT
RCC:   0                 ;RCV CHAR COUNT
RDA:   0                 ;RCV DATA ADDR.
XDA:   0                 ;XMIT DATA ADDR.

TKS:   177560
TKB:   177562
TPS:   177564
TPB:   177566

FULL.DUPLEX=000001
    
```

```

655
656
657
658 011106 000240
659 011110 017700 177730
660 011114 042700 177400
661 011120 013702 011006
662 011124 012722 015052
663 011130 013722 011010
664 011134 012722 014230
665 011140 013722 011010
666 011144 013704 011004
667 011150 013714 011012
668 011154 013702 011014
669 011160 042702 000001
670 011164 010264 000002
671 011170 012703 000017
672 011174 112764 000014 000005
673 011202 110364 000001
674 011206 005064 000006
675 011212 005303
676 011214 100372
677 011216 005014
678 011220 005064 000002
679 011224 005064 000004
680 011230 012703 000020
681 011234 052764 010000 000004
682 011242 042764 060000 000004
683 011250 005064 000006
684 011254 105264 000005
685 011260 005303
686 011262 001364
687 011264 112764 000012 000005
688 011272 012764 000040 000006
689 011300 005037 014220
690 011304 052764 000010 000002
691 011312 005037 011032
692 011316 005037 013274
693
694
695
696
697
698
699
700
701 011322
702 011322 013702 011022
703 011326 005003
704 011330 005203
705 011332 123722 011040
706 011336 001374
707 011340 005403
708 011342 010337 011062
709 011346 013702 011022
710 011352 005003
    
```

```

:*****
:      DQ11-X INTERFACE SERVICE ROUTINE
:*****
START:  NOP
        MOV     @SWR, R0      ;SETUP MODE IN R0
        BIC     #177400, R0   ;STRIP JUNK
        MOV     RIV, R2      ;SETUP
        MOV     #RISR, (R2)+  ;INTERRUPT
        MOV     PRIOR, (R2)+ ;VECTORS
        MOV     #XISR, (R2)+
        MOV     PRIOR, (R2)+
        MOV     BA, R4       ;SETUP BUS ADDR INDEX
        MOV     PARAM1, @RCSR ;SETUP VARIABLES
        MOV     PARAM2, R2
        BIC     #0001, R2
        MOV     R2, XCSR(R4) ;IN CSR'S
        MOV     #17, R3      ;SET-UP TO CLEAR ALL 16 LOCATIONS
        MOV     #14, REG(RCSR) ;OF THE SEQUENCE CONTROL REGISTER
1$:     MOV     R3, 1(RCSR)   ;CLEAR A LOCATION OF THE
        CLR     SEC(RCSR)    ;SEQUENCE CONTROL REGISTER
        DEC     R3          ;HAVE ALL 16 LOCATIONS BEEN CLEARED?
        BPL     1$         ;IF NO THEN BRANCH, OTHERWISE PROCEED ON
        CLR     @RCSR      ;CLEAR THE RX CSR
        CLR     XCSR(R4)    ;CLEAR TX CSR
        CLR     ERR(R4)     ;CLEAR THE ERROR REGISTER
        MOV     #16, R3     ;SET COUNTER
2$:     BIS     #BIT12, ERR(R4) ;SET WRITE ENABLE
        BIC     #60000, ERR(R4) ;CLEAR EXT MEM BITS
        CLR     SEC(R4)     ;CLEAR THE SECONDARY REGISTER
        INCB   REG(R4)     ;GET NEXT REGISTER
        DEC     R3         ;DONE YET??
        BNE     2$        ;KEEP CLEARING
        MOV     #12, REG(R4) ;SELECT THE MISC REGISTER
        MOV     #BIT5, SEC(R4) ;ISSUE A MASTER CLEAR.
        CLR     ERRORS
        BIS     #BIT3, XCSR(R4) ;ENABLE ERROR INTERUPTS
        CLR     TIME       ;RESET TIMER
        CLR     DELAY      ;RESET DELAY INDICATOR.
    
```

```

;ROUTINE TO FIGURE RX AND TX WORD COUNTS.
;
;FOR HIGH BAUD THE TX AND RX MESSAGES MUST BE IDENTICAL
;THE DATA WILL BE TRANSFERED AT A BURST MODE.
;THE TXWC AND RXWC SET FOR HOW MANY CHARS TO DEAL WITH.
;
;X.X:
    
```

```

X.X:   MOV     IXDA, R2
1$:   CLR     R3
        INC     R3
        CMP     TX.TERM, (R2)+
        BNE     1$
        NEG     R3
        MOV     R3, TXWC
        MOV     IXDA, R2
        CLR     R3
    
```

711 011354 005203
 712 011356 123722 011041
 713 011362 001374
 714 011364 010302
 715 011366 005403
 716 011370 010337 011064

2\$: INC R3
 CMPB RX.TERM,(R2)+
 BNE 2\$
 MOV R3,R2
 NEG R3
 MOV R3,RXWC

717
 718
 719

720

721

722

723

724

725

 : ROUTINE USED TO GOTO
 : SUBROUTINE DEPENDENT
 : ON MODE SELECTED.
 :*****

726 011374 005037 011032
 727 011400 005037 013274
 728 011404 005037 013300
 729 011410 032700 000001
 730 011414 001402
 731 011416 000137 011572
 732 011422 032700 000002
 733 011426 001402
 734 011430 000137 011464
 735 011434 032700 000010
 736 011440 001402
 737 011442 000137 011670
 738 011446 032700 000004
 739 011452 001402
 740 011454 000137 012120
 741 011460 000000
 742 011462 000776

GO: CLR TIME
 CLR DELAY
 CLR STOP
 BIT #OWO,MODE
 BEQ 1\$
 JMP \$OWO
 1\$: BIT #OWI,MODE
 BEQ 2\$
 JMP \$OWI
 2\$: BIT #ILB,MODE
 BEQ 3\$
 JMP \$IIB
 3\$: BIT #XLB,MODE
 BEQ 4\$
 JMP \$XLB
 4\$: HALT
 BR .-2

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757 011464 104416
 758 011466 004737 014556
 759 011472 032700 040000
 760 011476 001013
 761 011500 023727 011032 000100
 762 011506 103771
 763 011510 011402
 764 011512 016403 000002
 765 011516 104001
 766 011520 005037 011032

\$OWI: KBDIN
 JSR PC,STARTR
 1\$: BIT #RFLG,STAT
 BNE 2\$
 CMP TIME,#100
 BLO 1\$
 MOV @RCSR,R2
 MOV XCSR(R4),R3
 HLT 1
 CLR TIME

 : ROUTINE USED IF 'ONE WAY IN' MODE WAS SELECTED.
 : NOTE THAT WHEN IN THIS MODE HALF DUPLEX IS THE
 : ONLY MODE AVAILABLE.
 : 'ONE WAY IN' MEANS THAT ONLY THE RECEIVER IS
 : ENABLED. THE TRANSMITTER IS NEVER 'TURNED ON'.
 :*****

