

.REM \_

IDENTIFICATION

PRODUCT CODE: AC-E911C-MC  
PRODUCT NAME: CXDZBCO DZV11 MODULE  
PRODUCT DATE: SEPTEMBER 1978  
MAINTAINER: DEC/X11 SUPPORT GROUP

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 MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITALS COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (C) 1976,1978 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT:

DZB IS AN IOMOD THAT EXERCISES UP TO EIGHT (CONSECUTIVELY ADDRESSED) DZV11 ASYNCHRONOUS INTERFACES. IT USES MAINTENANCE MODE TO TRANSMIT AND RECEIVE A BINARY COUNT PATTERN OUTPUTTED AND RECEIVED IN 32 CHARACTER BURSTS. THE MAJOR PORTION OF THE ERROR CHECKING IS DEFERRED TO PRIORITY LEVEL 0. ALL DEVICES SELECTED FOR TEST ARE ACTIVATED AND RUN CONCURRENTLY. ALL FOUR LINES ARE RUN ON EACH SELECTED DEVICE.

2. REQUIREMENTS:

HARDWARE: AT LEAST ONE DZV11 INTERFACE; NO WRAPAROUND OR MAINTENANCE CABLE IS NEEDED

STORAGE: DZB REQUIRES:

1. DECIMAL WORDS: 1021
2. OCTAL WORDS: 01775
3. OCTAL BYTES: 3772

3. PASS DEFINITION:

ONE PASS OF THE DZB MODULE CONSISTS OF TRANSMITTING AND RECEIVING 8960. CHARACTERS FOR EACH LINE OF EACH DZV11 SELECTED

4. EXECUTION TIME:

EXECUTION TIME IS PROPORTIONAL TO THE BAUD RATE BUT SHOULD TAKE AN AVERAGE OF ONE MINUTE TO COMPLETE ONE PASS WHEN RUNNING ALONE ON A LSI-11 AT 9600. BAUD.

5. CONFIGURATION PARAMETERS:

DEFAULT PARAMETERS:

DVA: 1, VCT: 1, BR1: 4, BR2: 4, DVC: 1, SR1: 0

REQUIRED PARAMETERS:

AT CONFIGURATION TIME THE USER MUST SPECIFY:

DVA: ADDRESS OF FIRST DZV11 CSR REG.  
VCT: VECTOR ADDRESS OF FIRST DZV11  
DVC: NO OF DZV11'S IF GREATER THAN 1

6. DEVICE OPTION SETUP:

NONE REQUIRED

7. MODULE OPERATION:

START: DETERMINE IF ANY DEVICES ARE SELECTED. DO NOT RUN THE MODULE IF NO DEVICES ARE SELECTED. IF THERE ARE SELECTED DEVICES, INITIALIZE THE BINARY COUNT PATTERN AT 0. CONTINUE PROCESSING.

RESTR1: INITIALIZE THE ITERATION COUNTER TO 1120. DETERMINE IF ANY DZV11'S ARE SELECTED. LOAD THE INTERRUPT VECTORS TO POINT TO THE JSR LINKING TABLE.

SETUP2: INITIALIZE THE QUEUE POINTERS. CLEAR ALL THE BUFFERS AND QUEUES. CLEAR THE BUFFER ACCESS FLAG (LCKOUT) IN CASE IT WAS STILL SET BY A CONTROL C INTERRUPT OF THE PROGRAM.

ACTVATE: THIS SEGMENT INITIALIZES EACH DZV11 SELECTED. EIGHT BITS PER CHARACTER IS SELECTED. IF A BAUD RATE IS SELECTED, IT IS CALCULATED AND ASSIGNED. OTHERWISE THE DEFAULT RATE OF 9600 BAUD IS ASSUMED.

INITIAL: THE DATA PATTERN IS LOADED INTO THE TRANSMITTER BUFFER. IT IS A BINARY COUNT PATTERN WHICH, ON SUCCESSIVE ITERATIONS, BEGINS 0,10,20,....,177760,177770. THE NUMBER OF CHARACTERS TO BE TRANSMITTED IS CALCULATED. TRANSMITTER INTERRUPTS ARE ENABLED. ALL SELECTED TRANSMITTERS FOR EACH SELECTED DZV11 ARE ENABLED.

TMRSET: TMRCNT IS USED AS A MULTIPLYING FACTOR TO DETERMINE THE WAITING LENGTH FOR THE WATCHDOG TIMER. IT IS PRESENTLY SET AT 5 TO ALLOW SEVENTY-FIVE SECONDS TO ELAPSE BEFORE TAKING FURTHER ACTION.

TIMER: THIS IS THE WATCHDOG TIMER LOOP. IT IS CONTROLLED BY R4 AND TMRCNT. IF ALL DZV11'S SELECTED GENERATED BOTH TRANSMIT AND RECEIVE INTERRUPTS, THE APPROPRIATE BIT IN DONFLG FOR THAT DZV11 WILL BE CLEARED. IF THIS DOES NOT OCCUR IN THE GIVEN TIME, THE DEVICE NUMBER OF THE OFFENDING DZV11 WILL BE CALCULATED, AND THIS WILL BE REPORTED IN A MODULE MESSAGE. THE OFFENDING DEVICE IS DROPPED FROM THE EXERCISE. IF NO MORE DZV11'S ARE SELECTED, THE MODULE ITSELF IS DROPPED FROM THE RUN. IF MORE REMAIN TO BE EXERCISED, HOWEVER, CONTROL IS TRANSFERRED TO "FINISH."

- FINISH: CONTROL COMES HERE IF ALL SELECTED DZV11'S WERE SUCCESSFULLY EXERCISED OR IF MORE DZV11'S REMAIN AFTER ONE WAS HUNG. THE ITERATION COUNT IS DECREASED. IF THE COUNT DOES NOT REACH ZERO, CONTROL IS PASSED TO SETUP2 AND THE MODULE IS RUN AGAIN. WHEN THE COUNT REACHES ZERO, AN END OF PASS IS SIGNALLED.
- XMTINT: THIS SEGMENT SAVES A POINTER TO THE CSR ADDRESS OF THE INTERRUPTING DZV11 IN A FIRST-IN, FIRST-OUT, WRAPAROUND BUFFER, THE TRANSMITTER QUEUE. THE ENTRY POINTER IS THEN UPDATED TO POINT TO THE NEXT ENTRY IN THE QUEUE. SERVICE IS DEFERRED TO LEVEL ZERO PRIORITY.
- XMTSRV: THIS BLOCK FETCHES A POINTER TO A CSR ADDRESS FROM THE TRANSMITTER QUEUE, AND THE QUEUE IS UPDATED TO THE NEXT ENTRY. THE CSR IS TESTED TO DETERMINE WHAT KIND OF INTERRUPT OCCURRED. FALSE INTERRUPT IS REPORTED. IF EVERYTHING IS CORRECT, THE ADDRESS OF THE BYTE ASSOCIATED WITH THIS LINE (IN THE TRANSMITTER BUFFER) IS CALCULATED. A CHARACTER IS TRANSMITTED ON THIS LINE. TRANSMITTER INTERRUPTS ARE REENABLED FOR THIS DEVICE, IF MORE CHARACTERS ARE TO BE TRANSMITTED. IF NO MORE ARE TO BE SENT ON THIS LINE, A BIT POINTER IS BUILT AND THE TRANSMITTER IS DISABLED. IF ALL LINES ARE DISABLED, THE RECEIVER FOR THIS DZV11 IS ENABLED.
- RCVINT: THIS SEGMENT SAVES A POINTER TO THE CSR ADDRESS OF THE INTERRUPTING DZV11 IN THE RECEIVER QUEUE. IT DISABLES FURTHER INTERRUPTS FOR THIS DEVICE. IT UPDATES THE QUEUE ENTRY AND RESTORES THE VALUE OF R5 WHICH WAS SAVED BY THE JSR INSTRUCTION FROM THE LINKAGE TABLE.
- RCVSRV: THE FIRST TASK IS TO PREVENT VOLATILE REGISTER INFORMATION FROM BEING DESTROYED. THIS IS DONE BY TESTING A SEMAPHORE, "LCKOUT." IF IT IS SET, CONTROL IS RETURNED TO THE MONITOR TO WAIT FOR A WHILE. IF IT IS CLEAR, ACCESS IS PERMITTED. THE FLAG IS SET TO DENY OTHER ACCESSES TO THIS DEFERRED ROUTINE. A CSR ADDRESS IS OBTAINED FROM THE QUEUE, AND THE QUEUE ENTRY IS UPDATED. THE RECEIVER INTERRUPT AND INTERRUPT ENABLE ARE CLEARED. THE REGISTERS ARE SET UP TO RETRIEVE AS QUICKLY AS POSSIBLE THE DATA FROM THE DZV11 SILO. EACH FETCH IS CHECKED TO SEE IF THE INFORMATION IS VALID. IF IT IS NOT, THE REGISTERS ARE SAVED AND A BREAK LOOP IS USED TO ALLOW MORE TIME FOR VALID INFORMATION TO BECOME AVAILABLE. IF AFTER THE ALLOTTED TIME ALL THE CHARACTERS ARE STILL NOT RECEIVED, AN ERROR MESSAGE IS REPORTED. IN THE MESSAGE, TWO NUMBERS ARE GIVEN. THE FIRST IS THE NUMBER OF CHARACTERS THAT WERE TRANSMITTED (IN OCTAL). THE NEXT IS THE NUMBER OF CHARACTERS THAT WERE NOT RECEIVED (ALSO IN OCTAL).

