

DJ11

DJ11 LOGIC TEST
MD-11-DZDJA-E

EP-DZDJA-E-DL-B
COPYRIGHT © 72-77
FICHE 1 OF 1

OCT 1977
digital
MADE IN USA

The image displays a grid of 48 small logic test diagrams, arranged in 8 rows and 6 columns. Each diagram is a schematic representation of a circuit component or test point, showing various electronic symbols, lines, and labels. The diagrams are densely packed and cover the left two-thirds of the page. The right third of the page is mostly blank, with a few faint, illegible markings near the bottom right corner.

B01

ECF:DFWDRS0411

00010000 771026

IDENTIFICATION

TOP:DZDJAESG

000:0000

77:026
SE:000:

PRODUCT CODE: MAINDEC-11-DZDJA-E-D
PRODUCT NAME: DJ11 LOGIC TESTS
PROGRAM DATE: MAY 1977
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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) 1972, 1977 BY DIGITAL EQUIPMENT CORPORATION

CONTENTS

1	ABSTRACT
1	REQUIREMENTS
1	Equipment
1	Storage
1	Preliminary Programs
1	LOADING PROCEDURE
1	STARTING PROCEDURE
1	Control Switch Settings
1	Starting Address
1	Program and Operator Action
1	OPERATING PROCEDURE
1	Operational Switch Settings
1	Subroutine Abstracts
1	Program and Operator Action
1	ERRORS
1	Error Printout
1	Error Recovery
1	Error Counter
1	RESTRICTIONS
1	MISCELLANEOUS
1	Execution Time
1	Stack Pointer
1	Pass Counter
1	Power Fail
1	PROGRAM DESCRIPTION

1. ABSTRACT

This program tests the logic of the DJ11 Asynchronous Multiplexer in maintenance mode. It checks that all the control registers function properly, that interrupts occur at the right level, and that data can be transmitted and received correctly. This program does not test that the input and output lead connections are functional. (See MAINDEC-11-DZDJB, PROGRAMS 2 and 3 for on-line testing). The program should be run for at least 2 passes with all switches down.

NOTE: The program will autosize the sio alarm level of the DJ11 and will run at any alarm level set. and will run at any alarm level set.

2. REQUIREMENTS

2.1 Equipment

PDP-11 standard computer with console teletype
Up to 16 DJ11 asynchronous multiplexers.

2.2 Storage

This program uses all of BK, except ABS LOADER.

2.3 Preliminary Programs

None

3. LOADING PROCEDURE

Use standard procedure for ABS tapes.

4. STARTING PROCEDURE

4.1 Control Switch Settings

See 5.1 (all down for worst case testing)

4.2 Starting Address

The program should always be started at 200. It may be restarted at 1000 after all parameters have been selected.

4.3 Program and Operator Action

- 1) Load program into memory using ABS loader.
- 2) Load address 200.
- 3) If hardware switch register is available, set switches (see sec. 5.1), all down for worst case, press start.
- 4) If switch-less processor simply press start.
- 5) Enter parameters (see sec. 5.3) as they are requested.
- 6) The program will loop and bell will ring once every pass. EOP^m is also printed on each pass.
- 7) A minimum of two passes should always be run.

5. OPERATING PROCEDURE

5.1 Operational Switch Settings

At SA 200, all switches down is worst case testing. Each subtest will be looped upon until completion of 16 passes of that subtest. The bell will ring upon completion of a pass of the entire program.

The switch settings are:

```

SW<15> = 1 ..... HALT ON ERROR
SW<14> = 1 ..... SCOPE LOOP
SW<13> = 1 ..... INHIBIT PRINTOUT
SW<12> = 1 ..... PRINT SILO ALARM LEVEL (IN OCTAL)
SW<11> = 1 ..... INHIBIT ITERATIONS OF SUBTEST
SW<10> = 1 ..... BELL ON ERROR
          0 ..... BELL ON PASS COMPLETE
SW<09> = 1 ..... LOOP ON ERROR
SW<08> = 1 ..... LOOP ON TEST IN SW<7:0>

```

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

(i.e.) 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. ? 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 tG (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.

5.2 Subroutine Abstracts

5.2.1 SCOPE

This subroutine call (via a TRAP instruction) is placed between each subtest in the instruction section. It records the starting address of each subtest as it is being entered in location "LAD". If a scope loop is requested, the current subtest will be looped upon. SW<11> on a 1 inhibits iteration of subtests. The contents of "LAD" may be used to determine the last subtest successfully completed.

5.2.2 HLT

This routine (called by an EMT instruction) prints out an error message (see 6.1). If SW<9> is on a 1 and a HLT is executed, the subtest will be looped upon until 16 consecutive good passes are completed. To inhibit typeouts, put SW<13> on a 1. To ring the bell on an error, put SW<10> on a 1.

5.2.4 ALMCK

in the normal operation the "done" bit is set as each character is read into the fi/fo buffer (silo). but this "done" condition can be delayed to cause done on the 5, 9, or 17 character. this is done by cutting one of the jumpers (w1,w2,w3) on the m7285 control board. the program tests for this "silo alarm level" and if sw12 is set (1) it will print out the level at which each dj11 will set "done." the subroutine also adjusts the character counters to ensure that the maximum number of characters to be transferred is a multiple of the silo alarm level. this ensures that all data will be read out of the silo. done will not set if the number of characters in the fi/fo buffer is less than the silo alarm level. (note character present is set on each character in the buffer, regardless of the silo alarm level.)

5.2.5 TRAPCATCHER

A ".+2" - "HALT" sequence is repeated from 0 - 56 to detect

any unexpected traps and a ".+2" - "IOT" sequence is repeated from 60 - 776 to detect any unexpected interrupts. Thus any unexpected traps will HALT at the vector + 2. Any unexpected interrupts will result in an error "HLT" in "IOTRAP".

5.3 Program and Operator Action

The following requests are made to the operator at the beginning of the program. A detailed description of what is required for each parameter is given bellow.

- 1) "FIRST DJ11 ADDRESS: "
The CSR address of the first DJ11 you wish to test. Must be between 160000(8) and 177777(8). The default (carriage return) is to 160010(8). "P(REVIOUS)" selects the address previously selected.
- 2) "VECTOR ADDRESS: "
The receiver interrupt vector address of the first DJ11 you wish to test. Must be between 300(8) and 1000(8). The default (carriage return) is to 300(8). "P(REVIOUS)" selects the address previously selected.
- 3) "NO. OF DJ11'S: "
The number of DJ11 units you wish to test at one time. Must be between 1 and 16. The default (carriage return) is to 1. "P(REVIOUS)" selects the number of units previously selected.
- 4) "STANDARD CONFIGURATION? "
"Y(ES)" or default (carriage return) selects 8 level code, no parity. "N(O)" causes requests for code level and parity on all requested lines in groups of four. "P(REVIOUS)" selects the code levels and parities previously selected.
- 5) "CHAR LENGTH: "
The code level for the line group specified. Must be 5, 6, 7, or 8. The default (carriage return) is to 8 level code.
- 6) "PARITY (NO, ODD, EVEN): "
The type of parity selected for the line group specified. The default (carriage return) is to no parity.

E. ERRORS

E.1 Error Printout

The format is as follows:

ADR DJADR (R1) (R2) (R3) (R4)

Where:

ADR = Address of error HLT
 DJADR = CSR address of DJ11 under test
 (Rn) = Contents of general register "n". From none to four of these may be typed depending on the number following the HLT; e.g. HLT+3 would type (R1) thru (R3); HLT (by itself) would stop after typing ADR and DJADR.

To find the failing test, look at the listing above the address typed. In most cases the comment beside the HLT tells what was being checked and what was expected. Also, a list of the probable failing logic is given in the comments at the beginning of the test.

6.2 Error Recovery

Restart at 200 or 1000.

6.3 Error Counter

An error count is kept in "ERRORS". It can be cleared from the console, by restarting at 200, or by reloading the program.

7. RESTRICTIONS

If more than one DJ11 is tested at a time, the device addresses and the vector addresses must all be contiguous.

If this program is run with a monitor, i.e. ACT11 or DDP, the device addresses must follow the floating address convention. DJ11's will be first, starting at 160010,.

8. MISCELLANEOUS

8.1 Execution Time

Due to the various baud rates available and the ability to check up to 16 DJ11's at once, the execution time can be anywhere from 15 seconds to three and a half hours. The following typical times are for one DJ11 with all lines at the same speed, 8 level code, 2 stop bits, and no parity, on a PDP-11/20 without trace trapping. For multiple DJ11's, multiply these times by the number of units selected for test.

BALC RUN TIME

75 00:26:00
110 00:18:00
134.5 00:15:00
150 00:13:00
300 00:06:30
600 00:03:15
1200 00:01:45
1800 00:01:20
2400 00:00:55
4800 00:00:30
9600 00:00:15

B.2 Stack Pointer

Stack is initially set to 1200

B.3 Pass Count

A 32 bit (2 words) pass count is kept in pcnt. It can be cleared from the console, by restarting at 200, or by reloading the program.

B.4 Power Fail

Each test can be power failed with no errors. To use, start the test as usual and power down then up at any time. The routine should type "POWER" and restart the program with no other error typeouts.

9. PROGRAM DESCRIPTION

This program tests the logic of up to 16 DJ11 asynchronous data multiplexers in maintenance mode. It checks that all the control registers function properly, that interrupts occur at the right priority level, and that data can be transmitted and received correctly. The program has many subtests (the code between 2 SCOPE statements) which are run 16 times before continuing to the next. SW<11> on a 1 causes each subtest to be run only once. SW<9> on a 1 enables loop on error. The address "ICNT" contains the iteration count in the left byte and the test number in the right byte. All the subtests should be run sequentially by starting at 200 not by starting at the beginning of the subtest. To loop on a particular subtest, put the test number (see listing) in the right byte of the switch register and SW<8> on a 1. This test will be looped upon until SW<8> is put on a 0 or the right byte is changed. If the test is non-existent, the program will be run as usual. If more than one DJ11 is selected, all the subtests are

MAINDEC-11-DZDJA-E-D DJ11 Logic Tests
Description

SEG 0009

performed on one unit at a time. The bell will not ring and
EOP print, until a pass has been made on each of the DJ11's
selected for test.

2506	SWITCH SETTINGS
2573	DJ11 SPECIFICATIONS
2630	TABLES AND "SCRATCH PAD"
2686	SETUP AREA
4560	TST1: TEST IF CSR EXISTS
4561	TST2: TEST BIT1 OF CSR
4562	TST3: TEST BIT2 OF CSR
4563	TST4: TEST BIT6 OF CSR
4564	TST5: TEST BIT8 OF CSR
4565	TST6: TEST BIT10 OF CSR
4566	TST7: TEST BIT12 OF CSR
4567	TST10: TEST BIT14 OF CSR
4568	TST11: TEST RECEIVER ENABLE AND CMOS BITS OF CSR
4569	TST12: TEST BYTE ADDRESSING OF CSR
4570	TST13: TEST BIS AND BIC ON CSR
4571	TST14: TEST READ/WRITE BITS OF TCR
4572	TST15: TEST BYTE ADDRESSING OF TCR
4573	TST16: TEST BIS AND BIC ON TCR
4576	TST17: TEST TRANSMIT READY
4577	TST20: TEST TRANSMIT READY
4578	TST21: TEST MASTER TRANSMIT SCAN ENABLE
4579	TST22: TEST TRANSMIT INTERRUPT LEVEL
4580	TST23: TEST TRANSMIT INTERRUPT LEVEL
4581	TST24: TEST TRANSMIT INTERRUPT LEVEL
4582	TST25: TEST TRANSMIT INTERRUPT LEVEL
4583	TST26: TEST TRANSMIT INTERRUPT LEVEL
4584	TST27: TEST TRANSMIT INTERRUPT LEVEL
4585	TST30: TEST TRANSMIT INTERRUPT LEVEL
4586	TST31: TEST TRANSMIT INTERRUPT LEVEL
4589	TST32: TEST ALL OF LINE 0 TRANSMIT AND RECEIVE LOGIC
4590	TST33: TEST ALL OF LINE 1 TRANSMIT AND RECEIVE LOGIC
4591	TST34: TEST ALL OF LINE 2 TRANSMIT AND RECEIVE LOGIC
4592	TST35: TEST ALL OF LINE 3 TRANSMIT AND RECEIVE LOGIC
4595	TST36: TEST ALL OF LINE 4 TRANSMIT AND RECEIVE LOGIC
4596	TST37: TEST ALL OF LINE 5 TRANSMIT AND RECEIVE LOGIC
4597	TST40: TEST ALL OF LINE 6 TRANSMIT AND RECEIVE LOGIC
4598	TST41: TEST ALL OF LINE 7 TRANSMIT AND RECEIVE LOGIC
4601	TST42: TEST ALL OF LINE 8 TRANSMIT AND RECEIVE LOGIC
4602	TST43: TEST ALL OF LINE 9 TRANSMIT AND RECEIVE LOGIC
4603	TST44: TEST ALL OF LINE 10 TRANSMIT AND RECEIVE LOGIC
4604	TST45: TEST ALL OF LINE 11 TRANSMIT AND RECEIVE LOGIC
4607	TST46: TEST ALL OF LINE 12 TRANSMIT AND RECEIVE LOGIC
4608	TST47: TEST ALL OF LINE 13 TRANSMIT AND RECEIVE LOGIC
4609	TST50: TEST ALL OF LINE 14 TRANSMIT AND RECEIVE LOGIC
4610	TST51: TEST ALL OF LINE 15 TRANSMIT AND RECEIVE LOGIC
4620	TST52: TEST CLEAR MOS
4621	TST53: TEST TRANSMIT READY
4622	TST54: TEST SILO ALARM LEVEL / RECEIVER ENABLE
4623	TST55: TEST HALF DUPLEX
4624	TST56: TEST RECEIVER INTERRUPT LEVEL
4625	TST57: TEST RECEIVER INTERRUPT LEVEL
4626	TST60: TEST FI/FO OVERRUN
4627	TST61: TEST RECEIVER UART OVERRUN ON ALL LINES
4628	TST62: TEST READ/WRITE BITS OF BCSR
4629	TST63: TEST BREAKS ON LINE 0
4630	TST64: TEST BREAKS ON ALL LINES

4632	TST65: TEST RESET
4633	TST66: TEST ALL DATA ON EACH LINE
4640	BELL AND SCOPE ROUTINE
4692	SILO ALARM LEVEL ROUTEEN
4747	SCOPE LOOP HANDLER
4750	HLT ROUTINE (ERROR TYPEOUT)
4765	TELETYPE INPUT ROUTINES
4803	TTY INPUT ROUTINE
4805	OCTAL DUMP OF A WORD
4807	POWER DOWN AND UP ROUTINES
4809	TYPE ROUTINE
4826	AUTOMATIC SYSTEM SIZER

2563
2564
2565
2566
2567
2568
2569
2570
2571
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610

.TITLE MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS
.ENABLE ABS
.ENABLE AMA
;COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
;PROGRAM BY KEN CHAPMAN
;MODIFIED BY DAVID L. ADAMS MAY 1977

	SWITCH	USE
	SW15= 100000	;HALT ON ERROR
	SW14= 40000	;LOOP ON TEST
	SW13= 20000	;INHIBIT ERROR TYPEOUTS
	SW12= 10000	;PRINT SILO ALARM LEVEL
	SW11= 4000	;INHIBIT ITERATIONS
	SW10= 2000	;0 - BELL ON PASS COMPLETE
		;1 - BELL ON ERROR
	SW9= 1000	;LOOP ON ERROR
	SW8= 400	;LOOP ON TEST IN SW<7:0>
	.REM!	

DJ11 REGISTER BIT ASSIGNMENTS:

CONTROL STATUS REGISTER (CSR) XXXXX0
BIT0 RECEIVER ENABLE (READ/WRITE)
BIT1 HALF DUPLEX SELECT (READ/WRITE)
BIT2 MAINTENANCE (READ/WRITE)
BIT3 CLEAR MOS (WRITE ONLY)
BIT4 CLEAR MOS FLAG (READ ONLY)
BIT5 NOT USED
BIT6 RECEIVER INTERRUPT ENABLE (READ/WRITE)
BIT7 DONE (READ ONLY)
BIT8 MASTER TRANSMITTER SCAN ENABLE (READ/WRITE)
BIT9 NOT USED
BIT10 READ/WRITE BREAK REGISTER (READ/WRITE)
BIT11 NOT USED
BIT12 STATUS ENABLE (READ/WRITE)
BIT13 FI/FO OVERRUN (READ ONLY)
BIT14 MASTER TRANSMITTER INTERRUPT ENABLE (READ/WRITE)
BIT15 TRANSMITTER READY (READ ONLY)

RECEIVER BUFFER REGISTER (RBUF) XXXXX2 (READ ONLY)

BIT0-7 RECEIVED CHARACTER
BIT8-11 LINE NUMBER
BIT12 PARITY ERROR
BIT13 FRAMING ERROR
BIT14 UART OVERRUN ERROR
BIT15 CHARACTER PRESENT

TRANSMITTER CONTROL REGISTER (TCR) XXXXX4 (READ/WRITE)

BIT0-15 STOP THE SCANNER ON CORRESPONDING LINE

TRANSMITTER BUFFER (TBUF) XXXXX6

2611
2612
2613
2614
2615
2616
2617
2618
2619

BIT0-7 TRANSMITTED CHARACTER (WRITE ONLY)
BIT8-11 LINE NUMBER (READ ONLY)

BREAK CONTROL STATUS REGISTER (BCSR) XXXXX4 (BIT10 OF CSR SET) (READ/WRITE)

BIT0-15 TRNSMIT A BREAK ON CORRESPONDING LINE!

(1) 104400
(1) 104000
(1) 000004
(1) 177776
(1) 000000
(1) 000001
(1) 000002
(1) 000003
(1) 000004
(1) 000005
(1) 000005
(1) 000006
(1) 000007
(1) 000007
(1) 000001
(1) 000002
(1) 000004
(1) 000010
(1) 000020
(1) 000040
(1) 000100
(1) 000200
(1) 000400
(1) 001000
(1) 002000
(1) 004000
(1) 010000
(1) 020000
(1) 040000
(1) 100000
2620 000340
2621 000000
2622 000020
2623 001200

SCOPE= TRAP
HLT= EMT
TYPE= IOT
PS= 177776
R0= %0
R1= %1
R2= %2
R3= %3
R4= %4
R5= %5
TTY= %5
SP= %6
PC= %7
BELL= 7
BIT0= 1
BIT1= 2
BIT2= 4
BIT3= 10
BIT4= 20
BIT5= 40
BIT6= 100
BIT7= 200
BIT8= 400
BIT9= 1000
BIT10= 2000
BIT11= 4000
BIT12= 10000
BIT13= 20000
BIT14= 40000
BIT15= 100000
LEVEL7 = 340
OPEN = 0
DJMXNO=16.
STACK =1200

;MAX ALLOWED NUMBER OF DJ11'S

2625 000000 000000
 2626 000000 000000
 2627 000046 015354
 2628 000174 000000
 2629 000176 000000
 2630 000200 001402
 2631 001000 001000
 2632 001000 002330
 2633 001200
 2634
 2635
 2636
 2637
 2638
 2639
 2640
 2641
 2642
 2643
 2644 001200 000100
 2645 001300 000000
 2646 001302 000000
 2647 001304 000000
 2648 001306 000000
 2649 001310 000000
 2650 001312 000000
 2651 001314 000000 000000
 2652
 2653 001320 160010
 2654 001322 160012
 2655 001324
 2656 001324 160014
 2657 001326 160016
 2658
 2659 001330 000300
 2660 001332 000302
 2661 001334 000304
 2662 001336 000306
 2663
 2664 001340 160010
 2665 001342 000300
 2666 001344 000001
 2667 001346 000000
 2668 001350 000000
 2669 001352 000000

= 0
 0,0
 :TRAP CATCHER IN LOCATIONS 0 THRU 776
 :LOCATIONS 0 AND 2 CONTAIN "HALT" INSTRUCTIONS
 :LOCATIONS 4 THRU 56 CONTAIN ".+2" AND "HALT" IN EVERY VECTOR
 :LOCATIONS 60 THRU 776 CONTAIN ".+2" AND "IOT" IN EVERY VECTOR

= 46
 SENDAD

= 174
 DISPREG: 0
 SWREG: 0

= 200
 JMP BEGIN :200 ALWAYS IS THE STARTING ADDRESS
 = 1000
 JMP RESTART :RESTART ADDRESS

= 1200
 ADDRESS TABLES

"LENGTH" TABLE IS SET SO THE FIRST FOUR BYTES (2 WORDS) CONTAIN
 THE CHARACTER LENGTH FOR THE 4 LINE GROUPS OF THE FIRST DJ11, ETC.

"PARITY" TABLE HAS 4 WORDS, ONE FOR EACH LINE GROUP. THE FIRST WORD CONTAINS
 THE ODD PARITY FLAGS FOR LINES 0-3 OF ALL DJ11'S. THE NEXT WORD
 HAS PARITY FLAGS FOR LINES 4-7, ETC. EACH BIT IN THE WORD REPRESENTS
 EACH DJ11. BIT 0 IN EACH OF THE FOUR WORDS IS FOR DJ/0 TO BIT
 15 FOR DJ/15.

LENGTH: .BLKB 100
 PARITY: 0
 PARIT2: 0
 PARIT3: 0
 PARIT4: 0
 ICNT: 0
 ERRORS: 0
 PCNT: 0,0

:TABLE OF CHAR. LENGTHS (MASK)
 :ODD PARITY FLAGS FOR LINES 0-3
 :ODD PARITY FLAGS FOR LINES 4-7
 :ODD PARITY FLAGS FOR LINES 10-13
 :ODD PARITY FLAGS FOR LINES 14-17
 :ITERATION COUNT-LH, TEST NO.-RH
 :ERROR COUNT REGISTER
 :PASS COUNT REGISTER

CSR: 160010
 RBUF: 160012
 TCR:
 BCSR: 160014
 TBUF: 160016

:CONTROL STATUS REGISTER(DJ UNDER TEST)
 :RECEIVER BUFFER REGISTER(DJ UNDER TEST)
 :TRANSMITTER CONTROL REGISTER(DJ UNDER TEST)
 :BREAK STATUS REGISTER(DJ UNDER TEST)
 :TRANSMITTER BUFFER REGISTER(DJ UNDER TEST)

RCVVEC: 300
 RCVLVL: 302
 XMTVEC: 304
 XMTLVL: 306

:RECEIVER INTERRUPT VECTOR ADDRESS(DJ UNDER TEST)
 :TRANSMITTER INTERRUPT VECTOR ADDRESS(DJ UNDER TEST)

DEVADR: 160010
 VECADR: 300
 UNITS: 1
 TIMER: 0
 ALMFLG: 0
 TIMERA: 0

:FIRST DEVICE ADDRESS(SELECTED)
 :FIRST VECTOR ADDRESS(SELECTED)
 :NUMBER OF UNITS TO BE TESTED
 :USED TO SAVE RELATIVE TIMES
 :TEST FLAG FOR SILO ALARM LEVEL
 :TIME COUNTERS

C02

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 44-3
TABLES AND "SCRATCH PAD"

SEG 0015

2670 001354 000000
 2671 001356 000000
 2672 001360 000000
 2673
 2674 001362 000000
 2675 001364 000000
 2676 001366 000000
 2677 001370 160010
 2678 001372 000300
 2679
 (1) 001374 177570
 (1) 001376 177570
 (1)
 2680
 2681 001400 177777
 2682
 2683

TIMERS: 0
 COUNT: 0
 SUM: 0

 DJUUT: 0
 DJLEN: 0
 DJPAR: 0
 DEFADR: 160010
 DEFVEC: 300

 SWR: 177570
 DISPLAY: 177570

 FTIME: -1

: CHAR COUNTER
 : SUMATION OF ALARM LEVEL CHAR'S

 : UNIT NUMBER OF DJ11 UNDER TEST
 : CHAR MASK TABLE POINTER
 : UNIT FLAG (FOR ODD PARITY FLAG)
 : DEFAULT FIRST DEVICE ADDRESS
 : DEFAULT FIRST VECTOR ADDRESS


```

2688
2689 001402 012706 001200 BEGIN: MOV #STACK, SP ;SET UP STACK POINTER
2690 001406 004737 020124 JSR PC SUSWRR ;CHECK FOR SWITCH REGISTER
2691 001412 012700 000020 MOV #20, R0
2692 001416 012720 017204 MOV #IOTRAP, (R0)+ ;IOT VECTOR (20)
2693 001422 012720 000340 MOV #340, (R0)+
2694 001426 012720 017044 MOV #PDOWNS, (R0)+ ;POWER FAIL VECTOR (24)
2695 001432 012720 000340 MOV #340, (R0)+
2696 001436 012720 016074 MOV #EMTS, (R0)+ ;EMT VECTOR (30)
2697 001442 012720 000340 MOV #340, (R0)+
2698 001446 012720 015732 MOV #TRAPS, (R0)+ ;TRAP VECTOR (34)
2699 001452 012720 000340 MOV #340, (R0)+
2700 001456 005037 001312 CLR ERRORS ;CLEAR ERROR COUNTER
2701 001462 005037 001314 CLR PCNT ;CLEAR PASS COUNTER
2702 001466 005037 001316 CLR PCNT+2
2703 001472 005037 001350 CLR ALMFLG
2704 001476 005737 000042 TST #42 ;CHECK FOR ACT11 OR DDP PRESENT
2705 001502 001404 BEQ GETADR ;BRANCH IF NONE
2706 001504 004737 017704 JSR PC AUTO ;GO TO SUBROUTINE TO "MAP" DJ11 ON THE SYS
2707 001510 000137 002330 JMP RESTAR ;SKIP OPERATOR ACTION
2708
2709 001514 000004 017633 GETADR: TYPE, MT:ILE
2710 001520 000004 017344 TYPE, RETURN
2711 001524 000004 017352 TYPE, MSGADR ;TYPE "FIRST DJ11 ADDRESS"
2712 001530 004537 016316 JSR R5, READIN ;READ INPUT FROM TTY AND SAVE
2713 001534 001340 .WORD DEVADR ;IN DEVADR
2714 001536 001366 BNE GETADR ;BRANCH IF BAD INPUT
2715 001540 005737 001340 TST DEVADR
2716 001544 001003 BNE 1$
2717 001546 013737 001370 001340 MOV DEFADR, DEVADR
2718 001554 042737 000007 001340 1$: BIC #7, DEVADR
2719 001562 022737 160000 001340 CMP #160000, DEVADR
2720 001570 101351 BHI GETADR
2721
2722 001572 000004 017402 GETVEC: TYPE, MSGVEC ;TYPE "VECTOR ADDRESS:"
2723 001576 004537 016316 JSR R5, READIN ;READ INPUT FROM TTY AND SAVE
2724 001602 001342 .WORD VECADR ;IN VECADR
2725 001604 001372 BNE GETVEC ;BRANCH IF BAD INPUT
2726 001606 005737 001342 TST VECADR ;CHECK FOR CR
2727 001612 001003 BNE 1$
2728 001614 012737 000300 001342 MOV #300, VECADR ;SET TO FIRST FLOATING VECTOR
2729 001622 042737 000007 001342 1$: BIC #7, VECADR ;CLEAR TO MODULO 10
2730 001630 022737 000300 001342 CMP #300, VECADR ;CHECK FOR LOW LIMIT
2731 001636 003355 BGT GETVEC
2732 001640 022737 001000 001342 CMP #1000, VECADR ;CHECK FOR UPPER LIMIT
2733 001646 003751 BLE GETVEC
2734 001650 000004 017426 GETNUM: TYPE, MSGNUM ;TYPE "NUMBER OR UNITS:"
2735 001654 004737 016464 JSR PC, READS ;READ INPUT FROM THE TTY
2736 ; CHECK STRING FOR VALID DIGITS, TERMINATOR, AND 'P'
2737 001660 012702 016612 MOV #INPUT, R2 ;POINTS TO INPUT STRING
2738 001664 112201 MOV# (R2)+, R1 ;LOAD 1ST CHAR
2739 001666 001434 BEQ 1$ ;BRANCH IF IMMEDIATE TERMINATOR
2740 001670 122701 000120 CMPB #'P', R1 ;CHECK FOR A P
2741 001674 001443 BEQ CONFIG ;CONFIGURE MODE IF SO
2742 001676 120127 000071 CMPB R1, #71 ;MUST BE A VALID ASCII NUMBER
2743 001702 101362 BHI GETNUM ;ELSE RETRY