```

767 011524 000762          BR      1$
768
769 011526 032777 000200 177310 2$:  BIT      #NODAT,@SWR
770 011534 001002          BNE      3$
771 011536 004737 012510          JSR      PC,TESTD
772 011542 042700 040000          BIC      #RFLG,STAT
773 011546 032777 000020 177270          BIT      #LOOP,@SWR
774 011554 001405          BEQ      4$
775 011556 012737 011570 013276          MOV      #4$,BACK
776 011564 000137 012350          JMP      EOP
777 011570 000735          4$:  BR      $OWI
778
779
780
781
782
783
784
785
786
787
788 011572 104416          $OWO:  KBDIN
789 011574 004737 013302          JSR      PC,STARTX
790 011600 005037 011032          CLR      TIME
791 011604 032700 100000          1$:  BIT      #XFLG,STAT
792 011610 001013          BNE      2$
793 011612 023727 011032 000100          CMP      TIME,#100
794 011620 103771          BLO      1$
795 011622 011402          MOV      @RCSR,R2
796 011624 016403 000002          MOV      )CSR(R4),R3
797 011630 104001          HLT
798 011632 005037 011032          CLR      TIME
799 011636 000762          BR      1$
800 011640 042700 100000          2$:  BIC      #XF G,STAT
801 011644 032777 000020 177172          BIT      #LOOP,@SWR
802 011652 001405          BEQ      3$
803 011654 012737 011666 013276          MOV      #3$,BACK
804 011662 000137 012350          JMP      EOP
805 011666 000741          3$:  BR      $OWO
806
807
808
  
```

```

:*****
:  ROUTINE USED IF 'ONE WAY OUT' WAS SELECTED.
:  NOTE THAT WHEN IN THIS MODE HALF DUPLEX IS THE ONLY
:  MODE AVAILABLE.
:  'ONE WAY OUT' MEANS THAT ONLY THE TRANSMITTER IS
:  ENABLED. THE RECEIVER IS NEVER 'TURNED ON.'
:*****
  
```

```

809
810
811
812
813
814
815
816
817
818
819
820 011670 104416
821 011672 004737 014556
822 011676 005037 011032
823 011702 032700 040000
824 011706 001013
825 011710 023727 011032 000100
826 011716 103771
827 011720 011402
828 011722 016403 000002
829 011726 104001
830 011730 005037 011032
831 011734 000762
832 011736 032777 000200 177100
833 011744 001002
834 011746 004737 012510
835 011752 042700 040000
836 011756 032777 000020 177060
837 011764 001405
838 011766 012737 012000 013276
839 011774 000137 012350
840 012000 032777 000400 177036
841 012006 001416
842 012010 013702 011020
843 012014 013703 011022
844 012020 010337 011074
845 012024 112223
846 012026 001376
847 012030 112743 000177
848 012034 005203
849 012036 112723 000177
850 012042 105023
851 012044 005037 011032
852 012050 004737 013302
853 012054 032700 100000
854 012060 001013
855 012062 023727 011032 000100
856 012070 103771
857 012072 011402
858 012074 016403 000002
859 012100 104001
860 012102 005037 011032
861 012106 000762
862 012110 042700 100000
863 012114 000137 011670
  
```

```

*****
: ROUTINE USED IF INTERNAL LOOP BACK'' WAS SELECTED.
: NOTE THAT WHEN IN THIS MODE; HALF DUPLEX IS THE
: ONLY MODE AVAILABLE.
: 'INTERNAL LOOP BACK'' MEANS THAT THE RECEIVER IS 'TURNED ON''
: AND A COMPLETE MESSAGE IS RECEIVED. IF DATA IS TO BE CHECKED
: IT IS; IF 'END PASS'' IS DESIRED; IT IS GIVEN.
: THEN THE TRANSMITTER IS ENABLED. AFTER THE WHOLE MESSAGE
: IS TRANSMITTED; THE CYCLE IS REPETED AS ABOVE.
*****
  
```

```

$ILB: KBDIN
      JSR PC,STARTR
      CLR TIME
1$:   BIT #RFLG,STAT
      BNE 2$
      CMP TIME,#100
      BLO 1$
      MOV @RCSR,R2
      MOV XCSR(R4),R3
      HLT 1
      CLR TIME
      BR 1$
2$:   BIT #NODAT,@SWR
      BNE 3$
      JSR PC,TESTD
3$:   BIC #RFLG,STAT
      BIT #LOOP,@SWR
      BEQ 4$
      MOV #4$,BACK
      JMP EOP
4$:   BIT #400,@SWR ;USE EXTERNAL DATA?
      BEQ 7$ ;BR IF NO
      MOV IRDA,R2 ;SET POINTER
      MOV IXDA,R3 ;SET POINTER
      MOV R3,XDA ;SETUP XMIT DATA ADDR
      MOVB (R2)+,(R3)+ ;MOVE INPUT TO OUTPUT
      BNE -2 ;LOOP IF NOT ZERO CHAR
      MOVB #177,-(R3) ;INSERT A FILL CHAR
      INC R3 ;BUMP ADDRESS
      MOVB #177,(R3)+ ;INSERT ANOTHER FILL
      CLRB (R3)+ ;INSERT ZERO CHAR
7$:   CLR TIME
      JSR PC,STARTX
5$:   BIT #XFLG,STAT
      BNE 6$
      CMP TIME,#100
      BLO 5$
      MOV @RCSR,R2
      MOV XCSR(R4),R3
      HLT 1
      CLR TIME
      BR 5$
6$:   BIC #XFLG,STAT
      JMP $:.5
  
```

864
 865
 866
 867
 868
 869
 870
 871
 872
 873
 874
 875
 876
 877 012120 104416
 878 012122 032737 000001 011014
 879 012130 001402
 880 012132 004737 014556
 881 012136 004737 013302
 882 012142 005037 011032
 883 012146 032700 100000
 884 012152 001016
 885 012154 032700 040000
 886 012160 001024
 887 012162 023727 011032 000100
 888 012170 103766
 889 012172 011402
 890 012174 016403 000002
 891 012200 104001
 892 012202 005037 011032
 893 012206 000757
 894 012210 032737 000001 011014
 895 012216 001356
 896 012220 042700 100000
 897 012224 004737 014556
 898 012230 000746
 899 012232 032737 000001 011014
 900 012240 001420
 901 012242 032700 100000
 902 012246 001013
 903 012250 023727 011032 000100
 904 012256 103765
 905 012260 011402
 906 012262 016403 000002
 907 012266 104001
 908 012270 005037 011032
 909 012274 000756
 910 012276 042700 100000
 911 012302 042700 040000
 912 012306 005037 011032
 913 012312 032777 000200 176524
 914 012320 001002
 915 012322 004737 012510
 916 012326 032777 000020 176510
 917 012334 001671
 918 012336 012737 012120 013276
 919 012344 000137 012350

```

*****
ROUTINE USED IF 'EXTERNAL LOOP BACK' WAS SELECTED.
EITHER HALF OR FULL DUPLEX MAY BE SELECTED IN THIS MODE.
'EXTERNAL LOOP BACK' MEANS THAT THE TRANSMITTER IS FIRST
TURNED ON (IF HALF DUPLEX) AND THE WHOLE MESSAGE IS TRANSMITTED;
THEN THE RECEIVER IS ENABLED. AFTER THE WHOLE MESSAGE IS RECEIVED
DATA WILL THEN BE CHECKED IF DESIRED AND END PASS WILL
BE GIVEN IF DESIRED. THEN THE CYCLE IS REPEATED
AS ABOVE. IF RUNNING IN FULL DUPLEX THE PROGRAM
WAITS FOR BOTH THE RECEIVER AND TRANSMITTER TO
FINISH THEN RESTARTS THE RECEIVER AND TRANSMITTER.
*****
  
```