CKDATA: THIS SEGMENT INITIALIZES THE LINE CHECK BUFFER (LNCKBF) TO THE FIRST DATUM THAT WAS TRANSMITTED. THE DEVICE NUMBER IS SAVED FOR LATER USAGE. THE RECEIVED INFORMATION IS CHECKED FOR VALIDITY AND TRANSMISSION ERRORS. ERRORS ARE HANDLED BY THE "STATERR" (STATUS ERROR) AND "DERROR" (DATA ERROR) ROUTINES.

RCVDONE: THIS BLOCK CLEARS THE ACCESS SEMAPHORE TO ALLOW OTHER DEVICES TO USE THE LINE CHECK BUFFER. IT ALSO DISABLES THE DEVICE WITH A DEVICE CLEAR. IT THEN BUILDS A ONE BIT MASK USING RO AND THE CARRY BIT TO DELETE THE APPROPRIATE BIT IN THE WATCHDOG TIMER FLAG (DONFLG). WHEN THIS IS DONE, PROCESSING CONTROL IS RETURNED TO THE MONITOR.

#### SUBROUTINES

VCTLOAD: THIS ROUTINE IS CALLED IN "SETUP1". IT IS USED TO LOAD THE ADDRESS OF THE LINKING INSTRUCTION FOR INTERRUPT SERVICING INTO THE CORRESPONDING VECTOR SPACE. IT ALSO LOADS THE PRIORITY LEVEL AND THE DEVICE ADDRESS. THE LATTER IS LOADED INTO THE APPROPRIATE JSR TABLE ENTRY.

SAVREG: THIS ROUTINE SAVES THE FIVE VOLATILE INFORMATION REGISTERS IN A FIRST-IN, FIRST-OUT WRAPAROUND BUFFER, THE ERROR QUEUE.

GETREG: THIS ROUTINE RETRIEVES THOSE SAME REGISTERS.

GETLINE: THIS ROUTINE CALCULATES HOW MANY CHARACTERS WILL BE TRANSMITTED FOR EACH DEVICE DURING AN ITERATION OF THE PROGRAM. EIGHT CHARACTERS ARE SENT ON EACH SELECTED LINE IN ONE ITERATION. THE TOTAL COUNT IS STORED IN "XMTCNT".

BAUDRTE: THIS ROUTINE CALCULATES THE BAUD RATE, ASSIGNS IT, AND SELECTS 8 BITS/CHARACTER COMMUNICATION MODE. IF SR1=0, THE DEFAULT RATE OF 9600. BAUD IS ASSIGNED. THE BAUD RATE SELECTED IS DETERMINED BY THE LEAST SIGNIFICANT (RIGHTMOST) SET BIT IN SR1.

STATERR: THIS ROUTINE DETERMINES WHETHER AN ERROR INDICATED IN THE RECEIVED CHARACTER INFORMATION WAS AN OVERRUN ERROR, A FRAMING ERROR, OR A PARITY ERROR. THE DEVICE NUMBER OF THE ERRING DEVICE IS REPORTED AS STATC. CSRA WILL BE CLEAR.

DERROR: THIS ROUTINE REPORTS A DATA ERROR.

#### 8. OPERATOR OPTIONS

MODULE LOCATION DVID1 (APC=14) MAY BE MODIFIED (MOD. CMMD) TO EXERCISE ANY COMBINATION OF EIGHT DZV11'S.

MODULE LOCATION SR1 (APC=16) MAY BE MODIFIED TO SELECT A DIFFERENT BAUD RATE. THE FOLLOWING TABLE SHOULD BE USED:

| FOR THE BAUD RATE | SR1=  | LOC 276= |
|-------------------|-------|----------|
| 7200              | 1     | 2060     |
| 4800              | 2     | 2000     |
| 3600              | 4     | 1730     |
| 2400              | 10    | 1460     |
| 2200              | 20    | 1350     |
| 1800              | 40    | 1350     |
| 1200              | 100   | 1200     |
| 600               | 200   | 630      |
| 300               | 400   | 330      |
| 150               | 1000  | 150      |
| 134.5             | 2000  | 144      |
| 110               | 4000  | 120      |
| 75                | 10000 | 70       |
| 50                | 20000 | 45       |

(USING THESE VALUES WILL YIELD AN END OF PASS CLOSE TO ONE MINUTE FO EACH BAUD RATE.)

THE DEFAULT RATE IS 9600 BAUD(SR1=0).

MODULE LOCATION SLCTLIN MAY BE MODIFIED TO RUN ANY COMBINATION OF FOUR LINES. THE COMBINATION IS THEN RUN ON ALL SELECTED DEVICES. THE DEFAULT SELECTION IS ALL FOUR LINES.

NOTE: SLCTLIN FALLS ON A BYTE BOUNDARY!! BE SURE TO RESTORE THE BITS SET IN THE OTHER BYTE !!!

MODULE LOCATION ICONT MAY BE MODIFIED TO VARY THE PERIOD BETWEEN END OF PASS REPORTS.

MODULE LOCATION TMRSET +2 MAY BE MODIFIED TO VARY THE PERIOD OF THE WATCHDOG TIMER. IT IS PRESENTLY SET TO EXPIRE AFTER SEVENTY-FIVE SECONDS WHEN DZB IS RUNNING ALONE.

9. NON-STANDARD PRINTOUTS:

WHEN A STATUS ERROR IS DETECTED, DZB USES THE ERRORN CALL TO REPORT IT. THE FIRST NUMBER GIVEN IS THE NUMBER OF THE DEVICE (0 TO 7). THE SECOND IS THE CONTENTS OF THE READ BUFFER (DZRDBUF= CSR + 2, E.G., 160042).

WHEN ALL CHARACTERS ARE NOT RECEIVED, AN ERRORN CALL IS REPORTED. THE FIRST NUMBER IS THE NUMBER OF CHARACTERS EXPECTED. THE SECOND IS THE NUMBER OF CHARACTERS THAT WERE NOT RECEIVED.

ALL OTHER PRINTOUT IS STANDARD.

10. MNEMONICS  
-----

THE FOLLOWING INFORMATION SHOULD BE USEFUL IN UNDERSTANDING  
NAMES GIVEN TO VARIABLES IN THIS PROGRAM.