```

2744	001704	162701	000060		SUB	#60,R1		:STRIP ASCII CODE
2745	001710	003757			BLE	GETNUM		:SHOULDN'T BE ZERO OR NEG
2746								:1ST CHAR IS A VALID DIGIT, 1 THRU 9 - CHECK REST OF STRING
2747	001712	105722			TSTB	(R2)+		:2ND CHAR A TERMINATOR?
2748	001714	001423			BEQ	3\$:YES - R1 HAS THE FINAL # OF UNITS
2749	001716	105712			TSTB	(R2)		:NO - NXT CHAR MUST BE THE TERMINATOR
2750	001720	001353			BNE	GETNUM		:ELSE RETRY.
2751	001722	124227	000071		CMPB	-(R2),#71		:CHECK 2ND CHAR FOR A VALID NUMBER
2752	001726	101350			BHI	GETNUM		
2753	001730	111203			MOVB	(R2),R3		:MAKE IT MORE ACCESSIBLE
2754	001732	162703	000060		SUB	#60,R3		:STRIP ASCII CODE
2755	001736	100744			BMI	GETNUM		:SHOULDN'T BE NEGATIVE
2756								:BOTH LOW AND HIGH ORDER DIGITS ARE OK - CONVERT DECIMAL VALUE.
2757	001740	010146			MOV	R1,-(SP)		:SAVE IT FOR LATER
2758	001742	006301			ASL	R1		:MULT HI ORDER BY 8
2759	001744	006301			ASL	R1		
2760	001746	006301			ASL	R1		
2761	001750	006316			ASL	(SP)		:MULTIPLY IT BY 2
2762	001752	062601			ADD	(SP)+,R1		:RESULT IS (HI ORD)*10.
2763	001754	060301			ADD	R3,R1		:NOW ADD IN THE LOW ORDER
2764	001756	000402			BR	3\$		
2765	001760	012701	000001		IS: MOV	#1,R1		:LOAD DEFAULT VALUE HERE
2766					:R1 HAS	THE CONVERTED VALUE - COMPARE AGAINST MAX ALLOWED.		
2767	001764	020127	000020		3\$: CMP	R1,#DJMXNO		:TOO BIG?
2768	001770	101327			BHI	GETNUM		:YES - RETRY.
2769	001772	010137	001344		MOV	R1,UNITS		:VALUE OK - STORE RESULT
2770	001776	012737	000100	001362	MOV	#100,DJUUT		: "PRIME" UNIT UNDER TEST NUMBER
2771								
2772	002004	000004	017451		CONFIG: TYPE,	MSGCON		:TYPE "STANDARD CONFIG"
2773	002010	004737	016464		JSR	PC	READS	:READ INPUT FROM TTY
2774	002014	122737	000120	016612	CMPB	#'P	INPUT	:CHECK FOR "P"
2775	002022	001542			BEQ	RESTAR		:BRANCH IF PREVIOUS
2776	002024	012700	000044		MOV	#44,R0		:SET UP COUNTER
2777	002030	012701	001200		MOV	#LENGTH,R1		:POINT TO CHAR LEN TABLE
2778	002034	005021			IS: CLR	(R1)+		:PUT CHAR MASK FOR 8 IN CHAR TABLE
2779								:AND CLR PARITY TABLE
2780	002036	005300			DEC	R0		:COUNT DOWN
2781	002040	001375			BNE	IS		:BRANCH IF NOT DONE
2782	002042	105737	016612		TSTB	INPUT		:CHECK FOR CR
2783	002046	001530			BEQ	RESTAR		:BRANCH IF DEFAULT
2784	002050	122737	000131	016612	CMPB	#131	INPUT	:CHECK FOR "Y"
2785	002056	001524			BEQ	RESTAR		:BRANCH IF DEFAULT
2786	002060	122737	000116	016612	CMPB	#116	INPUT	:CHECK FOR "N"
2787	002066	001346			BNE	CONFIG		:BRANCH IF ILLEGAL ENTRY
2788	002070	005000			CLR	R0		:CLR UNIT COUNTER
2789	002072	005001			CLR	R1		:CLR LINE COUNTER
2790	002074	012702	001200		MOV	#LENGTH,R2		:SET UP POINTER TO CHAR MASK TABLE
2791	002100	012703	001300		MOV	#PARITY,R3		:SET UP POINTER TO PARITY TABLE
2792	002104	012704	000001		MOV	#1,R4		:SET UP MARKER
2793	002110	000004	017505		TYPLIN: TYPE,	MSGLIN		:TYPE "LINES"
2794	002114	010105			MOV	R1,TTY		:TYPE R1 IN OCTAL
2795	002116	004737	016672		JSR	PC,PRINTS		:AND SUPPRESS LEADING ZERO'S
2796	002122	000004	017516		TYPE,	MSGDAS		:TYPE A DASH (-)
2797	002126	062701	000003		ADD	#3,R1		:LAST LINE IN GROUP OF 4
2798	002132	010105			MOV	R1,TTY		:TYPE R1 IN OCTAL
2799	002134	004737	016672		JSR	PC,PRINTS		:AND SUPPRESS LEADING ZERO'S

F02

2798	002140	005201			INC	R1		;FIRST LINE NEXT GROUP
2799	002142	000004	017522		GETLEN: TYPE,	MSGLEN		;TYPE "CHAR LENGTH: "
2800	002146	005037	016610		CLR	INHRE		
2801	002152	004737	016464		JSR	PC	READS	;READ FROM THE TTY
2802	002156	105737	016612		TSTB	INPUT		;CHECK FOR CR (DEFAULT)
2803	002162	001421			BEQ	GETPAR		;BRANCH IF DEFAULT
2804	002164	105737	016613		TSTB	INPUT+1		;CHECK FOR BAD DATA
2805	002170	001364			BNE	GETLEN		;BRANCH IF BAD
2806	002172	162737	000060	016612	SUB	#60,	INPUT	;CONVERT FROM ASCII
2807	002200	112712	177777		MOVB	#-1,	(R2)	;SET UP MASK
2808	002204	106312		3\$:	ASLB	(R2)		;CLEAR MASK BIT BY BIT
2809	002206	103355			BCC	GETLEN		;BAD INPUT, > 8
2810	002210	105337	016612		DECB	INPUT		;COUNT
2811	002214	001373			BNE	2\$;BRANCH IF NOT DONE
2812	002216	132712	000037		BITB	#37,	(R2)	;CHECK FOR 5 CHAR
2813	002222	001347			BNE	GETLEN		;BAD INPUT, < 5
2814	002224	005202			INC	R2		;INC TO NEXT LINE GROUP
2815	002226	000004	017543		GETPAR: TYPE,	MSGPAR		;TYPE "PARITY (NO, ODD, EVEN):
2816	002232	005037	016610		CLR	INHRE		
2817	002236	004737	016464		JSR	PC READS		;READ INPUT FROM TTY
2818	002242	005737	016612		TST	INPUT		;CHECK FOR CR
2819	002246	001415			BEQ	3\$;BRANCH IF DEFAULT
2820	002250	122737	000116	016612	CMPB	#116, INPUT		;CHECK FOR "N"
2821	002256	001411			BEQ	3\$;BRANCH IF "NO"
2822	002260	122737	000105	016612	CMPB	#105, INPUT		;CHECK FOR "E"
2823	002266	001405			BEQ	3\$;BRANCH IF "EVEN"
2824	002270	122737	000117	016612	CMPB	#117, INPUT		;CHECK FOR "O"
2825	002276	001353			BNE	GETPAR		;BRANCH IF NONE
2826	002300	050413			BIS	R4,	(R3)	;SET THE ODD PARITY FLAG
2827	002302	005723		3\$:	TST	(R3)+		;POINT TO THE NEXT PARITY WORD
2828	002304	032701	000017		BIT	#17, R1		;CHECK FOR NEXT UNIT
2829	002310	001277			BNE	TYPLIN		;BRANCH IF NOT
2830	002312	012703	001300		MOV	#PARITY, R3		;RESET PARITY TABLE POINTER
2831	002316	006304			ASL	R4		;FLAG TO NEXT UNIT
2832	002320	005200			INC	R0		;COUNT UNITS
2833	002322	020037	001344		CMP	R0, UNITS		;CHECK FOR LAST
2834	002326	001270			BNE	TYPLIN		;BRANCH IF MORE

```

2836
2837 002330 000005          RESTAR: RESET          ;ISSUE RESET
2838 002332 012706 001200  MOV    #STACK, SP      ;SET UP STACK POINTER
2839 002336 005737 001400  TST   FTIME
2840 002342 001410          BEQ   CLRVEC
2841 002344 022737 000176 001374  CMP   #SWREG, SWR
2842 002352 001004          BNE   CLRVEC
2843 002354 004737 020204  JSR   PC, CNTLU
2844 002360 005037 001400  CLR   FTIME
2845 002364 012700 000300  CLRVEC: MOV #300,RO      ;BEGINNING OF FLOATING VECTORS
2846 002370 005720          IS: TST (RO)+
2847 002372 010060 177776  MOV   RO,-2(RO)      ;"-+2"
2848 002376 012720 000004  MOV   #IOT, (RO)+   ;"-IOT"
2849 002402 022700 001000  CMP   #1000,RO
2850 002406 001370          BNE   IS
2851 002410 062737 000010 001320  ADD   #10, CSR      ;UPDATE DEVICE ADDRESS TO NEXT UNIT
2852 002416 062737 000010 001330  ADD   #10, RCVVEC   ;UPDATE DEVICE VECTOR ADDRESS
2853 002424 062737 000004 001364  ADD   #4, DJLEN     ;MOVE CHAR TABLE POINTER
2854 002432 006337 001366          ASL   DJPAR        ;UPDATE PARITY FLAG MARKER
2855 002436 023737 001362 001344  CMP   DJUUT,UNITS
2856 002444 002416          BLT   2$
2857 002446 005037 001362          CLR   DJUUT
2858 002452 013737 001340 001320  MOV   DEVAOR,CSR
2859 002460 013737 001342 001330  MOV   VECADR,RCVVEC
2860 002466 012737 001200 001364  MOV   #LENGTH,DJLEN ;INIT CHAR TABLE POINTER
2861 002474 012737 000001 001366  MOV   #1,DJPAR     ;INIT PARITY FLAG MARKER
2862 002502 005237 001362          2$: INC  DJUUT
2863 002506 013701 001320          MOV   CSR, R1      ;SET UP ALL THE REGISTER ADDRESSES
2864 002512 062701 000002          ADD  #2, R1        ;ADD 2
2865 002516 010137 001322          MOV  R1, RBUF     ;SET UP RECEIVER BUFFER
2866 002522 062701 000002          ADD  #2, R1        ;ADD 2
2867 002526 010137 001324          MOV  R1, TCR      ;SET UP TRANSMITTER CONTROL REG
2868                                     AND  AND,BREAK     ;AND BREAK STATUS REG
2869 002532 062701 000002          ADD  #2, R1        ;ADD 2
2870 002536 010137 001326          MOV  R1, TBUF     ;SET UP TRANSMITTER BUFFER
2871 002542 013701 001330          MOV  RCVVEC, R1   ;POINTER FOR VECTOR SETUP
2872 002546 005721          TST  (R1)+        ;INC R1
2873 002550 010137 001332          MOV  R1, RCVLVL   ;SET INT LVL
2874 002554 005721          TST  (R1)+        ;INC R1
2875 002556 010137 001334          MOV  R1, XMTVEC   ;TRANSMITTER VECTOR
2876 002562 005721          TST  (R1)+        ;INC R1
2877 002564 010137 001336          MOV  R1, XMTLVL   ;XMT INT LVL ADP
2878 002570 005037 001310          CLR  ICNT
2879 002574 005037 016070          CLR  LAD
2880          SCOPE
2881
2882

```



```

(1) 002754 032777 177773 176336 BIT #177773, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 002762 001401 BEQ .+4 ;BRANCH IF OK
(1) 002764 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 002766 005077 176326 CLR @CSR ;CLEAR BIT2
(1) 002772 032777 000004 176320 BIT #BIT2, @CSR ;CHECK THAT BIT2 IS CLEARED
(1) 003000 001401 BEQ .+4 ;BRANCH IF OK
(1) 003002 104000 HLT ;CSR BIT2 FAILED TO CLEAR
(1)
(1) 003004 104400 SCOPE

```

4563

```

;*****
;TEST 4: TEST THAT CSR BIT6 CAN BE SET AND CLEARED
;PROBABLE FAULTY LOGIC: M7285 (D2-2) E4, E18, (D2-4) E47, E64
;*****

```

```

(1) 003006 012777 000100 176304 TST4: MOV #BIT6, @CSR ;SET BIT6
(1) 003014 032777 000100 176276 BIT #BIT6, @CSR ;CHECK THAT BIT6 IS SET
(1) 003022 001001 BNE .+4 ;BRANCH IF OK
(1) 003024 104000 HLT ;CSR BIT6 FAILED TO SET
(1)
(1) 003026 032777 177677 176264 BIT #177677, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 003034 001401 BEQ .+4 ;BRANCH IF OK
(1) 003036 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 003040 005077 176254 CLR @CSR ;CLEAR BIT6
(1) 003044 032777 000100 176246 BIT #BIT6, @CSR ;CHECK THAT BIT6 IS CLEARED
(1) 003052 001401 BEQ .+4 ;BRANCH IF OK
(1) 003054 104000 HLT ;CSR BIT6 FAILED TO CLEAR
(1)
(1) 003056 104400 SCOPE

```

4564

```

;*****
;TEST 5: TEST THAT CSR BIT8 CAN BE SET AND CLEARED
;PROBABLE FAULTY LOGIC: M7285 (D2-2) E6, E18, (D2-4) E47, E31
;*****

```

```

(1) 003060 012777 000400 176232 TST5: MOV #BIT8, @CSR ;SET BIT8
(1) 003066 032777 000400 176224 BIT #BIT8, @CSR ;CHECK THAT BIT8 IS SET
(1) 003074 001001 BNE .+4 ;BRANCH IF OK
(1) 003076 104000 HLT ;CSR BIT8 FAILED TO SET
(1)
(1) 003100 032777 177377 176212 BIT #177377, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 003106 001401 BEQ .+4 ;BRANCH IF OK
(1) 003110 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 003112 005077 176202 CLR @CSR ;CLEAR BIT8
(1) 003116 032777 000400 176174 BIT #BIT8, @CSR ;CHECK THAT BIT8 IS CLEARED
(1) 003124 001401 BEQ .+4 ;BRANCH IF OK
(1) 003126 104000 HLT ;CSR BIT8 FAILED TO CLEAR
(1)
(1) 003130 104400 SCOPE

```

4565

```

;*****
;TEST 6: TEST THAT CSR BIT10 CAN BE SET AND CLEARED
;PROBABLE FAULTY LOGIC: M7285 (D2-2) E30, E24, (D2-4) E47, E31
;*****

```

```

(1) ;*****
(1)
(1) 003132 012777 002000 176160 TST6: MOV #BIT10, @CSR ;SET BIT10
(1) 003140 032777 002000 176152 BIT #BIT10, @CSR ;CHECK THAT BIT10 IS SET
(1) 003146 001001 BNE .+4 ;BRANCH IF OK
(1) 003150 104000 HLT ;CSR BIT10 FAILED TO SET
(1)
(1) 003152 032777 175777 176140 BIT #175777, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 003160 001401 BEQ .+4 ;BRANCH IF OK
(1) 003162 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 003164 005077 176130 CLR @CSR ;CLEAR BIT10
(1) 003170 032777 002000 176122 BIT #BIT10, @CSR ;CHECK THAT BIT10 IS CLEARED
(1) 003176 001401 BEQ .+4 ;BRANCH IF OK
(1) 003200 104000 HLT ;CSR BIT10 FAILED TO CLEAR
(1)
(1) 003202 104400 SCOPE
4566

```

```

(1) ;*****
(1) ;TEST 7: TEST THAT CSR BIT12 CAN BE SET AND CLEARED
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E5, E1, (D2-4) E47, E31
(1) ;*****

```

```

(1) 003204 012777 010000 176106 TST7: MOV #BIT12, @CSR ;SET BIT12
(1) 003212 032777 010000 176100 BIT #BIT12, @CSR ;CHECK THAT BIT12 IS SET
(1) 003220 001001 BNE .+4 ;BRANCH IF OK
(1) 003222 104000 HLT ;CSR BIT12 FAILED TO SET
(1)
(1) 003224 032777 167777 176066 BIT #167777, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 003232 001401 BEQ .+4 ;BRANCH IF OK
(1) 003234 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 003236 005077 176056 CLR @CSR ;CLEAR BIT12
(1) 003242 032777 010000 176050 BIT #BIT12, @CSR ;CHECK THAT BIT12 IS CLEARED
(1) 003250 001401 BEQ .+4 ;BRANCH IF OK
(1) 003252 104000 HLT ;CSR BIT12 FAILED TO CLEAR
(1)
(1) 003254 104400 SCOPE
4567

```

```

(1) ;*****
(1) ;TEST 10: TEST THAT CSR BIT14 CAN BE SET AND CLEARED
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E3, E1, (D2-4) E47, E31
(1) ;*****

```

```

(1) 003256 012777 040000 176034 TST10: MOV #BIT14, @CSR ;SET BIT14
(1) 003264 032777 040000 176026 BIT #BIT14, @CSR ;CHECK THAT BIT14 IS SET
(1) 003272 001001 BNE .+4 ;BRANCH IF OK
(1) 003274 104000 HLT ;CSR BIT14 FAILED TO SET
(1)
(1) 003276 032777 137777 176014 BIT #137777, @CSR ;CHECK THAT NO OTHER BIT SET
(1) 003304 001401 BEQ .+4 ;BRANCH IF OK
(1) 003306 104000 HLT ;EXTRA BIT SET IN CSR
(1)
(1) 003310 005077 176004 CLR @CSR ;CLEAR BIT14
(1) 003314 032777 040000 175776 BIT #BIT14, @CSR ;CHECK THAT BIT14 IS CLEARED
(1) 003322 001401 BEQ .+4 ;BRANCH IF OK

```

(1) 003324 104000
(1)
(1) 003326 104400
4568

HLT ;CSR BIT14 FAILED TO CLEAR
SCOPE

;TEST 11: TEST THAT RECEIVER ENABLE (BIT0 OF THE CSR) CAN BE SET
AND CLEARED, AND THAT CLEAR MOS (BIT3) IS WRITE ONLY.
;PROBABLE FAULTY LOGIC: M7285 (D2-2) E26,E36,E7,E24, (D2-8) E17,E14,E15

(1) 003330 012777 000004 175762
(1) 003336 005005
(1) 003340 052777 000010 175752
(1) 003346 017701 175746
(1) 003352 032701 000010
(1) 003356 001401
(1) 003360 104001
(1)
(1) 003362 032701 000020
(1) 003366 001403
(1) 003370 105305
(1) 003372 001365
(1) 003374 104001
(1)
(1) 003376 022701 000004
(1) 003402 001401
(1) 003404 104001
(1)
(1) 003406 052777 000001 175704
(1) 003414 017701 175700
(1) 003420 032777 000001 175672
(1) 003426 001001
(1) 003430 104001
(1)
(1) 003432 022777 000005 175660
(1) 003440 001401
(1) 003442 104001
(1)
(1)
(1)
(1) 003444 042777 000001 175646
(1) 003452 017701 175642
(1) 003456 022777 000004 175634
(1) 003464 001401
(1) 003466 104001
(1)
(1)
(1)
(1)

TST11: MOV #BIT2, @CSR ;SET MAINTENANCE MODE (BIT2)
CLR R5 ;SET UP COUNTER
BIS #BIT3, @CSR ;SET CLEAR MOS (BIT3)
1\$: MOV @CSR, R1 ;SAVE CSR
BIT #BIT3, R1 ;CHECK CLEAR MOS (BIT3)
BEQ .+4 ;BRANCH IF OK
HLT+1 ;CLEAR MOS (BIT3) SET (WRITE-ONLY)
R1 = CONTENTS OF CSR
BIT #BIT4, R1 ;CHECK CLEAR MOS FLAG
BEQ 2\$;BRANCH IF CLEARED
DECB R5 ;WAIT FOR MOS TO CLEAR
BNE 1\$;BRANCH IF MORE TIME
HLT+1 ;CLEAR MOS FLAG (BIT4) FAILED TO CLEAR
R1 = CONTENTS OF CSR
2\$: CMP #BIT2, R1 ;CHECK THAT ONLY MAINTENANCE BIT SET
BEQ .+4 ;BRANCH IF OK
HLT+1 ;CLEAR MOS CLEARED MAINTENANCE
OR SET OTHER CSR BITS
R1 = CONTENTS OF CSR
BIS #BIT0, @CSR ;SET RECEIVER ENABLE
MOV @CSR, R1 ;SAVE CSR
BIT #BIT0, @CSR ;CHECK THAT RECEIVER ENABLE SET
BNE .+4 ;BRANCH IF OK
HLT+1 ;RECEIVER ENABLE FAILED TO SET
R1 = CONTENTS OF CSR
CMP #5, @CSR ;CHECK REST OF CSR
BEQ .+4 ;BRANCH IF OK
HLT+1 ;CSR ERROR
R1 = CONTENTS OF CSR
;NOTE: IF THE TTY MODULE IS BEING USED AND DONE (BIT7) IS SET,
; THE ERROR COULD BE DUE TO MAINTENANCE OR CLEAR MOS NOT WORKING.
BIC #BIT0, @CSR ;CLEAR RECEIVER ENABLE
MOV @CSR, R1 ;SAVE CSR
CMP #BIT2, @CSR ;CHECK CSR
BEQ .+4 ;BRANCH IF OK
HLT+1 ;RECEIVER ENABLE DIDN'T CLEAR
; OR OTHER CSR BIT SET
R1 = CONTENTS OF CSR

(1) 003470 104400
4569

SCOPE

;TEST 12: TEST THAT CSR RESPONDS PROPERLY TO BYTE COMMANDS
;PROBABLE FAULTY LOGIC: M7285 (D2-4) E47

```

(1) 003472 012777 052506 175620 TST12: MOV #052506,@CSR ;SET TEST NUMBER IN CSR
(1) 003500 105077 175614 CLR @CSR ;CLR EVEN BYTE
(1) 003504 017701 175610 MOV @CSR,R1 ;SAVE CSR
(1) 003510 022701 052400 CMP #052400,R1 ;CHECK CSR
(1) 003514 001401 BEQ .+4 ;BRANCH IF OK
(1) 003516 104001 HLT+1 ;EVEN BYTE CLR FAILED ON CSR
(1) ;R1 = CONTENTS OF CSR
(1) 003520 012777 052506 175572 MOV #052506,@CSR ;SET TEST NUMBER IN CSR
(1) 003526 005237 001320 INC CSR ;INC TO ODD BYTE
(1) 003532 105077 175562 CLR @CSR ;CLR ODD BYTE
(1) 003536 005337 001320 DEC CSR ;RESTORE TO EVEN
(1) 003542 017701 175552 MOV @CSR,R1 ;SAVE CSR
(1) 003546 022701 000106 CMP #000106,R1 ;CHECK CSR
(1) 003552 001401 BEQ .+4 ;BRANCH IF OK
(1) 003554 104001 HLT+1 ;ODD BYTE CLR FAILED ON CSR
(1) ;R1 = CONTENTS OF CSR

```

(1) 003556 104400 SCOPE

4570

```

(1) ;*****
(1) ;TEST 13: TEST THAT THE BIS AND BIC INSTRUCTIONS SET AND CLEAR R/W
(1) ;BITS OF CSR
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-4) E47
(1) ;*****

```

```

(1) 003560 005077 175534 TST13: CLR @CSR ;CLEAR THE CSR
(1) 003564 017701 175530 MOV @CSR,R1 ;CHECK AND SAVE CSR
(1) 003570 001401 BEQ .+4 ;BRANCH IF CLEARED OK
(1) 003572 104001 HLT+1 ;RESET FAILED TO CLR CSR
(1)
(1) 003574 052777 052506 175516 BIS #052506,@CSR ;SET ALL R/W BITS OF CSR
(1) 003602 022777 052506 175510 CMP #052506,@CSR ;CHECK THAT THEY GOT SET
(1) 003610 001401 BEQ .+4 ;BRANCH IF OK
(1) 003612 104000 HLT ;REG FAILED CMP

```

```

(1) 003614 042777 052506 175476 BIC #052506,@CSR ;CLEAR CSR
(1) 003622 017701 175472 MOV @CSR,R1 ;CHECK AND SAVE CSR
(1) 003626 001401 BEQ .+4 ;BRANCH IF CLEARED OK
(1) 003630 104001 HLT+1 ;CLR FAILED TO CLR CSR

```

(1) 003632 104400 SCOPE

4571

```

(1) ;*****
(1) ;TEST 14: TEST BITS OF TCR FOR READ/WRITE CAPABILITY
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-2) ALL, (D2-3) E8,E20,E21,E43,E41
(1) ;*****

```

```

(1) 003634 012777 177777 175462 TST14: MOV #177777,@TCR ;SET ALL BITS OF TCR
(1) 003642 017701 175456 MOV @TCR,R1 ;CHECK AND SAVE TCR
(1) 003646 022701 177777 CMP #177777,R1 ;CHECK THAT ALL THE BITS ARE SET
(1) 003652 001401 BEQ .+4 ;BRANCH IF OK
(1) 003654 104001 HLT+1 ;BIT(S) OF TCR FAILED TO SET
(1)
(1) 003656 005077 175442 CLR @TCR ;CLEAR TCR
(1) 003662 017701 175436 MOV @TCR,R1 ;CHECK THAT IT CLEARED AND SAVE
(1) 003666 001401 BEQ .+4 ;BRANCH IF CLR

```

(1) 003670 104001 HLT+1 ;BIT(S) OF TCR FAILED TO CLEAR
(1)
(1) 003672 104400 SCOPE

4572
(1) ;*****
(1) ; TEST 15: TEST THAT TCR RESPONDS PROPERLY TO BYTE COMMANDS
(1) ; PROBABLE FAULTY LOGIC: M7285 (D2-3) E41
(1) ;*****

(1) 003674 012777 177777 175422 TST15: MOV #177777,@TCR ;SET TEST NUMBER IN TCR
(1) 003702 105077 175416 CLR @TCR ;CLR EVEN BYTE
(1) 003706 017701 175412 MOV @TCR,R1 ;SAVE TCR
(1) 003712 022701 177400 CMP #177400,R1 ;CHECK TCR
(1) 003716 001401 BEQ .+4 ;BRANCH IF OK
(1) 003720 104001 HLT+1 ;EVEN BYTE CLR FAILED ON TCR
(1) ; R1 = CONTENTS OF TCR
(1) 003722 012777 177777 175374 MOV #177777,@TCR ;SET TEST NUMBER IN TCR
(1) 003730 005237 001324 INC TCR ;INC TO ODD BYTE
(1) 003734 105077 175364 CLR @TCR ;CLR ODD BYTE
(1) 003740 005337 001324 DEC TCR ;RESTORE TO EVEN
(1) 003744 017701 175354 MOV @TCR,R1 ;SAVE TCR
(1) 003750 022701 000377 CMP #000377,R1 ;CHECK TCR
(1) 003754 001401 BEQ .+4 ;BRANCH IF OK
(1) 003756 104001 HLT+1 ;ODD BYTE CLR FAILED ON TCR
(1) ; R1 = CONTENTS OF TCR

(1) 003760 104400 SCOPE

4573
(1) ;*****
(1) ; TEST 16: TEST THAT THE BIS AND BIC INSTRUCTIONS SET AND CLEAR R/W
(1) ; BITS OF TCR
(1) ; PROBABLE FAULTY LOGIC: M7285 (D2-3) E41
(1) ;*****

(1) 003762 005077 175336 TST16: CLR @TCR ;CLEAR THE TCR
(1) 003766 017701 175332 MOV @TCR,R1 ;CHECK AND SAVE TCR
(1) 003772 001401 BEQ .+4 ;BRANCH IF CLEARED OK
(1) 003774 104001 HLT+1 ;RESET FAILED TO CLR TCR
(1) 003776 052777 177777 175320 BIS #177777,@TCR ;SET ALL R/W BITS OF TCR
(1) 004004 022777 177777 175312 CMP #177777,@TCR ;CHECK THAT THEY GOT SET
(1) 004012 001401 BEQ .+4 ;BRANCH IF OK
(1) 004014 104000 HLT ;REG FAILED CMP
(1) 004016 042777 177777 175300 BIC #177777,@TCR ;CLEAR TCR
(1) 004024 017701 175274 MOV @TCR,R1 ;CHECK AND SAVE TCR
(1) 004030 001401 BEQ .+4 ;BRANCH IF CLEARED OK
(1) 004032 104001 HLT+1 ;CLR FAILED TO CLR TCR

(1) 004034 104400 SCOPE
4574 004036 012737 000001 001346 MOV #1,TIMER ;INITIALIZE TIMER
4575 004044 012737 004052 016070 MOV #.+6,LAD ;RESET LOOP ADDRESS

4576
(1) ;*****
(1) ; TEST 17: TEST THAT TRANSMIT READY (BIT15 OF CSR) SETS AND CLEARS
(1) ; WHEN TCR0 IS SET AND CLEARED.

```

(1) ;
(1) ;ALSO CHECK THAT THE RIGHT LINE NO. (0) APPEARS IN TBUF.
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-5) ALL, (D2-6) E23, E32, E33, E49, (D2-2) E3, E1
(1) ;*****
(1) 004052 012777 000400 175240 TST17: MOV #400, @CSR ;SET XMTR SCAN ENABLE
(1) 004060 052777 000001 175236 BIS #BIT0, @TCR ;SET XMTR CONTROL BIT, LINE 0
(1) 004066 005000 CLR RO ;SET UP WAIT COUNTER
(1) 004070 017701 175224 1$: MOV @CSR, R1 ;CHECK AND SAVE XMTR READY
(1) 004074 100404 BMI 2$ ;BRANCH IF SET OK
(1) 004076 005200 INC RO ;WAIT A WHILE
(1) 004100 001373 BNE 1$
(1) 004102 104001 HLT+1 ;XMTR READY FAILED TO SET
(1)
(1) 004104 000416 BR 3$ ;SKIP LINE # CHECK ON ERROR
(1)
(1) 004106 050037 001346 2$: BIS RO, TIMER ;SET UP TIMER
(1) 004112 017701 175210 MOV @TBUF, R1 ;SAVE LINE NUMBER
(1) 004116 105001 CLRB R1
(1) 004120 000301 SWAB R1
(1) 004122 022701 000000 CMP #0, R1 ;CHECK THAT THE XMTR STOPPED ON LINE 0
(1) 004126 001401 BEQ .+4 ;BRANCH IF OK
(1) 004130 104001 HLT+1 ;WRONG LINE NUMBER APPEARED IN TBUF
(1)
(1) 004132 017702 175162 MOV @CSR, R2 ;CHECK AND SAVE XMTR READY
(1) 004136 100401 BMI .+4 ;BRANCH IF OK
(1) 004140 104002 HLT+2 ;READING THE TBUF CLEARED XMTR READY
(1)
(1) 004142 042777 000001 175154 3$: BIC #BIT0, @TCR ;CLR XMTR CONTROL BIT, LINE 0
(1) 004150 017701 175144 MOV @CSR, R1 ;CHECK AND SAVE XMTR READY
(1) 004154 100001 BPL .+4 ;BRANCH IF OK
(1) 004156 104001 HLT+1 ;XMTR READY FAILED TO CLEAR WHEN
; XMTR CONTROL FOR LINE 0 WAS CLEARED
(1)
(1) 004160 104400 SCOPE
(1)
4577
(1) ;*****
(1) ;TEST_20: TEST THAT TRANSMIT READY (BIT15 OF CSR) SETS AND CLEARS
(1) ; WHEN EACH TCR BIT IS SET AND CLEARED.
(1) ;ALSO CHECK THAT THE RIGHT LINE NO. APPEARS IN TBUF.
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-5) ALL, (D2-6) E23, E32, E33, E49, (D2-2) E3, E1
(1) ;*****
(1) 004162 012777 000400 175130 TST20: MOV #400, @CSR ;SET XMTR SCAN ENABLE
(1) 004170 005001 CLR R1
(1) 004172 012704 000001 MOV #1, R4 ;SET UP MARKER
(1) 004176 050477 175122 4$: BIS R4, @TCR ;SET XMTR CONTROL BIT, EACH LINE
(1) 004202 005000 CLR RO ;SET UP WAIT COUNTER
(1) 004204 017702 175110 1$: MOV @CSR, R2 ;CHECK AND SAVE XMTR READY
(1) 004210 100404 BMI 2$ ;BRANCH IF SET OK
(1) 004212 005300 DEC RO ;WAIT A WHILE
(1) 004214 001373 BNE 1$
(1) 004216 104002 HLT+2 ;XMTR READY FAILED TO SET
(1) ;R1=LINE #
(1) ;R2=CSR