```

$XLB:  KBDIN
      BIT  #FULL.DUPLEX,PARAM2
      BEQ  1$
      JSR  PC,STARTR
      JSR  PC,STARTX
      CLR  TIME
      BIT  #XFLG,STAT
      BNE  3$
      BIT  #RFLG,STAT
      BNE  4$
      CMP  TIME,#100
      BLO  2$
      MOV  @RCSR,R2
      MOV  XCSR(R4),R3
      HLT  1
      CLR  TIME
      BR   2$
      BIT  #FULL.DUPLEX,PARAM2
      BNE  7$
      BIC  #XFLG,STAT
      JSR  PC,STARTR
      BR   2$
      BIT  #FULL.DUPLEX,PARAM2
      BEQ  8$
      BIT  #XFLG,STAT
      BNE  6$
      CMP  TIME,#100
      BLO  4$
      MOV  @RCSR,R2
      MOV  XCSR(R4),R3
      HLT  1
      CLR  TIME
      BR   4$
      BIC  #XFLG,STAT
      BIC  #RFLG,STAT
      CLR  TIME
      BIT  #NODAT,@SWR
      BNE  5$
      JSR  PC,TESTD
      BIT  #LOOP,@SWR
      BEQ  $XLB
      MOV  #XLB,BACK
      IMP  EOP
  
```

```

920
921
922
923
924
925
926 012350
927 012350 104414 000340
928 012354 016437 000002 012506
929 012362 042737 177737 012506
930 012370 042764 000040 000002
931 012376 012766 012436 000002
932 012404 010037 013260
933 012410 010137 013262
934 012414 010237 013264
935 012420 010337 013266
936 012424 010437 013270
937 012430 010537 013272
938 012434 000207
939
940 012436
941 012436 013700 013260
942 012442 013701 013262
943 012446 013702 013264
944 012452 013703 013266
945 012456 013704 013270
946 012462 013705 013272
947 012466 012737 177777 013274
948 012474 053764 012506 000002
949 012502 000177 000570
950 012506 000000
951
952
953
954
955
956
957
958 012510 013746 011056
959 012514 001413
960 012516 032777 020000 176320
961 012524 001007
962 012526 104400 012720
963 012532 004077 176272
964 012536 005746
965 012540 104400 013001
966 012544 013701 011022
967 012550 013702 011020
968 012554 122122
969 012556 001776
970 012560 123741 011040
971 012564 001453
972 012566 122742 000002
973 012572 001005
974 012574 010237 012602
975 012600 104400

```

```

:*****
:      ROUTINE TO RETURN
:      TO MONITOR FOR
:      END PASS.
:*****

EOP:
STPS,PRTY7 ;SET PS PRIORITY TO 7
MOV XCSR(R4),QTPIE ;SAVE TX CSR
BIC #^C<TIE>,QTPIE ;CLEAR ALL BUT TX IE.
BIC #TIE,XCSR(R4) ;CLEAR TX IE (EVEN IF IT WASN'T SET)
MOV #ENTER,2(SP) ;SET FOR RETURN IF SW 14=1
MOV R0,SAVR0 ;SAVE REGISTER 0
MOV R1,SAVR1 ;SAVE REGISTER 1
MOV R2,SAVR2 ;SAVE REGISTER 2
MOV R3,SAVR3 ;SAVE REGISTER 3
MOV R4,SAVR4 ;SAVE REGISTER 4
MOV R5,SAVR5 ;SAVE REGISTER 5
RTS PC ;RETURN TO CONTROL PROGRAM

ENTER:
MOV SAVR0,R0 ;RESTORE R0
MOV SAVR1,R1 ;RESTORE R1
MOV SAVR2,R2 ;RESTORE R2
MOV SAVR3,R3 ;RESTORE R3
MOV SAVR4,R4 ;RESTORE R4
MOV SAVR5,R5 ;RESTORE R5
MOV #-1,DELAY
BIS QTPIE,XCSR(R4) ;IF ORGINALLY SET; SET TX IE
JMP @BACK
QTPIE: 000000

:*****
:      SUBROUTINE TO CHECK
:      RECEIVER DATA.
:*****
TESTD: MOV ERDBR,-(SP) ;WAS THERE A RECEIVE ERROR?
BEQ TSTDAT ;BR IF NO
BIT #BIT13,@SWR ;INHIBIT PRINTOUTS?
BNE TSTDAT ;BR IF YES
TYPE ,MSG0 ;<15><12>THERE WAS A RECEIVE ERROR. RBUF
JSR R0,@B2016 ;PRINT CONTENTS OF RBUF
TST -(SP)
TYPE ,MSG1 ;<15><12>
TSTDAT: MOV IXDA,R1 ;SETUP XMIT DATA ADDR
MOV IRDA,R2 ;SETUP RCV DATA ADDR
SCAN4: CMPB (R1)+,(R2)+ ;DATA OK ?
BEQ SCAN4 ;BR IF OK
CMPB TX_TERM,-(R1) ;IS IT END OF DATA
BEQ TESTDX ;BR IF YES
CMPB #002,-(R2)
BNE 2$
MOV R2,1$
TYPE

```

```
976 012602 000000 1$: .WORD 0
977 012604 000443 BR TESTDX
978 012606 2$:
979 012606 105712 TSTB (R2)
980 012610 001441 BEQ TESTDX ;BR IF YES
981 012612 122721 000177 CMPB #177, (R1)+ ;IS IT FILL CHAR?
982 012616 001756 BEQ SCAN4 ;BR IF YES
983 012620 005301 DEC R1 ;BACKUP
984 012622 122722 000177 CMPB #177, (R2)+ ;IS IT FILL?
985 012626 001752 BEQ SCAN4 ;BR IF YES
986 012630 105742 TSTB -(R2) ;BACK UP POINTER
987 012632 123722 011015 CMPB PARAM2+1, (R2)+
988 012636 001746 BEQ SCAN4 ;BR IF CHAR WAS SYNC
989 012640 000240 SCANS: NOP ;DATA ERROR
990 012642 032777 020000 176174 BIT #BIT13,@SWR ;INHIBIT PRINTOUTS
991 012650 001016 BNE DERR ;BR IF YES
992 012652 104400 013004 TYPE ,MSG2 ;<15><12>RECEIVED DATA - <15><12>
993 012656 013737 011020 012666 MOV IRDA, RDAX ;SETUP DATA ADDRESS
994 012664 104400 TYPE ;PRINT RECEIVED DATA
995 012666 000000 RDAX: 0 ;RECEIVED DATA ADDR.
996 012670 104400 013031 TYPE ,MSG3 ;<15><12>DATA SHOULD BE<15><12>
997 012674 013737 011022 012704 MOV IXDA, .+10 ;SETUP ADDR.
998 012702 104400 TYPE ;PRINT GOOD DATA
999 012704 011022 IXDA
1000 012706 111103 DERR: MOVB (R1),R3 ;SETUP XMIT DATA
1001 012710 114202 MOVB -(R2),R2 ;SETUP RCV DATA
1002 012712 104007 HLT+7 ;DATA ERROR HALT
1003 012714 005726 TESTDX: TST (SP)+ ;POP STACK
1004 012716 000207 RTS PC ;RETURN FROM SUB/ROUT
1005
1006 012720 005015 044124 051105 MSG0: .ASCIZ <15><12>/THERE WAS A RECEIVER ERROR. REGISTER (SEL 2) =/
(1) 013001 015 000012 MSG1: .ASCIZ <15><12>
(1) 013004 005015 042522 042503 MSG2: .ASCIZ <15><12>/RECEIVED DATA = /<15><12>
(1) 013031 015 042012 052101 MSG3: .ASCIZ <15><12>/DATA SHOULD BE/<15><12>
(1) 013054 005015 046120 040505 MSG4: .ASCII <15><12>/PLEASE MAKE CONNECTION (DIAL NUMBER)./
(1) 013123 015 053412 042510 .ASCIZ <15><12>/WHEN CONNECTION COMPLETE; HIT CONTINUE SWITCH./<15><12>
(1) 013206 005015 046120 040505 MSG5: .ASCIZ <15><12>/PLEASE MAKE CONNECTION (DIAL NUMBER)./<15><12>
(1) .EVEN
(1) 013260 000000 SAVR0: 0
1007 013262 000000 SAVR1: 0
1008 013264 000000 SAVR2: 0
1009 013266 000000 SAVR3: 0
1010 013270 000000 SAVR4: 0
1011 013272 000000 SAVR5: 0
1012 013274 000000 DELAY: 0
1013 013276 000000 BACK: C
1014 013300 000000 STOP: 0
1015
```