XMT REFERS TO THE TRANSMITTER  
RCV REFERS TO THE RECEIVER  
ERR REFERS TO ANYTHING TO DO WITH ERROR HANDLING  
FLG REFERS TO A SOFTWARE FLAG, USUALLY A BIT FLAG  
QUE REFERS TO A FIRST IN, FIRST OUT BUFFER  
TMR REFERS TO SOFTWARE TIMING FUNCTIONS  
CNT REFERS TO A WORD USED AS A COUNTER  
QP REFERS TO A POINTER ASSOCIATED WITH A QUEUE BUFFER  
I IS AN INSERTION POINTER, O IS AN OUTPUT POINTER  
LN REFERS TO SOMETHING INVOLVING A GIVEN LINE  
XM IS ANOTHER REFERENCE TO TRANSMITTER  
CT GENERALLY REFERS TO A COUNT

E.G: XMTQPO=TRANSMITTER QUEUE POINTER OUT

OTHERS ARE BASICALLY SELF EXPLANATORY.

```

;*****LIST BIN*****
;*****
000000 055104 041502 040 BEGIN:
000005 000000 MODNAM: .ASCII /DZBC / ;MODULE NAME.
000006 000001 XFLAG: .BYTE OPEN ;USED TO KEEP TRACK OF WBUF USAGE
000010 000001 ADDR: 1+0 ;1ST DEV
000012 000001 VECTOR: 1+0 ;1ST DEVICE VECTOR.
000013 200 BR1: .BYTE PRTY4+0 ;1ST BR LEVEL.
000014 000001 BR2: .BYTE PRTY4+0 ;2ND BR LEVEL.
000016 000000 SV1: +1 ;DEVICE INDICATOR 1.
000020 000000 SR1: OPEN ;SWITCH REGISTER 1
000022 000000 SR2: OPEN ;SWITCH REGISTER 2
000024 000000 SR3: OPEN ;SWITCH REGISTER 3
000026 140000 SR4: OPEN ;SWITCH REGISTER 4
;*****
000030 000314 STAT: 140000 ;STATUS WORD.
000032 000224 INIT: START ;MODULE START ADDR.
000034 000000 SPOINT: MODSP ;MODULE STACK POINTER.
000036 002140 PASCNT: 0 ;PASS COUNTER.
000040 000000 ICONT: 2140 ;# OF ITERATIONS PER PASS=2140
000042 000000 ICONF: 0 ;LOC TO COUNT ITERATIONS
000044 000000 SDFCNT: 0 ;LOC TO SAVE TOTAL SOFT ERRORS
000046 000000 HRDCNT: 0 ;LOC TO SAVE TOTAL HARD ERRORS
000050 000000 SOPPAS: 0 ;LOC TO SAVE SOFT ERRORS PER PASS
000052 000000 HRDPAS: 0 ;LOC TO SAVE HARD ERRORS PER PASS
000054 000000 SYSCNT: 0 ;# OF SYS ERRORS ACCUMULATED
000056 000000 RANDUM: 0 ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000058 000000 CONFIG: 0
000060 000000 RES1: 0 ;RESERVED FOR MONITOR USE
000062 000000 RES2: 0 ;RESERVED FOR MONITOR USE
000064 000000 SVR0: OPEN ;LOC TO SAVE R0.
000066 000000 SVR1: OPEN ;LOC TO SAVE R1.
000070 000000 SVR2: OPEN ;LOC TO SAVE R2.
000072 000000 SVR3: OPEN ;LOC TO SAVE R3.
000074 000000 SVR4: OPEN ;LOC TO SAVE R4.
000076 000000 SVR5: OPEN ;LOC TO SAVE R5.
000078 000000 SVR6: OPEN ;LOC TO SAVE R6.
00100 000000 CSRA: OPEN ;ADDR OF CURRENT CSR.
00102 000000 CSRB: OPEN ;ADDR OF GOOD DATA, OR
00104 000000 ACSR: OPEN ;CONTENTS OF CSR.
00106 000000 WASADR: OPEN ;ADDR OF BAD DATA, OR
00108 000000 ASAT: OPEN ;STATUS REG CONTENTS.
00110 000000 ERRTYP: 0 ;TYPE OF ERROR
00112 000000 ASB: OPEN ;EXPECTED DATA.
00114 000000 AWAS: OPEN ;ACTUAL DATA.
00116 000000 RSTRT: RSTRT ;RESTART ADDRESS AFTER END OF PASS
00118 000000 WDR: OPEN ;WORDS TO MEMORY PER ITERATION
00120 000000 WDRP: OPEN ;WORDS FROM MEMORY PER ITERATION
00122 000142 IDNUM: 142 ;# OF INTERRUPTS PER ITERATION
000224 000000 MODSP: 142 ;MODULE IDENTIFICATION NUMBER=142
;*****
390 000002 ;*****RBUF=2***** ;DEFINITION OF MODE 6 OFFSET TO THE READ BUFFER

```

```

391 000002 LPR=2 ;MODE 6 OFFSET TO THE LINE PARAMETER REGISTER
392 000004 TCR=4 ;OFFSET TO THE TRANSMITTER CONTROL REGISTER
393 000006 MSR=6 ;OFFSET TO THE MODEM STATUS REGISTER
394 000008 TDR=6 ;OFFSET TO THE TRANSMITTER DATA REGISTER
395 000010 MAINT=BIT3 ;ENABLE MAINTENANCE MODE BIT
396 000020 DCLR=BIT4 ;DEVICE MASTER CLEAR
397 000040 MSENAB=BIT5 ;ENABLE THE MASTER SCANNER (DEVICE GO)
398 000100 RIE=BIT6 ;RECEIVER INTERRUPT ENABLE
399 010000 SILOEN=BIT12 ;SILO ALARM INTERRUPT ENABLE
400 040000 TIE=BIT14 ;TRANSMITTER INTERRUPT ENABLE
401 010000 RCVON=BIT12 ;TURN THE RECEIVER ON (RECEIVER GO)
402 000030 EIGHT=BIT4IBIT3 ;EIGHT BITS/CHARACTER SELECTION