```



```

(1) ; THEN MASTER TRAN SCAN ENABLE
(1) 004356 042777 000001 174740 55: BIC #BIT0, @TCR ; CLEAR IT OUT
(1) 004364 017701 174730 MOV @CSR, R1 ; CHECK AND SAVE TRAN READY
(1) 004370 100001 BPL .+4 ; BRANCH IF OK
(1) 004372 104001 HLT+1 ; TRAN READY DIDN'T CLEAR.
(1) 004374 104400 SCOPE
(1)
+579
(1) ; *****
(1) ; TEST 22: TEST THAT INTERRUPT DOES NOT OCCUR AT LEVEL 7
(1) ; PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
(1) ; *****
(1) 004376 012777 004452 174730 TST22: MOV #ISR22, @XMTVEC ; SET UP XMTR INTERRUPT VECTOR
(1) 004404 012777 000340 174724 MOV #340, @XMTLVL ; AT LEVEL 7
(1) 004412 042737 000340 177776 BIC #340, @PS ; CLEAR PS LEVEL
(1) 004420 052737 000340 177776 BIS #340, @PS ; SET PS TO LEVEL 7
(1) 004426 012777 040400 174664 MOV #040400, @CSR ; SET TRAN MASTER INT. ENABLE
(1) 004434 012777 000001 174662 MOV #BIT0, @TCR ; SET TRAN CONTROL BIT, LINE 0
(1) 004442 017701 174652 15: MOV @CSR, R1 ; WAIT
(1) 004446 100375 BPL 15
(1) 004450 000404 BR END22 ; OK, BRANCH IF NO INTERRUPT
(1) 004452 104000 ISR22: HLT ; SHOULDN'T HAVE INTERRUPTED AT LEVEL 7
(1) 004454 012716 004462 MOV #END22, (SP) ; MOVE NEW RTI ADR ONTO STACK
(1) 004460 000002 RTI
(1)
(1) 004462 013777 001336 174644 END22: MOV XMTLVL, @XMTVEC
(1) 004470 012777 000004 174640 MOV #IOT, @XMTLVL
(1) 004476 005077 174622 CLR @TCR
(1) 004502 005077 174612 CLR @CSR
(1) 004506 042737 000340 177776 BIC #340, @PS
(1)
(1) 004514 104400 SCOPE
(1)
+580
(1) ; *****
(1) ; TEST 23: TEST THAT INTERRUPT DOES NOT OCCUR AT LEVEL 6
(1) ; PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
(1) ; *****
(1) 004516 012777 004572 174610 TST23: MOV #ISR23, @XMTVEC ; SET UP XMTR INTERRUPT VECTOR
(1) 004524 012777 000340 174604 MOV #340, @XMTLVL ; AT LEVEL 7
(1) 004532 042737 000340 177776 BIC #340, @PS ; CLEAR PS LEVEL
(1) 004540 052737 000300 177776 BIS #300, @PS ; SET PS TO LEVEL 6
(1) 004546 012777 040400 174544 MOV #040400, @CSR ; SET TRAN MASTER INT. ENABLE
(1) 004554 012777 000001 174542 MOV #BIT0, @TCR ; SET TRAN CONTROL BIT, LINE 0
(1) 004562 017701 174532 15: MOV @CSR, R1 ; WAIT
(1) 004566 100375 BPL 15
(1) 004570 000404 BR END23 ; OK, BRANCH IF NO INTERRUPT
(1) 004572 104000 ISR23: HLT ; SHOULDN'T HAVE INTERRUPTED AT LEVEL 6
(1) 004574 012716 004602 MOV #END23, (SP) ; MOVE NEW RTI ADR ONTO STACK
(1) 004600 000002 RTI

```

```

(1) 004602 013777 001336 174524
(1) 004610 012777 000004 174520
(1) 004616 005077 174502
(1) 004622 005077 174472
(1) 004626 042737 000340 177776
(1)
(1) 004634 104400
(1)

```

```

END23: MOV XMTLVL, @XMTVEC
      MOV #IOT, @XMTLVL
      CLR @TCR
      CLR @CSR
      BIC #340, @#PS
      SCOPE

```

4581

```

(1)
(1)
(1)
(1)
(1) 004636 012777 004712 174470
(1) 004644 012777 000340 174464
(1) 004652 042737 000340 177776
(1) 004660 052737 000240 177776
(1) 004666 012777 040400 174424
(1) 004674 012777 000001 174422
(1) 004702 017701 174412
(1) 004706 100375
(1) 004710 000404
(1)

```

```

;*****
;TEST 24: TEST THAT INTERRUPT DOES NOT OCCUR AT LEVEL 5
;PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
;*****
TST24: MOV #ISR24, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
      MOV #340, @XMTLVL ;AT LEVEL 7
      BIC #340, @#PS ;CLEAR PS LEVEL
      BIS #240, @#PS ;SET PS TO LEVEL 5
      MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
      MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
    BPL IS
    BR END24 ;OK, BRANCH IF NO INTERRUPT

```

```

(1) 004712 104000
(1) 004714 012716 004722
(1) 004720 000002
(1)

```

```

ISR24: HLT ;SHOULDN'T HAVE INTERRUPTED AT LEVEL 5
      MOV #END24, (SP) ;MOVE NEW RTI ADR ONTO STACK
      RTI

```

```

(1) 004722 013777 001336 174404
(1) 004730 012777 000004 174400
(1) 004736 005077 174362
(1) 004742 005077 174352
(1) 004746 042737 000340 177776
(1)
(1) 004754 104400
(1)

```

```

END24: MOV XMTLVL, @XMTVEC
      MOV #IOT, @XMTLVL
      CLR @TCR
      CLR @CSR
      BIC #340, @#PS
      SCOPE

```

4582

```

(1)
(1)
(1)
(1)
(1) 004756 012777 005034 174350
(1) 004764 012777 000340 174344
(1) 004772 042737 000340 177776
(1) 005000 052737 000200 177776
(1) 005006 012777 040400 174304
(1) 005014 012777 000001 174302
(1) 005022 017701 174272
(1) 005026 100375
(1) 005030 104000
(1) 005032 000403
(1)

```

```

;*****
;TEST 25: TEST THAT INTERRUPT OCCURS AT LEVEL 4
;PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
;*****
TST25: MOV #ISR25, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
      MOV #340, @XMTLVL ;AT LEVEL 7
      BIC #340, @#PS ;CLEAR PS LEVEL
      BIS #200, @#PS ;SET PS TO LEVEL 4
      MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
      MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
    BPL IS
    HLT ;SHOULDN'T HAVE INTERRUPTED AT LEVEL 4
    BR END25 ;CONTINUE

```

```

(1) 005034 012716 005042
(1) 005040 000002
(1)

```

```

ISR25: MOV #END25, (SP) ;MOVE NEW RTI ADR ONTO STACK
      RTI

```

E03

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-10
TST25: TEST TRANSMIT INTERRUPT LEVEL

SEQ 0030

```
(1) 005042 013777 001336 174264
(1) 005050 012777 000004 174260
(1) 005056 005077 174242
(1) 005062 005077 174232
(1) 005066 042737 000340 177776
(1)
(1) 005074 104400
```

```
END25: MOV XMTLVL, @XMTVEC
MOV #IOT, @XMTLVL
CLR @TCR
CLR @CSR
BIC #340, @#PS
```

SCOPE

4583

```
*****
:TEST 26: TEST THAT INTERRUPT OCCURS AT LEVEL 3
:PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
*****
```

```
(1) 005076 012777 005154 174230
(1) 005104 012777 000340 174224
(1) 005112 042737 000340 177776
(1) 005120 052737 000140 177776
(1) 005126 012777 040400 174164
(1) 005134 012777 000001 174162
(1) 005142 017701 174152
(1) 005146 100375
(1) 005150 104000
(1) 005152 000403
```

```
TST26: MOV #ISR26, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
MOV #340, @XMTLVL ;AT LEVEL 7
BIC #340, @#PS ;CLEAR PS LEVEL
BIS #140, @#PS ;SET PS TO LEVEL 3
MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
BPL IS ;SHOULD HAVE INTERRUPTED AT LEVEL 3
HLT ;CONTINUE
BR END26
```

```
(1) 005154 012716 005162
(1) 005160 000002
```

```
ISR26: MOV #END26, (SP) ;MOVE NEW RTI ADR ONTO STACK
RTI
```

```
(1) 005162 013777 001336 174144
(1) 005170 012777 000004 174140
(1) 005176 005077 174122
(1) 005202 005077 174112
(1) 005206 042737 000340 177776
```

```
END26: MOV XMTLVL, @XMTVEC
MOV #IOT, @XMTLVL
CLR @TCR
CLR @CSR
BIC #340, @#PS
```

```
(1) 005214 104400
```

SCOPE

4584

```
*****
:TEST 27: TEST THAT INTERRUPT OCCURS AT LEVEL 2
:PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
*****
```

```
(1) 005216 012777 005274 174110
(1) 005224 012777 000340 174104
(1) 005232 042737 000340 177776
(1) 005240 052737 000100 177776
(1) 005246 012777 040400 174044
(1) 005254 012777 000001 174042
(1) 005262 017701 174032
(1) 005266 100375
(1) 005270 104000
(1) 005272 000403
```

```
TST27: MOV #ISR27, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
MOV #340, @XMTLVL ;AT LEVEL 7
BIC #340, @#PS ;CLEAR PS LEVEL
BIS #100, @#PS ;SET PS TO LEVEL 2
MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
BPL IS ;SHOULD HAVE INTERRUPTED AT LEVEL 2
HLT ;CONTINUE
BR END27
```

```
(1) 005274 012716 005302
(1) 005300 000002
```

```
ISR27: MOV #END27, (SP) ;MOVE NEW RTI ADR ONTO STACK
RTI
```

```

(1) 005302 013777 001336 174024
(1) 005310 012777 000004 174020
(1) 005316 005077 174002
(1) 005322 005077 173772
(1) 005326 042737 000340 177776
(1)
(1) 005334 104400

```

```

END27: MOV XMTLVL, @XMTVEC
MOV #IOT, @XMTLVL
CLR @TCR
CLR @CSR
BIC #340, @#PS
SCOPE

```

4585

```

(1)
(1)
(1)
(1)
(1)
(1)
(1) 005336 012777 005414 173770
(1) 005344 012777 000340 173764
(1) 005352 042737 000340 177776
(1) 005360 052737 000040 177776
(1) 005366 012777 040400 173724
(1) 005374 012777 000001 173722
(1) 005402 017701 173712
(1) 005406 100375
(1) 005410 104000
(1) 005412 000403

```

```

:*****
:TEST 30: TEST THAT INTERRUPT OCCURS AT LEVEL 1
:PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
:*****
TST30: MOV #ISR30, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
MOV #340, @XMTLVL ;AT LEVEL 7
BIC #340, @#PS ;CLEAR PS LEVEL
BIS #040, @#PS ;SET PS TO LEVEL 1
MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
BPL IS ;SHOULD HAVE INTERRUPTED AT LEVEL 1
HLT ;CONTINUE
BR END30

```

```

(1) 005414 012716 005422
(1) 005420 000002
(1)
(1) 005422 013777 001336 173704
(1) 005430 012777 000004 173700
(1) 005436 005077 173662
(1) 005442 005077 173652
(1) 005446 042737 000340 177776

```

```

ISR30: MOV #END30, (SP) ;MOVE NEW RTI ADR ONTO STACK
RTI
END30: MOV XMTLVL, @XMTVEC
MOV #IOT, @XMTLVL
CLR @TCR
CLR @CSR
BIC #340, @#PS

```

4586

```

(1) 005454 104400
(1)
(1)
(1)
(1)
(1) 005456 012777 005534 173650
(1) 005464 012777 000340 173644
(1) 005472 042737 000340 177776
(1) 005500 052737 000000 177776
(1) 005506 012777 040400 173604
(1) 005514 012777 000001 173602
(1) 005522 017701 173572
(1) 005526 100375
(1) 005530 104000
(1) 005532 000403

```

```

:*****
:TEST 31: TEST THAT INTERRUPT OCCURS AT LEVEL 0
:PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
:*****
TST31: MOV #ISR31, @XMTVEC ;SET UP XMTR INTERRUPT VECTOR
MOV #340, @XMTLVL ;AT LEVEL 7
BIC #340, @#PS ;CLEAR PS LEVEL
BIS #000, @#PS ;SET PS TO LEVEL 0
MOV #040400, @CSR ;SET TRAN MASTER INT. ENABLE
MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
IS: MOV @CSR, R1 ;WAIT
BPL IS ;SHOULD HAVE INTERRUPTED AT LEVEL C
HLT ;CONTINUE
BR END31

```

```

(1) 005534 012716 005542
(1) 005540 000002

```

```

ISR31: MOV #END31, (SP) ;MOVE NEW RTI ADR ONTO STACK
RTI

```



```

(1) 005542 013777 001336 173564
(1) 005550 012777 000004 173560
(1) 005556 005077 173542
(1) 005562 005077 173532
(1) 005566 042737 000340 177776
(1)
(1) 005574 104400
(1)
4587 005576 005037 001346
4588 005602 012737 005610 016070
4589

```

```

END31: MOV XMTLVL, @XMTVEC
MOV #IOT, @XMTLVL
CLR @TCR
CLR @CSR
BIC #340, @#PS

SCOPE

CLR TIMER ; INITIALIZE TIMER
MOV #.+6, LAD ; RESET LOOP ADDRESS

```

```

*****
TEST 32: TEST THAT LINE 0 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
3$: CHECKS THAT NO ERRORS IN FI/FO
4$: CHECKS THAT RIGHT LINE # (0) IN FI/FO
5$: CHECKS FOR RIGHT CHARACTER LENGTH
6$: CHECKS THAT CORRECT DATA WAS RECEIVED
7$: CHECKS THAT CHARACTER PRESENT CLEARS
8$: CHECKS THAT DONE CLEARS
PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
*****

```

```

INITIALIZE
DEVICE"CSR"REGISTER

TST32: JSR PC, @#INITD SET:
;BIT2 = MAINTENANCE
;BIT3 = CLEAR MOS
;BIT8 = MASTER XMTR SCAN ENB

```

```

;WAIT FOR MOS TO CLEAR

SET:
;BIT0 = RECEIVER ENABLE

```

```

(1) 005614 052777 000001 173502
(1) 005622 017701 173472
(1) 005626 100375
(1) 005630 012777 000377 173470
(1) 005636 005000
(1) 005640 105305
(1) 005642 001376
(1) 005644 017701 173452
(1) 005650 100405
(1) 005652 005200
(1) 005654 001371
(1) 005656 017702 173436
(1) 005662 104002
(1)
(1)
(1) 005664 050037 001346
(1) 005670 032701 070000
(1) 005674 001401
(1) 005676 104001

```

```

LOP32: BIS #BIT0, @TCR ;SET XMTR CONTROL BIT, LINE 0
MOV @CSR, R1 ;WAIT FOR XMTR READY
BPL LOP32
MOV #377, @TBUF ;SEND A RUBOUT
CLR R0 ;CLEAR COUNTER
1$: DECB R5 ;SHORT WAIT LOOP
BNE 1$
MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
BMI 3$ ;BRANCH WHEN FOUND
INC R0 ;TIME COUNTER
BNE 1$ ;BRANCH IF NOT TIME-OUT
MOV @CSR, R2 ;SAVE CSR
HLT+2 ;CHARACTER READY DIDN'T SET
;R1 = CONTENTS OF RBUF
;R2 = CONTENTS OF CSR
;SAVE THE TIMER
3$: BIS R0, TIMER
BIT #70000, R1 ;CHECK FOR ERROR BITS
BEQ .+4 ;BRANCH IF NONE
HLT+1 ;ERROR IN RECEIVED CHAR
;R1 = CONTENTS OF RBUF
;BIT14=UART OVERRUN
;BIT13=FRAMING ERROR

```

H03

MAINDEC-11-DZDJA-E
DZDJA.E.P11

DJ11 LOGIC TESTS
08-SEP-77 15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-13
TST32: TEST ALL OF LINE 0 TRANSMIT AND RECEIVE LOGIC

SEG 0033

(1)				45:	MOV	R1	R2	:	BIT12=PARITY ERROR
(1)	005700	010102			BIC	#170377,	R2	:	DUPLICATE DATA WORD
(1)	005702	042702	170377		SWAB	R2		:	MASK LINE #
(1)	005706	000302			CMPB	#0.,	R2	:	LINE # IN LOW BYTE
(1)	005710	122702	000000		BEQ	+.4		:	CHECK LINE #
(1)	005714	001401			HLT+1			:	BRANCH IF OK
(1)	005716	104001						:	WRONG LINE # RECEIVED
(1)								:	R1 = CONTENTS OF RBUF
(1)								:	BIT58-11 = LINE #
(1)	005720	117702	173440	55:	MOV	2DJLEN,	R2	:	GET MASK OF CHARACTER
(1)	005724	130201			BITB	R2,	R1	:	CHECK CHAR LENGTH.
(1)	005726	001401			BEQ	+.4		:	BRANCH IF OK
(1)	005730	104002			HLT+2			:	WRONG CHARACTER LENGTH
(1)								:	R1=DATA FROM FI/FO
(1)								:	R2=MASK (BITS SET NOT EXPECTED)
(1)	005732	105102		65:	COMB	R2		:	REVERSE THE MASK
(1)	005734	120102			CMPB	R1,	R2	:	CHECK THE ACTUAL DATA
(1)	005736	001401			BEQ	+.4		:	BRANCH IF OK
(1)	005740	104002			HLT+2			:	WRONG CHAR LEN OR DATA ERROR
(1)								:	R1=DATA FROM FI/FO (COMPLETE WORD)
(1)								:	R2=DATA (LOW BYTE) EXPECTED
(1)	005742	017701	173354	75:	MOV	2RBUF,	R1	:	READ FI/FO
(1)	005746	100001			BPL	+.4		:	BRANCH IF CHAR PRESENT NOT SET
(1)	005750	104001			HLT+1			:	CHARACTER PRESENT STAYED SET
(1)								:	R1 = CONTENTS OF RBUF
(1)	005752	017701	173342	85:	MOV	2CSR,	R1	:	SAVE THE CSR
(1)	005756	022701	100405		CMP	#100405,	R1	:	CHECK THE CSR
(1)	005762	001401			BEQ	+.4		:	BRANCH IF OK
(1)	005764	104001			HLT+1			:	DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1)								:	R1 = CONTENTS OF CSR
(1)	005766	005077	173332		CLR	2TCR		:	CLEAR TCR
(1)	005772	005077	173322		CLR	2CSR		:	CLEAR CSR
(1)	005776	104400			SCOPE			:	

4590

```
*****
TEST 33: TEST THAT LINE 1 CAN TRANSMIT AND
          RECEIVE A CHARACTER. (377)
          15: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
          35: CHECKS THAT NO ERRORS IN FI/FO
          45: CHECKS THAT RIGHT LINE # (1) IN FI/FO
          55: CHECKS FOR RIGHT CHARACTER LENGTH
          65: CHECKS THAT CORRECT DATA WAS RECEIVED
          75: CHECKS THAT CHARACTER PRESENT CLEARS
          85: CHECKS THAT DONE CLEARS
PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD DC3 SERIES
*****
```

```
INITIALIZE
DEVICE"CSR"REGISTER
```

006000 004737 015420

TST33: JSR PC, 2#INIT0

```
SET:
:BIT2 = MAINTENANCE
:BIT3 = CLEAR MOS
:BIT8 = MASTER XMTR SCAN ENB
```

:WAIT FOR MOS TO CLEAR

```
SET:
:BIT0 = RECEIVER ENABLE
```

```

(1) ;
(1) 006004 052777 000002 173312 ; BIS #BIT1, @TCR ;SET XMTR CONTROL BIT, LINE 1
(1) 006012 017701 173302 LOP33: MOV @CSR, R1 ;WAIT FOR XMTR READY
(1) 006016 100375 BPL LOP33
(1) 006020 012777 000377 173300 MOV #377, @RBUF ;SEND A RUBOUT
(1) 006026 005000 CLR R0 ;CLEAR COUNTER
(1) 006030 105305 15: DECB R5 ;SHORT WAIT LOOP
(1) 006032 001376 BNE 15
(1) 006034 017701 173262 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 006040 100405 BMI 35 ;BRANCH WHEN FOUND
(1) 006042 005200 INC R0 ;TIME COUNTER
(1) 006044 001371 BNE 15 ;BRANCH IF NOT TIME-OUT
(1) 006046 017702 173246 MOV @CSR, R2 ;SAVE CSR
(1) 006052 104002 HLT+2 ;CHARACTER READY DIDN'T SET
(1) ; R1 = CONTENTS OF RBUF
(1) ; R2 = CONTENTS OF CSR
(1) 006054 050037 001346 35: BIS R0, TIMER ;SAVE THE TIMER
(1) 006060 032701 070000 BIT #70000, R1 ;CHECK FOR ERROR BITS
(1) 006064 001401 BEQ .+ ;BRANCH IF NONE
(1) 006066 104001 HLT+1 ;ERROR IN RECEIVED CHAR
(1) ; R1 = CONTENTS OF RBUF
(1) ; BIT14=UART OVERRUN
(1) ; BIT13=FRAMING ERROR
(1) ; BIT12=PARITY ERROR
(1) 006070 010102 45: MOV R1, R2 ;DUPLICATE DATA WORD
(1) 006072 042702 170377 BIC #170377, R2 ;MASK LINE #
(1) 006076 000302 SWAB R2 ;LINE # IN LOW BYTE
(1) 006100 122702 000001 CMPB #1., R2 ;CHECK LINE #
(1) 006104 001401 BEQ .+4 ;BRANCH IF OK
(1) 006106 104001 HLT+1 ;WRONG LINE # RECEIVED
(1) ; R1 = CONTENTS OF RBUF
(1) ; BITS8-11 = LINE #
(1) 006110 117702 173250 55: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
(1) 006114 130201 BITB R2, R1 ;CHECK CHAR LENGTH.
(1) 006116 001401 BEQ .+4 ;BRANCH IF OK
(1) 006120 104002 HLT+2 ;WRONG CHARACTER LENGTH
(1) ; R1=DATA FROM FI/FO
(1) ; R2=MASK (BITS SET NOT EXPECTED)
(1) 006122 105102 65: COMB R2 ;REVERSE THE MASK
(1) 006124 120102 CMPB R1, R2 ;CHECK THE ACTURAL DATA
(1) 006126 001401 BEQ .+4 ;BRANCH IF OK
(1) 006130 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
(1) ; R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ; R2=DATA (LOW BYTE) EXPECTED
(1) 006132 017701 173164 75: MOV @RBUF, R1 ;READ FI/FO
(1) 006136 100001 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
(1) 006140 104001 HLT+1 ;CHARACTER PRESENT STAYED SET
(1) ; R1 = CONTENTS OF RBUF
(1) 006142 017701 173152 85: MOV @CSR, R1 ;SAVE THE CSR
(1) 006146 022701 100405 CMP #100405, R1 ;CHECK THE CSR
(1) 006152 001401 BEQ .+4 ;BRANCH IF OK
(1) 006154 104001 HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ; R1 = CONTENTS OF CSR
(1) 006156 005077 173142 CLR @TCR ;CLEAR TCR
(1) 006162 005077 173132 CLR @CSR ;CLEAR CSR
(1) 006166 104400 SCOPE

```


K03

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-16
TST34: TEST ALL OF LINE 2 TRANSMIT AND RECEIVE LOGIC

SEG 0036

```

(1) 006304 130201 BITB R2, R1 ;CHECK CHAR LENGTH.
(1) 006306 001401 BEQ .+4 ;BRANCH IF OK
(1) 006310 104002 HLT+2 ;WRONG CHARACTER LENGTH
(1) ;R1=DATA FROM FI/FO
(1) ;R2=MASK (BITS SET NOT EXPECTED)
(1) 006312 105102 65: COMB R2 ;REVERSE THE MASK
(1) 006314 120102 CMPB R1, R2 ;CHECK THE ACTUAL DATA
(1) 006316 001401 BEQ .+4 ;BRANCH IF OK
(1) 006320 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
(1) ;R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ;R2=DATA (LOW BYTE) EXPECTED
(1) 006322 017701 172774 75: MOV @RBUF, R1 ;READ FI/FO
(1) 006326 100001 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
(1) 006330 104001 HLT+1 ;CHARACTER PRESENT STAYED SET
(1) ;R1 = CONTENTS OF RBUF
(1) 006332 017701 172762 85: MOV @CSR, R1 ;SAVE THE CSR
(1) 006336 022701 100405 CMP #100405, R1 ;CHECK THE CSR
(1) 006342 001401 BEQ .+4 ;BRANCH IF OK
(1) 006344 104001 HLT+1 ;DONE DON'T CLEAR OR OTHER CSR ERROR
(1) ;R1 = CONTENTS OF CSR
(1) 006346 005077 172752 CLR @TCR ;CLEAR TCR
(1) 006352 005077 172742 CLR @CSR ;CLEAR CSR
(1) 006356 104400 SCOPE

```

4592

```

*****
:TEST 35: TEST THAT LINE 3 CAN TRANSMIT AND
: RECEIVE A CHARACTER. (377)
: 15: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 35: CHECKS THAT NO ERRORS IN FI/FO
: 45: CHECKS THAT RIGHT LINE # (3) IN FI/FO
: 55: CHECKS FOR RIGHT CHARACTER LENGTH
: 65: CHECKS THAT CORRECT DATA WAS RECEIVED
: 75: CHECKS THAT CHARACTER PRESENT CLEARS
: 85: CHECKS THAT DONE CLEARS
:PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; JART CARD D03 SERIES
*****

```

INITIALIZE
DEVICE"CSR"REGISTER

```

(1) 006360 004737 015420 TST35: JSR PC, @INITD SET:
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER XMTR SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR
(1) ;SET:
(1) ;BIT0 = RECEIVER ENABLE
(1) 006364 052777 000010 172732 LOP35: BIS #BIT3, @TCR ;SET XMTR CONTROL BIT, LINE 3
(1) 006372 017701 172722 MOV @CSR, R1 ;WAIT FOR XMTR READY
(1) 006376 100375 BPL LOP35
(1) 006400 012777 000377 172720 MOV #377, @TBUF ;SEND A RUBOUT
(1) 006406 005000 CLR R0 ;CLEAR COUNTER
(1) 006410 105305 15: DECB R5 ;SHORT WAIT LOOP
(1) 006412 001376 BNE 15
(1) 006414 017701 172702 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 006420 100405 BMI 35 ;BRANCH WHEN FOUND

```

L03

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-17
TEST ALL OF LINE 3 TRANSMIT AND RECEIVE LOGIC

SEG 0037

(1)	006422	005200			INC	R0			: TIME COUNTER
(1)	006424	001371			BNE	1\$: BRANCH IF NOT TIME-OUT
(1)	006426	017702	172666		MOV	@CSR,	R2		: SAVE CSR
(1)	006432	104002			HLT+2				: CHARACTER READY DIDN'T SET
(1)									: R1 = CONTENTS OF RBUF
(1)									: R2 = CONTENTS OF CSR
(1)	006434	050037	001346	3\$:	BIS	R0,	TIMER		: SAVE THE TIMER
(1)	006440	032701	070000		BIT	#70000,	R1		: CHECK FOR ERROR BITS
(1)	006444	001401			BEQ	.+4			: BRANCH IF NONE
(1)	006446	104001			HLT+1				: ERROR IN RECEIVED CHAR
(1)									: R1 = CONTENTS OF RBUF
(1)									: BIT14=UART OVERRUN
(1)									: BIT13=FRAMING ERROR
(1)									: BIT12=PARITY ERROR
(1)	006450	010102		4\$:	MOV	R1,	R2		: DUPLICATE DATA WORD
(1)	006452	042702	170377		BIC	#170377,	R2		: MASK LINE #
(1)	006456	000302			SWAB	R2			: LINE # IN LOW BYTE
(1)	006460	122702	000003		CMPB	#3.,	R2		: CHECK LINE #
(1)	006464	001401			BEQ	.+4			: BRANCH IF OK
(1)	006466	104001			HLT+1				: WRONG LINE # RECEIVED
(1)									: R1 = CONTENTS OF RBUF
(1)									: BITS8-11 = LINE #
(1)	006470	117702	172670	5\$:	MOVB	@JLEN,	R2		: GET MASK OF CHARACTER
(1)	006474	130201			BITB	R2,	R1		: CHECK CHAR LENGTH.
(1)	006476	001401			BEQ	.+4			: BRANCH IF OK
(1)	006500	104002			HL +2				: WRONG CHARACTER LENGTH
(1)									: R1=DATA FROM FI/FO
(1)									: R2=MASK (BITS SET NOT EXPECTED)
(1)	006502	105102		6\$:	COMB	R2			: REVERSE THE MASK
(1)	006504	120102			CMPB	R1,	R2		: CHECK THE ACTUAL DATA
(1)	006506	001401			BEQ	.+4			: BRANCH IF OK
(1)	006510	104002			HLT+2				: WRONG CHAR LEN OR DATA ERROR
(1)									: R1=DATA FROM FI/FO (COMPLETE WORD)
(1)									: R2=DATA (LOW BYTE) EXPECTED
(1)	006512	017701	172604	7\$:	MOV	@RBUF,	R1		: READ FI/FO
(1)	006516	100001			BPL	.+4			: BRANCH IF CHAR PRESENT NOT SET
(1)	006520	104001			HLT+1				: CHARACTER PRESENT STAYED SET
(1)									: R1 = CONTENTS OF RBUF
(1)	006522	017701	172572	8\$:	MOV	@CSR,	R1		: SAVE THE CSR
(1)	006526	022701	100405		CMP	#100405,	R1		: CHECK THE CSR
(1)	006532	001401			BEQ	.+4			: BRANCH IF OK
(1)	006534	104001			HLT+1				: DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1)									: R1 = CONTENTS OF CSR
(1)	006536	005077	172562		CLR	@TCR			: CLEAR TCR
(1)	006542	005077	172552		CLR	@CSR			: CLEAR CSR
(1)	006546	104400			SCOPE				
4593	006550	005237	001364		INC	@JLEN			
4594	006554	012737	006562	016070	MOV	#.+6,	LAD		