```
1016 :*****
1017 :INITIALIZE TRANSMITTER.
1018 :PURPOSE OF THIS ROUTINE IS TO SEND OUT FIVE
1019 :SYNC CHARS. THE SYNC CHAR TO BE SENT IS FOUND
1020 :IN LOCATION PARAM2+1 (HIGH BYTE OF PARAM2).
1021 :WHEN THE SYNC CHARS. HAVE BEEN TRANSMITTED
1022 :THE TRANSMITTER WILL BE ENABLED FOR REAL DATA
1023 :TRANSFER. CHARACTERS WILL BE TRANSMITTED ONE AT A TIME.
1024 :*****
1025
1026 013302 005737 013274 STARTX: TST DELAY ;IF SW04=1 & SW14=0 WAIT BEFORE TURNING TX ON
1027 013306 001416 BEQ NDLY ;NO GO AHEAD AND TURN ON TX
1028 013310 005037 014214 CLR TEMP1
1029 013314 012737 000007 014216 MOV #7,TEMP2
1030 013322 062737 000001 014214 ADD #1,TEMP1
1031 013330 001374 BNE -6
1032 013332 005337 014216 DEC TEMP2
1033 013336 001371 BNE -14
1034 013340 005037 013274 CLR DELAY ;ZERO DELAY
1035 013344 005037 011032 NDLY: CLR TIME
1036 013350 042764 000300 000002 BIC #300,XCSR(R4) ;CLEAR BOTH DONE BITS FROM TX
1037 013356 052764 000010 000002 BIS #BIT3,XCSR(R4) ;ENABLE ERRORS
1038 013364 032764 000004 000002 BIT #BIT2,XCSR(R4) ;WHERE IS THE POINTER POINTING
1039 013372 001403 BEQ 1$ ;BR IF PRI IS NEXT.
1040 013374 004737 015426 JSR PC,GETPRI ;GO TOGGLE TO GET PRIMARY.
1041 013400 000740 BR STARTX ;GOTO BEGGINING
1042 013402 012701 000005 1$: MOV #5,R1 ;SET FOR FIVE SYNC CHARS.
1043 013406 012705 014222 MOV #SYNC,R5 ;SET POINTER LOCATION
1044 013412 113725 011015 MOV#B PARAM2+1,(R5)+ ;LOAD IN SYNC CHARS.
1045 013416 005301 DEC R1 ;ALL DONE??
1046 013420 001374 BNE -6 ;BRANCH IF NOT DONE
1047 013422 105015 CLR#B (R5) ;SET LAST BYTE TO ZERO
1048
1049 ;NOW THE SYNC CHARS ARE LOADED IN CORE FOR THE DQ11 TO
1050 ;PICK UP AND TRANSMIT OUT.
1051
1052 013424 112764 000012 000005 MOV#B #12,REG(R4) ;SELECT MISC REGISTER
1053 013432 012764 004000 000006 MOV #4000,SEC(R4) ;SET FOR EIGHT BITS PER CHAR.
1054 013440 112764 000002 000005 MOV#B #2,REG(R4) ;SELECT THE TX BA PRI.
1055 013446 012764 014222 000006 MOV #SYNC,SEC(R4) ;LOAD TX BA PRI. WITH SYNC ADDR.
1056 013454 105264 000005 INCB REG(R4) ;SELECT THE TX WC PRI.
1057 013460 012764 177773 000006 MOV #-5,SEC(R4) ;SET FOR FIVE CHARS.
1058 013466 042700 100000 BIC #XFLG,STAT ;RESET TX COMPLETE FLAG
1059 013472 052764 001000 000002 BIS #DTR,XCSR(R4) ;SET DATA TERMINAL READY
1060 013500 005737 013300 TST STOP ;FIRST TIME HERE?
1061 013504 001004 BNE 16$ ;BR IF NO
1062 013506 104400 013206 TYPE ,MSG5 ;MAKE CONNECTION
1063 013512 005137 013300 COM STOP
1064 013516 032764 002000 000002 16$: BIT #MRDY,XCSR(R4)
1065 013524 001037 BNE 2$
1066 013526 032764 002000 000002 8$: BIT #MRDY,XCSR(R4)
1067 013534 001017 BNE 3$
1068 013536 023727 011032 000036 CMP TIME,#36 ;HAVE 30 SEC ELAPSED YET
1069 013544 103770 BLO 8$ ;NO NOT YET
1070 013546 011402 MOV @XCSR,R2 ;LOAD FOR TYPEOUT
1071 013550 016403 000002 MOV XCSR(R4),R3 ;LOAD FOR TYPEOUT
```

```

1072 013554 032777 010000 175262 BIT #SW12,@SWR ;INHIBIT PRINTOUTS?
1073 013562 001001 BNE 12$ ;BR IF YES
1074 013564 104002 HLT+2 ;TYPE 'WAITING TO TRANSMIT'' MESSAGE.
1075 013566 005037 011032 12$: CLR TIME
1076 013572 000755 BR 8$
1077 013574 005037 014214 3$: CLR TEMP1
1078 013600 012737 000005 014216 MOV #5,TEMP2
1079 013606 062737 000001 014214 ADD #1,TEMP1
1080 013614 001374 BNE -6
1081 013616 005337 014216 DEC TEMP2
1082 013622 001371 BNE -14
1083 013624 032737 000001 011014 2$: BIT #FULL.DUPLEX,PARAM2
1084 013632 001023 BNE 9$
1085 013634 032764 010000 000002 10$: BIT #10000,XCSR(R4) ;IS CARRIER UP?
1086 013642 001417 BEQ 9$
1087 013644 023727 011032 000036 CMP TIME,#36 ;30 SECONDS UP ?
1088 013652 103770 BLO 10$ ;NOT YET
1089 013654 011402 MOV @RCSR,R2 ;PREPARE TYPE OUT
1090 013656 016403 000002 MOV XCSR(R4),R3 ;AS ABOVE
1091 013662 032777 010000 175154 BIT #SW12,@SWR ;INHIBIT PRINTOUTS?
1092 013670 001001 BNE 13$ ;BR IF YES
1093 013672 104001 HLT 1 ;TYPE 'WAITING ''
1094 013674 005037 011032 13$: CLR TIME ;ZERO TIMER
1095 013700 000755 BR 10$
1096 013702 005037 011032 9$: CLR TIME ;SET TIME=0
1097 013706 052764 000400 000002 BIS #RQTS,XCSR(R4)
1098 013714 032764 020000 000002 11$: BIT #CTS,XCSR(R4)
1099 013722 001017 BNE 6$
1100 013724 023727 011032 000036 CMP TIME,#36 ;30 SECONDS UP ??
1101 013732 103770 BLO 11$ ;NOT YET
1102 013734 011402 MOV @RCSR,R2 ;PREPARE TYPE OUT
1103 013736 016403 000002 MOV XCSR(R4),R3 ;AS ABOVE
1104 013742 032777 010000 175074 BIT #SW12,@SWR ;INHIBIT PRINTOUTS?
1105 013750 001001 BNE 14$ ;BR IF YES
1106 013752 104002 HLT 2 ;TYPE 'WAITING ''
1107 013754 005037 011032 14$: CLR TIME ;ZERO TIMER
1108 013760 000755 BR 11$
1109
1110
1111 ;TEST AND SETUP FOR HI BAUD RATE. NOTE THE MESSAGES ;:++D
1112 ;MUST BE THE SAME LENGTH IN HI BAUD.
1113
1114 013762 032737 000002 011014 6$: BIT #HI.BAUD,PARAM2 ;HAS HI BAUD BEEN SELECTED?
1115 013770 001414 BEQ 17$ ;BR IF NO
1116 013772 112764 000006 000005 MOV #6,REG(R4) ;SEL SEC TXBA
1117 014000 013764 011022 000006 MOV IXDA,SEC(R4) ;LOAD SEC TXBA WITH DATTA
1118 014006 112764 000007 000005 MOV #7,REG(R4) ;SEL SEC TXWC
1119 014014 013764 011062 000006 MOV TXWC,SEC(R4) ;SFT TXWC FOR MAX CHAR XFER RATE
1120
1121
1122 ;HERE NOW BECAUSE CLEAR TO SEND HAS BEEN SET
1123 ;NOW SEND THE SYNC CHARS.
1124
1125 014022 052764 000001 000002 17$: BIS #.GO,XCSR(R4) ;SET THE GO BIT
1126 014030 005037 011032 CLR TIME ;SET TIME TO ZERO
1127 014034 105764 000002 4$: TSTB XCSR(R4) ;IS TX DONE WITH SYNC

```