```



```
403 ;THESE ARE THE PROGRAM PARAMETERS
404
405
406 000224* 000000 LINPAR: OPEN ;SCRATCHWORD USED TO LOAD LINE PARAMETER REGISTERS
407 000226* 000000 THRCNT: OPEN ;COUNTER FOR WATCHDOG TIMER
408 000230* 000000 RCVTMR: OPEN ;RECEIVER BREAK LOOP TIMER
409 000232* 000000 DVCNMR: OPEN ;NUMBER OF DEVICE BEING PROCESSED
410 000234* 000000 XMTCNT: OPEN ;COUNTER OF TOTAL NUMBER OF TRANSMISSIONS
411 000236* 000040 LIXMCT: BLKB 32. ;TRANSMISSION COUNTERS FOR EACH DEVICE
412 000276* 000000 BGNDDATA: OPEN ;TWO COPIES OF THE FIRST CHARACTER IN TRANSMIT PATTERN
413 000300* 000000 RCVWORD: OPEN ;USED TO REPORT RECEIVER ERROR
414 000302* 000000 MISS: OPEN ;USED TO REPORT NUMBER OF CHARACTERS MISSING
415 000304* 000 DATA: BYTE OPEN ;CHARACTER BUILDING BYTE
416 000306* 000 SELECT: .BYTE OPEN ;ACTIVE DEVICE SELECTION PARAMETER
417 000308* 017 SLCYLN: .BYTE 17 ;ACTIVE LINE SELECTION PARAMETER
418 000310* 000 DDFLNG: .BYTE OPEN ;WATCHDOG FLAG FOR BUSY DEVICES
419 000312* 000 RCVDATA: .BYTE OPEN ;BUFFER FOR CHARACTER CHECKING
420 000314* 000 LCKOUT: .BYTE OPEN ;BUFFER ACCESS FLAG
421 000316* 000000 TEMP1: .WORD 0 ;TEMPORARY STORAGE FOR ASCII CONVERSION
422 .EVEN
```

```
423 ;START ROUTINE. DETERMINE IF ANY DZV'S ARE SELECTED
424 ;IF SO, BEGIN MODULE PROCESSING... IF NOT, DROP THE MODULE FROM THE RUN
425
426 000314* 105067 177764 START: CLR B DATA ;INITIALIZE THE DATA PATTERN WORD
427 000320* 116767 177470 177757 MOV B DVID1,SELECT ;COPY THE DEVICE SELECTION PARAMETER. ARE
428 ;ANY DEVICES SELECTED?
429 000326* 001002 BNE RSTRT ;IF SO, BEGIN PROCESSING. IF NOT, DROP THE MODULE
430 000330* 104410 000000* DROP: ENDS,BEGIN ;DROP THE MODULE
431
432 ;INITIALIZE THE NUMBER OF PASSES TO BE MADE. SET UP THE DZV11 INTERRUPT
433 ;VECTORS TO INTERRUPT IN THE SUBROUTINE LINKING TABLE. CALL SUBROUTINE
434 ;VCTLOAD FOR THIS PURPOSE.
435
436 000334* 116701 177745 RSTRT: MOV B SELECT,R1 ;COPY THE DEVICE SELECTION PARAMETER INTO R1
437 000340* 001773 BEQ DROP ;IF NO DEVICES SELECTED (ALL FLAGS CLEAR) DROP THE
438 ;MODULE
439 000342* 016700 177442 MOV VECTOR,R0 ;LOAD THE VECTOR ADDRESS IN R0
440 000344* 016702 177434 MOV ADDR,R2 ;LOAD R2 WITH ADDRESS OF FIRST DZ11
441 000350* 012703 002324* MOV #LNKTAB,R3 ;POINT R3 TO THE BEGINNING OF JSR LINKING TABLE
442 000352* 000241 CLC ;MAKE SURE CARRY BIT IS CLEAR BEFORE ROTATION
443 000354* 106001 RORB R1 ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
444 000356* 103410 BCS SETUP2 ;IF THE FLAG IS SET, GO SET UP THE VECTORS
445 000358* 001416 BEQ #10,R0 ;IF NO FLAGS ARE LEFT, GO SET UP THE BUFFERS
446 000360* 062700 ADD ;IF MORE FLAGS ARE SET, ADJUST POINTERS. THE
447 ;VECTOR POINTER IS THE POINTER...
448 000372* 062703 000020 ADD #20,R3 ;AND THE ADDRESS POINTER...
449 000374* 062702 000010 ADD #10,R2 ;GO SET UP THE NEXT DZV11 ADDRESSES
450 000402* 000766 BR 1S ;CALL THE VECTOR SETUP ROUTINE FOR RECEIVER
451 000404* 004567 001344 JSR R5,VCTLOAD ;VECTOR, PASSING THE RECEIVER BR LEVEL AS THE
452 000410* 000013 BR2 ;ARGUMENT
453 000412* 004567 001336 JSR R5,VCTLOAD ;SET UP THE TRANSMITTER VECTOR PASSING THE
454 000416* 000012 BR1 ;TRANSMITTER BR LEVEL AS THE ARGUMENT
455 000420* 000766 BR 2S ;GO SETUP THE NEXT DZV11
456
457 ;THIS BLOCK RESETS ALL THE QUEUE POINTERS, CLEARS THE TRANSMITTER TEXT BUFFER, THE
458 ;RECEIVER BUFFERS, AND THE QUEUES. THIS IS THE BEGINNING OF THE ITERATIVE PART OF THE
459 ;PROGRAM.
460
461 000422* 012767 002524* 003262 SETUP2: MOV #XMTQUE,XMTQPI ;POINT THE TRANSMIT QUEUE ENTRY (IN) POINTER
462 000430* 012767 002524* 003256 MOV #XMTQUE,XMTQPO ;TO THE BEGINNING OF THE QUEUE
463 000432* 012767 002544* 003246 MOV #RCVQUE,RCVQPI ;POINT THE TRANSMIT QUEUE RETRIEVAL (OUT)
464 000434* 012767 002544* 003246 MOV #RCVQUE,RCVQPO ;POINTER TO THE BEGINNING OF THE QUEUE
465 000436* 012767 002564* 003236 MOV #ERRQUE,ERRQPI ;SET UP THE RECEIVER QUEUE POINTERS
466 000438* 012703 002524* 003236 MOV #ERRQUE,ERRQPO ;SETUP THE ERROR QUEUE POINTERS
467 000440* 012704 MOV #XMTQUE,R3 ;POINT R3 TO THE BEGINNING OF THE QUEUE AREA
468 000442* 005023 CLR (R3)+,R4 ;USING R4 AS A COUNTER CLEAR
469 000444* 001375 DEC R4 ;TRANSMITTER, RECEIVERS, AND ERROR QUEUES....
470 000446* 001375 BNE 1S ;TRANSMITTER TEXT BUFFER...
471 000448* 005304 CLR R5 ;RECEIVER SILO BUFFERS
472 000450* 001375 BNE 1S ;CLEAR THE BUFFER ACCESS FLAG
473 000452* 105067 177691 CLR LCKOUT ;CLEAR THE CHARACTER COUNTER FOR EACH LINE
474 000454* 00236* MOV #LNXMCT,R3 ;POINT TO THE CHARACTER COUNTER FOR EACH LINE
475
476
477
478
```

```

479 000514 012704 000040
480 000524 011773 000010
481 000524 005304
482 000526 001374
483
484
485
486
487 000530 116700 177551
488 000534 117773 177447
489 000540 016701 177242
490 000544 106000
491 000546 103404
492 000550 01426
493 000552 062701 000010
494 000556 000772 000020
495 000560 012711 000004
496 000564 012702 000004
497 000570 012767 010030 177426
498 000576 004767 001324
499 000602 052711 000010
500 000606 016701 177412 000002
501 000614 005302
502 000616 001755
503 000620 005267 177400
504 000624 000770
505
506
507
508
509 000626 012701 002646
510 000632 116700 177446
511 000636 110067 177434
512 000642 110067 177431
513
514 000646 012702 000040
515 000652 110021
516 000654 005302
517 000656 001375
518 000660 062700 000010
519 000664 110067 177414
520 000670 116700 177411
521 000674 016701 177106
522 000700 004767 001162
523 000704 00221
524 000706 106000
525 000710 103404
526 000712 001411
527 000714 002701 000010
528 000720 00772
529 000722 052711 040040
530 000726 116701 177354 000004
531 000734 000767
532

```

```

533
534
535
536
537
538
539 000736 012767 000005 177262
540 000744 005004
541 000746
542 000746 104407 000000
543 000752 104407 000000
544 000756 105767 177325
545
546 000762 001437
547 000764 005304
548 000766 001367 177232
549 000770 005367
550 000774 001363
551
552 000776 116703 177305
553
554 001002 140367 177277
555 001006 006003
556
557 001010 103402
558
559 001012 005204
560
561 001014 000774
562 001016 010467 177270
563
564
565
566 001022 104420 000000 000312
567 001030 003756
568
569 001032 004767 000734
570 001036 104403 000000 003742
571 001044 004767 000760
572 001050 105767 177231
573 001054 001002
574
575 001056 104410 000000
576
577
578
579 001062
580 001062 104413 000000
581
582 001066 000167 177436
583
584
585
586
587
588