(1) :*****
 (1) :TEST 36: TEST THAT LINE 4 CAN TRANSMIT AND
 (1) : RECEIVE A CHARACTER. (377)
 (1) : 1\$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
 (1) : 3\$: CHECKS THAT NO ERRORS IN FI/FO
 (1) : 4\$: CHECKS THAT RIGHT LINE # (4) IN FI/FO
 (1) : 5\$: CHECKS FOR RIGHT CHARACTER LENGTH

M03

MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS
DZDJA.E.P11 08-SEP-77 15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-18
TST36: TEST ALL OF LINE 4 TRANSMIT AND RECEIVE LOGIC

SEG 0038

```

(1)      :      65: CHECKS THAT CORRECT DATA WAS RECEIVED
(1)      :      75: CHECKS THAT CHARACTER PRESENT CLEARS
(1)      :      85: CHECKS THAT DONE CLEARS
(1)      : PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD DC3 SERIES
(1)      : *****
(1)      :
(1)      :      INITIALIZE
(1)      :      DEVICE"CSR"REGISTER
(1)      :
(1) 006562 004737 015420 TST36: JSR    PC,    @#INITD  SET:
(1)      :                               ;BIT2 = MAINTENANCE
(1)      :                               ;BIT3 = CLEAR MOS
(1)      :                               ;BIT8 = MASTER XMTR SCAN ENB
(1)      :
(1)      : ;WAIT FOR MOS TO CLEAR
(1)      :
(1)      : SET:
(1)      : ;BIT0 = RECEIVER ENABLE
(1)      :
(1) 006566 052777 000020 172530 LOP36: BIS    #BIT4, @TCR  ;SET XMTR CONTROL BIT, LINE 4
(1) 006574 017701 172520          MOV    @CSR,  R1      ;WAIT FOR XMTR READY
(1) 006600 100375          BPL    LOP36
(1) 006602 012777 000377 172516          MOV    #377, @TBUF  ;SEND A RUBOUT
(1) 006610 005000          CLR    RD          ;CLEAR COUNTER
(1) 006612 105305          15:   DECB   R5      ;SHORT WAIT LOOP
(1) 006614 001376          BNE    15
(1) 006616 017701 172500          MOV    @RBUF,R1    ;WAIT FOR CHAR. PRES.
(1) 006622 100405          BMI    35          ;BRANCH WHEN FOUND
(1) 006624 005200          INC    RD          ;TIME COUNTER
(1) 006626 001371          BNE    15          ;BRANCH IF NOT TIME-OUT
(1) 006630 017702 172464          MOV    @CSR,  R2    ;SAVE CSR
(1) 006634 104002          HLT+2 ;CHARACTER READY DIDN'T SET
(1)      : ;R1 = CONTENTS OF RBUF
(1)      : ;R2 = CONTENTS OF CSR
(1)      : ;SAVE THE TIMER
(1) 006636 050037 001346          35:   BIS    RD,TIMER
(1) 006642 032701 070000          BIT    #70000, R1  ;CHECK FOR ERROR BITS
(1) 006646 001401          BEQ    .+4         ;BRANCH IF NONE
(1) 006650 104001          HLT+1 ;ERROR IN RECEIVED CHAR
(1)      : ;R1 = CONTENTS OF RBUF
(1)      : ;BIT14=UART OVERRUN
(1)      : ;BIT13=FRAMING ERROR
(1)      : ;BIT12=PARITY ERROR
(1) 006652 010102          45:   MOV    R1,    R2    ;DUPLICATE DATA WORD
(1) 006654 042702 170377          BIC    #170377,R2  ;MASK LINE#
(1) 006660 000302          SWAB  R2          ;LINE # IN LOW BYTE
(1) 006662 122702 000004          CMPB  #4,    R2    ;CHECK LINE #
(1) 006666 001401          BEQ    .+4         ;BRANCH IF OK
(1) 006670 104001          HLT+1 ;WRONG LINE # RECEIVED
(1)      : ;R1 = CONTENTS OF RBUF
(1)      : ;BITS8-11 = LINE #
(1) 006672 117702 172466          55:   MOVB  @DJLEN, R2   ;GET MASK OF CHARACTER
(1) 006676 130201          BITB  R2,    R1    ;CHECK CHAR LENGTH.
(1) 006700 001401          BEQ    .+4         ;BRANCH IF OK
(1) 006702 104002          HLT+2 ;WRONG CHARACTER LENGTH
(1)      : ;R1=DATA FROM FI/FO
(1)      : ;R2=MASK (BITS SET NOT EXPECTED)
(1) 006704 105102          65:   COMB  R2
(1) 006706 120102          CMPB  R1,    R2    ;REVERSE THE MASK
(1) 006710 001401          BEQ    .+4         ;CHECK THE ACTURAL DATA
(1)      : ;BRANCH IF OK

```

```

(1) 006712 104002          HLT+2          ;WRONG CHAR LEN OR DATA ERROR
(1)                               ;R1=DATA FROM FI/FO (COMPLETE WORD)
(1)                               ;R2=DATA (LOW BYTE) EXPECTED
(1) 006714 017701 172402  7$:  MOV   @RBUF, R1      ;READ FI/FO
(1) 006720 100001          BPL   .+4      ;BRANCH IF CHAR PRESENT NOT SET
(1) 006722 104001          HLT+1      ;CHARACTER PRESENT STAYED SET
(1)                               ;R1 = CONTENTS OF RBUF
(1) 006724 017701 172370  8$:  MOV   @CSR, R1      ;SAVE THE CSR
(1) 006730 022701 100405  CMP   #100405,R1 ;CHECK THE CSR
(1) 006734 001401          BEQ   .+4      ;BRANCH IF OK
(1) 006736 104001          HLT+1      ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1)                               ;R1 = CONTENTS OF CSR
(1) 006740 005077 172360  CLR   @TCR      ;CLEAR TCR
(1) 006744 005077 172350  CLR   @CSR      ;CLEAR CSR
(1) 006750 104400          SCOPE

```

4596

```

(1) *****
(1) TEST 37: TEST THAT LINE 5 CAN TRANSMIT AND
(1) RECEIVE A CHARACTER. (377)
(1) 1$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
(1) 3$: CHECKS THAT NO ERRORS IN FI/FO
(1) 4$: CHECKS THAT RIGHT LINE # (5) IN FI/FO
(1) 5$: CHECKS FOR RIGHT CHARACTER LENGTH
(1) 6$: CHECKS THAT CORRECT DATA WAS RECEIVED
(1) 7$: CHECKS THAT CHARACTER PRESENT CLEARS
(1) 8$: CHECKS THAT DONE CLEARS
(1) PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
(1) *****

```

```

(1) INITIALIZE
(1) DEVICE"CSR"REGISTER
(1) TST37: JSR   PC, @#INITD SET:
(1)                               ;BIT2 = MAINTENANCE
(1)                               ;BIT3 = CLEAR MOS
(1)                               ;BITB = MASTER XMTR SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR
(1)                               SET:
(1)                               ;BITD = RECEIVER ENABLE
(1) 006756 052777 000040 172340 LOP37: BIS   @BITS, @TCR ;SET XMTR CONTROL BIT, LINE 5
(1) 006764 017701 172330          MOV   @CSR, R1      ;WAIT FOR XMTR READY
(1) 006770 100375          BPL   LOP37
(1) 006772 012777 000377 172326  MOV   @377, @TBUF ;SEND A RUBOUT
(1) 007000 005000          CLR   R0          ;CLEAR COUNTER
(1) 007002 105305          1$:  DECB  R5          ;SHORT WAIT LOOP
(1) 007004 001376          BNE   1$
(1) 007006 017701 172310          MOV   @RBUF,R1    ;WAIT FOR CHAR. PRES.
(1) 007012 100405          BMI   3$          ;BRANCH WHEN FOUND
(1) 007014 005200          INC   R0          ;TIME COUNTER
(1) 007016 001371          BNE   1$          ;BRANCH IF NOT TIME-OUT
(1) 007020 017702 172274          MOV   @CSR, R2    ;SAVE CSR
(1) 007024 104002          HLT+2          ;CHARACTER READY DIDN'T SET
(1)                               ;R1 = CONTENTS OF RBUF
(1)                               ;R2 = CONTENTS OF CSR
(1) 007026 050037 001346  3$:  BIS   R0,TIMER    ;SAVE THE TIMER
(1) 007032 032701 070000          BIT   #70000, R1 ;CHECK FOR ERROR BITS

```


007036 001401
007040 104001

BEG .+4
HLT+1

: BRANCH IF NONE
: ERROR IN RECEIVED CHAR
: R1 = CONTENTS OF RBUF
: BIT14=UART OVERRUN
: BIT13=FRAMING ERROR
: BIT12=PARITY ERROR
: DUPLICATE DATA WORD

007042 010102
007044 042702 170377
007050 000302
007052 122702 000005
007056 001401
007060 104001

4S: MOV R1 R2
BIC #170377, R2
SWAB R2
CMPB #5., R2
BEQ .+4
HLT+1

: MASK LINE #
: LINE # IN LOW BYTE
: CHECK LINE #
: BRANCH IF OK
: WRONG LINE # RECEIVED
: R1 = CONTENTS OF RBUF
: BITS8-11 = LINE #

007062 117702 172276
007066 130201
007070 001401
007072 104002

5S: MOVB #0JLEN, R2
BITB R2, R1
BEQ .+4
HLT+2

: GET MASK OF CHARACTER
: CHECK CHAR LENGTH.
: BRANCH IF OK
: WRONG CHARACTER LENGTH
: R1=DATA FROM FI/FO
: R2=MASK (BITS SET NOT EXPECTED)

007074 105102
007076 120102
007100 001401
007102 104002

6S: COMB R2
CMPB R1, R2
BEQ .+4
HLT+2

: REVERSE THE MASK
: CHECK THE ACTUAL DATA
: BRANCH IF OK
: WRONG CHAR LEN OR DATA ERROR
: R1=DATA FROM FI/FO (COMPLETE WORD)
: R2=DATA (LOW BYTE) EXPECTED

007104 017701 172212
007110 100001
007112 104001

7S: MOV #RBUF, R1
BPL .+4
HLT+1

: READ FI/FO
: BRANCH IF CHAR PRESENT NOT SET
: CHARACTER PRESENT STAYED SET
: R1 = CONTENTS OF RBUF

007114 017701 172200
007120 022701 100405
007124 001401
007126 104001

8S: MOV #CSR, R1
CMP #100405, R1
BEQ .+4
HLT+1

: SAVE THE CSR
: CHECK THE CSR
: BRANCH IF OK
: DONE DIDN'T CLEAR OR OTHER CSR ERROR
: R1 = CONTENTS OF CSR

007130 005077 172170
007134 005077 172160
007140 104400

CLR #TCR
CLR #CSR
SCOPE

: CLEAR TCR
: CLEAR CSR

+597

TEST 40: TEST THAT LINE 6 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
1S: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
3S: CHECKS THAT NO ERRORS IN FI/FO
4S: CHECKS THAT RIGHT LINE # (6) IN FI/FO
5S: CHECKS FOR RIGHT CHARACTER LENGTH
6S: CHECKS THAT CORRECT DATA WAS RECEIVED
7S: CHECKS THAT CHARACTER PRESENT CLEARS
8S: CHECKS THAT DONE CLEARS

PROBABLE FAULTY LOGIC: M7285 (02-7) ALL; M7279 ALL; UART CARD 003 SERIES

INITIALIZE
DEVICE"CSR"REGISTER

007142 004737 015420

TEST 40: JSR PC, #INITD SET:
:BIT2 = MAINTENANCE

```

(1)                                     ;BIT3 = CLEAR MOS
(1)                                     ;BIT8 = MASTER XMTR SCAN ENB
(1)                                     ;
(1)                                     SET:
(1)                                     ;BIT0 = RECEIVER ENABLE
(1)                                     ;
(1) 007146 052777 000100 172150 LOP40: BIS #BIT6, @TCR ;SET XMTR CONTROL BIT, LINE 6
(1) 007154 017701 172140 LOP40: MOV @CSR, R1 ;WAIT FOR XMTR READY
(1) 007160 100375 LOP40: BPL LOP40
(1) 007162 012777 000377 172136 LOP40: MOV #377, @RBUF ;SEND A RUBOUT
(1) 007170 005000 LOP40: CLR R0 ;CLEAR COUNTER
(1) 007172 105305 1S: DECB R5 ;SHORT WAIT LOOP
(1) 007174 001376 1S: BNE 1S
(1) 007176 017701 172120 LOP40: MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 007202 100405 3S: BMI 3S ;BRANCH WHEN FOUND
(1) 007204 005200 LOP40: INC R0 ;TIME COUNTER
(1) 007206 001371 1S: BNE 1S ;BRANCH IF NOT TIME-OUT
(1) 007210 017702 172104 LOP40: MOV @CSR, R2 ;SAVE CSR
(1) 007214 104002 LOP40: HLT+2 ;CHARACTER READY DIDN'T SET
(1)                                     ;R1 = CONTENTS OF RBUF
(1)                                     ;R2 = CONTENTS OF CSR
(1) 007216 050037 001346 3S: BIS R0, TIMER ;SAVE THE TIMER
(1) 007222 032701 070000 3S: BIT #70000, R1 ;CHECK FOR ERROR BITS
(1) 007226 001401 3S: BEQ .+4 ;BRANCH IF NONE
(1) 007230 104001 3S: HLT+1 ;ERROR IN RECEIVED CHAR
(1)                                     ;R1 = CONTENTS OF RBUF
(1)                                     ;BIT14=UART OVERRUN
(1)                                     ;BIT13=FRAMING ERROR
(1)                                     ;BIT12=PARITY ERROR
(1) 007232 010102 4S: MOV R1, R2 ;DUPLICATE DATA WORD
(1) 007234 042702 170377 4S: BIC #170377, R2 ;MASK LINE#
(1) 007240 000302 4S: SWAB R2 ;LINE # IN LOW BYTE
(1) 007242 122702 000006 4S: CMPB #6, R2 ;CHECK LINE #
(1) 007246 001401 4S: BEQ .+4 ;BRANCH IF OK
(1) 007250 104001 4S: HLT+1 ;WRONG LINE # RECEIVED
(1)                                     ;R1 = CONTENTS OF RBUF
(1)                                     ;BITS8-11 = LINE #
(1) 007252 117702 172106 5S: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
(1) 007256 130201 5S: BITB R2, R1 ;CHECK CHAR LENGTH.
(1) 007260 001401 5S: BEQ .+4 ;BRANCH IF OK
(1) 007262 104002 5S: HLT+2 ;WRONG CHARACTER LENGTH
(1)                                     ;R1=DATA FROM FI/FO
(1)                                     ;R2=MASK (BITS SET NOT EXPECTED)
(1) 007264 105102 6S: COMB R2 ;REVERSE THE MASK
(1) 007266 120102 6S: CMPB R1, R2 ;CHECK THE ACTUAL DATA
(1) 007270 001401 6S: BEQ .+4 ;BRANCH IF OK
(1) 007272 104002 6S: HLT+2 ;WRONG CHAR LEN OR DATA ERROR
(1)                                     ;R1=DATA FROM FI/FO (COMPLETE WORD)
(1)                                     ;R2=DATA (LOW BYTE) EXPECTED
(1) 007274 017701 172022 7S: MOV @RBUF, R1 ;READ FI/FO
(1) 007300 100001 7S: BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
(1) 007302 104001 7S: HLT+1 ;CHARACTER PRESENT STAYED SET
(1)                                     ;R1 = CONTENTS OF RBUF
(1) 007304 017701 172010 8S: MOV @CSR, R1 ;SAVE THE CSR
(1) 007310 022701 100405 8S: CMP #100405, R1 ;CHECK THE CSR
(1) 007314 001401 8S: BEQ .+4 ;BRANCH IF OK

```

```

(1) 007316 104001 HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ;R1 = CONTENTS OF CSR
(1) 007320 005077 172000 CLR JTCR ;CLEAR TCR
(1) 007324 005077 171770 CLR JCSR ;CLEAR CSR
(1) 007330 104400 SCOPE

```

4598

```

(1) *****
(1) TEST 41: TEST THAT LINE 7 CAN TRANSMIT AND
(1) RECEIVE A CHARACTER. (377)
(1) 1S: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
(1) 3S: CHECKS THAT NO ERRORS IN FI/FO
(1) 4S: CHECKS THAT RIGHT LINE # (?) IN FI/FO
(1) 5S: CHECKS FOR RIGHT CHARACTER LENGTH
(1) 6S: CHECKS THAT CORRECT DATA WAS RECEIVED
(1) 7S: CHECKS THAT CHARACTER PRESENT CLEARS
(1) 8S: CHECKS THAT DONE CLEARS
(1) PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD 003 SERIES
(1) *****

```

```

(1) INITIALIZE
(1) DEVICE CSR REGISTER
(1) TST41: JSR PC, JINITD SET:
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER XMTR SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR
(1) SET:
(1) ;BIT0 = RECEIVER ENABLE
(1) 007336 052777 000200 171760 LOP41: BIS #BIT7, JTCR ;SET XMTR CONTROL BIT, LINE 7
(1) 007344 017701 171750 MOV JCSR, R1 ;WAIT FOR XMTR READY
(1) 007350 100375 BPL LOP41
(1) 007352 012777 000377 171746 MOV #377, JTBUF ;SEND A RUBOUT
(1) 007360 005000 CLR R0 ;CLEAR COUNTER
(1) 007362 105305 1S: DECB R5 ;SHORT WAIT LOOP
(1) 007364 001376 ANE 1S
(1) 007366 017701 171730 MOV JTBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 007372 100405 BMI 3S ;BRANCH WHEN FOUND
(1) 007374 005200 INC R0 ;TIME COUNTER
(1) 007376 001371 1S: BNE 1S ;BRANCH IF NOT TIME-OUT
(1) 007400 017702 171714 MOV JCSR, R2 ;SAVE CSR
(1) 007404 104002 HLT+2 ;CHARACTER READY DIDN'T SET
(1) ;R1 = CONTENTS OF RBUF
(1) ;R2 = CONTENTS OF CSR
(1) 007406 050037 001346 3S: BIS R0, TIMER ;SAVE THE TIMER
(1) 007412 032701 070000 BIT #70000, R1 ;CHECK FOR ERROR BITS
(1) 007416 001401 BEQ .+4 ;BRANCH IF NONE
(1) 007420 104001 HLT+1 ;ERROR IN RECEIVED CHAR
(1) ;R1 = CONTENTS OF RBUF
(1) ;BIT14=UART OVERRUN
(1) ;BIT13=FRAMING ERROR
(1) ;BIT12=PARITY ERROR
(1) 007422 010102 4S: MOV R1, R2 ;DUPLICATE DATA WORD
(1) 007424 042702 170377 BIC #170377, R2 ;MASK LINE#
(1) 007430 000302 SWAB R2 ;LINE # IN LOW BYTE
(1) 007432 :22702 000007 CMPB #7., R2 ;CHECK LINE #

```

E04

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77 15:06

DJ11 LOGIC TESTS
TST41: TEST ALL OF LINE 7 TRANSMIT AND RECEIVE LOGIC

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-23

SEG 0043

```

(1) 007436 001401 BEQ .+4 ; BRANCH IF OK
(1) 007440 104001 HLT+1 ; WRONG LINE # RECEIVED
(1) ; R1 = CONTENTS OF RBUF
(1) ; BITS8-11 = LINE #
(1) 007442 117702 171716 55: MOVB @DJLEN, R2 ; GET MASK OF CHARACTER
(1) 007446 130201 BITB R2, R1 ; CHECK CHAR LENGTH.
(1) 007450 001401 BEQ .+4 ; BRANCH IF OK
(1) 007452 104002 HLT+2 ; WRONG CHARACTER LENGTH
(1) ; R1=DATA FROM FI/FO
(1) ; R2=MASK (BITS SET NOT EXPECTED)
(1) 007454 105102 65: COMB R2 ; REVERSE THE MASK
(1) 007456 120102 CMPB R1, R2 ; CHECK THE ACTUAL DATA
(1) 007460 001401 BEQ .+4 ; BRANCH IF OK
(1) 007462 104002 HLT+2 ; WRONG CHAR LEN OR DATA ERROR
(1) ; R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ; R2=DATA (LOW BYTE) EXPECTED
(1) 007464 017701 171632 75: MOV @RBUF, R1 ; READ FI/FO
(1) 007470 100001 BPL .+4 ; BRANCH IF CHAR PRESENT NOT SET
(1) 007472 104001 HLT+1 ; CHARACTER PRESENT STAYED SET
(1) ; R1 = CONTENTS OF RBUF
(1) 007474 017701 171620 85: MOV @CSR, R1 ; SAVE THE CSR
(1) 007500 022701 100405 CMP #100405, R1 ; CHECK THE CSR
(1) 007504 001401 BEQ .+4 ; BRANCH IF OK
(1) 007506 104001 HLT+1 ; DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ; R1 = CONTENTS OF CSR
(1) 007510 005077 171610 CLR @TCR ; CLEAR TCR
(1) 007514 005077 171600 CLR @CSR ; CLEAR CSR
(1) 007520 104400 SCOPE
4599 007522 005237 001364 INC DJLEN
4600 007526 012737 007534 016070 MOV #.+6, LAD
4601

```

```

*****
: TEST 42: TEST THAT LINE 8 CAN TRANSMIT AND
: RECEIVE A CHARACTER. (377)
: 15: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
: 35: CHECKS THAT NO ERRORS IN FI/FO
: 45: CHECKS THAT RIGHT LINE # (8) IN FI/FO
: 55: CHECKS FOR RIGHT CHARACTER LENGTH
: 65: CHECKS THAT CORRECT DATA WAS RECEIVED
: 75: CHECKS THAT CHARACTER PRESENT CLEARS
: 85: CHECKS THAT DONE CLEARS
: PROBABLE FAULTY LOGIC: M7285 (02-7) ALL; M7279 ALL; UART CARD 003 SERIES
*****

```

INITIALIZE
DEVICE"CSR"REGISTER

```

(1) 007534 004737 015420 TST42: JSR PC, @INITD SET:
(1) ; BIT2 = MAINTENANCE
(1) ; BIT3 = CLEAR MOS
(1) ; BIT8 = MASTER XMTR SCAN ENB
(1) ; WAIT FOR MOS TO CLEAR
(1) ; SET:
(1) ; BIT0 = RECEIVER ENABLE
(1) 007540 052777 000400 171556 BIS #BIT8, @TCR ; SET XMTR CONTROL BIT, LINE 8
(1) 007546 017701 171546 LOP42: MOV @CSR, R1 ; WAIT FOR XMTR READY

```

F04

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-24
TST42: TEST ALL OF LINE 8 TRANSMIT AND RECEIVE LOGIC

SEG 0044

```

(1) 007552 100375          BPL      LOP42
(1) 007554 012777 000377 171544  MOV      #377,  @TBUF ; SEND A RUBOUT
(1) 007562 005000          CLR      R0 ; CLEAR COUNTER
(1) 007564 105305          DECB    R5 ; SHORT WAIT LOOP
(1) 007566 001376          BNE     1$
(1) 007570 017701 171526  MOV      @RBUF,R1 ; WAIT FOR CHAR. PRES.
(1) 007574 100405          BMI     3$ ; BRANCH WHEN FOUND
(1) 007576 005200          INC     R0 ; TIME COUNTER
(1) 007600 001371          BNE     1$ ; BRANCH IF NOT TIME-OUT
(1) 007602 017702 171512  MOV      @CSR,  R2 ; SAVE CSR
(1) 007606 104002          HLT+2 ; CHARACTER READY DIDN'T SET
(1) ; R1 = CONTENTS OF RBUF
(1) ; R2 = CONTENTS OF CSR
(1) 007610 050037 001346 3$:  BIS      R0,TIMER ; SAVE THE TIMER
(1) 007614 032701 070000  BIT      #70000,R1 ; CHECK FOR ERROR BITS
(1) 007620 001401          BEQ     .+4 ; BRANCH IF NONE
(1) 007622 104001          HLT+1 ; ERROR IN RECEIVED CHAR
(1) ; R1 = CONTENTS OF RBUF
(1) ; BIT14=UART OVERRUN
(1) ; BIT13=FRAMING ERROR
(1) ; BIT12=PARITY ERROR
(1) 007624 010102          MOV      R1,    R2 ; DUPLICATE DATA WORD
(1) 007626 042702 170377  BIC      #170377,R2 ; MASK LINE #
(1) 007632 000302          SWAB   R2 ; LINE # IN LOW BYTE
(1) 007634 122702 000010  CMPB    #8.,  R2 ; CHECK LINE #
(1) 007640 001401          BEQ     .+4 ; BRANCH IF OK
(1) 007642 104001          HLT+1 ; WRONG LINE # RECEIVED
(1) ; R1 = CONTENTS OF RBUF
(1) ; BITS8-11 = LINE #
(1) 007644 117702 171514 5$:  MOVB    @DJLEN, R2 ; GET MASK OF CHARACTER
(1) 007650 130201          BITB   R2,    R1 ; CHECK CHAR LENGTH.
(1) 007652 001401          BEQ     .+4 ; BRANCH IF OK
(1) 007654 104002          HLT+2 ; WRONG CHARACTER LENGTH
(1) ; R1=DATA FROM FI/FO
(1) ; R2=MASK (BITS SET NOT EXPECTED)
(1) 007656 105102          COMB   R2 ; REVERSE THE MASK
(1) 007660 120102          CMPB   R1,    R2 ; CHECK THE ACTUAL DATA
(1) 007662 001401          BEQ     .+4 ; BRANCH IF OK
(1) 007664 104002          HLT+2 ; WRONG CHAR LEN OR DATA ERROR
(1) ; R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ; R2=DATA (LOW BYTE) EXPECTED
(1) 007666 017701 171430 7$:  MOV      @RBUF, R1 ; READ FI/FO
(1) 007672 100001          BPL     .+4 ; BRANCH IF CHAR PRESENT NOT SET
(1) 007674 104001          HLT+1 ; CHARACTER PRESENT STAYED SET
(1) ; R1 = CONTENTS OF RBUF
(1) 007676 017701 171416 8$:  MOV      @CSR,  R1 ; SAVE THE CSR
(1) 007702 022701 100405  CMP      #100405,R1 ; CHECK THE CSR
(1) 007706 001401          BEQ     .+4 ; BRANCH IF OK
(1) 007710 104001          HLT+1 ; DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ; R1 = CONTENTS OF CSR
(1) 007712 005077 171406          CLR     @TCR ; CLEAR TCR
(1) 007716 005077 171376          CLR     @CSR ; CLEAR CSR
(1) 007722 104400          SCOPE

```

+602

:TEST 43: TEST THAT LINE 9 CAN TRANSMIT AND


```

(1) ;R1=DATA FROM FI/FO
(1) ;R2=MASK (BITS SET NOT EXPECTED)
(1) 010046 105102 6S: COMB R2 ;REVERSE THE MASK
(1) 010050 120102 CMPB R1 R2 ;CHECK THE ACTUAL DATA
(1) 010052 001401 BEQ .+4 ;BRANCH IF OK
(1) 010054 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
(1) ;R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ;R2=DATA (LOW BYTE) EXPECTED
(1) 010056 017701 171240 7S: MOV @RBUF, R1 ;READ FI/FO
(1) 010062 100001 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
(1) 010064 104001 HLT+1 ;CHARACTER PRESENT STAYED SET
(1) ;R1 = CONTENTS OF RBUF
(1) 010066 017701 171226 8S: MOV @CSR, R1 ;SAVE THE CSR
(1) 010072 022701 100405 CMP #100405, R1 ;CHECK THE CSR
(1) 010076 001401 BEQ .+4 ;BRANCH IF OK
(1) 010100 104001 HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ;R1 = CONTENTS OF CSR
(1) 010102 005077 171216 CLR @TCR ;CLEAR TCR
(1) 010106 005077 171206 CLR @CSR ;CLEAR CSR
(1) 010112 104400 SCOPE

```

4603

```

*****
TEST 44: TEST THAT LINE 10 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
1S: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
3S: CHECKS THAT NO ERRORS IN FI/FO
4S: CHECKS THAT RIGHT LINE # (10) IN FI/FO
5S: CHECKS FOR RIGHT CHARACTER LENGTH
6S: CHECKS THAT CORRECT DATA WAS RECEIVED
7S: CHECKS THAT CHARACTER PRESENT CLEARS
8S: CHECKS THAT DONE CLEARS
PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; JAPY CARD DC3 SERIES
*****