```

1128 014040 100417          BMI      5$          ;BR IF YES
1129 014042 023727 011032 000036      CMP      TIME,#36      ;HAVE 30 SECOND GONE BY??
1130 014050 103771          BLO      4$          ;BR IF NOT YET
1131 014052 011402          MOV      @RCSR,R2      ;LOAD FOR TYPE OUT
1132 014054 016403 000002          MOV      XCSR(R4),R3   ;LOAD FOR TYPE OUT
1133 014060 032777 010000 174756      BIT      #SW12,@SWR    ;INHIBIT PRINTOUTS?
1134 014066 001001          BNE      15$         ;BR IF YES
1135 014070 104002          HLT      2           ;TYPE WAITING...
1136 014072 005037 011032 15$:      CLR      TIME        ;ZERO TIMER
1137 014076 000756          BR       4$          ;GO WAIT FOR DONE
1138
1139          ;NOW TRANSMITTER DONE HAS SET AND THE SYNC CHARS
1140          ;HAVE BEEN SENT OUT. THE TRANSMITTER IS TO BE
1141          ;ENABLED AND THE DATA WILL BE TRANSFERED.
1142
1143 014100 032737 000002 011014 5$:      BIT      #HI.BAUD,PARAM2 ;HAS HI BAUD BEEN SELECTED?
1144 014106 001033          BNE      7$          ;BR IF YES
1145 014110 112764 000002 000005      MOVB     #2,REG(R4)    ;SELECT TX BA PRI
1146 014116 013764 011022 000006      MOV      IXDA,SEC(R4) ;LOAD BA
1147 014124 062764 000002 000006      ADD      #2,SEC(R4)   ;POINT BA TWO HIGHER
1148 014132 112764 000003 000005      MOVB     #3,REG(R4)    ;SELECT TX WC PRI
1149 014140 012764 177776 000006      MOV      #-2,SEC(R4) ;SET TWO CHARS AT A TIME
1150 014146 112764 000006 000005      MOVB     #6,REG(R4)    ;SELECT TX BA SEC
1151 014154 013764 011022 000006      MOV      IXDA,SEC(R4) ;LOAD TX BA SEC WITH DATA POINTER
1152 014162 112764 000007 000005      MOVB     #7,REG(R4)    ;SELECT TX WC SEC
1153 014170 012764 177776 000006      MOV      #-2,SEC(R4) ;SET WC WITH -2
1154
1155          ;THE DQ11 IS TOTALLY ENABLE TO GO. THE TRANSMISSION
1156          ;WILL BEGIN ON THE SECONDAY (THE SYNC CHARS WENT ON
1157          ;THE PRIMARY). THE DQ11 WILL INTERUPT AFTER EACH CHAR
1158          ;AND CONTINUE TRANSMITTING WHILE THE END CHAR IS
1159          ;SEARCHED FOR.
1160
1161 014176 042764 000300 000002 7$:      BIC      #300,XCSR(R4) ;CLEAR ALL TX DONES
1162 014204 052764 000041 000002          BIS      #IE+.GO,XCSR(R4)
1163
1164 014212 000207          RTS      PC          ;SET INTERUPT ENABLE AND GO
1165          ;GO BACK TO MAIN PROGRAM
1166
1167          ;SYNC CHARACTER BUFFER AREA.
1168 014214 000000          TEMP1: 0
1169 014216 000000          TEMP2: 0
1170 014220 000000          ERRORS: 0
1171 014222 026 026          SYNC:  .BYTE 26,26
1172 014224 026 026          .BYTE 26,26
1173 014226 026 026          .BYTE 26,26
    
```

```

1174
1175
1176
1177
1178
1179
1180
1181
1182 014230 000240 1.5 XISR: NOP ;LOCATION SET FOR HALT INSTRUCTION IN DEBUGGING.
1183 014232 005764 000004 4.4 TST ERR(R4) ;IS THE DQ11 ERROR FLAG SET??
1184 014236 100012 2.6 BPL 2$ ;NOT SET GOOD SO FAR.
1185 014240 016403 000002 5.0 MOV XCSR(R4),R3
1186 014244 011402 3.8 MOV @RCSR,R2
1187 014246 104000 9.3 HLT 0 ;DQ11 ERROR FLAG SET. EXAMINE REGISTERS FOR ERRO
1188 014250 016437 000004 014220 6.4 MOV ERR(R4),ERRORS ;SAVE DQ ERROR REG
1189 014256 042764 000377 000004 7.0 BIC #377,ERR(R4)
1190 014264 011402 3.8 2$: MOV @RCSR,R2 ;PREPARE FOR ERROR TYPEOUT
1191 014266 016403 000002 5.0 MOV XCSR(R4),R3 ;PREPARE FOR ERROR TYPEOUT
1192 014272 032764 000300 000002 6.5 BIT #300,XCSR(R4) ;IS EITHER TX DONE SET??
1193 014300 001001 2.6 BNE .+4 ;GOOD EITHER PRI OR SEC DONE IS SET
1194 014302 104000 9.3 HLT 0 ;REPORT ERROR. INTERRUPT WAS TO
1195 ;BE CAUSED BY TX DONE; AND TX IS NOT DONE.
1196 014304 032737 000002 011014 5.3 BIT #HI.BAUD,PARAM2 ;IS HIGH BAUD RATE SELECTED?
1197 014312 001402 2.6 BEQ 1$ ;BR IF NO
1198 014314 005746 4.4 TST -(SP) ;FAKE STACK
1199 014316 000474 2.6 BR HBTCHK ;XFER ALL DONE
1200 014320 032764 000004 000002 6.5 1$: BIT #BIT2,XCSR(R4) ;WHERE IS POINTER??
1201 014326 001424 2.6 BEQ TX.SEC ;SECONDARY IS NEXT.
1202 014330 112764 000002 000005 6.4 TX.PRI: MOVB #2,REG(R4) ;SELECT TX BA PRIMARY
1203 014336 016401 000006 5.0 MOV SEC(R4),R1 ;GET NEXT ADDRESS TO TRANSMIT FROM
1204 014342 062764 000002 000006 6.4 ADD #2,SEC(R4) ;UPDATE CURRENT ADDRESS.
1205 014350 004737 014450 5.8 JSR PC,TX.CK ;GO CHECK FOR END CHAR.
1206
1207 ;IF I COME BACK FROM THE SUBROUTINE THAT MEANS THAT THE
1208 ;END CHAR WAS NOT FOUND AND THE TRANSMISSON GOES ON.
1209
1210 014354 112764 000003 000005 6.4 MOVB #3,REG(R4) ;SELECT TX WC SEC
1211 014362 012764 177776 000006 6.4 MOV #-2,SEC(R4) ;LOAD WITH A -2
1212 014370 042764 000200 000002 7.0 BIC #BIT7,XCSR(R4) ;CLEAR PRI DONE FROM TX
1213 014376 000002 4.8 RTI ;LEAVE TX ISR
1214
1215 ;THE ABOVE ROUTINE SERVICED THE INTERRUPT IF THE
1216 ;PRIMARY REGISTERS CAUSED THE INTERRUPT.
1217
1218 014400 112764 000006 000005 6.4 TX.SEC: MOVB #6,REG(R4) ;SELECT THE TX BA SECONDARY.
1219 014406 016401 000006 5.0 MOV SEC(R4),R1 ;GET ADDRESS POINTER.
1220 014412 062764 000002 000006 6.4 ADD #2,SEC(R4) ;UPDATE CURRENT ADDRESS.
1221 014420 004737 014450 5.8 JSR PC,TX.CK ;GO CHECK THE LAST CHARACTER.
1222
1223 ;JUST LIKE ABOVE; IF I COME BACK FROM THE ABOVE SUBROUTIN
1224 ;THAT MEANS THE THE LAST CHAR HASN'T BEEN TRANSMITTED
1225 ;AND THAT TRANSMISSON SHOULD CONTINUE.
1226
1227 014424 112764 000007 000005 6.4 MOVB #7,REG(R4) ;SELECT THE TX WC SEC
1228 014432 012764 177776 000006 6.4 MC #-2,SEC(R4) ;LOAD WITH A -2
1229 014440 042764 000100 000002 7.0 BIC #BIT6,XCSR(R4) ;CLEAR SEC DONE

```