```

```
589 001072* 010577 002614 XMTINT: MOV R5,@XMTQPI ;LOAD THE OFFSET TO THE CSR INTO TRANSMITTER QUEUE
590 001076* 062767 000000 ADD R2,XMTQPI ;UPDATE THE QUEUE RETRIEVAL POINTER
591 001102* 022767 002542* 002606 CME #XMTQUE+16,XMTQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
592 001102* 022767 002542* 002606 BHS IS ;IF NO, CONTINUE PROCESSING
594 001114* 012767 002524* 002570 MOV #XMTQUE,XMTQPI ;IF YES, RESET THE POINTER TO QUEUE BEGINNING
595 001122* 012605 (SP),R5 ;RESTORE THE PREVIOUS R5 VALUE
596 1$: MOV (SP),R5 ;RESTORE THE PREVIOUS VALUE OF R5
597 -----
598 ;PIRQS,BEGIN,XMTRSV ; QUEUE UP TO CONTINUE AT XMTRSV AND RTI
599 -----
599 001132* 017701 002556 XMTRSV: MOV @XMTQD,R1 ;FETCH THE OFFSET FROM THE QUEUE
600 001134* 052767 000000 ADD R2,XMTQD ;UPDATE THE QUEUE RETRIEVAL POINTER
601 001144* 052767 002542* 002542 CME #XMTQUE+16,XMTQD ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
602 001152* 103003 BHS IS ;IF NO, CONTINUE PROCESSING
603 001154* 012767 002524* 002532 MOV #XMTQUE,XMTQD ;IF YES, RESET POINTER TO BEGINNING OF THE QUEUE
604 001162* 011100 (R1),R0 ;LOAD CSR ADDRESS INTO R0
605 001164* 005710 TST (R0) ;IS THIS A VALID INTERRUPT?
606 001166* 100420 BMI ZS ;IF YES, GO ENABLE RECEIVER INTERRUPT
607 001170* 010067 176704 MOV (R0),CSRA ;IF NO, SET UP THE CSR ADDRESS FOR ERROR REPORTING
608 001174* 011067 176702 MOV (R0),ACSR ;SET UP THE CONTENTS
609 001200* 004767 000566 JSR PC,SAVREG ;SAVE THE REGISTERS BEFORE ERROR MESSAGE
610 001204* 012767 000011 176674 MOV #1,ERRTYP
611 *****
611 001212* 104405 000000* 000000 HDRDR,REGTNULL ;IF INTERRUPT REQUEST FROM TRANSMITTER
612 *****
613 -----
614 001220* 004767 000604 JSR PC,GETREG ;RESTORE THE REGISTERS
615 001224* 104400 000000* EXITS,BEGIN ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
616 2$: MOV BIC #1,R0,R2 ;GET THE LINE NUMBER FROM THE HIGH BYTE OF CSR
617 001234* 042767 000001* BIC #7774,R2 ;ISOLATE THE LINE NUMBER IN R2
618 001240* 016103 MOV #2,R1,R3 ;GET THE NUMBER OF THIS DEVICE
619 001244* 006303 ASL R3 ;MULTIPLY THE DEVICE NUMBER BY FOUR
620 001250* 060203 ADD R2,R3 ;CALCULATE THIS LINE TABLE OFFSET
621 001252* 042710 040000 BIC #1,R0 ;DISABLE INTERRUPTS BEFORE AN OVERRUN OCCURS
622 001256* 105763 000236* TSTB LNXMCT(R3) ;ALL CHARAC. TRANSMITTED?
623 001264* 112763 000010 000236* MOV BIC #58,LNXMCT(R3) ;IF SO, RESET THE COUNT OF CHARACTERS
624 001272* 012703 000001* MOV #1,R3 ;SET A BIT POINTER IN R3
625 001276* 005302 DEC R2 ;USE R2 AS COUNT OF ROTATIONS TO MAKE
626 001280* 006303 ASL R2 ;THE RIGHT INTERFERE AT THE RIGHT LINE?
627 001302* 006303 ASL R3 ;IF NOT, POINT IT TO THE NEXT LINE
628 001304* 000774 BR 3S ;GO SEE IF THIS IS THE LINE
629 001306* 040360 BIC R3,TCR(R0) ;IF SO, DISABLE TRANSMISSIONS ON THIS LINE
630 001314* 052710 000100 BIC #1,R0 ;IF TRANSMITTER IS ACTIVE, GO EXIT
631 001320* 000411 BR 7S ;OTHERWISE, ENABLE THE RECEIVER
632 001322* 116360 MOVB XMTBUF(R3),TDR(R0) ;SKIP RESETTING THE INTERRUPT ENABLE
633 001334* 105263 000236* YENB LNXMCT(R3) ;LOAD THE CHAR INTO TRANS BUFFER
634 001340* 052710 040000* BR 7S ;CREATE A CHARACTER
635 001342* 104400 000000* BIC #1,R0 ;COUNT A CHARAC. FOR THIS LINE
636 001344* 104400 000000* BIC #1,R0 ;RESTART TRANS. INTERRUPTS
637 -----
638 7$: EXITS,BEGIN ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
639 -----
640
```

```
641 ;RECEIVER INTERRUPT SERVICE ROUTINE
642 ;THIS ROUTINE STORES A POINTER TO THE CSR ADDRESS OF THE INTERRUPTING DEVICE IN THE
643 ;RECEIVER QUEUE. SERVICE IS DEFERRED TO PRIORITY LEVEL 0. AT DEFERRED SERVICE (RCVSRV),
644 ;THE OFFSET IS FETCHED FROM THE QUEUE. THE INTERRUPT AND INTERRUPT ENABLE BITS ARE
645 ;CLEARED AND THE SILO IS EMPTIED INTO THE SOFTWARE BUFFER. IF ALL CHARACTERS
646 ;WERE NOT RECEIVED THIS IS REPORTED. CHECK THE BUFFER AND REPORT DATA ERRORS. IF
647 ;ANOTHER RECEIVER IS ALREADY USING THE LINE CHECK BUFFER (LNCB), SUBSEQUENT
648 ;RECEIVERS WILL NOT BE PERMITTED ACCESS UNTIL THE FIRST IS COMPLETE. IF THERE
649 ;ARE ANY CHARACTER ERRORS REPORT THEM. WHEN ALL DATA IS CHECKED, IT RELEASES
650 ;THE LINE CHECK BUFFER AND CLEAR THE CORRESPONDING BIT IN DONFLC.
651 -----
652 653 001350* 010577 002342 RCVINT: MOV R5,RCVQPI ;STORE THE OFFSET IN THE RETRIEVAL QUEUE
654 001354* 042735 000100 BIC #R1E,@(R5)+ ;DISABLE THIS RECEIVER'S INTERRUPTS BEFORE
655 ;ITS OTHER LINES OVERRUN THE QUEUES
656 001360* 062767 000002 002330 ADD #2,RCVQPI ;UPDATE THE QUEUE ENTRY POINTER
657 001366* 022767 002562* 002322 CME #RCVQUE+16,RCVQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
658 001374* 103003 BHS IS ;IF NO, CONTINUE PROCESSING
659 001376* 012767 002544* 002312 MOV #RCVQUE,RCVQPI ;IF YES, RESET THE POINTER TO QUEUE BEGINNING
660 001404* 012605 (SP),R5 ;RESTORE THE PREVIOUS VALUE OF R5
661 -----
662 ;PIRQS,BEGIN,RCVSRV ; QUEUE UP TO CONTINUE AT RCVSRV AND RTI
663 -----
663 001414* 105767 176671 RCVSRV: TSTB LCKOUT ;TEST THE BUFFER LOCK FLAG. IS ACCESS PERMITTED?
664 001420* 001405 BEQ 11S ;IF YES, GO SET THE LOCK AND CHECK THE DATA
665 001422* 104407 000000* BREAKS,BEGIN ;TEMPORARY RETURN TO MONITOR.
666 001426* 104407 000000* BREAKS,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
667 001432* 000770 BR RCVSRV ;TRY TO ACCESS AGAIN
668 001434* 105167 176651 11S: COMB LCKOUT ;DENY OTHER ACCESSES TO THE BUFFER
669 001440* 017700 002254 MOV @RCVQD,R0 ;FETCH THE OFFSET FROM THE QUEUE
670 001444* 052767 000002 002246 ADD #2,RCVQD ;UPDATE THE QUEUE RETRIEVAL POINTER
671 001452* 022767 002562* 002240 CME #RCVQUE+16,RCVQD ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
672 001460* 103003 BHS IS ;IF NO, CONTINUE PROCESSING
673 001462* 012767 002544* 002230 MOV #RCVQUE,RCVQD ;IF YES, RESET THE POINTER TO THE QUEUE BEGINNING
674 001470* 012901 (R0),R1 ;LOAD THE CSR ADDRESS INTO R1
675 001472* 042721 030300 BIC #30300,(R1)+ ;DISABLE RECEIVER INTERRUPT AND INTERRUPT ENABLE
676 -----
677 678 001476* 011002 MOV (R0),R2 ;LOAD THE SOFTWARE SILO BUFFER ADDRESS
679 001500* 012767 002000 176522 MOV #2000,RCVTMR ;SET THE RECEIVER BREAK LOOP TIMER
680 001506* 016703 176522 MOV #MTCNT,R3 ;SET THE COUNT FOR STORING THE DATA
681 001512* 011122 MOV (R1),(R2)+ ;STORE A WORD IN THE SOFTWARE SILO
682 001514* 002003 BGE 3S ;IF THE DATA WASN'T VALID, GO ALLOW A LITTLE TIME FOR IT
683 001516* 005303 DEC R2 ;IF IT WAS VALID, REDUCE THE COUNT.
684 001520* 005303 BNE ZS ;IF NOT ZERO, GO STORE THE NEXT WORD
685 001522* 000434 BR CKDATA ;IF ALL CHARACTERS RECEIVED, GO CHECK THEM
686 001524* 005742 TST (R2) ;RESET R2 TO THE PROPER BUFFER LOCATION
687 001526* 04767 000240 JSR PC,SAVREG ;SAVE THE REGISTER VALUES
688 001536* 104407 000000* BREAKS,BEGIN ;TEMPORARY RETURN TO MONITOR.
689 001536* 104407 000000* BREAKS,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
690 001542* 004767 000262 JSR PC,GETREG ;RESTORE THE REGISTER VALUES
691 001546* 005367 176456 DEC RCTMR ;HAS ENOUGH BEEN PERMITTED FOR A CHARACTER?
692 001552* 100357 ZS ;IF NO, TRY TO GET ONE
693 001554* 014067 176320 MOV -(R0),CSRA ;LOAD THE DEVICE ADDRESS
694 001560* 013067 176316 MOV @R0,ACSR ;LOAD THE CSR CONTENTS
695 001574* 016367 076512* MOV #MIS,CSRA ;REPORT THE NUMBER OF CHARACTERS MISSING
696 001570* 004767 000176 JSR PC,SAVREG ;SAVE THE REGISTERS
```

```

DZBCC.P11 12-OCT-78 11:58
697 001574 012767 000014 176304 MOV #14,ERRTP
698 *****
699 001602 104405 000000 003726 4RDERS,BEGIN,TAB1 ;DID NOT RECEIVE ALL TRANSMITTED CHARACTERS
700 *****
701 001610 004767 000214 JSR PC,GETREG
702
703 ;THIS ROUTINE PERMITS ONE DEVICE TO HAVE ITS DATA CHECKED.
704 ;THE DATA IS CHECKED
705 ;FOR EACH LINE OF THE DEVICE. DATA ERRORS ARE REPORTED.
706
707 001614 012705 003706 CKDATA: MOV #LNCKBF,R5 ;POINT TO THE BEGINNING OF THE BUFFER
708 001620 016725 176452 MOV BGNDA,(R5)+ ;SET UP FOUR LINE CHECK BYTES
709 001624 016725 176446 MOV BGNDA,(R5)+
710 001630 011007 MOV (R0),R2 ;LOAD THE SOFTWARE SILO ADDRESS IN R2
711 001632 016067 MOV 10(R0),DVCNMBR ;SAVE THE NUMBER OF THIS DEVICE
712 001640 016703 176370 MOV XMTCNT,R3 ;LOAD COUNT TO COMPARE CHARACTERS
713 001644 012200 3$: MOV (R2),R0 ;FETCH A WORD FROM THE SILO INTO R0
714 001646 002024 BGE RCVDONE ;IF IT'S INVALID, GO CLEAN UP BUFFERS
715 001650 032700 BIT #7000,R0 ;ARE THERE ANY TRANSMISSION ERRORS?
716 001654 001402 BEQ 4$ ;IF NO, GO DETERMINE THE LINE NUMBER
717 001656 004767 000322 JSR PC,STATERR ;IF YES, GO DETERMINE THE ERROR TYPE
718 001662 110067 176422 4$: MOV R0,RCVDATA ;SAVE THE RECEIVED CHARACTER
719 001666 000261 SHLB ;PUT THE LINE NUMBER IN R0'S LOWER BYTE
720 001670 042700 177774 BIC #177774,R0 ;ISOLATE THE LINE NUMBER IN R0
721 001674 126067 003706 176406 CMPB LNCKBF(R0),RCVDATA ;IS THE DATA CORRECT?
722 001702 001407 BEQ 5$ ;IF YES, GO CHECK THE NEXT CHARACTER
723 001704 003767 000336 JSR PC,ERRROR ;IF NO, GO REPORT A DATA ERROR
724 001710 105260 003706 5$: INCB LNCKBF(R0) ;SET UP THE NEXT CHARACTER FOR THIS LINE
725 001714 005303 DEC R3 ;REDUCE THE COUNT, ARE ALL CHARACTERS CHECKED?
726 001716 001352 BNE 3$ ;IF NO, GO CHECK THE NEXT CHARACTER.
727
728 ;THIS ROUTINE UNLOCKS THE LINE CHECK BUFFER TO PERMIT ACCESS BY OTHER DEVICES.
729 ;IT THEN COMPUTES THE LINE NUMBER AND CLEARS THE APPROPRIATE FLAG FROM THE WATCHDOG
730 ;TIMER BYTE.
731
732 001720 105067 176365 RCVDONE:CLRB LCKOUT ;RESET THE BUFFER ACCESS SEMAPHORE
733 001724 012741 000020 MOV #CLR,-(R1) ;CLEAR THIS DEVICE AND DISABLE IT
734 001730 005000 CLR R0 ;THE DELETION FLAG WILL PROPAGATE IN R0
735 001734 000261 SFC ;USE THE CARRY BIT TO BUILD THE DELETION FLAG
736 001736 006100 176270 1$: ROL R0 ;MOVE THE FLAG TO THE NEXT DEVICE
737 001738 005367 DEC DVCNMBR ;WHEN THIS NUMBER IS NEGATIVE, THE CORRECT
738 001742 002374 BGE RCVDONE ;DEVICE IS POINTED TO.
739 001744 176337 R1CB R0,DONFLG ;CLEAR THIS FLAG
740 001750 104400 000000 EXIT$,BEGIN ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
741