```

INITIALIZE
DEVICE"CSR"REGISTER

```

(1) ;SET:
(1) 010114 004737 015420 TST44: JSR PC, @INITD ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BITB = MASTER XMTR SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR
(1) ;SET:
(1) ;BIT0 = RECEIVER ENABLE
(1) 010120 052777 002000 171176 LOP44: BIS @BIT10, @TCR ;SET XMTR CONTROL BIT. LINE 10
(1) 010126 017701 171166 MOV @CSR, R1 ;WAIT FOR XMTR READY
(1) 010132 100375 BPL LOP44
(1) 010134 012777 000377 171164 MOV #377, @TBUF ;SEND A RUBOUT
(1) 010142 005000 CLR RO ;CLEAR COUNTER
(1) 010144 105305 1S: DECB R5 ;SHORT WAIT LOOP
(1) 010146 001376 BNE 1S
(1) 010150 017701 171146 MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 010154 100405 BMI 3S ;BRANCH WHEN FOUND
(1) 010156 005200 RO ;TIME COUNTER
(1) 010160 001371 BNE 1S ;BRANCH IF NOT TIME-OUT
(1) 010162 017702 171132 MOV @CSR, R2 ;SAVE CSR

```

```

(1) 010166 104002 HLT+2 ; CHARACTER READY DIDN'T SET
(1) ; R1 = CONTENTS OF RBUF
(1) ; R2 = CONTENTS OF CSR
(1) 010170 050037 001346 3S: BIS R0, TIMER ; SAVE THE TIMER
(1) 010174 032701 070000 BIT #70000, R1 ; CHECK FOR ERROR BITS
(1) 010200 001401 BEQ .+4 ; BRANCH IF NONE
(1) 010202 104001 HLT+1 ; ERROR IN RECEIVED CHAR
(1) ; R1 = CONTENTS OF RBUF
(1) ; BIT14=UART OVERRUN
(1) ; BIT13=FRAMING ERROR
(1) ; BIT12=PARITY ERROR
(1) 010204 010102 4S: MOV R1, R2 ; DUPLICATE DATA WORD
(1) 010206 042702 170377 BIC #170377, R2 ; MASK LINE #
(1) 010212 000302 SWAB R2 ; LINE # IN LOW BYTE
(1) 010214 122702 000012 CMPB #10., R2 ; CHECK LINE #
(1) 010220 001401 BEQ .+4 ; BRANCH IF OK
(1) 010222 104001 HLT+1 ; WRONG LINE # RECEIVED
(1) ; R1 = CONTENTS OF RBUF
(1) ; BITS8-11 = LINE #
(1) 010224 117702 171134 5S: MOVB @DJLEN, R2 ; GET MASK OF CHARACTER
(1) 010230 130201 BITB R2, R1 ; CHECK CHAR LENGTH.
(1) 010232 001401 BEQ .+4 ; BRANCH IF OK
(1) 010234 104002 HLT+2 ; WRONG CHARACTER LENGTH
(1) ; R1=DATA FROM FI/FO
(1) ; R2=MASK (BITS SET NOT EXPECTED)
(1) 010236 105102 6S: COMB R2 ; REVERSE THE MASK
(1) 010240 120102 CMPB R1, R2 ; CHECK THE ACTUAL DATA
(1) 010242 001401 BEQ .+4 ; BRANCH IF OK
(1) 010244 104002 HLT+2 ; WRONG CHAR LEN OR DATA ERROR
(1) ; R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ; R2=DATA (LOW BYTE) EXPECTED
(1) 010246 017701 171050 7S: MOV @RBUF, R1 ; READ FI/FO
(1) 010252 100001 BPL .+4 ; BRANCH IF CHAR PRESENT NOT SET
(1) 010254 104001 HLT+1 ; CHARACTER PRESENT STAYED SET
(1) ; R1 = CONTENTS OF RBUF
(1) 010256 017701 171036 8S: MOV @CSR, R1 ; SAVE THE CSR
(1) 010262 022701 100405 CMP #100405, R1 ; CHECK THE CSR
(1) 010266 001401 BEQ .+4 ; BRANCH IF OK
(1) 010270 104001 HLT+1 ; DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ; R1 = CONTENTS OF CSR
(1) 010272 005077 171026 CLR @TCR ; CLEAR TCR
(1) 010276 005077 171016 CLR @CSR ; CLEAR CSR
(1) 010302 104400 SCOPE

```

4604

```

*****
TEST 45: TEST THAT LINE 11 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
1S: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
3S: CHECKS THAT NO ERRORS IN FI/FO
4S: CHECKS THAT RIGHT LINE # (11) IN FI/FO
5S: CHECKS FOR RIGHT CHARACTER LENGTH
6S: CHECKS THAT CORRECT DATA WAS RECEIVED
7S: CHECKS THAT CHARACTER PRESENT CLEARS
8S: CHECKS THAT DONE CLEARS
PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
*****

```



```

(1)
(1)
(1)
(1)
(1) 010304 004737 015420
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 010310 052777 004000 171006
(1) 010316 017701 170776
(1) 010322 100375
(1) 010324 012777 000377 170774
(1) 010332 005000
(1) 010334 105305
(1) 010336 001376
(1) 010340 017701 170756
(1) 010344 100405
(1) 010346 005200
(1) 010350 001371
(1) 010352 017702 170742
(1) 010356 104002
(1)
(1)
(1) 010360 050037 001346
(1) 010364 032701 070000
(1) 010370 001401
(1) 010372 104001
(1)
(1)
(1)
(1)
(1) 010374 010102
(1) 010376 042702 170377
(1) 010402 000302
(1) 010404 122702 000013
(1) 010410 001401
(1) 010412 104001
(1)
(1)
(1)
(1)
(1) 010414 117702 170744
(1) 010420 130201
(1) 010422 001401
(1) 010424 104002
(1)
(1)
(1)
(1)
(1) 010426 105102
(1) 010430 120102
(1) 010432 001401
(1) 010434 104002
(1)
(1)
(1)
(1)
(1) 010436 017701 170660
(1) 010442 100001

```

```

INITIALIZE
DEVICE"CSR"REGISTER
TST45: JSR PC, @#INITD
;WAIT FOR MOS TO CLEAR
L0P45: BIS #BIT11, @TCR ;SET XMTR CONTROL BIT, LINE 11
MOV @CSR, R1 ;WAIT FOR XMTR READY
BPL L0P45
MOV #377, @RBUF ;SEND A RUBOUT
CLR R0 ;CLEAR COUNTER
1$: DEC R5 ;SHORT WAIT LOOP
BNE 1$
MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
BMI 3$ ;BRANCH WHEN FOUND
INC R0 ;TIME COUNTER
BNE 1$ ;BRANCH IF NOT TIME-OUT
MOV @CSR, R2 ;SAVE CSR
HLT+2 ;CHARACTER READY DIDN'T SET
;R1 = CONTENTS OF RBUF
;R2 = CONTENTS OF CSR
3$: BIS R0, TIMER ;SAVE THE TIMER
BIT #70000, R1 ;CHECK FOR ERROR BITS
BEQ .+4 ;BRANCH IF NONE
HLT+1 ;ERROR IN RECEIVED CHAR
;R1 = CONTENTS OF RBUF
;BIT14=UART OVERRUN
;BIT13=FRAMING ERROR
;BIT12=PARITY ERROR
4$: MOV R1, R2 ;DUPLICATE DATA WORD
BIC #170377, R2 ;MASK LINE#
SWAB R2 ;LINE # IN LOW BYTE
CMPB #11., R2 ;CHECK LINE #
BEQ .+4 ;BRANCH IF OK
HLT+1 ;WRONG LINE # RECEIVED
;R1 = CONTENTS OF RBUF
;BITS8-11 = LINE #
5$: MOV @DJLEN, R2 ;GET MASK OF CHARACTER
BITB R2, R1 ;CHECK CHAR LENGTH.
BEQ .+4 ;BRANCH IF OK
HLT+2 ;WRONG CHARACTER LENGTH
;R1=DATA FROM FI/FO
;R2=MASK (BITS SET NOT EXPECTED)
6$: COMB R2 ;REVERSE THE MASK
CMPB R1, R2 ;CHECK THE ACTUAL DATA
BEQ .+4 ;BRANCH IF OK
HLT+2 ;WRONG CHAR LEN OR DATA ERROR
;R1=DATA FROM FI/FO (COMPLETE WORD)
;R2=DATA (LOW BYTE) EXPECTED
7$: MOV @RBUF, R1 ;READ FI/FO
BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET

```



```

(1) ;BIT14=UART OVERRUN
(1) ;BIT13=FRAMING ERROR
(1) ;BIT12=PARITY ERROR
(1) 010576 010102 45: MOV R1 R2 ;DUPLICATE DATA WORD
(1) 010600 042702 170377 BIC #170377,R2 ;MASK LINE#
(1) 010604 000302 SWAB R2 ;LINE # IN LOW BYTE
(1) 010606 122702 000014 CMPB #12., R2 ;CHECK LINE #
(1) 010612 001401 BEQ .+4 ;BRANCH IF OK
(1) 010614 104001 HLT+1 ;WRONG LINE # RECEIVED
(1) ;R1 = CONTENTS OF RBUF
(1) ;BITS8-11 = LINE #
(1) 010616 117702 170542 55: MOVB @DJLEN, R2 ;GET MASK OF CHARACTER
(1) 010622 130201 BITB R2, R1 ;CHECK CHAR LENGTH.
(1) 010624 001401 BEQ .+4 ;BRANCH IF OK
(1) 010626 104002 HLT+2 ;WRONG CHARACTER LENGTH
(1) ;R1=DATA FROM FI/FO
(1) ;R2=MASK (BITS SET NOT EXPECTED)
(1) 010630 105102 65: COMB R2 ;REVERSE THE MASK
(1) 010632 120102 CMPB R1, R2 ;CHECK THE ACTUAL DATA
(1) 010634 001401 BEQ .+4 ;BRANCH IF OK
(1) 010636 104002 HLT+2 ;WRONG CHAR LEN OR DATA ERROR
(1) ;R1=DATA FROM FI/FO (COMPLETE WORD)
(1) ;R2=DATA (LOW BYTE) EXPECTED
(1) 010640 017701 170456 75: MOV @RBUF, R1 ;READ FI/FO
(1) 010644 100001 BPL .+4 ;BRANCH IF CHAR PRESENT NOT SET
(1) 010646 104001 HLT+1 ;CHARACTER PRESENT STAYED SET
(1) ;R1 = CONTENTS OF RBUF
(1) 010650 017701 170444 85: MOV @CSR, R1 ;SAVE THE CSR
(1) 010654 022701 100405 CMP #100405,R1 ;CHECK THE CSR
(1) 010660 001401 BEQ .+4 ;BRANCH IF OK
(1) 010662 104001 HLT+1 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1) ;R1 = CONTENTS OF CSR
(1) 010664 005077 170434 CLR @TCR ;CLEAR TCR
(1) 010670 005077 170424 CLR @CSR ;CLEAR CSR
(1) 010674 104400 SCOPE

```

4608

```

(1) ;*****
(1) ;TEST 47: TEST THAT LINE 13 CAN TRANSMIT AND
(1) ;RECEIVE A CHARACTER. (377)
(1) ;15: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
(1) ;35: CHECKS THAT NO ERRORS IN FI/FO
(1) ;45: CHECKS THAT RIGHT LINE # (13) IN FI/FO
(1) ;55: CHECKS FOR RIGHT CHARACTER LENGTH
(1) ;65: CHECKS THAT CORRECT DATA WAS RECEIVED
(1) ;75: CHECKS THAT CHARACTER PRESENT CLEARS
(1) ;85: CHECKS THAT DONE CLEARS
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD DC3 SERIES
(1) ;*****
(1) ;
(1) ; INITIALIZE
(1) ; DEVICE"CSR"REGISTER
(1) ;
(1) ;TST47: JSR PC, @INITD SET:
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER XMTR SCAN ENB
(1) ;
(1) ;WAIT FOR MOS TO CLEAR

```

```

(1)
(1)
(1)
(1) 010702 052777 020000 170414
(1) 010710 017701 170404
(1) 010714 100375
(1) 010716 012777 000377 170402
(1) 010724 005000
(1) 010726 105305
(1) 010730 001376
(1) 010732 017701 170364
(1) 010736 100405
(1) 010740 005200
(1) 010742 001371
(1) 010744 017702 170350
(1) 010750 104002
(1)
(1) 010752 050037 001346
(1) 010756 032701 070000
(1) 010762 001401
(1) 010764 104001
(1)
(1) 010766 010102
(1) 010770 042702 170377
(1) 010774 000302
(1) 010776 122702 000015
(1) 011002 001401
(1) 011004 104001
(1)
(1) 011006 117702 170352
(1) 011012 130201
(1) 011014 001401
(1) 011016 104002
(1)
(1) 011020 105102
(1) 011022 120102
(1) 011024 001401
(1) 011026 104002
(1)
(1) 011030 017701 170266
(1) 011034 100001
(1) 011036 104001
(1)
(1) 011040 017701 170254
(1) 011044 022701 100405
(1) 011050 001401
(1) 011052 104001
(1)
(1) 011054 005077 170244

```

;BIT0 = RECEIVER ENABLE
 ;SET XMTR CONTROL BIT, LINE 13
 ;WAIT FOR XMTR READY
 ;SEND A RUBOUT
 ;CLEAR COUNTER
 ;SHORT WAIT LOOP
 ;WAIT FOR CHAR. PRES.
 ;BRANCH WHEN FOUND
 ;TIME COUNTER
 ;BRANCH IF NOT TIME-OUT
 ;SAVE CSR
 ;CHARACTER READY DIDN'T SET
 ;R1 = CONTENTS OF RBUF
 ;R2 = CONTENTS OF CSR
 ;SAVE THE TIMER
 ;CHECK FOR ERROR BITS
 ;BRANCH IF NONE
 ;ERROR IN RECEIVED CHAR
 ;R1 = CONTENTS OF RBUF
 ;BIT14=UART OVERRUN
 ;BIT13=FRAMING ERROR
 ;BIT12=PARITY ERROR
 ;DUPLICATE DATA WORD
 ;MASK LINE #
 ;LINE # IN LOW BYTE
 ;CHECK LINE #
 ;BRANCH IF OK
 ;WRONG LINE # RECEIVED
 ;R1 = CONTENTS OF RBUF
 ;BITS8-11 = LINE #
 ;GET MASK OF CHARACTER
 ;CHECK CHAR LENGTH.
 ;BRANCH IF OK
 ;WRONG CHARACTER LENGTH
 ;R1=DATA FROM FI/FO
 ;R2=MASK (BITS SET NOT EXPECTED)
 ;REVERSE THE MASK
 ;CHECK THE ACTURAL DATA
 ;BRANCH IF OK
 ;WRONG CHAR LEN OR DATA ERROR
 ;R1=DATA FROM FI/FO (COMPLETE WORD)
 ;R2=DATA (LOW BYTE) EXPECTED
 ;READ FI/FO
 ;BRANCH IF CHAR PRESENT NOT SET
 ;CHARACTER PRESENT STAYED SET
 ;R1 = CONTENTS OF RBUF
 ;SAVE THE CSR
 ;CHECK THE CSR
 ;BRANCH IF OK
 ;DONE DIDN'T CLEAR OR OTHER CSR ERROR
 ;R1 = CONTENTS OF CSR
 ;CLEAR TCR

(1) 011060 005077 170234
(1) 011064 104400

CLR @CSR ;CLEAR CSR
SCOPE

4609

(1) ;*****
(1) ;TEST 50: TEST THAT LINE 14 CAN TRANSMIT AND
(1) ; RECEIVE A CHARACTER. (377)
(1) ; 1\$: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
(1) ; 3\$: CHECKS THAT NO ERRORS IN FI/FO
(1) ; 4\$: CHECKS THAT RIGHT LINE # (14) IN FI/FO
(1) ; 5\$: CHECKS FOR RIGHT CHARACTER LENGTH
(1) ; 6\$: CHECKS THAT CORRECT DATA WAS RECEIVED
(1) ; 7\$: CHECKS THAT CHARACTER PRESENT CLEARS
(1) ; 8\$: CHECKS THAT DONE CLEARS
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES
(1) ;*****

(1) ; INITIALIZE
(1) ; DEVICE"CSR"REGISTER
(1) ;
(1) ; TST50: JSR PC, @#INITD ;SET:
(1) ; ;BIT2 = MAINTENANCE
(1) ; ;BIT3 = CLEAR MOS
(1) ; ;BIT8 = MASTER XMTR SCAN ENB ,
(1) ;
(1) ; WAIT FOR MOS TO CLEAR
(1) ; SET:
(1) ; ;BIT0 = RECEIVER ENABLE

(1) 011072 052777 040000 170224 ; BIS #BIT14, @TCR ;SET XMTR CONTROL BIT, LINE 14
(1) 011100 017701 170214 ; MOV @CSR, R1 ;WAIT FOR XMTR READY
(1) 011104 100375 ; BPL LOP_J
(1) 011106 012777 000377 170212 ; MOV #377, @TBUF ;SEND A RUBOUT
(1) 011114 005000 ; CLR RO ;CLEAR COUNTER
(1) 011116 105305 ; 1\$: DECB R5 ;SHORT WAIT LOOP
(1) 011120 001376 ; BNE 1\$
(1) 011122 017701 170174 ; MOV @RBUF, R1 ;WAIT FOR CHAR. PRES.
(1) 011126 100405 ; BMI 3\$;BRANCH WHEN FOUND
(1) 011130 005200 ; INC RO ;TIME COUNTER
(1) 011132 001371 ; BNE 1\$;BRANCH IF NOT TIME-OUT
(1) 011134 017702 170160 ; MOV @CSR, R2 ;SAVE CSR
(1) 011140 104002 ; HLT+2 ;CHARACTER READY DIDN'T SET

(1) ; R1 = CONTENTS OF RBUF
(1) ; R2 = CONTENTS OF CSR
(1) 011142 050037 001346 ; 3\$: BIS RO, TIMER ;SAVE THE TIMER
(1) 011146 032701 070000 ; BIT #70000, R1 ;CHECK FOR ERROR BITS
(1) 011152 001401 ; BEQ .+4 ;BRANCH IF NONE
(1) 011154 104001 ; HLT+1 ;ERROR IN RECEIVED CHAR
(1) ; R1 = CONTENTS OF RBUF
(1) ; BIT14=UART OVERRUN
(1) ; BIT13=FRAMING ERROR
(1) ; BIT12=PARITY ERROR

(1) 011156 040102 ; 4\$: MOV R1, R2 ;DUPLICATE DATA WORD
(1) 011160 042702 170377 ; BIC #170377, R2 ;MASK LINE #
(1) 011164 000302 ; SWAB R2 ;LINE # IN LOW BYTE
(1) 011166 122702 000016 ; CMPB #14., R2 ;CHECK LINE #
(1) 011172 001401 ; BEQ .+4 ;BRANCH IF OK
(1) 011174 104001 ; HLT+1 ;WRONG LINE # RECEIVED
(1) ; R1 = CONTENTS OF RBUF

011176 117702 170162
011202 130201
011204 001401
011206 104002

55: MOVB @JLEN, R2
BITB R2, R1
BEQ .+4
HLT+2

:BITS8-11 = LINE #
:GET MASK OF CHARACTER
:CHECK CHAR LENGTH.
:BRANCH IF OK
:WRONG CHARACTER LENGTH
:R1=DATA FROM FI/FO
:R2=MASK (BITS SET NOT EXPECTED)

011210 105102
011212 120102
011214 001401
011216 104002

65: COMB R2
CMPB R1, R2
BEQ .+4
HLT+2

:REVERSE THE MASK
:CHECK THE ACTUAL DATA
:BRANCH IF OK
:WRONG CHAR LEN OR DATA ERROR
:R1=DATA FROM FI/FO (COMPLETE WORD)
:R2=DATA (LOW BYTE) EXPECTED

011220 017701 170076
011224 100001
011226 104001

75: MOV @RBUF, R1
BPL .+4
HLT+1

:READ FI/FO
:BRANCH IF CHAR PRESENT NOT SET
:CHARACTER PRESENT STAYED SET
:R1 = CONTENTS OF RBUF

011230 017701 170064
011234 022701 100405
011240 001401
011242 104001

85: MOV @CSR, R1
CMP #100405, R1
BEQ .+4
HLT+1

:SAVE THE CSR
:CHECK THE CSR
:BRANCH IF OK
:DONE DIDN'T CLEAR OR OTHER CSR ERROR
:R1 = CONTENTS OF CSR

011244 005077 170054
011250 005077 170044
011254 104400

CLR @TCR
CLR @CSR
SCOPE

:CLEAR TCR
:CLEAR CSR

4610

TEST 51: TEST THAT LINE 15 CAN TRANSMIT AND
RECEIVE A CHARACTER. (377)
15: CHECKS THAT CHAR PRESENT IS IN FI/FO IN REASONABLE TIME.
35: CHECKS THAT NO ERRORS IN FI/FO
45: CHECKS THAT RIGHT LINE # (15) IN FI/FO
55: CHECKS FOR RIGHT CHARACTER LENGTH
65: CHECKS THAT CORRECT DATA WAS RECEIVED
75: CHECKS THAT CHARACTER PRESENT CLEARS
85: CHECKS THAT DONE CLEARS

PROBABLE FAULTY LOGIC: M7285 (D2-7) ALL; M7279 ALL; UART CARD D03 SERIES

INITIALIZE
DEVICE CSR REGISTER

011256 004737 015420

TST51: JSR PC, @INITD

SET
:BIT2 = MAINTENANCE
:BIT3 = CLEAR MOS
:BIT8 = MASTER XMTR SCAN ENB

:WAIT FOR MOS TO CLEAR

SET
:BIT0 = RECEIVER ENABLE

011262 052777 100000 170034
011270 017701 170024
011274 100375
011276 012777 000377 170022
011304 005000
011306 105305
011310 00137E

LOP51: BIS #BIT15, @TCR
MOV @CSR, R1
BPL LOP51
MOV #377, @RBUF
CLR R0
15: DECB R5
BNE 15

:SET XMTR CONTROL BIT. LINE 15
:WAIT FOR XMTR READY
:SEND A RUBOUT
:CLEAR COUNTER
:SHORT WAIT LOOP

(1)	011312	017701	170004		MOV	0RBUF, R1		WAIT FOR CHAR. PRES.
(1)	011316	100405			BMI	35		BRANCH WHEN FOUND
(1)	011320	005200			INC	RO		TIME COUNTER
(1)	011322	001371			BNE	15		BRANCH IF NOT TIME-OUT
(1)	011324	017702	167770		MOV	0CSR, R2		SAVE CSR
(1)	011330	104002			HLT+2			CHARACTER READY DIDN'T SET
(1)								R1 = CONTENTS OF RBUF
(1)								R2 = CONTENTS OF CSR
(1)	011332	050037	001346	35:	BIS	RO, TIMER		SAVE THE TIMER
(1)	011336	032701	070000		BIT	#70000, R1		CHECK FOR ERROR BITS
(1)	011342	001401			BEQ	+.4		BRANCH IF NONE
(1)	011344	104001			HLT+1			ERROR IN RECEIVED CHAR
(1)								R1 = CONTENTS OF RBUF
(1)								BIT14=UART OVERRUN
(1)								BIT13=FRAMING ERROR
(1)								BIT12=PARITY ERROR
(1)	011346	010102		45:	MOV	R1, R2		DUPLICATE DATA WORD
(1)	011350	042702	170377		BIC	#170377, R2		MASK LINE #
(1)	011354	000302			SWAB	R2		LINE # IN LOW BYTE
(1)	011356	122702	000017		CMPB	#15., R2		CHECK LINE #
(1)	011362	001401			BEQ	+.4		BRANCH IF OK
(1)	011364	104001			HLT+1			WRONG LINE # RECEIVED
(1)								R1 = CONTENTS OF RBUF
(1)								BIT8-11 = LINE #
(1)	011366	117702	167772	55:	MOVB	0DJLEN, R2		GET MASK OF CHARACTER
(1)	011372	130201			BITB	R2, R1		CHECK CHAR LENGTH.
(1)	011374	001401			BEQ	+.4		BRANCH IF OK
(1)	011376	104002			HLT+2			WRONG CHARACTER LENGTH
(1)								R1=DATA FROM FI/FO
(1)								R2=MASK (BITS SET NOT EXPECTED)
(1)	011400	105102		65:	COMB	R2		REVERSE THE MASK
(1)	011402	120102			CMPB	R1, R2		CHECK THE ACTUAL DATA
(1)	011404	001401			BEQ	+.4		BRANCH IF OK
(1)	011406	104002			HLT+2			WRONG CHAR LEN OR DATA ERROR
(1)								R1=DATA FROM FI/FO (COMPLETE WORD)
(1)								R2=DATA (LOW BYTE) EXPECTED
(1)	011410	017701	167706	75:	MOV	0RBUF, R1		READ FI/FO
(1)	011414	100001			BPL	+.4		BRANCH IF CHAR PRESENT NOT SET
(1)	011416	104001			HLT+1			CHARACTER PRESENT STAYED SET
(1)								R1 = CONTENTS OF RBUF
(1)	011420	017701	167674	85:	MOV	0CSR, R1		SAVE THE CSR
(1)	011424	022701	100405		CMP	#100405, R1		CHECK THE CSR
(1)	011430	001401			BEQ	+.4		BRANCH IF OK
(1)	011432	104001			HLT+1			DONE DIDN'T CLEAR OR OTHER CSR ERROR
(1)								R1 = CONTENTS OF CSR
(1)	011434	005077	167664		CLR	0TCR		CLEAR TCR
(1)	011440	005077	167654		CLR	0CSR		CLEAR CSR
(1)	011444	104400			SCOPE			
+E11	011446	162737	000003	001364	SUB	#3, DJLEN		WORSE CASE TIME, ONE CHARACTER
+E12	011454	005237	001346		INC	TIMER		DUP TIMER
+E13	011460	013700	001346		MOV	TIMER, RO		
+E14	011464	006200			ASR	RO		/2
+E15	011466	006200			ASR	RO		/4
+E16	011470	005200			INC	RO		+1
+E17	011472	006137	001346		ROL	TIMER		TIMER * 2
+E18	011476	060037	001346		ADD	RO, TIMER		2.25 TIMES ONE CHARACTER TIME

4619 011502 012737 011510 016070
4620

MOV #.+6, LAD ;RESET LOOP ADDRESS

:TEST 52: TEST THAT CHARACTER PRESENT (BIT15) OF RBUF
IS CLEARED (BY CLEAR MOS).
:PROBABLE FAULTY LOGIC: M7285 (D2-B) E17,E14,E15; M7279; M7280

: INITIALIZE

(1) 011510 004737 015420

†ST52: JSR PC,@INITD
:SET:
:BIT2 = MAINTENANCE
:BIT3 = CLEAR MOS
:BIT8 = TRANS SCAN ENABLE
:WAIT FOR BIT4 = MOS CLEAR

(1) 011514 012777 000001 167602
(1) 011522 012704 000040
(1) 011526 017701 167566
(1) 011532 100375
(1) 011534 010477 167566
(1) 011540 005304
(1) 011542 001371

:SET: BIT0 = RECEIVER ENABLE
MOV @BIT0,@TCR ;TRANS CONTROL LINED
MOV @40,R4 ;SET UP COUNTER TO OUTPUT 32 CHAR'S
15: MOV @CSR,R1 ;WAIT FOR TRANS READY
BPL 15
MOV R4,@TBUF ;TRANSMIT COUNT
DEC R4 ;COUNT DOWN
BNE 15

(1) 011544 105777 167550
(1) 011550 100375

25: TSTB @CSR ;MAKE SURE DONE IS SET
BPL 25

(1) 011552 052777 000010 167540
(1) 011560 032777 000020 167532
(1) 011566 001374

35: BIS @BIT3,@CSR ;CLEAR MOS
BIT @BIT4,@CSR ;WAIT FOR CLR MOS TO FINISH
BNE 35

(1) 011570 017701 167526
(1) 011574 100001
(1) 011576 104001
(1) 011600 017701 167514
(1) 011604 022701 100405
(1) 011610 001401
(1) 011612 104001

MOV @RBUF,R1 ;CHECK RBUF AND SAVE
BPL .+4
HLT+1
MOV @CSR,R1 ;SAVE CSR
CMP @100405,R1 ;CHECK CSR
BEQ .+4 ;BRANCH IF OK
HLT+1 ;CSR ERROR. POSSIBILITIES:
;(1) DONE DIDN'T CLEAR
;(2) TRANSMITTER UART DIDN'T CLR.
;R1=CONTENTS OF CSR

(1) 011614 013700 001346
(1) 011620 105305
(1) 011622 001376

45: MOV TIMER, R0 ;SET UP TIMER
DECB R5 ;SHORT WAIT LOOP
BNE 45

(1) 011624 017701 167470
(1) 011630 105701
(1) 011632 100001
(1) 011634 104001

MOV @CSR,R1 ;SAVE CSR
TSTB R1 ;CHECK FOR DONE
BPL .+4 ;BRANCH IF OK
HLT+1 ;DONE CAME UP!
;MOS MUST NOT HAVE CLEARED
;TIMER COUNT