```

1230 014446 000002          4.8          RTI          ;LEAVE HERE
1231 014450 123741 011040  5.9 TX.CK:  CMPB    TX.TERM,-(R1) ;WAS THAT THE LAST CHARACTER.
1232 014454 001427          2.6          BEQ    XISRDN  ;ISR IS DONE IF BR IS MADE
1233 014456 123741 011040  5.9          CMPB    TX.TERM,-(R1) ;LAST CHAR?
1234 014462 001424          2.6          BEQ    XISRDN  ;BR IF YES
1235 014464 032777 000100 174352 7.7 MONDAT: BIT    #BIT6,@SWR ;CHECK FOR DATA MONITOR.
1236 014472 001405          2.6          BEQ    1$      ;DON'T MONITOR DATA
1237 014474 105777 174402  5.6          TSTB   @TPS    ;TTY READY??
1238 014500 100002          2.6          BPL    1$      ;TTY NOT READY GO ON WITH TEST
1239 014502 111177 174376  7.6          MOVB   (R1),@TPB ;TYPE CHARACTER
1240 014506 000207          3.5 1$:      RTS    PC      ;GO BACK TO ISR
1241 014510 032777 000100 174326 7.7 HBTCHK: BIT    #BIT6,@SWR ;CHECK FOR DATA MONITOR
1242 014516 001406          2.6          BEQ    XISRDN  ;DON'T MONITOR
1243 014520 105777 174356  5.6          TSTB   @TPS    ;IS TTY READY
1244 014524 100003          2.6          BPL    XISRDN  ;TTY NOT READY GO ON WITH TEST
1245 014526 112777 000124 174350 7.6          MOVB   #'T,@TPB ;TYPE 'T'
1246 014534 052700 100000  3.8 XISRDN: BIS   #XFLG,STAT ;SET TX COMPLETE FLAG
1247 014540 042764 000441 000002 7.0          BIC    #IE+RQTS+.GO,XCSR(R4)
1248          ;CLEAR ALL CONDITIONS FOR TX
1249 014546 005037 011032  3.7          CLR    TIME    ;ZERO TIMER
1250 014552 005726          3.2          TST    (SP)+   ;POP SUBROUTINE PC FROM STACK
1251 014554 000002          4.8 TXOUT:  RTI          ;LEAVE HERE FOR GOOD.
1252

```

```
1253 :*****
1254 :INITIALIZE RECEIVER ROUTINE
1255 :*****
1256
1257 014556 000240          STARTR: NOP          ;LEAVE OPEN FOR TEST PURPOSES.
1258 014560 042714 000300 BIC      #300,@RCSR    ;CLEAR ALL DONES
1259 014564 032714 000004 BIT      #BIT2,@RCSR   ;CHECK FOR POINTER TO SEC.
1260 014570 001403          BEQ      1$           ;BR IF POINTING TO PRI
1261 014572 004737 015426 JSR      PC,GETPRI    ;POINT REGISTERS TO PRI
1262 014576 000767          BR        STARTR      ;BEGIN AGAIN
1263 014600 113737 011015 014222 1$:  MOVB   PARAM2+1,SYNC  ;GET SYNC CHAR
1264 014606 113737 011015 014223  MOVB   PARAM2+1,SYNC+1 ;GET SECOND SYNC CHAR
1265 014614 112764 000011 000005  MOVB   #11,REG(R4)    ;SELECT SYNC REGISTER
1266 014622 013764 014222 000006  MOV     SYNC,SEC(R4)  ;LOAD SYNC REGISTER
1267 014630 112764 000000 000005  MOVB   #0,REG(R4)    ;SELECT RX BA PRI.
1268 014636 013764 011020 000006  MOV     IRDA,SEC(R4) ;LOAD RX BA FOR DATA
1269 014644 105264 000005          INCB   REG(R4)      ;GET WC REGISTER
1270 014650 012764 177776 000006  MOV     #-2,SEC(R4)  ;SET FOR ONE CHAR.
1271 014656 112764 000004 000005  MOVB   #4,REG(R4)    ;SELECT RX BA SEC.
1272 014664 013764 011020 000006  MOV     IRDA,SEC(R4) ;LOAD RX BA SEC.
1273 014672 062764 000002 000006  ADD     #2,SEC(R4)   ;UPDATE DATA POINTER BY ONE
1274 014700 105264 000005          INCB   REG(R4)      ;GET WC REGISTER
1275 014704 012764 177776 000006  MOV     #-2,SEC(R4)  ;SET FOR TWO CHAR.
1276 014712 112764 000012 000005  MOVB   #12,REG(R4)   ;SELECT MISC REGISTER
1277 014720 012764 004000 000006  MOV     #4000,SEC(R4) ;SELECT EIGHT BITS PER CHAR.
1278 014726 012737 000750 011070  MOV     #750,RCC     ;SET FOR MAX 750 CHARS TO BE RXED
1279 014734 042700 040000          BIC     #RFLG,STAT   ;RESET RX COMPLETE FLAG
1280 014740 005037 011054          CLR    ERCSR
1281 014744 005037 011056          CLR    ERDBR
1282 014750 032737 000002 011014  BIT     #HI.BAUD,PARAM2 ;IS HIGH BAUD RATE SELECTED?
1283 014756 001413          BEQ     2$           ;BR IF NO
1284 014760 112764 000001 000005  MOVB   #1,REG(R4)    ;SELECT THE RX WC PRI
1285 014766 013764 011064 000006  MOV     RXWC,SEC(R4) ;RECEIVE CHARS MAX.
1286 014774 112764 000005 000005  MOVB   #5,REG(R4)    ;SELECT THE RX WC SEC.
1287 015002 005064 000006          CLR    SEC(R4)     ;ZERO THE RX WC SECONDARY.
1288 015006 052764 001000 000002 2$:  BIS     #DTR,XCSR(R4) ;SET DATA TERMINAL READY.
1289 015014 005737 013300          TST    STOP        ;FIRST TIME HERE?
1290 015020 001004          BNE     3$
1291 015022 104400 013206          TYPE  ,MSG5
1292 015026 005137 013300          COM   STOP
1293 015032
1294 015032 000240          3$:  NOP
1295 015034 000240          NOP
1296 015036 000240          NOP
1297 015040 000240          NOP
1298 015042 000240          NOP
1299 015044 052714 000041          BIS     #IE+.GO,@RCSR ;SET INTERRUPT ENABLE AND GO
1300 015050 000207          RTS     PC          ;GO TO MAINLINE.....
1301
1302 :THE ABOVE SHOULD HAVE PREPARED THE RECEIVER TO
1303 :RECEIVE ALL THE CHARS NEEDED FOR TESTING.
1304 :THE RECEIVER WILL INTERRUPT AFTER EACH CHAR IS RECEIVED.
1305 :CHECKING FOR THE END CHARACTER.
1306
```

1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362

015052 000240
015054 005764 000004
015060 100012
015062 011402
015064 016403 000002
015070 104000
015072 016437 000004 014220
015100 042764 000377 000004
015106 005337 000070
015112 001004
015114 000005
015116 104006
015120 000000
015122 000776
015124 011402
015126 016403 000002
015132 032714 000300
015136 001001
015140 104000
015142 032737 000002 011014
015150 001404
015152 042714 000041
015156 005746
015160 000507
015162 032714 000004
015166 001423
015170 112764 000000 000005
015176 016401 000006
015202 062764 000002 000006
015210 004737 015304

1.5
4.4
2.6
.8
.0
9.3
6.4
7.0
3.7
2.6
1.5
9.3
1.8
2.6
3.8
5.0
5.3
2.6
9.3
5.3
2.6
5.8
4.4
2.6
5.3
2.6
6.4
5.0
6.4
5.8
6.4
5.8
6.4
6.4
5.8
6.4
6.4
5.8
6.4
5.8
6.4
5.0
6.4
5.8
6.4
5.8