```

```

DZBCC.P11 12-OCT-78 11:58
742 ;SUBROUTINES
743 ;-----
744
745 ;THE FIRST IS THE VECTOR LOADING SUBROUTINE. THE VECTOR ADDRESS IS PASSED IN R0,
746 ;THE DEVICE ADDRESS IS PASSED IN R2. THE LINKING TABLE ADDRESS IS PASSED IN R3.
747 ;THE BUS REQUEST PRIORITY LEVEL IS A PARAMETER TAKEN INDIRECTLY THROUGH R5.
748
749 001754 010320 VCTLOAD:MOV R3,(R0)+ ;LOAD THE ADDRESS OF THE LINKING INSTRUCTION
750 001756 113520 MOVB @R5+,(R0)+ ;AND THE PRIORITY LEVEL INTO THE VECTOR
751 001760 005200 INC R0 ;POINT R0 TO THE NEXT VECTOR ADDRESS
752 001762 023233 CMP (R3),(R3)+ ;POINT R3 TO THE CSR INSERT LOCATION
753 001764 010223 MOV R2,(R3)+ ;LOAD THE DEVICE ADDRESS IN THE LINK TABLE
754 001766 005723 TST (R3)+ ;ALIGN R3 FOR THE NEXT DEVICE
755 001770 000205 RTS R5 ;RETURN TO CALLING BLOCK
756
757 ;THIS ROUTINE SAVES THE REGISTER VALUES IN THE ERROR QUEUE, A FIRST-IN,
758 ;FIRST-OUT DISCIPLINE WRAPAROUND BUFFER.
759
760 001772 016705 001724 SAVREG: MOV ERRQPI,R5 ;USE R5 FOR INDEXING CAPABILITY
761 001776 010025 MOV R0,(R5)+ ;SAVE R0...
762 002000 010125 MOV R1,(R5)+ ;SAVE R1...
763 002004 010352 MOV R2,(R5)+ ;SAVE R2...
764 002008 010352 MOV R3,(R5)+ ;SAVE R3...
765 002006 010425 MOV R4,(R5)+ ;AND R4
766 002010 022705 002644 CMP #ERRQUE+60,R5 ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
767 002014 103007 BHIS 1$ ;IF NO, GO RELOAD THE POINTER
768 002016 012705 MOV #ERRQUE,R5 ;RESET R5 TO THE QUEUE BEGINNING
769 002022 010567 001674 1$: MOV R5,ERRQPI ;SET THE ENTRY POINTER TO NEXT ENTRY POINT
770 002026 000207 RTS PC ;RETURN TO THE CALLING BLOCK
771
772 ;THIS ROUTINE RETRIEVES REGISTER VALUES FROM THE ERROR QUEUE
773
774 002030 016705 001670 GETREG: MOV ERRQPI,R5 ;USE R5 FOR INDEXING CAPABILITY
775 002034 012560 MOV (R5),R0 ;RETRIEVE R0...
776 002036 012501 MOV (R5),R1 ; R1...
777 002040 012502 MOV (R5),R2 ; R2...
778 002042 012503 MOV (R5),R3 ; R3...
779 002044 012504 MOV (R5),R4 ; AND R4
780 002046 022705 002644 CMP #ERRQUE+60,R5 ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
781 002052 103007 BHIS 1$ ;IF NO, GO RELOAD THE POINTER
782 002054 012705 MOV #ERRQUE,R5 ;RESET R5 TO THE QUEUE BEGINNING
783 002060 010567 001640 1$: MOV R5,ERRQPI ;SET THE RETRIEVAL POINTER TO THE NEXT FETCH POINT.
784 002064 000207 RTS PC ;RETURN TO THE CALLING BLOCK
785
786 ;THIS ROUTINE CALCULATES HOW MANY CHARACTERS ARE BEING TRANSMITTED IN EACH PATTERN
787
788 002066 116702 176214 GETLINE:MOVB SLCTLIN,R2 ;COPY THE LINES SELECTED PARAMETER
789 002072 005067 CLR XMTCNT ;RESET THE TOTAL TRANSMISSION COUNT
790 002076 106907 176130 1$: ROBB R2 ;GET A SELECTION BIT INTO THE CARRY BIT
791 002100 005567 ADC XMTCNT ;ADD IT TO TOTAL NUMBER OF ACTIVE LINES
792 002104 105702 R2 TSTB R2 ;ARE 4 LINES CHECKED?
793 002106 001373 BNE 1$ ;IF NOT, GO CHECK THE REST
794 002110 006367 ASL XMTCNT ;NOW MULTIPLY BY 8
795 002114 006367 ASL XMTCNT ;DO THIS BY SHIFTING THREE TIMES
796 002120 006367 ASL XMTCNT ;WE NOW KNOW HOW MANY CHARACTERS ARE IN EACH BURST
797 002124 000207 RTS PC ;RETURN TO THE CALLING SEGMENT