(1) 011636 005300
(1) 011640 001367

DEC R0
BNE 45


```

(1)
(1) 012004 017701 167312 MOV @RBUF, R1 ;CHECK RBUF FOR CHAR PRESENT
(1) 012010 100401 BMI .+4 ;BRANCH IF CHAR PRESENT
(1) 012012 104001 HLT+1 ;CHAR PRESENT MISSING
(1) ;R1 = CONTENTS OF RBUF
(1) 012014 022701 100001 CMP #100001,R1 ;CHECK THE DATA
(1) 012020 001401 BEQ .+4 ;BRANCH IF OK
(1) 012022 104001 HLT+1 ;RECEIVER ERROR
(1) ;R1 = CNTENTS OF RBUF
(1) 012024 017701 167272 MOV @RBUF, R1 ;CHECK RBUF FOR SECOND CHAR
(1) 012030 100401 BMI .+4 ;BRANCH IF CHAR PRESENT
(1) 012032 104001 HLT+1 ;CHAR PRESENT MISSING
(1) ;R1 = CONTENTS OF RBUF
(1) 012034 022701 100002 CMP #100002,R1 ;CHECK THE DATA
(1) 012040 001401 BEQ .+4 ;BRANCH IF OK
(1) 012042 104001 HLT+1 ;RECEIVER ERROR
(1) ;R1 = CNTENTS OF RBUF
(1) 012044 017701 167252 MOV @RBUF, R1 ;CHECK FOR NO MORE CHARACTERS
(1) 012050 100001 BPL .+4 ;BRANCH IF CHAR PRESENT CLEARED
(1) 012052 104001 HLT+1 ;CHAR PRESENT NOT CLEAR!
(1) ;R1 = CONTENTS OF RBUF
(1) 012054 017701 167240 MOV @CSR, R1 ;SAVE CSR
(1) 012060 022701 100405 CMP #100405,R1 ;CHECK CSR
(1) 012064 001401 BEQ .+4 ;BRANCH IF OK
(1) 012066 104001 HLT+1 ;CSR ERROR
(1) ;R1 = CONTENTS OF CSR
(1) 012070 005077 167230 CLR @TCR ;CLEAR TCR
(1) 012074 005077 167220 CLR @CSR ;CLEAR CSR
(1) 012100 104400 SCOPE

```

4622

```

*****
TEST 54: TEST THE SILO ALARM LEVEL AT WHICH DONE WILL SET.
TEST THAT RECEIVER ENABLE ON A 0 INHIBITS CHARACTER PRESENT.
PROBABLE FAULTY LOGIC: M7285 (D2-7) E32, E15; M7279 (D11-3), E19, E23
*****

```

```

INITIALIZE
DEVICE"CSR"REGISTER

```

```

(1) 012102 004737 015420 TST54: JSR PC,@INITO ;SET:
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER XMTR SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR

```

```

(1) 012106 052777 000001 167210 BIS #BIT0, @TCR ;SET:
(1) 012114 005737 001350 TST ALMFLG ;BIT0 = RECEIVER ENABLE
(1) 012120 001005 BNE 115 ;TRANS CONTROL, LINE 0
(1) 012122 004737 015464 JSR PC,ALMCK ;HAS THE ALARM LEVEL BEEN TESTED?
(1) 012126 012737 000001 001350 MOV #1,ALMFLG ;YES
(1) 012134 012777 000414 167156 115: MOV #414, @CSR ;NO, GO DO IT
(1) ;SET THE FLAG
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER TRAN SCAN ENB
(1) 012142 032777 000020 167150 105: BIT #BIT4, @CSR ;WAIT FOR MOS TO CLEAR
(1) 012150 001374 BNE 105

```

G05

MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS
DZDJA.E.P11 08-SEP-77 15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-38
TST54: TEST SILO ALARM LEVEL / RECEIVER ENABLE

SEG 0058

```

(1) ;OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
(1) ;LEVEL TO ALLOW DONE TO SET.
(1) 012152 013702 001356      MOV     COUNT,R2      ;SET CHAR COUNT
(1) 012156 017701 167136      15:    MOV     @CSR,  R1      ;WAIT FOR XMTR READY
(1) 012162 100375                BPL     15
(1) 012164 012777 000252 167134  MOV     @252,  @TBUF  ;SEND AN "*"
(1) 012172 005302                DEC     R2
(1) 012174 001370                BNE     15
(1) 012176 013700 001346      MOV     TIMER,  R0      ;SET UP TIMER
(1) 012202 105305                25:    DECB   R5          ;SHORT WAIT LOOP
(1) 012204 001376                BNE     25
(1) 012206 017701 167106      MOV     @CSR,  R1      ;SAVE CSR FOR TYPING
(1) 012212 105701                TSTB   R1              ;CHECK FOR DONE
(1) 012214 100002                BPL     35              ;BRANCH IF NOT SET
(1) 012216 104001                HLT+1                    ;DONE SET WHEN RCV ENB CLR
(1) ;RI=CONTENTS OF CSR
(1) 012220 000402                BR      45
(1) 012222 005300                35:    DEC     R0          ;TIMER COUNT
(1) 012224 001366                BNE     25              ;BRANCH IF MORE TIMER
(1) 012226 017701 167070      45:    MOV     @RBUF,  R1      ;CHECK AND SAVE FI/FO
(1) 012232 100001                BPL     .+4              ;BRANCH IF OK
(1) 012234 104001                HLT+1                    ;CHARACTER PRESENT IN FI/FO
(1) ;RI=DATA FROM FI/FO
(1) 012236 005277 167056      INC     @CSR          ;SET RECEIVER ENABLE
(1) 012242 017701 167054      55:    MOV     @RBUF,  R1      ;CHECK FOR CHARACTER PRESENT
(1) 012246 100403                BMI     65              ;BRANCH IF OK
(1) 012250 105300                DECB   R0              ;SHORT TIMER
(1) 012252 001373                BNE     55
(1) 012254 104001                HLT+1                    ;CHARACTER PRESENT MISSING
(1) ;RI = CONTENTS OF RBUF
(1) 012256 005077 167042      65:    CLR     @TCR          ;CLR TRANS CONTROL REG
(1) 012262 104400                SCOPE

```

+623

```

*****
;TEST 55: TEST THAT HALF DUPLEX (BIT1) DISABLES THE RECEIVER UARTS.
;PROBABLE FAULTY LOGIC: M7285 (D2-4) E32, E17, E22, (D2-2) E5, E1
*****

```

```

(1) 012264 004737 015410      TST55: JSR     PC,      @INITC ;INITIALIZE
(1) ;BIT1 = HALF DUPLEX
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT8 = MASTER TRAN SCAN ENB
(1) ;WAIT FOR MOS TO CLEAR
(1) ;BIT0 = RECEIVER ENABLE
(1) ;SET XMTR CONTROL BIT, LINE0
(1) 012270 012777 000001 167026  MOV     @BIT0, @TCR
(1) ;OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
(1) ;LEVEL TO ALLOW DONE TO SET.
(1) 012276 013702 001356      MOV     COUNT,R2      ;SET CHAR COUNT
(1) 012302 017701 167012      15:    MOV     @CSR,  R1      ;WAIT FOR XMTR READY
(1) 012306 100375                BPL     15
(1) 012310 012777 000252 167010  MOV     @252,  @TBUF  ;SEND AN "*"
(1) 012316 005302                DEC     R2
(1) 012320 001370                BNE     15

```

H05

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-39
TST55: TEST HALF DUPLEX

SEG 0059

```

(1) 012322 013700 001346      MOV    TIMER, R0      ;SET UP TIMER
(1) 012326 105305      25:  DECB    R5        ;SHORT WAIT LOOP
(1) 012330 001376      BNE    25
(1) 012332 017701 166762      MOV    @CSR, R1      ;SAVE CSR
(1) 012336 105701      TSTB   R1            ;CHECK FOR DONE
(1) 012340 100002      BPL    35            ;BRANCH IF NOT SET
(1) 012342 104001      HLT+1                ;DONE SET WHEN HALF DUPLEX (BIT1), SET
                                ;R1=CONTENTS OF CSP
(1) 012344 000402      BR     45
(1) 012346 005300      35:  DEC    R0        ;TIMER COUNT
(1) 012350 001366      BNE    25            ;BRANCH IF MORE TIMER
(1) 012352 017701 166744      45:  MOV    @RBUF, R1    ;CHECK AND SAVE FI/FO
(1) 012356 100001      BPL    .+4           ;BRANCH IF OK
(1) 012360 104001      HLT+1                ;CHARACTER PRESENT IN FI/FO
                                ;R1=DATA FROM FI/FO
(1) 012362 042777 000002 166730      BIC    #BIT1, @CSR   ;CLEAR HALF DUPLEX BIT
(1) 012370 000240      NOP
(1) 012372 000240      NOP
(1) 012374 000240      NOP
(1) 012376 000240      NOP
(1) 012400 017701 166716      55:  MOV    @RBUF, R1    ;CHECK FOR CHAR PRESENT
(1) 012404 100001      BPL    .+4           ;BRANCH IF CHAR NOT PRESENT
(1) 012406 104001      HLT+1                ;CHAR PRESENT AFTER H/D CLEARED
                                ;R1 = CONTENTS OF RBUF
(1) 012410 005077 166710      CLR    @TCR         ;CLR TRANS CONTROL REG
(1) 012414 104400      SCOPE

```

+624

```

*****
;TEST 56: TEST THAT RECEIVER INTERRUPT DOES NOT OCCUR AT LEVEL 5
;PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
*****

```

```

(1) 012416 012777 012514 166704 TST56: MOV    #ISR56, @RCVVEC ;SET UP XMTR INTERRUPT VECTOR
(1) 012424 012777 000340 166700      MOV    #340, @RCVLVL ;AT LEVEL 7
(1) 012432 042737 000340 177776      BIC    #340, @PS     ;CLEAR PS LEVEL
(1) 012440 052737 000240 177776      BIS    #240, @PS     ;SET PS TO LEVEL 5
(1) 012446 004737 015400      JSR    PC, @INITB   ;SET:
                                ;BIT2 = MAINTENANCE
                                ;BIT3 = CLEAR MOS
                                ;BIT6 = RECEIVER INTERRUPT ENABLE
                                ;BIT8 = MASTER TRANS SCAN ENABLE
                                ;WAIT FOR MOS TO CLEAR
                                ;BIT0 = RECEIVER ENABLE
(1) 012452 012777 000001 166644      MOV    #BIT0, @TCR  ;SET TRAN CONTROL BIT, LINE 0
                                ;OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
                                ;LEVEL TO ALLOW DONE TO SET.
(1) 012460 013702 001356      MOV    COUNT, R2    ;SET CHAR COUNT
(1) 012464 017701 166630      15:  MOV    @CSR, R1     ;WAIT FOR TRAN RDY
(1) 012470 100375      BPL    15
(1) 012472 012777 000025 166626      MOV    #25, @TBUF   ;SEND #25
(1) 012500 005302      DEC    R2
(1) 012502 001370      BNE    15
(1) 012504 105777 166610      25:  TSTB   @CSR        ;WAIT FOR DONE

```

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-40
TST56: TEST RECEIVER INTERRUPT LEVEL

SEG 0060

```

(1) 012510 100375 BPL 25
(1) 012512 000404 BR END56 ;OK, BRANCH IF NO INTERRUPT
(1)
(1) 012514 104000 ISR56: HLT ;SHOULDN'T HAVE INTERRUPTED AT LEVEL 5
(1) 012516 012716 012524 MOV #END56,(SP) ;MOVE NEW RTI ADR ONTO STACK
(1) 012522 000002 RTI
(1)
(1) 012524 017701 166572 END56: MOV @RBUF, R1 ;READ THE CHARACTER
(1) 012530 100401 BMI .+4 ;BRANCH IF CHAR PRESENT
(1) 012532 104001 HLT+1 ;CHAR PRESENT MISSING
(1) ;R1 = CONTENTS OF RBUF
(1) 012534 022701 100025 CMP #100025,R1 ;CHECK THE DATA
(1) 012540 001401 BEQ .+4 ;BRANCH IF OK
(1) 012542 104001 HLT+1 ;RECEIVED DATA ERROR
(1) ;R1 = CONTENTS OF RBUF
(1) 012544 013777 001332 166556 MOV RCVLVL, @RCVVEC
(1) 012552 012777 000004 166552 MOV #IOT, @RCVLVL
(1) 012560 005077 166540 CLR @TCR
(1) 012564 005077 166530 CLR @CSR
(1) 012570 042737 000340 177776 BIC #340, @PS
(1)
(1) 012576 104400 SCOPE

```

+625

```

(1) ;*****
(1) ;TEST 57: TEST THAT RECEIVER INTERRUPT OCCURES AT LEVEL 4
(1) ;PROBABLE FAULTY LOGIC: M7821 WIRING, PROPER PRIORITY CHIP
(1) ;*****
(1)

```

```

(1) 012600 012777 012700 166522 TST57: MOV #ISR57, @RCVVEC ;SET UP XMTR INTERRUPT VECTOR
(1) 012606 012777 000340 166516 MOV #340, @RCVLVL ;AT LEVEL 7
(1) 012614 042737 000340 177776 BIC #340, @PS ;CLEAR PS LEVEL
(1) 012622 052737 000200 177776 BIS #200, @PS ;SET PS TO LEVEL 4
(1) 012630 004737 015400 JSR PC, @INITB ;SET:
(1) ;BIT2 = MAINTENANCE
(1) ;BIT3 = CLEAR MOS
(1) ;BIT6 = RECEIVER INTERRUPT ENABLE
(1) ;BIT8 = MASTER TRANS SCAN ENABLE
(1) ;WAIT FOR MOS TO CLEAR
(1) ;BIT0 = RECEIVER ENABLE
(1) 012634 012777 000001 166462 MOV #BIT0, @TCR ;SET TRAN CONTROL BIT, LINE 0
(1) ;OUTPUT THE # OF CHARACTERS NECESSARY FOR THE SILO ALARM
(1) ;LEVEL TO ALLOW DONE TO SET.
(1) 012642 013702 001356 MOV COUNT, R2 ;SET CHAR COUNT
(1) 012646 017701 166446 IS: MOV @CSR, R1 ;WAIT FOR TRAN RDY
(1) 012652 100375 BPL 15
(1) 012654 012777 000025 166444 MOV #25, @TBUF ;SEND #25
(1) 012662 005302 DEC R2
(1) 012664 001370 BNE 15
(1) 012666 105777 166426 ZS: TSTB @CSR ;WAIT FOR DONE
(1) 012672 100375 BPL 25
(1) 012674 104000 HLT ;SHOULDN'T HAVE INTERRUPTED AT LEVEL 4
(1) 012676 000403 BR END57 ;CONTINUE
(1)
(1) 012700 012716 012706 ISR57: MOV #END57,(SP) ;MOVE NEW RTI ADR ONTO STACK
(1) 012704 000002 RTI

```

```

(1)
(1) 012706 017701 166410
(1) 012712 100401
(1) 012714 104001
(1)
(1) 012716 022701 100025
(1) 012722 001401
(1) 012724 104001
(1)
(1) 012726 013777 001332 166374
(1) 012734 012777 000004 166370
(1) 012742 005077 166356
(1) 012746 005077 166346
(1) 012752 042737 000340 177776
(1)
(1) 012760 104400

```

```

END57: MOV @RBUF, R1 ; READ THE CHARACTER
      BMI .+4 ; BRANCH IF CHAR PRESENT
      HLT+1 ; CHAR PRESENT MISSING
              ; R1 = CONTENTS OF RBUF
      CMP #100025,R1 ; CHECK THE DATA
      BEQ .+4 ; BRANCH IF OK
      HLT+1 ; RECEIVED DATA ERROR
              ; R1 = CONTENTS OF RBUF
      MOV RCVLVL, @RCVVEC
      MOV #IOT, @RCVLVL
      CLR @TCR
      CLR @CSR
      BIC #340, @#PS
      SCOPE

```

4626

```

*****
: TEST 60: TEST FI/FO OVERRUN
: THE FI/FO BUFFER SHOULD HOLD 64 CHARACTERS.
: PROBABLE FAULTY LOGIC: M7285 (D1-7) E32, E17, E22 (D2-2) E5, E1
*****

```

```

: INITIALIZE
: DEVICE CSR REGISTER

```

```

(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 012762 004737 015420
(1)
(1)
(1)
(1)
(1)
(1)
(1) 012766 012777 177777 166330
(1) 012774 012700 000100
(1) 013000 017701 166314
(1) 013004 100375
(1) 013006 000377 166314
(1) 013012 032701 020000
(1) 013016 001401
(1) 013020 104001
(1)
(1) 013022 005300
(1) 013024 001365
(1)
(1) 013026 013700 001346
(1) 013032 017701 166262
(1) 013036 100375
(1) 013040 105305
(1) 013042 001376
(1) 013044 017701 166250
(1) 013050 100401
(1) 013052 104001
(1)
(1) 013054 005300

```

```

TST60: JSR PC, @#INITD SET:
; BIT2 = MAINTENANCE
; BIT3 = CLEAR MOS
; BIT8 = MASTER XMTR SCAN END

; WAIT FOR MOS TO CLEAR
SET:
; BIT0 = RECEIVER ENABLE

:
MOV #177777, @TCR ; TRANS CONTROL BIT, ALL LINES
MOV #100, R0 ; SET UP COUNTER - 64. CHAR FI/FO BUFF
MOV @CSR, R1 ; SAVE AND WAIT FOR TRANS READY
BPL IS
SWAB @TBUF ; TRANSMIT LINE # ON LINE
BIT #BIT13, R1 ; CHECK FI/FO OVERRUN
BEQ .+4 ; BRANCH IF OK
HLT+1 ; FI/FO OVERRUN TOO SOON
; R1=CONTENTS OF CSR
; COUNT DOWN
DEC R0
BNE IS

2$: MOV TIMER, R0 ; SET UP TIMER
MOV @CSR, R1 ; WAIT FOR XMTR READY
BPL 2$
3$: DECB R5 ; SHORT WAIT LOOP
BNE 3$
MOV @CSR, R1 ; SAVE CSR FOR THE RECORD
BMI .+4 ; BRANCH IF TRANS READY
HLT+1 ; TRANS READY MISTERIOUSLY DISAPPEARED
; R1=CONTENTS OF CSR
DEC R0 ; TIME 3 CHARACTER LENGTHS

```

TST60: TEST FI/FO OVERRUN

(1) 013056 001370
 (1)
 (1)
 (1)
 (1) 013060 105701
 (1) 013062 100401
 (1) 013064 104001
 (1)
 (1) 013066 022701 100605
 (1) 013072 001401
 (1) 013074 104001
 (1)
 (1)
 (1)
 (1)
 (1)

```

      BNE      3$           ;BRANCH IF MORE TIME
;FI/FO SHOULD NOW BE FULL
      TSTB     R1           ;CHECK THAT DONE IS SET
      BMI     .+4          ;BRANCH IF OK
      HLT+1                    ;DONE DIDN'T COME UP!
;R1 = CONTENTS OF CSR
      CMP      #100605,R1   ;CHECK THAT FI/FO NOT OVERRUN
      BEQ     .+4          ;BRANCH IF OK
      HLT+1                    ;FI/FO OVERRUN SET
;OR SOME OTHER CSR ERROR
;R1=CONTENTS OF CSR

```

```

;*****
;TEST 60A: TEST THAT FI/FO OVERRUN COMES UP WHEN 65TH CHARACTER
;          IS RECEIVED WITHOUT READING FI/FO
;*****

```

(1) 013076 000377 166224
 (1) 013102 013700 001346
 (1) 013106 105305
 (1) 013110 001376
 (1) 013112 017701 166202
 (1) 013116 032701 020000
 (1) 013122 001003
 (1) 013124 005300
 (1) 013126 001367
 (1) 013130 104001
 (1)
 (1) 013132 022701 120605
 (1) 013136 001401
 (1) 013140 104001
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)

```

T60A: SWAB     @TBUF           ;SEND 65TH CHARACTER
      MOV     TIMER, R0       ;SET UP TIMER
11$:  DECB     R5              ;SHORT WAIT LOOP
      BNE     11$
      MOV     @CSR,R1        ;SAVE CSR
      BIT     #BIT13,R1      ;CHECK FI/FO OVERRUN
      BNE     12$           ;BRANCH WHEN SET
      DEC     R0             ;TIMER
      BNE     11$           ;BRANCH IF MORE TIME
      HLT+1                    ;FI/FO OVERRUN DIDN'T COMEUP
;R1 = CONTENTS OF CSR
12$:  CMP      #120605,R1    ;CHECK TOTAL CSR
      BEQ     .+4           ;BRANCH IF OK
      HLT+1                    ;SOMETHING IN CSR FOULED UP
;R1=CONTENTS OF CSR

```

```

;*****
;TEST 60B. TEST THAT READING THE RECEIVER BUFFER CAUSES FI/FO
;          OVERRUN TO CLEAR.
;          NOTE: BECAUSE OF TIMING OF THE FI/FO, FI/FO OVERRUN CAN COME
;                BACK UP AFTER READING ONE CHARACTER, SO A SECOND MUST
;                BE READ TO INSURE THAT FI/FO OVERRUN IS CLEAR.
;*****

```

(1) 013142 012700 000002
 (1) 013146 017701 166150
 (1) 013152 100401
 (1) 013154 104001
 (1)
 (1) 013156 032701 070000
 (1) 013162 001401
 (1) 013164 104001
 (1)
 (1)
 (1)
 (1)
 (1)

```

T60B: MOV      #2, R0        ;SET UP COUNTER - 2 CHARACTERS
21$:  MOV     @RBUF, R1      ;CHECK AND SAVE FIRST CHAR IN FI/FO
      BMI     .+4           ;BRANCH IF CHAR PRESENT
      HLT+1                    ;CHARACTER PRESENT GONE!
;CHECK RECEIVER ERRORS
      BIT     #070000,R1    ;BRANCH IF OK
      BEQ     .+4           ;RECEIVER ERROR
      HLT+1                    ;R1=CONTENTS OF RBUF
;BIT14=UART OVERRUN
;BIT13=FRAMMING ERROR
;BIT12=PARITY ERROR

```

```

(1) 013166 010102      MOV    R1,    R2      ;PUT LINE # IN R2
(1) 013170 000302      SWAB   R2
(1) 013172 042702      BIC    #177760,R2
(1) 013176 120102      CMPB   R1,    R2      ;CHECK DATA (=LINE#)
(1) 013200 001401      BEQ    .+4           ;BRANCH IF OK
(1) 013202 104001      HLT+1              ;WRONG DATA RECEIVED
(1)                                     ;R1=FI/FO DATA
(1)                                     ;(DATA SHOULD=LINE#)
(1) 013204 005300      22$:  DEC    R0      ;COUNT CHARACTERS
(1) 013206 001403      BEQ    24$          ;BRANCH WHEN DONE
(1) 013210 105305      23$:  DECB   R5      ;SHORT WAIT LOOP - GIVE FI/FO TIME
(1) 013212 001376      BNE    23$
(1) 013214 000754      BR     21$          ;GO READ ANOTHER
(1) 013216 017701      166076
(1) 013222 022701      100605      24$:  MOV    @CSR, R1      ;SAVE CSR
(1) 013226 001401      CMP    #100605,R1  ;CHECK THAT FI/FO OVERRUN CLEARED
(1) 013230 104001      BEQ    .+4           ;BRANCH IF OK
(1)                                     ;FI/FO OVERRUN DIDN'T CLR
(1)                                     ;OR SOMEOTHER CSR PROBLEM
(1)                                     ;R1=CONTENTS OF CSR

```

```

*****
;TEST 60C:      TEST THAT FI/FO OVERRUN INTERRUPT
;                DOESN'T OCCUR WHEN THE PROCESSOR IS AT LEVEL 5
*****

```

```

(1) 013232 042737 000340 177776 T60C: BIC    #340, @#PS ;CLEAR PSW
(1) 013240 052737 000240 177776      BIS    #240, @#PS ;SET PROCESSOR TO LEVEL 5
(1) 013246 012777 013336 166054      MOV    #32$, @RCVVEC ;SET UP RECEIVER INTERRUPT VEC
(1) 013254 012777 000340 166050      MOV    #340, @RCVLVL
(1) 013262 052777 010000 166030      BIS    #BIT12, @CSR ;SET STATUS ENABLE
(1) 013270 017701 166024
(1) 013274 022701 110605      MOV    @CSR, R1      ;SAVE CSR
(1) 013300 001401      CMP    #110605,R1  ;CHECK CSR
(1) 013302 104001      BEQ    .+4           ;CSR ERROR
(1)                                     ;R1=CONTENTS OF CSR
(1) 013304 000377 166016      SWAB   @TBUF        ;SEND LINE #
(1) 013310 005777 166004      30$:  TST    @CSR        ;WAIT FOR TRANSMITTER READY
(1) 013314 100375
(1) 013316 000377 166004      BPL    30$
(1) 013322 017701 165772      SWAB   @TBUF        ;SEND LINE #
(1) 013326 032701 020000      31$:  MOV    @CSR, R1      ;SAVE CSR
(1) 013332 001773      BIT    #BIT13, R1  ;WAIT FOR FI/FO OVERRUN
(1) 013334 000410      BEQ    31$
(1)                                     BR     33$          ;SKIP ISR
(1) 013336 017701 165756      32$:  MOV    @CSR, R1      ;SAVE CSR
(1) 013342 104001      HLT+1              ;INTERRUPT OCCURRED AT LEVEL 5
(1) 013344 005777 165752      TST    @RBUF        ;"POP" ONE CHARACTER
(1) 013350 012716 013362      MOV    #34$, (SP)  ;RESET RETURN ADDRESS
(1) 013354 000002      RTI                ;RETURN
(1) 013356 012700 000002      33$:  MOV    #2, R0      ;SET UP COUNTER - 2 CHARACTERS
(1) 013362 017701 165734      34$:  MOV    @RBUF, R1   ;READ ONE CHARACTER
(1) 013366 100401      BMI    .+4           ;BRANCH IF CHARACTER PRESENT
(1) 013370 104001      HLT+1

```



```

(1)
(1) 013372 032701 070000      BIT      #70000, R1      ;CHECK ERRORS
(1) 013376 001401           BEQ      .+4
(1) 013400 104001           HLT+1
(1)
(1) 013402 010102           MOV      R1, R2       ;DUP DATA
(1) 013404 000302           SWAB    R2
(1) 013406 042702 177760     BIC      #177760, R2   ;CLR ALL BUT LINE #
(1) 013412 120102           CMPB    R1, R2       ;CHECK DATA
(1) 013414 001401           BEQ      .+4
(1) 013416 104001           HLT+1
(1)
(1) 013420 005300           DEC      RO           ;COUNT CHARACTERS
(1) 013422 003403           BLE     36$          ;BRANCH IF DONE
(1) 013424 105305           35$:   DECB    R5     ;SHORT WAIT LOOP
(1) 013426 001376           BNE     35$
(1) 013430 000754           BR      34$          ;GO READ ANOTHER CHARACTER
(1)
(1) 013432 017701 165662     36$:   MOV      @CSR, R1   ;SAVE CSR
(1) 013436 022701 110605     CMP      #110605, R1 ;CHECK THAT FI/FO OVERRUN CLEARED
(1) 013442 001401           BEQ      .+4         ;BRANCH IF OK
(1) 013444 104001           HLT+1             ;FI/FO OVERRUN DIDN'T CLR
(1)                                     ;OR SOMEOTHER CSR PROBLEM
(1)                                     ;R1=CONTENTS OF CSR
(1)
(1)                                     ;*****
(1) TEST 600:      TEST THAT FI/FO OVERRUN INTERRUPT
(1)                                     ; OCCURS WHEN PROCESSOR IS AT LEVEL 4
(1)                                     ;*****
(1)
(1) 013446 042737 000340 177776   T600:  BIC      #340, @#PS ;CLEAR PROCESSOR LEVEL
(1) 013454 052737 000200 177776   BIS      #200, @#PS ;SET PROCESSOR TO LEVEL 4
(1) 013462 012777 013546 165640   MOV      #42$, @RCVVEC ;SET UP RECEIVER INTERRUPT VEC
(1) 013470 017701 165624           MOV      @CSR, R1    ;SAVE CSR
(1) 013474 022701 110605           CMP      #110605, R1 ;CHECK CSR
(1) 013500 001401           BEQ      .+4
(1) 013502 104001           HLT+1
(1)
(1) 013504 000377 165616           SWAB    @TBUF        ;SEND LINE #
(1) 013510 005777 165604           TST     @CSR         ;WAIT FOR TRANSMITTER READY
(1) 013514 100375           BPL     40$
(1) 013516 000377 165604           SWAB    @TBUF        ;SEND LINE #
(1) 013522 017701 165572           MOV      @CSR, R1    ;SAVE CSR
(1) 013526 032701 020000           BIT     #BIT13, R1   ;WAIT FOR FI/FO OVERRUN
(1) 013532 001773           BEQ     41$
(1) 013534 104001           HLT+1             ;INTERUPT DIDN'T OCCURE WHEN OVERRUN SET
(1)                                     ;R1 = CONTENTS OF CSR
(1) 013536 052777 000010 165554     BIS     #BIT3, @CSR  ;CLEAR MOS
(1) 013544 000470           BR      45$         ;SKIP TO THE END
(1)
(1) 013546 017701 165546           MOV      @CSR, R1    ;SAVE CSR
(1) 013552 022701 130605           CMP      #130605, R1 ;CHECK CSR
(1) 013556 001401           BEQ     .+4         ;BRANCH IF OK
(1) 013560 104001           HLT+1             ;CSR ERROR
(1)                                     ;R1 = CONTENTS OF CSR
(1) 013562 012700 000101           MOV     #101, RO     ;SET UP COUNTER - 65 CHARACTERS