```
*****  
:RECEIVER INTERRUPT SERVICE ROUTINE.  
*****  
  
:THIS ROUTINE WILL SERVICE THE PRIMARY AND SECONDARY  
:REGISTERS WHEN THEY INTERRUPT.  
:CHECKING FOR THE END CHAR IS PERFORMED.  
  
RISR:  NOP ;LEAVE SPACE FOR BEBUGGING  
      TST  ERR(R4) ;CHECK THE DQ11 ERROR FLAG  
      BPL  2$ ;BR IF ERROR FLAG NOT SET  
      MOV  @RCSR,R2  
      MOV  XCSR(R4),R3  
      HLT  0 ;CHECK ERROR REGISTER FOR ERROR.  
      MOV  ERR(R4),ERRORS  
      BIC  #377,ERR(R4)  
2$:   DEC  RCC ;CHECK THE BUFFER SIZE  
      BNE  1$ ;BR IF OK  
      RESET ;STOP THE SHOW  
      HLT+6 ;RECEIVER BUFFER FULL  
      HALT ;STOP EVERY THING  
      BR   .-2 ;DISABLE CONT SWITCH  
1$:   MOV  @RCSR,R2 ;PREPARE FOR ERROR TYPEOUT  
      MOV  XCSR(R4),R3 ;PREPARE FOR ERROR TYPEOUT  
      BIT  #300,@RCSR ;IS EITHER DONE SET??  
      BNE  .+4 ;BR IF AT LEAST ONE DONE IS SET  
      HLT  0 ;NOBODY IS DONE. BUT AN INTERRUPT OCCURED.  
      BIT  #HI.BAUD,PARAM2 ;HIGH BAUD??  
      BEQ  3$ ;BR IF NO  
      BIC  #IE+.GO,@RCSR ;CLEAR GO AND INTERUPT ENABLE  
      TST  -(SP) ;FAKE STACK  
      BR   HBRCHK ;XFER ALL DONE  
3$:   BIT  #BIT2,@RCSR ;WHERE IS THE POINTER.  
      BEQ  RX.SEC ;POINTING TO SECONDARY.  
RX.PRI: MOV  #0,REG(R4) ;SELECT THE RX BA PRIMARY  
      MOV  SEC(R4),R1 ;GET THE ADDRESS OF DATA.  
      ADD  #2,SEC(R4) ;UPDATE CURRENT ADDRESS  
      JSR  PC,RX.CK ;GO CHECK THE DATA.  
  
:IF I RETURN BACK FROM THE ABOVE SUBROUTINE THE END CHAR  
:WAS NOT FOUND AND RECEIVING WILL CONTINUE.  
  
      MOV  #1,REG(R4) ;SELECT THE RX WC PRI.  
      MOV  #-2,SEC(R4) ;LOAD WITH -2  
      BIC  #BIT7,@RCSR ;CLEAR RX DONE  
      RTI ;LEAVE HERE NOW.  
  
:THE ABOVE ROUTINE WAS FOR THE PRIMARY REGISTER INTERUPT.  
:THE BELOW ROUTINE IS FOR THE SECONDARY REGISTERS.  
  
RX.SEC: MOV  #4,REG(R4) ;SELECT THE RX BA SEC.  
      MOV  SEC(R4),R1 ;GET DATA ADDRESS  
      ADD  #2,SEC(R4) ;UPDATE CURRENT ADDRESS  
      JSR  PC,RX.CK ;GO CHECK THE DATA
```

```

1363                                     ;IF I COME BACK THAT MEANS THAT THE END CHARACTER WASN'T
1364                                     ;FOUND AND I SHOULD CONTINUE RECEIVING.
1365
1366 015262 112764 000005 000005 6.4      MOVB   #5,REG(R4)      ;SELECT THE RX WC SEC
1367 015270 012764 177776 000006 6.4      MOV    #-2,SEC(R4)   ;LOAD WITH A -2
1368 015276 042714 000100 5.8      BIC    #BIT6,@RCSR   ;CLEAR RX DONE
1369 015302 000002 4.8      RTI                      ;GO BACK TO MAINLINE ....
1370
1371                                     ;THE BELOW ROUTINE CHECKS THE DATA FOR END CHAR.
1372
1373 015304 142741 000200 7.0      RX.CK: BICB   #BIT7,-(R1)  ;CLEAR PARITY BIT.
1374 015310 142741 000200 7.0      BICB   #BIT7,-(R1)  ;CLEAR PARITY BIT.
1375 015314 123721 011041 4.7      CMPB   RX.TERM,(R1)+ ;WAS THE CHAR =001
1376 015320 001403 2.6      BEQ    RXISRDN      ;BR IF 001
1377 015322 123721 011041 4.7      CMPB   RX.TERM,(R1)+ ; 001?
1378 015326 001012 2.6      BNE    NO.12       ; NO 001
1379 015330
1380 015330 042714 000041 5.8      RXISRDN: BIC    #IE+.GO,@RCSR ;CLEAR GO AND INTERRUPT ENABLE
1381 015334 042714 000300 5.8      BIC    #300,@RCSR   ;CLEAR ALL RX DONES
1382 015340 052700 040000 3.8      BIS    #RFLG,STAT  ;SET RX COMPLETE FLAG
1383 015344 005037 011032 3.7      CLR    TIME        ;ZERO TIMER
1384 015350 005726 3.2      TST    (SP)+       ;POP SUBROUTINE POINTER.
1385 015352 000002 4.8      RTI                      ;ALL DONE GO HOME.....
1386 015354 032777 000040 173462 7.7      NO.12: BIT    #BIT5,@SWR ;MONITOR RX DATA??
1387 015362 001405 2.6      BEQ    1$          ;DON'T MONITOR
1388 015364 105777 173512 5.6      TSTB   @TPS        ;TTY READY??
1389 015370 100002 2.6      BPL    1$          ;NOT READY GO ON WITH TEST
1390 015372 114177 173506 8.8      MOVB   -(R1),@TPB  ;PRINT CHARACTER
1391 015376 000207 3.5      1$:    RTS    PC    ;GO TO RX ISR
1392 015400 032777 000040 173436 7.7      HBRCHK: BIT    #BIT5,@SWR ;MONITOR RX DATA?
1393 015406 001750 2.6      BEQ    RXISRDN    ;BR IF NO
1394 015410 105777 173466 5.6      TSTB   @TPS        ;TTY READY?
1395 015414 100345 2.6      BPL    RXISRDN    ;NO GO ON WITH TEST
1396 015416 112777 000122 173460 7.6      MOVB   #'R,@TPB   ;TYPE 'R'
1397 015424 000741 2.6      BR     RXISRDN

```