```

```

798
799 ;THIS ROUTINE CALCULATES THE BAUD RATE FROM THE SRI OPTION. IF SRI IS 0, THE DEFAULT
800 ;RATE OF 9600. IS ASSIGNED. THE LEAST SIGNIFICANT BIT IS THE ONLY ONE CONSIDERED.
801 ;THE BAUD RATE IS THEN LOADED INTO THE LINE PARAMETER WORD.
802
803 002126 005767 175664 BAUDRATE:TST SRI ;IS A SPECIFIC RATE SELECTED?
804 002132 001414 BEQ 35 ;IF NO, GO ASSIGN THE DEFAULT RATE
805 002134 015704 MOV SRI,R4 ;IF YES, COPY SRI
806 002140 012703 MOV #5,R3 ;SET UP THE BIT CONFIGURATION FOR A BAUD RATE SELECTION
807 002144 006004 ROR R4 ;ISOLATE THE NEXT BIT IN THE CARRY BIT
808 002146 103403 BCS 25 ;IF THIS IS THE BIT PUT TOGETHER THE BAUD RATE
809 002150 005303 DEC R3 ;IF NOT, CALCULATE THE NEXT BIT CONFIGURATION
810 002152 100413 BMI 45 ;IF NONE OF THE BITS WERE SET, RETURN TO THE CALLING PRO
811 002154 000773 BR 15 ;GO CHECK THE NEXT ALTERNATIVE
812 002156 150367 BLSB R3,LINPAR+1 ;PLACE THE BAUD RATE IN THE LINE PARAMETER REGISTER
813 002162 000207 RTS PC ;RETURN TO THE CALLING PROCEDURE
814 002164 052767 007000 176032 3S: RIS #7000,LINPAR ;SET THE DEFAULT RATE(9600. BAUD)
815 002172 104407 000000 BREAKS,BEGIN ;TEMPORARY RETURN TO MONITOR.
816 002176 104407 000000 BREAKS,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
817 002202 000207 RTS PC ;RETURN TO CALLING PROCEDURE
818
819 ;THIS ROUTINE DETERMINES WHAT TYPE OF ERROR WAS INDICATED BY THE SILO AND REPORTS EACH
820
821 002204 010067 176070 STATERR:MOV R0,RCVWORD ;COPY THE INFORMATION RECEIVED
822 002210 011167 MOV (R1),CSRA ;GET THE CONTENTS OF THE CONTROL REGISTER
823 002214 010167 MOV R1,ACSR ;REPORT THE CSR CONTENTS
824 002220 004767 JSR PC,SAVREG ;SAVE THE REGISTERS
825 002224 012767 MOV #7,CRTVP ;*****
826
827 002232 104405 000000 003734* HDRERS,BEGIN,TAB2 ;DEVICE RECEIVER ERROR
828
829 002240 004767 JSR PC,GETREG ;RESTORE THE REGISTERS
830 002244 000207 RTS ;RETURN TO CALLING PROCEDURE
831
832 ;THIS ROUTINE CALCULATES THE BUFFER ADDRESSES OF THE INCORRECT DATA AND REPORTS THE ERRO
833
834 002246 004767 DERROR: JSR PC,SAVREG ;SAVE THE PRESENT REGISTER VALUES
835 002252 005741 TST -(R1) ;POINT TO THE CSR ADDRESS AGAIN
836 002256 010167 MOV R1,CSRA ;SHOW WHICH DEVICE IT WAS
837 002260 116767 MOVB LNCKB(R0),ASB ;LOAD THE CORRECT DATA VALUE
838 002266 116767 MOVB RCVDATA,AWAS ;LOAD THE ACTUAL DATA VALUE
839 002274 005742 TST -(R2) ;GET THE SILO ADDRESS OF THE DATA
840 002278 010267 MOV R2,WASADR ;LOAD IT FOR REPORTING
841 002302 062700 ADD HLCKB,R0 ;CALCULATE THE CORRECT VALUE ADDRESS
842 002306 010067 MOV R0,SBADR ;REPORT IT
843
844 002312 104404 000000* DATERS,BEGIN ;DATA ERROR!!!
845
846 002316 004767 JSR PC,GETREG ;RESTORE THE REGISTERS
847 002322 000207 RTS ;RETURN TO THE CALLING PROCEDURE
848
849 ;LINKAGE TABLE
850 ;EACH ENTRY CONSISTS OF A JSR INSTRUCTION FOLLOWED BY A LOCATION INTO WHICH THE CSR IS
851 ;LOADED, FOLLOWED BY THE SILO BUFFER ADDRESS, FOLLOWED BY A JSR FOR THE TRANSMITTER
852 ;INTERRUPT, THE CSR ADDRESS WORD, AND THE NUMBER OF THE INTERRUPTING DEVICE
853