```

```

(1) 013566 017701 165530 43$: MOV  @RBUF, R1 ;READ ONE CHARACTER
(1) 013572 100401 BMI  .+4 ;BRANCH IF CHARACTER PRESENT
(1) 013574 104001 HLT+1
(1)
(1) 013576 032701 070000 BIT  #70000, R1 ;CHECK ERRORS
(1) 013602 001401 BEQ  .+4
(1) 013604 104001 HLT+1
(1)
(1) 013606 010102 MOV  R1, R2 ;DUP DATA
(1) 013610 000302 SWAB R2
(1) 013612 042702 177760 BIC  #177760, R2 ;CLR ALL BUT LINE #
(1) 013616 120102 CMPB R1, R2 ;CHECK DATA
(1) 013620 001401 BEQ  .+4
(1) 013622 104001 HLT+1
(1)
(1) 013624 017701 165470 MOV  @CSR, R1 ;SAVE CSR
(1) 013630 005300 DEC  RO
(1) 013632 001420 BEQ  46$ ;GET OUT IF ALL CHAR'S READ
(1) 013634 022700 000100 CMP  #100, RO ;CHECK FOR FIRST CHAR READ
(1) 013640 001752 BEQ  43$ ;SKIP CSR CHECK ON FIRST CHAR
(1) 013642 020037 001356 CMP  RO, COUNT ;IF CHAR'S LEFT IN SILO IS LESS THEN
(1) ;ALARM LEVEL, DONE WILL GO AWAY.
(1) 013646 002405 BLT  44$
(1) 013650 022701 110605 CMP  #110605, R1 ;CHECK CSR
(1) 013654 001401 BEQ  .+4
(1) 013656 104001 HLT+1
(1)
(1) 013660 000742 BR  43$
(1)
(1) 013662 022701 110405 44$: CMP  #110405, R1 ;CHECK CSR
(1) 013666 001401 BEQ  .+4 ;BRANCH IF OK
(1) 013670 104001 HLT+1 ;DONE DIDN'T GO AWAY
(1) ;OR OTHER CSR ERROR
(1) 013672 000735 BR  43$
(1)
(1) 013674 017701 165422 46$: MOV  @RBUF, R1 ;READ A CHARACTER
(1) 013700 100001 BPL  .+4 ;BRANCH IF NO CHAR. PRES
(1) 013702 104001 HLT+1 ;CHAR. PRESENT!
(1)
(1) 013704 013777 001332 165416 MOV  RCVLVL, @RCVVEC
(1) 013712 012777 000004 165412 MOV  #IOT, @RCVLVL
(1) 013720 012716 013726 MOV  #45$, (SP) ;RESET RETURN ADDRESS ON STACK
(1) 013724 000002 RTI ;RESTORE PSW
(1) 013726 005077 165372 45$: CLR  @TCR
(1) 013732 005077 165362 CLR  @CSR
(1) 013736 042737 000340 177776 BIC  #340, @#PS ;LOWER PROCESSOR STATUS
(1)
(1) 013744 104400 SCOPE
(1)
4627
(1) ;*****
(1) ;TEST 61: TEST THAT UART OVERRUN IS DETECTED ON ALL LINES
(1) ;PROBABLE FAULTY LOGIC: M7285 (D2-2) E3; E1; M7279; M7280
(1) ;*****
(1)
(1) 013746 012777 000414 165344 TST61: MOV  #414, @CSR ;BIT2 = MAINTENANCE

```

B06

MAINDEC-11-DZDJA-E
DZDJA.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

TST61:

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-46
TEST RECEIVER UART OVERRUN ON ALL LINES

SEG 0066

```

(1)                                     ;BIT3 = CLEAR MOS
(1)                                     ;BITB = TRANS SCAN ENABLE
(1) 013754 032777 000020 165336 10$: BIT    #BIT4,  @CSR    ;WAIT FOR MOS TO CLEAR
(1) 013762 001374                                     ;
(1) 013764 005001 CLR    R1                                     ;SET UP LINE COUNTER
(1) 013766 012777 000001 165330 MOV    #1,   @TCR    ;TRANS CONTROL LINE 0
(1) 013774 017702 165320 LOP61: MOV    @CSR,  R2    ;WAIT FOR TRANS READY
(1) 014000 100375                                     ;
(1) 014002 012777 000001 165316 MOV    #1,   @TBUF   ;SEND #1
(1) 014010 017702 165304 2$: MOV    @CSR,  R2    ;WAIT FOR TRANS READY
(1) 014014 100375                                     ;
(1) 014016 012777 000002 165302 MOV    #2,   @TBUF   ;SEND #2
(1) 014024 013700 001346 3$: MOV    TIMER, R0    ;SET UP TIMER
(1) 014030 105305                                     ;SHORT TIME LOOP
(1) 014032 001376 BNE    3$                                     ;
(1) 014034 017702 165260 MOV    @CSR,  R2    ;SAVE CSR FOR THE RECORD
(1) 014040 000400 BR    .+2    ;NOP FOR TIMING
(1) 014042 005300 DEC    R0    ;TIMER COUNT
(1) 014044 001371 BNE    3$                                     ;
(1) 014046 005277 165246 INC    @CSR                                     ;SET RECEIVER ENABLE
(1) 014052 017702 165242 MOV    @CSR,  R2    ;SAVE CSR
(1) 014056 032702 000001 BIT    #BIT0, R2    ;CHECK RECEIVER ENABLE
(1) 014062 001001 BNE    .+4    ;BRANCH IF OK
(1) 014064 104002 HLT+2    ;RECEIVER ENABLE FAILED TO SET
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF CSR
(1) 014066 013700 001346 MOV    TIMER, R0    ;SET UP TIMER
(1) 014072 105305 4$: DEC    R5    ;WAIT SHORT TIME LOOP
(1) 014074 001376 BNE    4$                                     ;
(1) 014076 017702 165220 MOV    @RBUF, R2    ;CHECK FOR CHAR PRES
(1) 014102 100403 BMI    5$                                     ;OK
(1) 014104 005300 DEC    R0    ;TIMER COUNT
(1) 014106 001371 BNE    4$                                     ;
(1) 014110 104002 HLT+2    ;NO CHARACTER PRESENT
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014112 032702 040000 5$: BIT    #BIT14, R2    ;CHECK FOR UART OVERRUN
(1) 014116 001001 BNE    .+4    ;BRANCH IF OK
(1) 014120 104002 HLT+2    ;UART OVERRUN MISSING
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014122 122702 000002 CMP    #2,   R2    ;CHECK THE DATA
(1) 014126 001401 BEQ    .+4    ;BRANCH IF OK
(1) 014130 104002 HLT+2    ;DATA ERROR - 3RD CHAR OVERRUNS 2ND
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014132 010203 MOV    R2,   R3    ;DUP DATA
(1) 014134 000303 SWAB  R3                                     ;
(1) 014136 042703 177700 BIC    #177700, R3    ;MASK ALL BIT LINE #, ERROR BITS
(1) 014142 020103 CMP    R1,   R3    ;CHECK LINE #, ERRORS
(1) 014144 001401 BEQ    .+4    ;BRANCH IF OK
(1) 014146 104002 HLT+2    ;LINE # OR OTHER RBUF ERROR
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014150 017702 165146 7$: MOV    @RBUF, R2    ;CHECK FOR MORE DATA
(1) 014154 100002 BPL    8$    ;BRANCH IF OK

```

```

(1) 014156 104002          HLT+2          ;EXTRA DATA IN FI/FO!
(1)                                ;R1 = LINE #
(1)                                ;R2 = CONTENTS OF RBUF
(1) 014160 000773          BR            75
(1) 014162 005377 165132    BS:  DEC        @CSR          ;CLEAR RECEIVER ENABLE
(1) 014166 017702 165126    MOV        @CSR, R2        ;SAVE CSR
(1) 014172 022702 100404    CMP        #100404,R2      ;CHECK CSR
(1) 014176 001401          BEQ         .+4            ;BRANCH IF OK
(1) 014200 104002          HLT+2          ;CSR ERROR
(1)                                ;R1 = LINE NUMBER
(1)                                ;R2 = CONTENTS OF CSR
(1) 014202 005201          INC        R1            ;COUNT LINES
(1) 014204 006377 165114    ASL        @TCR          ;GO TO NEXT LINE
(1) 014210 103271          BCC        LOP61        ;BRANCH BACK IF MORE LINES
(1) 014212 005077 165102    CLK        @CSR
(1) 014216 104400          SCOPE

```

+628

```

:*****
:TEST 62:      TEST BITS OF BCSR FOR READ/WRITE CAPABILITY
:PROBABLE FAULTY LOGIC:  M7285 (D2-2) E5, E1, (D2-3) E16, E2, E19, E35
:*****

```

```

(1) 014220 012777 002010 165072 TST62: MOV        #002010,@CSR    ;SET CSR
(1) 014226 012777 177777 165070    MOV        #177777,@BCSR   ;SET ALL BITS OF BCSR
(1) 014234 017701 165064          MOV        @BCSR, R1      ;CHECK AND SAVE BCSR
(1) 014240 022701 177777          CMP        #177777,R1     ;CHECK THAT ALL THE BITS ARE SET
(1) 014244 001401          BEQ         .+4            ;BRANCH IF OK
(1) 014246 104001          HLT+1          ;BIT(S) OF BCSR FAILED TO SET
(1) 014250 005077 165050          CLR        @BCSR         ;CLEAR BCSR
(1) 014254 017701 165044          MOV        @BCSR, R1     ;CHECK THAT IT CLEARED AND SAVE
(1) 014260 001401          BEQ         .+4            ;BRANCH IF CLR
(1) 014262 104001          HLT+1          ;BIT(S) OF BCSR FAILED TO CLEAR
(1) 014264 104400          SCOPE

```

+629

```

:*****
:TEST 63:      TEST THAT LINED CAN TRANSMIT AND RECEIVE A BREAK
:                ALSO CHECKS FRAMING ERROR(RBUF BIT13)
:                ALSO CHECKS PARITY ERROR(RBUF BIT12)
:                IF ODD PARITY IS SELECTED
:PROBABLE FAULTY LOGIC:  M7285 (D2-2) E5, E1, (D2-3) E16, E2, E19, E35
:*****

```

```

(1) 014266 004737 015370    TST63: JSR        PC,      @INITA ;INITIALIZE
(1)                                ;BIT2 = MAINTENANCE
(1)                                ;BIT3 = CLEAR MOS
(1)                                ;BIT10= R/W BCSR
(1)                                ;WAIT FOR MOS TO CLEAR
(1)                                ;BIT0 = RECEIVER ENABLE
(1) 014272 012777 000001 165024    MOV        #1, @BCSR     ;SEND BREAKS, LINE 0
(1) 014300 013700 001346    MOV        TIMER, RC    ;SET JP TIMER

```

```

(1) 014304 105305 15: DEC8 R5 ;SHORT WAIT LOOP
(1) 014306 001376 BNE 15
(1) 014310 017701 165006 MOV @RBUF, R1 ;SAVE CHAR PRES
(1) 014314 100403 BMI 25 ;BRANCH WHEN FOUND
(1) 014316 005200 INC R0 ;WAIT A WHILE
(1) 014320 001371 BNE 15
(1) 014322 104001 HLT+1 ;CHAR PRES NEVER CAME UP
;R1=CONTENTS OF RBUF

(1) 014324 032701 020000 25: BIT #020000,R1 ;CHECK FOR FRAMING ERROR
(1) 014330 001001 BNE .+4 ;BRANCH IF OK
(1) 014332 104001 HLT+1 ;FRAMING ERROR NOT UP
;R1=CONTENTS OF RBUF

(1) 014334 133737 001366 001300 BITB DJPAR, PARITY ;CHECK ODD PARITY FLAG
(1) 014342 001404 BEQ 35 ;BRANCH IF NOT
(1) 014344 032701 010000 BIT #010000,R1 ;CHECK PARITY ERROR
(1) 014350 001005 BNE 45 ;
(1) 014352 104001 HLT+1 ;ODD PARITY SHOULD CAUSE PARITY ERROR
;R1=CONTENTS OF RBUF

(1) 014354 032701 010000 35: BIT #010000,R1 ;CHECK PARITY ERROR
(1) 014360 001401 BEQ .+4 ;BRANCH IF OK
(1) 014362 104001 HLT+1 ;EVEN PARITY OR NO PARITY
;SHOULDN'T CAUSE PARITY ERROR
;R1=CONTENTS OF RBUF

(1) 014364 032701 040000 45: BIT #040000,R1 ;CHECK UART OVERRUN
(1) 014370 001401 BEQ .+4 ;BRANCH IF OK
(1) 014372 104001 HLT+1 ;UART OVERRUN SET!!
;R1=CONTENTS OF RBUF

(1) 014374 032701 007400 BIT #007400,R1 ;CHECK LINE #
(1) 014400 001401 BEQ .+4 ;BRANCH IF OK
(1) 014402 104001 HLT+1 ;WRONG LINE# IN FI/FO
;R1=CONTENTS OF RBUF

(1) 014404 105701 TSTB R1 ;CHECK DATA
(1) 014406 001401 BEQ .+4 ;BRANCH IF OK
(1) 014410 104001 HLT+1 ;WRONG DATA RECEIVED
;R1=CONTENTS OF RBUF

(1) 014412 017701 164704 MOV @RBUF, R1 ;READ FI/FO AGAIN
(1) 014416 100001 BPL .+4 ;BRANCH IF OK
(1) 014420 104001 HLT+1 ;EXTRA CHAR IN FI/FO
;R1 = CONTENTS OF RBUF

(1) 014422 005077 164676 55: CLR @BCSR ;CLEAR BREAK CONTROL REG
(1) 014426 105305 DEC8 R5 ;SHORT WAIT LOOP-REGISTER A MARK
(1) 014430 001376 BNE 55
(1) 014432 10440C SCOPE

```

+E3C

```

*****
:TEST 64: TEST THAT EACH LINE CAN TRANSMIT AND RECEIVE A BREAK
: ALSO CHECKS FRAMING ERROR(RBUF BIT13)
: ALSO CHECKS PARITY ERROR(RBUF BIT12)
: IF ODD PARITY IS SELECTED
:PROBABLE FAULTY LOGIC: M7285 (D2-2) E5, E1, (D2-3) E16, E2, E19, E35
*****

```

E06

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

TST64:

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-49
TEST BREAKS ON ALL LINES

SEQ 0069

```

(1)
(1) 014434 004737 015370      TST64: JSR      PC,      @INITA ; INITIALIZE
(1)                                     ; BIT2 = MAINTENANCE
(1)                                     ; BIT3 = CLEAR MOS
(1)                                     ; BIT10= R/W BCSR
(1)                                     ; WAIT FOR MOS TO CLEAR
(1)                                     ; BIT0 = RECEIVER ENABLE
(1)                                     ; SET UP LINE COUNTER
(1) 014440 005001                CLR      R1
(1) 014442 012704 000001        MOV      #1,      R4
(1) 014446 013700 001346        LOP64: MOV      TIMER, R0
(1) 014452 050477 164646        BIS      R4,      @BCSR ; SET UP LINE MARKER
(1) 014456 105305                BNE     R5,      @BCSR ; SET BREAK CONTROL BIT, LINE # IN R:
(1) 014460 001376                BNE     R5,      @BCSR ; SHORT WAIT LOOP
(1) 014462 017702 164634        MOV      @RBUF,  R2 ; READ RBUF FOR CHAR PRES
(1) 014466 100403                BMI     R5,      @BCSR ; BRANCH WHEN FOUND
(1) 014470 005200                INC     R0 ; WAIT A WHILE
(1) 014472 001371                BNE     R5,      @BCSR
(1) 014474 104002                HLT+2 ; CHAR PRES NEVER CAME UP
(1)                                     ; R1 = LINE #
(1)                                     ; R2 = CONTENTS OF RBUF
(1)
(1) 014476 032702 020000        25:    BIT      #020000,R2 ; CHECK FOR FRAMING ERROR
(1) 014502 001001                BNE     .+4 ; BRANCH IF OK
(1) 014504 104002                HLT+2 ; FRAMING ERROR NOT UP
(1)                                     ; R1 = LINE #
(1)                                     ; R2 = CONTENTS OF RBUF
(1)
(1) 014506 010103                MOV      R1,      R3 ; GET LINE #
(1) 014510 006203                ASR     R3 ; /4 TO GET RID OF BITS 1 & 0
(1) 014512 006203                ASR     R3
(1) 014514 006303                ASL     R3 ; SHIFT BACK FOR EVEN ADDR OF PARITY TABLE
(1) 014516 133763 001266 001300  BITB    DJPAR,PARITY(3) ; CHECK ODD PARITY FLAG
(1) 014524 001404                BEQ     R5,      @BCSR ; BRANCH IF NOT
(1) 014526 032702 010000        BIT      #010000,R2 ; CHECK PARITY ERROR
(1) 014532 001005                BNE     R5,      @BCSR
(1) 014534 104002                HLT+2 ; ODD PARITY SHOULD CAUSE PARITY ERROR
(1)                                     ; R1 = LINE #
(1)                                     ; R2 = CONTENTS OF RBUF
(1)
(1) 014536 032702 010000        35:    BIT      #010000,R2 ; CHECK PARITY ERROR
(1) 014542 001401                BEQ     R5,      @BCSR ; BRANCH IF OK
(1) 014544 104002                HLT+2 ; EVEN PARITY OR NO PARITY
(1)                                     ; SHOULDN'T CAUSE PARITY ERROR
(1)                                     ; R1 = LINE #
(1)                                     ; R2 = CONTENTS OF RBUF
(1)
(1) 014546 032702 040000        45:    BIT      #040000,R2 ; CHECK UART OVERRUN
(1) 014552 001401                BEQ     .+4 ; BRANCH IF OK
(1) 014554 104002                HLT+2 ; UART OVERRUN SET!!
(1)                                     ; R1 = LINE #
(1)                                     ; R2 = CONTENTS OF RBUF
(1)
(1) 014556 010203                MOV      R2,      R3 ; DUP DATA
(1) 014560 000303                SWAB   R3 ; LINE # IN LOW BYTE
(1) 014562 042703 177760        BIC     #17760,R3 ; MASK ALL BUT LINE #
(1) 014566 020103                CMP     R1,      R3 ; CHECK FOR RIGHT LINE #
(1) 014570 001401                BEQ     .+4 ; BRANCH IF OK
(1) 014572 104002                HLT+2 ; WRONG LINE # IN FI/FC

```

```

(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1)
(1)
(1) 014574 105702          TSTB   R2          ;CHECK DATA
(1) 014576 001401          BEQ    .+4         ;BRANCH IF OK
(1) 014600 104002          HLT+2        ;WRONG DATA RECEIVED
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014602 017702 164514  MOV    @RBUF, R2   ;READ FI/FO AGAIN
(1) 014606 100001          BPL    .+4         ;BRANCH IF OK
(1) 014610 104002          HLT+2        ;EXTRA CHAR IN FI/FO
(1)                                     ;R1 = LINE #
(1)                                     ;R2 = CONTENTS OF RBUF
(1) 014612 005077 164506  CLR    @BCSR       ;CLEAR BREAK CONTROL REG
(1) 014616 005201          INC    R1          ;COUNT LINES
(1) 014620 006304          ASL   R4           ;UPDATE LINE MARKER
(1) 014622 103311          BCC   LOP64       ;BRANCH IF MORE LINES
(1)
(1) 014624 005077 164470  CLR    @CSR        ;CLEAR CSR
(1) 014630 104400          SCOPE
(1)
4631 014632 012737 000002 016072  MOV    #2, TIMES
4632
(1)
(1) *****
(1) TEST 65: TEST THAT RESET CLEARS ALL BUFFERS
(1) NOTE: THE FI/FO BUFFER IS NOT COMPLETELY CLEARED
(1) BY RESET; ONLY CHARACTER PRESENT IS CLEARED.
(1) PROBABLE FAULTY LOGIC: M7285 (D2-B) E13
(1) *****
(1)
(1) 014640 052737 000340 177776  TST65: BIS    #340,@#PS ;SET PROCESSOR TO LEVEL 7
(1) 014646 012777 177777 164450  MOV    #177777,@TCR ;SET ALL TCR BITS
(1) 014654 012777 052507 164436  MOV    #052507,@CSR ;SET ALL R/W BITS OF CSR
(1) 014662 012777 177777 164434  MOV    #177777,@BCSR ;SET ALL BREAK CONTROL BITS (SEND BREAKS)
(1) ;NOTE: ALL LINES SHOULD BE SENDING BREAKS, BUT NONE SHOULD BE RECEIVING
(1) BECAUSE THE HALF DUPLEX BIT IS SET
(1) 014670 012777 177777 164430  MOV    #177777,@TBUF ;LOADING TRANS BUFF WHEN BREAK BIT SET
(1) ; SHOULD DO NOTHING
(1) 014676 000005          RESET        ;CLEAR THE WORLD
(1) 014700 013737 001320 014710  MOV    CSR,IS     ;CHECK CSR AND SAVE
(1) 014706 013701          MOV    @ (PC)+,R1
(1) 014710 000000 15:    000000
(1) 014712 001401          BEQ    .+4
(1) 014714 104001          HLT+1
(1)
(1) 014716 017701 164402          MOV    @TCR,R1   ;CHECK TCR AND SAVE
(1) 014722 001401          BEQ    .+4
(1) 014724 104001          HLT+1
(1)
(1) 014726 017701 164372          MOV    @BCSR,R1 ;CHECK BCSR AND SAVE
(1) 014732 001401          BEQ    .+4
(1) 014734 104001          HLT+1
(1)
(1) 014736 017701 164360          MOV    @RBUF,R1 ;CHECK RBUF AND SAVE
(1) 014742 :0000:          BPL    .+4

```

```

(1) 014744 104001
(1)
(1) 014746 017701 164354
(1) 014752 001401
(1) 014754 104001
(1)
(1) 014756 104400
(1)

```

```

HLT+1
MOV @TBUF,R1 ;CHECK TBUF AND SAVE
BEQ .+4
HLT+1
SCOPE

```

+633

```

:*****
:TEST 66: SEND A BINARY COUNT PATTERN ON EACH LINE
:PROBABLE FAULTY LOGIC: COULD BE ALMOST ANYWHERE!
:*****

```

```

(1) 014760 005001
(1) 014762 012703 100000
(1) 014766 012777 015164 164334
(1) 014774 012777 000240 164330
(1) 015002 042737 000340 177776
(1) 015010 052737 000200 177776
(1) 015016 012700 000001
(1) 015022 012777 010514 164270
(1)
(1)
(1)
(1)

```

```

TST66: CLR R1 ;SET UP LINE COUNTER
MOV #100000,R3 ;SET UP RCV DATA
MOV #ISR66,@RCVVEC ;SET UP RECEIVER INTERRUPT VECTOR
MOV #240,@RCVLVL
BIC #340,@PS ;CLEAR PROCESSOR PRIORITY
BIS #200,@PS ;SET PRIORITY TO 4
MOV #1,R0 ;SET UP LINE MARKER
MOV #010514,@CSR ;BIT2 = MAINTENANCE
;BIT3 = CLR MOS
;BIT6 = RECEIVER INTERRUPT ENB
;BIT8 = TRANS SCAN ENABLE
;BIT12= STATUS ENABLE
;WAIT FOR MOS TO CLEAR

```

```

(1) 015030 032777 000020 164262 10$: BIT #BIT4,@CSR
(1) 015036 001374 BNE 10$
(1) 015040 052777 000001 164252 BIS #1,@CSR
(1) 015046 005004 1$: CLR R4 ;SET THE RCV EN BIT
(1) 015050 010077 164250 MOV R0,@TCR ;SET FOR OUTPUT DATA
(1) 015054 005777 164240 2$: TST @CSR ;TRANS CONTROL, ONE LINE AT A TIME
(1) 015060 100375 BPL 2$ ;WAIT FOR TRANS READY
(1) 015062 010477 164240 MOV R4,@TBUF ;SEND DATA
(1) 015066 105204 INCB R4 ;BINARY COUNT
(1) 015070 123704 001360 CMPB SUM,R4 ;MAKE SURE THE CORECT NUMBER IS OUTPUTED
(1) 015074 001367 BNE 2$
(1) 015076 147704 164262 3$: BICB @DJLEN,R4 ;SET FOR MAX. RECIEVER COUNT
(1) 015102 120403 CMPB R4,R3 ;WAIT FOR RECEIVER DONE
(1) 015104 001374 BNE 3$
(1) 015106 017702 164206 MOV @CSR,R2 ;SAVE CSR
(1) 015112 022702 110505 CMP #110505,R2 ;CHECK CSR
(1) 015116 001401 BEQ .+4 ;BRANCH IF OK
(1) 015120 104002 HLT+2 ;CSR ERROR
;R1=LINE #
;R2=CONTENTS OF CSR

```

```

(1) 015122 017702 164174 MOV @RBUF,R2 ;CHECK CHARACTER PRESENT
(1) 015126 100001 BPL .+4 ;BRANCH IF OK
(1) 015130 104002 HLT+2 ;CHARACTER PRESENT SET!!
;R1=LINE #
;R2=CONTENTS OF CSR

```

```

(1) 015132 105003 CLRB R3 ;RESET EXPECTED DATA
(1) 015134 062703 000400 ADC #400,R3 ;UPDATE LINE # IN EXPECTED DATA
(1) 015140 005201 INC R1 ;UPDATE LINE #
(1) 015142 032701 000003 BIT #3,R1 ;CHECK FOR FOURTH LINE
(1) 015146 001002 BNE #5 ;BRANCH IF NOT

```


H06

MAINDEC-11-DZDJA-E
CZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

TST66: MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-52
TEST ALL DATA ON EACH LINE

SEG 0072

(1)	015150	005237	001364		INC	DJLEN		; MOVE CHARACTER LENGTH POINTER
(1)	015154	000241		45:	CLC			
(1)	015156	006300			ASL	RO		; UPDATE LINE MARKER
(1)	015160	103332			BCC	15		; BRANCH IF MORE
(1)	015162	000416			BR	END66		; SKIP ISR
(1)	015164	017702	164132	ISR66:	MOV	RBUF, R2		; READ FIRST DATA
(1)	015170	100401			BMI	115		; BRANCH IF CHARACTER PRESENT
(1)	015172	104003			HLT+3			; INTERRUPT BUT NO CHAR PRESENT
(1)								; R1=LINE #
(1)								; R2=CONTENTS OF RBUF
(1)								; R3=EXPECTED DATA
(1)	015174	147703	164164	115:	BICB	DJLEN, R3		; MASK CHAR. LENGTH
(1)	015200	020203			CMP	R2, R3		; CHECK THE DATA
(1)	015202	001401			BEG	.+4		; BRANCH IF OK
(1)	015204	104003			HLT+3			; DATA ERROR
(1)								; R1=LINE #
(1)								; R2=CONTENTS OF RBUF
(1)								; R3=EXPECTED DATA
(1)	015206	105203			INCB	R3		; UPDATE EXPECTED DATA
(1)	015210	017702	164106		MOV	RBUF, R2		; READ MORE DATA
(1)	015214	100767			BMI	115		; BRANCH IF MORE
(1)	015216	000002		125:	RTI			; RETURN
(1)	015220	162737	000004	001364	END66:	4, DJLEN		; RESET CHAR LENGTH POINTER
(1)	015226	013777	001332	164074	MOV	RCV LVL, RCVVEC		; RESTORE RECEIVER INT. VEC
(1)	015234	012777	000004	164070	MOV	RCV LVL, RCVVEC		
(1)	015242	005077	164056		CLR	TCR		; CLEAR TCR
(1)	015246	005077	164046		CLR	CSR		; CLEAR CSR
(1)	015252	104400			SCOPE			
4634	015254	012737	000020	016072	MOV	20, TIMES		
4635	015262	023737	001362	001344	CMP	DJUT, UNITS		; CHECK FOR LAST UNIT
4636	015270	002004			BGE	DONE		; BRANCH IF LAST UNIT
4637	015272	005037	001350		CLR	ALMFLG		; CLEAR FLAG FOR NEXT UNIT
4638	015276	000137	002330		JMP	RESTAR		; JUMP IF NOT
4639								
4640	015302				DONE:			
(1)	015302	004737	020054		JSR	PC, KBDINT		
(1)	015306	062737	000001	001316	ADD	1, PCNT+2		; ADD 1 TO THE PASS COUNT
(1)	015314	005537	001314		ADC	PCNT		; MAKE IT DOUBLE PREC.
(1)	015320	000004	017336		TYPE	MEOP		; END OF PASS INDICATOR
(1)	015324	032777	002000	164042	BIT	SW10, SWR		; RING THE BELL?
(1)	015332	001004			BNE	45		; NO!
(1)	015334	000004	000007		TYPE	BELL		; RING THE BELL
(1)	015340	000004	000177		TYPE	177		; TYPE A FILLER FOR 11/05
(1)	015344	013700	000042		45:	42, RO		; GET MONITOR ADDRESS
(1)	015350	001405			BEG	35		; IF NONE
(1)	015352	000005			RESET			; RESET AND
(1)		015354			SENDAD =			
(1)	015354	004710			JSR	7, (0)		; GO TO MONITOR
(1)	015356	000240			NOP			; SAVE ROOM
(1)	015360	000240			NOP			; FOR
(1)	015362	000240			NOP			; ACT11
(1)	015364	000137	002330	35:	JMP	RESTAR		; RETURN

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
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690

INITIALIZATION ROUTINE

DEVICE CSR REGISTER

ON FAILURE: REGISTER 0 CONTAINS ERROR ADDRESS

SET:
BIT01 = HALF DUPLEX
BIT02 = MAINTENANCE
BIT03 = CLEAR MOS
BIT06 = RECEIVER INTERRUPT ENABLE
BIT08 = MASTER XMTR SCAN ENB
BIT10 = R/W BCSR

WAIT FOR MOS TO CLEAR
SET:
BIT00 = RECEIVER ENABLE

```

INITA: MOV #2014, @CSR          SET
        BR   INTR              ;BIT(S)2,3,10 THEN 0
INITB: MOV #514, @CSR          ;BIT(S)2,3,6,8 THEN 0
        BR   INTR
INITC: MOV #416, @CSR          ;BIT(S)1,2,3,8 THEN 0
        BR   INTR
INITD: MOV #414, @CSR          ;BIT(S)2,3,8 THEN 0
INIR:  CLR  R0
IS:    INC  R0                  ;ANTI HANG
        BNE 2$                 ;ROUTINE
        MOV (SP), R0           ;RECORD SUBROUTINE CALL RETURN
        SUB #2, R0             ;FORM CALL ADDRESS FOR DISPLAY
HLT    ;BIT#4 OF DEVICE CSR FAILED TO CLEAR
2$:    BIT  #BIT4, @CSR         ;TEST HAS MOS CLEARED
        BNE 1$                 ;NO BRANCHES
        SET:
4686  015454 052777 000001 163636 BIS  #1, @CSR          ;BIT00 RECEIVE ENABLE
4687  015462 000207          RTS  PC                ;CONTINUE