```
1398      ; ROUTINE TO GET TRANSMITTER OR RECEIVER ONTO
1399      ; THE PRIMARY REGISTERS IF THEY ARE ON THE SECONDARY.
1400
1401 015426 032764 000004 000002 GETPRI: BIT    #BIT2,XCSR(R4) ;IS THE TX ON THE SEC??
1402 015434 001424          BEQ    1$          ;BR IF NO
1403 015436 112764 000012 000005      MOVB   #12,REG(R4) ;SELECT THE MISC REG
1404 015444 012764 004010 000006      MOV    #4010,SEC(R4) ;SET EIGHT BITS AND TEST LOOP FOR CLK
1405 015452 112764 000007 000005      MOVB   #7,REG(R4) ;SELECT TX WC SEC
1406 015460 012764 177777 000006      MOV    #-1,SEC(R4) ;TX ONE CHAR
1407 015466 052764 000001 000002      BIS    #.GO,XCSR(R4) ;SET GO
1408 015474 032764 000100 000002      BIT    #BIT6,XCSR(R4) ;HANG HERE FOR SEC DONE
1409 015502 001774          BEQ    -6          ;KEEP WAITING FOR TX DONE SEC.
1410 015504 000750          BR     GETPRI      ;GO RECHECK AND CKECK RX
1411
1412 015506 032714 000004          1$:   BIT    #BIT2,@RCSR ;IS RX ON SECONDARY REG
1413 015512 001430          BEQ    2$          ;BR IF ON PRI.
1414 015514 112764 000012 000005      MOVB   #12,REG(R4) ;SELECT MISC REGISTERS
1415 015522 012764 004010 000006      MOV    #4010,SEC(R4) ;EIGHT BITS AND TEST LOOP (CLK)
1416 015530 112764 000005 000005      MOVB   #5,REG(R4) ;SELECT RX WC SEC
1417 015536 012764 177777 000006      MOV    #-1,SEC(R4) ;RX ONE CHAR
1418 015544 112764 000004 000005      MOVB   #4,REG(R4) ;SEL RX BA SEC.
1419 015552 012764 015624 000006      MOV    #NO.DAT,SEC(R4) ;LOAD RX BA SEC.
1420 015560 052714 010001          BIS    #BIT12+.GO,@RCSR ;SET GO!!!!!!!+ACTIVE
1421 015564 032714 000100          BIT    #BIT6,@RCSR ;HANG HERE FOR RX SEC DONE.
1422 015570 001775          BEQ    -4          ;KEEP WAITING
1423 015572 000715          BR     GETPRI      ;GO CHECK EVERY ONE
1424
1425 015574 112764 000012 000005      2$:   MOVB   #12,REG(R4) ;SELECT MISC REGISTER
1426 015602 042764 004010 000006      BIC    #4010,SEC(R4) ;CLEAR EIGHT BITS AND TEST LOOP
1427 015610 042714 000300          BIC    #300,@RCSR ;CLEAR RX DONES
1428 015614 042764 000300 000002      BIC    #300,XCSR(R4) ;CLEAR TX DONES
1429 015622 000207          RTS    PC          ;GO HOME
1430 015624 000000          NO.DAT: 0
1431
1431 015626 005015 042522 042503 MFULL: .ASCIIZ <15><12>/RECEIVER BUFFER FILLED. ERROR../
1431 015670 005015 042522 042503 MRXDA: .ASCIIZ <15><12>/RECEIVED DATA - /<15><12>
1431 015715 015 05'012 04'505 MRXSB: .ASCIIZ <15><12>/RECEIVED DATA SHOULD BE /<15><12>
```

1432

000001

.EVEN
.END

STPS = 104414	599#	927													
SWR 011044	624#	659	769	773	801	832	836	840	913	916	960	990	1072		
SW12 = 010000	1091	1104	1133	1235	1241	1386	1392								
SXCSR 011050	599#	1072	1091	1104	1133										
SYNC 014222	636#														
TEMP1 014214	1043	1055	1171#	1263*	1264*	1266									
TEMP2 014216	1028*	1030*	1077*	1079*	1168#										
TESTD 012510	1029*	1032*	1078*	1081*	1169#										
TESTDX 012714	771	834	915	958#											
TIE = 000040	971	977	980	1003#											
TIME 011032	599#	929	930												
	616#	691*	726*	761	766*	790*	797	798*	822*	825	830*	851*	855		
	860*	882*	887	892*	903	908*	912*	1035*	1068	1075*	1087	1094*	1096*		
	1100	1107*	1126*	1129	1136*	1249*	1383*								
TKB 011100	650#														
TKS 011076	649#														
TPB 011104	652#	1239*	1245*	1390*	1396*										
TPS 011102	651#	1237	1243	1388	1394										
TSTDAT 012544	959	961	966#												
TXOUT 014554	1251#														
TXWC 011062	642#	708*	1119												
TX.CK 014450	1205	1221	1231#												
TX.PRI 014330	1202#														
TX.SEC 014400	1201	1218#													
TX.TER 011040	619#	705	970	1231	1233										
TYPE = 104400	599#	962	965	975	992	994	996	998	1062	1291					
XCC 011066	644#														
XCSR = 000002	599#	670*	678*	690*	764	796	828	858	890	906	928	930*	948*		
	1036*	1037*	1038	1059*	1064	1066	1071	1085	1090	1097*	1098	1103	1125*		
	1127	1132	1161*	1162*	1185	1191	1192	1200	1212*	1229*	1247*	1288*	1320		
	1331	1401	1407*	1408	1428*										
XDA 011074	647#	844*													
XFLG = 100000	631#	791	800	853	862	883	896	901	910	1058	1246				
XISR 014230	664	1182#													
XISRDN 014534	1232	1234	1242	1244	1246#										
XLB 000004	599#	738													
XWAIT = 104412	599#														
X.X 011322	701#														
\$ILB 011670	737	820#	863												
\$OWI 011464	734	757#	777												
\$OWO 011572	731	788#	805												
\$XLB 012120	740	877#	917	918											
.	603#	742	846	997*	1031	1033	1046	1080	1082	1193	1329	1333	1409		
.	1422														
.GO - 000001	599#	1125	1162	1247	1299	1337	1380	1407	1420						

. ABS. 015752 000

ERRORS DETECTED: 0

DSKZ:CZDQJD,DSKZ:CZDQOD,SEQ=DSKZ:ITEP1.MAC,DSKZ:CZDQOD.P11
 RUN-TIME: 4 5 .3 SECONDS
 RUN-TIME RATIO: 172/10-16.0
 CORE USED: 16K (31 PAGES)

DOCUMENT PAGES: 34