```

```

854 002324 004567 177020 LNKTAB: JSR R5,RCVINT ;LINK FOR RECEIVER 0
855 002330 000000 0 SILO0
856 002334 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 0
857 002340 000000 0
858 002342 000000 0
859 002344 004567 177000 JSR R5,RCVINT ;LINK FOR RECEIVER 1
860 002346 000000 0 SILO1
861 002352 003006 176512 JSR R5,XMTINT ;LINK FOR TRANSMITTER 1
862 002354 004567 0
863 002356 000000 1 SILO2
864 002360 004567 JSR R5,RCVINT ;LINK FOR RECEIVER 2
865 002364 000000 0 SILO2
866 002370 004567 176760 JSR R5,XMTINT ;LINK FOR TRANSMITTER 2
867 002374 000000 0
868 002376 003109 176472 JSR R5,RCVINT ;LINK FOR RECEIVER 3
869 002382 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 3
870 002400 000000 0 SILO3
871 002402 000002 176740 JSR R5,RCVINT ;LINK FOR RECEIVER 4
872 002404 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 4
873 002410 000000 0 SILO4
874 002412 003206 176452 JSR R5,RCVINT ;LINK FOR RECEIVER 5
875 002414 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 5
876 002420 000000 0 SILO5
877 002422 000003 176720 JSR R5,RCVINT ;LINK FOR RECEIVER 6
878 002424 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 6
879 002430 000000 0 SILO6
880 002432 003306 176432 JSR R5,RCVINT ;LINK FOR RECEIVER 7
881 002434 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 7
882 002440 000000 0 SILO7
883 002442 000004 176700 JSR R5,RCVINT ;LINK FOR RECEIVER 8
884 002444 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 8
885 002450 000000 0
886 002452 003406 176412 JSR R5,RCVINT ;LINK FOR RECEIVER 9
887 002454 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 9
888 002460 000000 5
889 002462 000005 176660 JSR R5,RCVINT ;LINK FOR RECEIVER 10
890 002464 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 10
891 002470 000000 0 SILO6
892 002472 003506 176372 JSR R5,RCVINT ;LINK FOR RECEIVER 11
893 002474 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 11
894 002500 000000 0 SILO7
895 002502 000006 176640 JSR R5,RCVINT ;LINK FOR RECEIVER 12
896 002504 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 12
897 002510 000000 0
898 002512 003606 176352 JSR R5,RCVINT ;LINK FOR RECEIVER 13
899 002514 004567 JSR R5,XMTINT ;LINK FOR TRANSMITTER 13
900 002520 000000 7
901 002522 000007 ;THESE ARE THE QUEUES.
902
903 XMTQUE: -BLKW 8. ;TRANSMITTER SERVICE QUEUE
904 RCVQUE: -BLKW 8. ;RECEIVER SERVICE QUEUE
905 ERRQUE: -BLKW 25. ;ERROR SERVICE QUEUE
906
907 ;THESE ARE THE BUFFERS
908
909

```

```

910
911 002646* 000040 XMTBUF: .BLKB 32. ;TRANSMITTER BUFFERS, ONE CHARACTER FOR EACH LINE
912 002706* 000040 SILO0: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 0
913 003006* 000040 SILO1: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 1
914 003106* 000040 SILO2: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 2
915 003206* 000040 SILO3: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 3
916 003306* 000040 SILO4: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 4
917 003406* 000040 SILO5: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 5
918 003506* 000040 SILO6: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 6
919 003606* 000040 SILO7: .BLKW 32. ;RECEIVER BUFFER FOR DEVICE 7
920 003706* 000040 LNCKBF: .BLKB 4. ;BUFFER FOR CHECKING INDIVIDUAL LINE DATA
921
922 ;THESE ARE THE QUEUE POINTERS
923
924 XMTQPI: OPEN ;TRANSMITTER QUEUE ENTRY (IN) POINTER
925 XMTQPO: OPEN ;TRANSMITTER QUEUE RETRIEVAL POINTER (OUT)
926 RCVQPI: OPEN ;RECEIVER QUEUE POINTERS
927 RCVQPO: OPEN
928 ERROPI: OPEN ;ERROR QUE POINTERS
929 ERROPO: OPEN
930
931 ;THESE ARE THE ASCII MESSAGES AND ERROR PRINTOUT TABLES
932
933 003726* 000234* TAB1: XMTCNT ;TABLE 1 IS USED TO REPORT XMTCNTING CHAPACTERS
934 ;FIRST, THE NUMBER TRANSMITTED IS GIVEN
935 003730* 000302* MISS ;THEN THE NUMBER OF CHARACTERS NOT RECEIVED
936 003734* 177777 TAB2: DVCNMBR ;TERMINATOR FOR TABLE 1
937 003736* 000233* RCVWORD ;TABLE 2 IS USED TO REPORT RECEIVER ERRORS
938 003736* 000300* ;WORD CONTAINING THE ERRORS
939 003740* 177777 HUNG: M1 ;TERMINATOR FOR TABLE 2
940 003742* 003746* M1 177777
941 003744* 177777
942
943 003746* 042504 044526 042503 M1: .ASCII "DEVICE "
944 003752* 026040 M2: .BLKB 6
945 003754* 000006 M3: .ASCIZ " HUNG"
946 003764* 044040 ;.EVEN
947 ;.END
948 000001

```

```

ACSR 000102R 377# 608* 694* 823*
ACTVAT 000530R 487# 582
ADDR 000106R 343# 442 489 521
ADDR22= 001006 398#
ASB 000106R 381# 837*
ASTAT 000104R 379#
AWAS 000106R 383# 838#
BADDR 002126R 492# 803#
BEGIN 000000R 340# 431 542 543 566 570 575 580 597 612 615 640 662
        666# 667 688 699 709
        511* 512* 708
BCNDAT 000276R 412#
BIT0 = 000001 396#
BIT1 = 000002 390#
BIT10 = 002000 390#
BIT11 = 000001 390#
BIT12 = 010000 399 401
BIT13 = 020000 390#
BIT14 = 040000 390# 400
BIT15 = 100000 390#
BIT2 = 000004 390#
BIT3 = 000010 395 402
BIT4 = 000020 390# 402
BIT5 = 000040 390# 397
BIT6 = 000100 390# 398
BIT7 = 000200 390#
BIT8 = 000400 390#
BIT9 = 001000 390#
BREAKS= 104407 390# 542 543 666 667 688 689 815 816
BR1 000012R 345# 457
BR2 000013R 346# 454
BTODS = 104411 390#
CDATA = 104412 390#
CKDATA 001614R 685 707#
CONFIG 000056R 362#
CSRA 000100R 373# 607* 693* 822* 836*
DATA 000304R 415# 426* 510 519*
DATCK = 104411 390#
DATERS = 104402 390# 844
DCR 000020 396# 495 733
DERRDR 002246R 723# 834#
DONFLG 000307R 418# 544 552 739*
DRDP 000330R 430# 435
DVCNMB 000330R 409# 711* 737* 937
DVID1 = 000014R 347# 427
EIGHT = 000030 402# 497
ENDIT = 104413 390# 580
ENWS 000000 431
ERRQPI 003722R 476# 575
ERRQPO 003724R 471# 760 769* 928#
ERRQUE 002564R 470 471 774 783* 929#
ERRRYP 000106R 380# 610* 766 768 780 782 907#
EXITS = 104400 390# 615 640 740
FINISH 001062R 545 573 579#
GETLIN = 002066R 362# 788#
GETTAB = 104410 360#
GETREC 002030R 571 614 690 701 774# 829 846

```



|        |           |      |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
|--------|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|--|--|--|--|--|--|--|
| TIMER  | 000744R   | 540# | 550  |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| IMRCNT | 000226R   | 407# | 539* | 549* |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| IMRSET | 000736R   | 526# | 539# |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| TRPDFD | 000022    | 390# |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| VCTLOA | 001754R   | 453# | 456  | 749# |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| VECTOR | 000010R   | 344# | 441  |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| WASADR | 000104R   | 378# | 840* |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| WDFR   | 000116R   | 384# |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| WDTD   | 000114R   | 384# |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| XFLAG  | 000005R   | 342# |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| XMTBUF | 002546R   | 506# | 635  | 636* | 911# |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| XMTCNT | 000234R   | 410# | 680  | 712  | 789# |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| XMTINT | 001072R   | 590# | 857  | 863  | 869  | 791* | 794* | 795* | 796* | 933  |      |      |      |      |  |  |  |  |  |  |  |
| XMTQPI | 003712R   | 464# | 590* | 591* | 592  | 875  | 881  | 887  | 893  | 899  |      |      |      |      |  |  |  |  |  |  |  |
| XMTQPD | 003714R   | 466# | 599  | 600* | 601  | 594* | 594# |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| XMTQUE | 002524R   | 464  | 466  | 472  | 592  | 603* | 601  | 603  | 905# |      |      |      |      |      |  |  |  |  |  |  |  |
| XMTSRV | 001132R   | 597  | 599# |      |      | 601  | 601  |      |      |      |      |      |      |      |  |  |  |  |  |  |  |
| .      | = 003772R | 411# | 905# | 906# | 907# | 911# | 912# | 913# | 914# | 915# | 916# | 917# | 918# | 919# |  |  |  |  |  |  |  |
|        |           | 920# | 945# |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |

. ABS. 000000 000  
003772 001

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

XDZBCO, XDZBCO/SOL/CRF:SYN=DDXCOM, XDZBCO  
RUN-TIME: 23.4 SECONDS  
RUN-TIME RATIO: 19/5=3.2  
CORE USED: 7K (13 PAGES)