```

```

4694
4695 015464 013701 001320 ALMCK: MOV CSR,R1 ;GET CSR ADDR INTO R1
4696 015470 012761 000001 000004 MOV #14(R1) ;SET LINE 0 IN THE TCR
4697 015476 052711 000004 BIS #BIT2,(R1) ;SET THE MAINT BIT
4698 015502 005037 001356 CLR COUNT
4699 015506 005037 001360 CLR SUM
4700 015512 005037 001352 CLR TIMERA
4701 015516 012737 000200 001354 2$: MOV #200,TIMERB ;SET UP TIME CONSTANTS
4702 015524 005711 TST (R1) ;WAIT FOR TRANSFER READY BIT
4703 015526 100373 BPL 2$
4704 015530 112761 000377 000006 MOVB #377,6(R1) ;OUTPUT A CHAR TO TBUF
4705 015536 005237 001356 INC COUNT ;COUNT EACH CHAR
4706 015542 105711 1$: TSTB (R1) ;CHECK FOR DONE IN THE CSR
4707 015544 100405 BMI 3$ ;IF SET GET OUT OF THE LOOP
4708 015546 004537 015676 JSR R5,TIME ;GIVE DONE TIME TO SET
4709 015552 000773 BR 1$ ;RETURN TO TEST FOR DONE AGAIN
4710 015554 000760 BR 2$ ;RETURN TO OUTPUT ANOTHER CHAR
4711 015556 000742 BR ALMCK ;ERROR RETURN TRY AGAIN
4712 015560 042711 000001 001356 3$: BIC #BIT0,(R1) ;TURN OFF RCV ENABLE
4713 015564 022737 000001 001356 CMP #1,COUNT ;IF SILO LEVEL SET FOR 1 THEN GET OUT
4714 015572 001414 BEQ 4$
4715 015574 063737 001356 001360 7$: ADD COUNT,SUM ;CONTINUE TO A COUNT OF 256
4716 015602 023727 001360 000377 CMP SUM,#377
4717 015610 002771 BLT 7$
4718 015612 001403 BEQ 10$ ;IF EQUAL USE IT
4719 015614 163737 001356 001360 SUB COUNT,SUM ;IF GREATER SUBTRACT 1 COUNT SO IT'S LESS
4720 015622 000403 BR 11$ ;ALL SET GET OUT
4721 015624 012737 000377 001360 4$: MOV #377,SUM ;PUT SUM TO MAX VALUE
4722 015632 004737 020054 11$: JSR PC,KBDINT ;GET THE SWITCH REGISTER
4723 015636 032777 010000 163530 BIT #SW12,JSWR ;PRINT ALARM LEVEL?
4724 015644 001413 BEQ 13$ ;NO
4725 015646 000004 017576 TYPE,MALARM
4726 015652 010105 MOV R1,TTY ;YES, PRINT CSR FIRST
4727 015654 004737 016662 JSR PC,PRINTR
4728 015660 000004 017516 TYPE,MSGDAS
4729 015664 013705 001356 MOV COUNT,TTY ;PRINT ALARM LEVEL
4730 015670 004737 016662 JSR PC,PRINTR
4731 015674 000207 13$: RTS PC
4732
4733
4734
4735 015676 105237 001352 TIME: INCB TIMERA ;INCREMENT THROUGH ONE WORD
4736 015702 001012 BNE 1$ ;GO TEST FOR DONE AGAIN
4737 015704 005337 001354 DEC TIMERB ;MAKE TIMERB LARGER IF FAST PROCESSOR
4738 015710 001007 BNE 1$
4739 015712 023727 001356 000022 CMP COUNT,#22 ;HAVE OUTPUTTED 18 TIMES
4740 015720 001002 BNE 2$ ;NO, GO OUTPUT ANOTHER CHAR
4741 015722 104001 HLT+1 ;YES,DONE DID NOT SET AFTER 18 OUTPUTS
4742 ;R1 = CSR
4743 015724 005725 TST (R5)+ ;SET R5 FOR ERROR RETURN
4744 015726 005725 2$: TST (R5)+ ;SET R5 FOR NEXT OUTPUT RETURN
4745 015730 000205 1$: RTS R5 ;RETURN FROM ABOVE OR RETEST DONE

```

K06

MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS
DZDJA.E.P11 08-SEP-77 15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-55
SCOPE LOOP HANDLER

SEQ 0075

4747

; \$SCOPE SCOPE LOOP HANDLER

(1) ; THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
(1) ; LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
(1)

(1) ; "SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
(1) ; RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"
(1)

(1)	015732	004737	020054	TRAPS:	JSR	PC,	KBDINT	
(1)	015736	032777	000400	163430	BIT	#SW8, 2SWR		; LOOP ON SPEC. TEST?
(1)	015744	001404			BEQ	1\$; NO LOOP ON SPEC. TEST
(1)	015746	127737	163422	001310	CMPB	2SWR, ICNT		; ON RIGHT TEST? *SW7-0*
(1)	015754	001434			BEQ	OVER\$; NOT RIGHT TEST
(1)	015756	032777	040000	163410	1\$: BIT	#SW14, 2SWR		; LOOP ON TEST?
(1)	015764	001026			BNE	KIT\$; LOOP ON TEST IS SET
(1)	015766	032777	004000	163400	BIT	#SW11, 2SWR		; KILL ITERATIONS
(1)	015774	001012			BNE	SVLAD\$; YES - KILL ITERATIONS
(1)	015776	105737	001311		TSTB	ICNT+1		; FIRST ONE?
(1)	016002	001404			BEQ	2\$; BRANCH IF FIRST
(1)	016004	123737	016072	001311	CMPB	TIMES, ICNT+1		; DONE?
(1)	016012	001013			BNE	KIT\$; BRANCH IF NOT
(1)	016014	112737	000001	001311	2\$: MOVB	#1, ICNT+1		; FIRST ITERATION
(1)	016022	105237	001310		SVLAD\$: INCB	ICNT		; COUNT TEST NUMBERS
(1)	016026	011637	016070		MOV	(6), LAD		; SAVE LOOP ADDRESS
(1)	016032	013737	001310	001376	MOV	ICNT, 2\$DISPLAY		; DISPLAY TEST NO. AND ITERATION COUNT
(1)	016040	000002			RTI			; RETURN
(1)	016042	105237	001311		KIT\$: INCB	ICNT+1		; INC THE ITERATION COUNT
(1)	016046	013737	001310	001376	OVER\$: MOV	ICNT, 2\$DISPLAY		; SET UP DISPLAY
(1)	016054	005737	016070		TST	LAD		; FIRST ONE?
(1)	016060	001760			BEQ	SVLAD\$; YES
(1)	016062	013716	016070		MOV	LAD, (6)		; FUDGE RETURN ADDRESS
(1)	016066	000002			RTI			; FIXES PS
(1)	016070	000000			LAD:	0		; LOOP ADDRESS
(1)	016072	000020			TIMES:	20		; RUN 20 TIMES

4749
4750

; SHLT ERROR TIMEOUT HANDLER

(1)
(1)
(1)
(1)
(1)
(1)
(1)

; THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
; ADDRESS OF THE "HLT". IT ALSO COUNTS THE NUMBER OF ERRORS
; AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
; "HALT" ON ERROR, AND INHIBIT TIMEOUTS. AN OPTIONAL ARGUMENT
; (HLT+3) WILL BE PLACED IN "HLTCTS:" FOR ADITIONAL TYPEOUTS.

(1) 016074 004737 020054
(1) 016100 032777 002000 163266
(1) 016106 001402
(1) 016110 000004 000007
(1) 016114 005237 001312
(1) 016120 032777 020000 163246
(1) 016126 001026
(2) 016130 000004 016134
(1) 016140 011637 016244
(1) 016144 162737 000002 016244
(1) 016152 117737 000066 016242
(2) 016160 013705 016244
(2) 016164 004737 016662
(2) 016170 000004 016174
(1) 016200 004737 016246
(1) 016204 005777 163164
(1) 016210 100001
(1) 016212 000000
(1) 016214 004737 020054
(1) 016220 032777 001000 163146
(1) 016226 001001
(1) 016230 000002
(1) 016232 105037 001311
(1) 016236 000137 016042
(1)
(1) 016242 000000
(1) 016244 000000

EMTS: JSR PC, KBDINT
BIT #SW10, ASWR
BEQ 1\$
TYPE BELL
1\$: INC ERRORS
BIT #SW13, ASWR
BNE 2\$
TYPE +2
MOV (6), HLTADR
SUB #2, HLTADR
MOVB #HLTADR, HLTCTS
MOV HLTADR, TTY
JSR PC, PRINTR
TYPE +2
2\$: JSR PC, ERRORS
TST ASWR
BPL +4
HALT
JSR PC, KBDINT
BIT #SW9, ASWR
BNE +4
RTI
CLRB ICNT+1
JMP KITS

; BELL ON ERROR?
; NO - SKIP
; RING BELL
; COUNT THE NUMBER OF ERRORS
; SKIP TYPEOUT IF SET
; SKIP TYPEOUTS
; ASCIZ (15)<(12)
; PUT ADDRESS OF INSTRUCTION ON STACK
; FUDGE ADDRESS
; GET HLT ARGUMENT
; TYPE HLTADR IN OCTAL
; TYPE LEADING ZERO'S
; ASCIZ " "
; GO TO USER ERROR ROUTINE
; HALT ON ERROR
; SKIP IF CONTINUE
; HALT ON ERROR!
; CHECK FOR INHIBIT LOOP ON ERROR
; SKIP IF LOOP ON ERROR
; RETURN
; CLEAR ITERATION COUNT
; LOOP ON TEST UNTIL NO ERRORS
; HLT ARGUMENT
; LAST HLT INSTRUCTION EXECUTED

4751
4752
4753
(1)
4754
4755
4756
4757
4758
4759
4760
4761
4762

ERRORS:
MOV CSR, TTY
JSR PC, PRINTR
BIC #7700, 2\$
1\$: DECB HLTCTS
BMI 3\$
ADD #100, 2\$
TYPE, SPACE
2\$: MOV %0, TTY
JSR %7, PRINTR
BR 1\$
3\$: RTS PC

; TYPE CSR IN OCTAL
; TYPE LEADING ZERO'S
; TYPE REGISTER X IN OCTAL

```

4767
4768 ;SUBROUTINE TO SAVE INPUT AS OCTAL NUMBER
4769
4770 016316 012737 000001 016610 READIN: MOV #1,INHRE
4771 016324 004737 016464 JSR PC, READS ;GO READ TTY UNTIL CR
4772 016330 005037 016610 CLR INHRE
4773 016334 010146 MOV R1,-(6) ;PUSH R1 ON STACK
(2) 016336 010246 MOV R2,-(6) ;PUSH R2 ON STACK
(2) 016340 010346 MOV R3,-(6) ;PUSH R3 ON STACK
4774 016342 012501 MOV (R5)+,R1
4775 016344 012737 000020 020264 MOV #20,CNT
4776 016352 012702 016612 MOV #INPUT,R2
4777 016356 122712 000120 CMPB #120,(R2) ;CHECK FOR "P"
4778 016362 001425 BEQ 3$
4779 016364 005011 CLR (R1)
4780 016366 112203 1$: MOVB (R2)+,R3
4781 016370 120327 000015 CMPB R3,#15
4782 016374 001420 BEQ 3$
4783 016376 162703 000060 SUB #60,R3
4784 016402 032703 177770 BIT #177770,R3
4785 016406 001013 BNE 3$ ;BRANCH IF BAD DATA
4786 016410 006311 ASL (R1)
4787 016412 103410 BCS 2$
4788 016414 006311 ASL (R1)
4789 016416 103406 BCS 2$
4790 016420 006311 ASL (R1)
4791 016422 103404 BCS 2$
4792 016424 050311 BIS R3,(R1)
4793 016426 005337 020264 DEC CNT
4794 016432 000755 BR 1$
4795 016434 000244 2$: CLZ ;MAKE SURE Z-BIT IS CLR
4796 016436 013737 177776 016462 3$: MOV @#PS, PSTEMP ;SAVE CONDITION CODES
4797 016444 012603 MOV (6)+,R3 ;POP STACK INTO R3
(2) 016446 012602 MOV (6)+,R2 ;POP STACK INTO R2
(2) 016450 012601 MOV (6)+,R1 ;POP STACK INTO R1
4798 016452 013737 016462 177776 MOV PSTEMP,@#PS ;RESTORE CONDITION CODES
4799 016460 000205 RTS R5
4800
4801 016462 000000 PSTEMP: 0 ;TEMPORARY STORAGE FOR PS
4802
4803 016464 010346 READS: MOV R3,-(6) ;SAVE R3
(1) 016466 012703 016612 1$: MOV #INPUT,R3 ;GET ADDRESS
(1) 016472 022703 016632 2$: CMPB #INPUT+20,R3 ;BUFFER FULL?
(1) 016476 001415 BEQ 4$ ;YES - TYPE "?"
(1) 016500 105737 177560 TSTB @#177560 ;WAIT FOR
(1) 016504 100375 BPL -4 ;A CHARACTER
(1) 016506 113713 177562 MOVB @#177562,(3) ;GET CHARACTER
(1) 016512 142713 000200 BICB #200,(3) ;GET RID OF JUNK
(1) 016516 122713 000177 CMPB #177,(3) ;IS IT A RUBOUT
(1) 016522 001403 BEQ 4$ ;SKIP IF NOT
(1) 016524 122713 000025 CMPB #25,(3)
(1) 016530 001006 BNE 3$
(1) 016532 4$:
(2) 016532 000004 016536 TYPE +2 ;ASCIZ "?*(15)(12)"= "
(1) 016544 000750 BR 1$ ;ZAP THE BUFFER AND LOOP
(1) 016546 111337 017334 3$: MOVB (3),.TYPE ;SET UP FOR TYPING

```

MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS
DZDJA.E.P11 08-SEP-77 15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-58
TTY INPUT ROUTINE

SEQ 0078

(1)	016552	000004	017334		;;PE	;;TYPE	;;ECHO IT
(1)	016556	122723	000015		CMPB	#15,(3)+	;;CHECK FOR RETURN
(1)	016562	001343			BNE	2\$;;LOOP IF NOT RETURN
(1)	016564	005737	016610		TST	INHRE	
(1)	016570	001401			BEQ	5\$	
(1)	016572	000402			BR	6\$	
(1)	016574	105063	177777	5\$:	CLRB	-1(3)	;;ZAP RETURN (THE 15)
(1)	016600	000004	000012	6\$:	TYPE	12	;;TYPE A LINE FEED
(1)	016604	012603			MOV	(6)+,R3	;;RESTORE R3
(1)	016606	000207			RTS	PC	;;RETURN
(1)							
(1)	016610	000000		INHRE:	0		
(1)	016612	000020		INPUT:	.BLKW 20		;;TTY INPUT AREA

4805

; SOCIAL OCTAL TYPEOUT ROUTINE

; THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
; ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, TYPE AN 18 BIT ADDRESS, OR TYPE
; THE 16 BITS. IT IS CALLED VIA THE DUMP, SDUMP, DUMP18, OR BITYPE MACRO'S.

(1)	016652	012737	170101	017020	BITYPS:	MOV	#170101, .PR	; SET BIT FLAG AMS 16. CHARACTER COUNT
(1)	016660	000411				BR	.PTIT	; NOW TYPE IT IN BIT FORM
(1)	016662	112737	000301	017020	PRINTR:	MOVB	#1, .PR	; SET ZERO FILL SWITCH
(1)	016670	000402				BR	+.6	; SKIP
(1)	016672	005037	017020		PRINTS:	CLR	.PR	; SUPPRESS LEADING ZERO'S
(1)	016676	112737	177772	017021		MOVB	#-6, .PR+1	; SET COUNT
(1)	016704	010446			.PTIT:	MOV	R4, -(6)	; SAVE R4
(1)	016706	012704	017022			MOV	#.PR+2, R4	; SET POINTER TO FIRST ASCII CHAR.
(1)	016712	105014				CLRB	(4)	; CLEAR FIRST BYTE
(1)	016714	000411				BR	.PRF	; ROTATE FIRST BIT
(1)	016716	105014			.PRL:	CLRB	(4)	; CLEAR BYTE OF CHARACTER
(1)	016720	032737	000100	017020		BIT	#100, .PR	; BIT TYPING MODE?
(1)	016726	001004				BNE	.PRF	; YES - SKIP 2 ROTATES
(1)	016730	006105				ROL	TTY	; ROTATE BIT INTO C
(1)	016732	106114				ROLB	(4)	; PACK IT
(1)	016734	006105				ROL	TTY	; ROTATE BIT INTO C
(1)	016736	106114				ROLB	(4)	; PACK IT
(1)	016740	006105			.PRF:	ROL	TTY	; ROTATE BIT INTO C
(1)	016742	106114				ROLB	(4)	; PACK IT
(1)	016744	105714				TSTB	(4)	; IS IT ZERO?
(1)	016746	001402				BEQ	+.6	; SKIP INC
(1)	016750	105237	017020			INCB	.PR	; SET FILL SWITCH
(1)	016754	105737	017020			TSTB	.PR	; CHECK FILL SWITCH
(1)	016760	001402				BEQ	+.6	; SKIP BITSET
(1)	016762	152724	000060			BISB	#'0, (4)+	; MAKE INTO ASCII CHAR
(1)	016766	105237	017021			INCB	.PR+1	; INC COUNT
(1)	016772	001351				BNE	.PRL	; REPEAT
(1)	016774	022704	017022			CMP	#.PR+2, R4	; EMPTY BUFFER?
(1)	017000	001002				BNE	+.6	; SKIP IF NOT
(1)	017002	112724	000060			MOVB	#'0, (4)+	; LOAD 1 ZERO
(1)	017006	105014				CLRB	(4)	; NULL TERMINATOR
(1)	017010	000004	017022			TYPE	.PR+2	; TYPE IT
(1)	017014	012604				MOV	(6)+, R4	; RESTORE R4
(1)	017016	000207				RTS	PC	; RETURN
(1)	017020	000012			.PR:	.BLKW	12	; COUNT, SWITCH, AND OUTPUT BUFFER


```

4807 017044 012777 017172 000126 PDOWN$ : MOV      #ILLUP, @PUVECS ; SET FOR FAST UP
(1) 017052 012777 000340 000122 : MOV      #340, @PUVECS+2 ; PRIO:7
(3) 017060 010046 : MOV      R0, -(6) ; PUSH R0 ON STACK
(3) 017062 010146 : MOV      R1, -(6) ; PUSH R1 ON STACK
(3) 017064 010246 : MOV      R2, -(6) ; PUSH R2 ON STACK
(3) 017066 010346 : MOV      R3, -(6) ; PUSH R3 ON STACK
(3) 017070 010446 : MOV      R4, -(6) ; PUSH R4 ON STACK
(3) 017072 010546 : MOV      R5, -(6) ; PUSH R5 ON STACK
(1) 017074 010637 017176 : MOV      SP, SAVR6 ; SAVE SP
(1) 017100 012777 017110 000072 : MOV      @PUPS, @PUVECS ; SET UP VECTOR
(1) 017106 000000 : HALT ; WAIT FOR PF

(1) 017110 013706 017176 PUPS: : MOV      .SAVR6, SP ; GET SP
(1) 017114 005001 : CLR      R1 ; WAIT LOOP FOR THE TTY
(1) 017116 005201 1$ : INC      R1 ; WAIT FOR THE INC
(1) 017120 001376 : BNE     1$ ; OF WORD
(3) 017122 012605 : MOV     (6)+, R5 ; POP STACK INTO R5
(3) 017124 012604 : MOV     (6)+, R4 ; POP STACK INTO R4
(3) 017126 012603 : MOV     (6)+, R3 ; POP STACK INTO R3
(3) 017130 012602 : MOV     (6)+, R2 ; POP STACK INTO R2
(3) 017132 012601 : MOV     (6)+, R1 ; POP STACK INTO R1
(3) 017134 012600 : MOV     (6)+, R0 ; POP STACK INTO R0
(1) 017136 012737 017044 000024 : MOV     @PDOWN$+2, @24 ; SET UP THE POWER DOWN VECTOR
(1) 017144 012737 000340 000026 : MOV     #340, @26 ; PRIO:7
(2) 017152 000004 017156 : TYPE   .+2 ; .ASCIZ (<15><12>"POWER"
(1) 017166 000137 002330 : JMP     RESTAR ; JMP TO USER ADDRESS

(1) 017172 000000 ILLUP: : HALT ; THE POWER UP SEQUENCE WAS STARTED
(1) 017174 000776 : BR     .-2 ; BEFORE THE POWER DOWN WAS COMPLETE

(1) 017176 000000 .SAVR6: 0 ; PUT THE SP HERE
(1) 017200 000024 000026 PUVECS: 24, 26 ; POWER UP VECTOR
    
```


E07

MAINDEC-11-DZDJA-E
DZDJA.E.P11 08-SEP-77

DJ11 LOGIC TESTS
15:06

MACY11 30(1046) 08-SEP-77 15:09 PAGE 53-62
TYPE ROUTINE

SEG 0082

	017410	051117	040440	042104
	017416	042522	051523	020072
	017424	000040		
4816	017426	005015	047516	020056
	017434	043117	042040	030512
	017442	223461	035123	020040
	017450	000		
4817	017451	015	051412	040524
	017456	042116	051101	020104
	017464	047503	043116	043511
	017472	051125	052101	047511
	017500	037516	020040	000
4818	017505	015	046012	047111
	017512	051505	000040	
4819	017516	026440	000040	
4820	017522	005015	044103	051101
	017530	046040	047105	052107
	017536	035110	020040	000
4821	017543	015	050012	051101
	017550	052111	024131	047516
	017556	020054	042117	026104
	017564	042440	042526	024516
	017572	020072	000040	
4822	017576	005015	044523	047514
	017604	040440	040514	046522
	017612	046040	053105	046105
	017620	043040	051117	041440
	017626	051123	005015	000
4823	017633	015	046412	044501
	017640	042116	041505	030455
	017646	026461	055104	045104
	017654	026501	020105	042040
	017662	030512	020061	047514
	017670	044507	020103	042524
	017676	052123	006523	000012

MSGNUM: .ASCIZ <15><12>"NO. OF DJ11'S: "

MSGCOX: .ASCIZ <15><12>"STANDARD CONFIGURATION? "

MSGLIN: .ASCIZ <15><12>"LINES "

MSGDAS: .ASCIZ " - "

MSGLEN: .ASCIZ <15><12>"CHAR LENGTH: "

MSGPAR: .ASCIZ <15><12>"PARITY(NO, OOD, EVEN): "

MALARM: .ASCIZ <15><12>"SILO ALARM LEVEL FOR CSR"<15><12>

MTITLE: .ASCIZ <15><12>"MAINDEC-11-DZDJA-E DJ11 LOGIC TESTS"<15><12>

4824				
4828				
4829				
4830				
4831				
4832				
4833				
4834				
4835	017704	012700	160000	
4836	017710	012702	000001	
4837	017714	012737	000002	000006
4838	017722	005001		
4839	017724	000261		
4840	017726	005710		
4841	017730	103404		
4842	017732	062700	000010	
4843	017736	005201		
4844	017740	000771		
4845				
4846	017742	005352		
4847	017744	000405		

.EVEN

;SUBROUTINE TO AUTOMATICLY DETERMINE THE NUMBER OF DJ11'S ON THE SYSTEM
AND WHERE THEIR INTERRUPT VECTORS ARE.

;THIS ROUTINE IS ONLY USED IF LOC 42 IS NOT ZERO, AS WHEN THE PROGRAM IS
BEING RUN UNDER ACT11 OR DDP MONITOR CONTROL.

;NOTE: SOME OF THE LOGIC MUST BE FUNCTIONAL OR THIS ROUTINE WILL BOMB!

AUTO: MOV #160000,R0 ;START AT NO RESPONSE BASE DJ11 -1018,
MOV #1,R2 ;COUNTER OF NON RESPONSE OR DJ11 OR DONE
MOV #RTI, #6 ;RTI WHEN TIME-OUT

55: CLR R1 ;SET UP COUNTER
15: SEC ;SET CARRY
TST (R0) ;CHECK FOR ANY DJ11'S
BCS 25 ;BRANCH IF IT TIMED OUT
ADD #10, R0 ;POINT TO NEXT DJ11 ADDRESS
INC R1 ;COUNT DJ11'S
BR 15 ;LOOK FOR MORE

25: DEC R2 ;COUNT DOWN DEVICES
BMI 75 ;BRANCH IF DONE

```

4848 017746 062700 000010      ADD    #10,    R0      ;POINT TO FIRST DJ11
4849 017752 010037 001340      MOV    R0,    DEVADR ;SAVE FIRST DJ11 ADDRESS
4850 017756 000761              BR     5$          ;GO COUNT DJ11'S
4851
4852 017760 005037 000006      7$:   CLR    2#6          ;RESTORE TIME-OUT CATCHER
4853 017764 010137 001344      MOV    R1,    UNITS  ;SAVE COUNT
4854 017770 001003              BNE   3$          ;BRANCH IF NOT ZERO
4855 017772 104000              HLT                   ;REPORT THAT NO DJ11'S WERE FOUND
4856 017774 000137 015302      JMP    DONE         ;EXIT THIS PROGRAM
4857
4858                               ;ROUTINE TO DETERMINE VECTOR ADDRESSES
4859
4860 020000 013746 000020      3$:   MOV    2#20, -(SP) ;SAVE IOT VECTOR ON THE STACK
4861 020004 012737 020040 000020      MOV    #45, 2#20    ;RESET IOT VECTOR
4862 020012 013701 001340      MOV    DEVADR, R1   ;GET FIRST DJ ADR
4863 020016 012721 040400      MOV    #40400, (R1)+ ;SET CSR
4864
4865                               ;BIT8 = TRANS SCAN ENABLE
4866 020022 005721              TST   (R1)+         ;BIT14 = TRANS INTERRUPT ENABLE
4867 020024 012721 000001      MOV    #1, (R1)+   ;INC POINTER
4868 020030 000001              WAIT                   ;SET TCR, LINE 0
4869 020032 012637 000020      MOV    (SP)+, 2#20 ;WAIT FOR AN INTERRUPT
4870 020036 000207              RTS    PC           ;RESTORE IOT VECTOR
4871
4872 020040 162716 000010      4$:   SUB    #10, (SP)   ;REPOSITION ADDRESS TO RVC VEC
4873 020044 011637 001342      MOV    (SP),  VECADR ;SAVE FIRST VECTOR
4874 020050 022626              CMP   (SP)+, (SP)+ ;RESET STACK FROM IOT
4875 020052 000002              RTI                   ;RETURN FROM INTERRUPT - RESTORE STATUS
4876
4877
(1) 020054 022737 000176 001374 KBDINT: CMP    #SWREG, SWR
(1) 020062 001016              BNE   1$
(1) 020064 005037 020122              CLR   TMP1
(1) 020070 113737 177562 020122      MOVB  177562, TMP1
(1) 020076 142737 000200 020122      BICB  #200, TMP1
(1) 020104 122737 000007 020122      CMPB  #7, TMP1
(1) 020112 001002              BNE   1$
(1) 020114 004737 020204              JSR   PC, CNTLU
(1) 020120 000207              1$:   RTS    PC
(1)
(1) 020122 000000      TMP1:  0
(1)
+878
(1) 020124 013746 000006      SUSWRR: MOV   6, -(SP)
(1) 020130 013746 000004      MOV   4, -(SP)
(1) 020134 012737 020154 000004      MOV   #15, 4
(1) 020142 022777 177777 161224      CMP   #-1, 2SWR
(1) 020150 001402              BEQ   2$
(1) 020152 000407              BR    3$
(1) 020154 022626              1$:   CMP   (SP)+, (SP)+
(1) 020156 012737 000176 001374      2$:   MOV   #SWREG, SWR
(1) 020164 012737 000174 001376      MOV   #DISPRG, DISPLAY
(1) 020172 012637 000004      3$:   MOV   (SP)+, 4
(1) 020176 012637 000006      MOV   (SP)+, 6
(1) 020202 000207              RTS   PC

```

4879

```

(1) 020204 022737 000176 001374 CNTLU: CMP #SWREG,SWR
(1) 020212 001023 BNE 1$
(1) 020214 000004 020276 TYPE SWREQ
(2) 020220 013705 000176 MOV SWREG,TTY ;TYPE SWREG IN OCTAL
(2) 020224 004737 016662 JSR PC,PRINTR ;TYPE LEADING ZERO'S
(1) 020230 000004 020266 TYPE NEWIS
(1) 020234 004537 016316 JSR RS,READIN
(1) 020240 020122 .WORD TMP1
(1) 020242 001360 BNE CNTLU
(1) 020244 022737 000020 020264 CMP #20,CNT
(1) 020252 001403 BEQ 1$
(1) 020254 013777 020122 161112 MOV TMP1,2SWR
(1) 020262 000207 1$: RTS PC
(1) 020264 000000 CNT: 0
(1) 020266 020040 042516 036527 NEWIS: .ASCIZ " NEW= "
(1) 020274 000040 SWREQ: .ASCIZ <15><12>"SWR= "
(1) 020276 005015
(1) 020304 000040
4980 000001 .END

```


N07

MAINDEC-11-DZDJA-E

DJ11 LOGIC TESTS

MACY11 30(1046)

08-SEP-77 15:09 PAGE 55-1

DZDJA.E.P11

08-SEP-77 15:06

CROSS REFERENCE TABLE -- MACRO NAMES

SEG 0091

\$SCOPE	654#	4747
\$SETUP	311#	
\$SRAT	191#	
\$SMDOC	103#	2571
\$SMROF	2497#	2679
\$SMRRR	2442#	4878
\$STRAP	978#	
\$TYPE	919#	4809
\$URAT	242#	
\$XR0Y	3300#	4577
\$XR0YC	3604#	4621
\$XR0YD	3253#	4576
\$XUOR	4211#	4627
.SCOP	374#	
.SCOPE	345#	

. ABS. 020306 000

ERRORS DETECTED: 0

DZDJA.E, DZDJA.E/CRF+DZDJA.E.MAC, DZDJA.E.P11

RUN-TIME: 9 13 .6 SECONDS

RUN-TIME RATIO: 253/24=10.3

CORE USED: 32K (63 PAGES)

B08