EK-KXTCA-HR-001

KXT11-CA ROM Listing

Prepared by Educational Services
of
Digital Equipment Corporation

First Edition, June 1984

The information in this software is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

Digital assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital.

CONTENTS

INTRODUCTION

This is a general overview of the KXTNT KXT11-CA-macro V05.00 native firmware implementation.

The firmware was written in several modules which were assembled separately and linked to produce the complete native firmware. For purposes of generating a listing for publication, a command procedure was developed which converts these modules into a single module which, when assembled, yields a listing that contains the actual addresses in the address column and the operand field.

The following is a list of each of the modules and a brief description of the function provided by each. A more detailed description proceeds each module:

ABAUD (Auto Baud) - Performs the auto baud function for serial ODT

ACCUM (Accumulate Number) - Accumulates a 16-bit number from octal input

ADDREG (Get Address/Register Number) - Inputs an address or register number for serial ODT

BSUPV (Boot Supervisor) - Controls the boot process

CHKCHR (Check Character) - Polls for input from console line

DATCOM (Data/Command) - Inputs data and commands for serial ODT

DECTST (Decode Test) - Decodes Q-Bus controlled test commands

DEPOS (Deposit) - Performs the deposit function for serial and Q-Bus ODTs

DINIT (Device Initialization) - Initializes all local I/O devices

DISPST (Display String) - Sends a print string to the console serial port

DISPWD (Display Word) - Converts a 16-bit word to an ASCII string and sends the string out to the console serial port

DMALD (DMA Load) - Controls the execution of the DMA load command

ENTRY (Entry Points) - Defines the two entry points to the native firmware (173000, 173004)

EXAMIN (Examine) - Performs the ODT examine function

EXTEST (Execute Test) - Executes each of the self tests

GETIN (Get Input) - Waits for and gets input from console serial line

GOPROC (Go/Proceed) - Performs both the Go and Proceed function of ODT

INIT (Initialization) - Initializes the local I/O

KXTDEF (KXT Definitions) - Defines the KXT specific constants used by the native firmware, KUI and MicroPower

LBTEST (Loop Back Test) - Performs all the self tests executed when in the loop back test mode

NFDEF (Native Firmware Definitions) - Defines all general native firmware related constants

PUTEST (Power-Up Tests) - Performs the appropriate power-up self tests

PWRUP (Power-Up Interrupt Handler) - Top level routine entered by Power-Up interrupt (173000)

QCOMIN (Q-Bus Command Interpreter) - Decode Q-Bus commands passed to first word of two-port RAM

QODTM (Q-Bus ODT Monitor) - Controls Q-Bus ODT

RESTRT (Restart Interrupt Handler) - Top level routine entered by Restart interrupt (173004)

SAVERG (Save Registers) - Save the user context of the CPU registers on entry of ODT

SETLED (Set LEDs) - Controls the LED display

SLFTS0 (Self Test Overlay 0) - Normally mapped section of self tests (tests 1 through 7)

SLFTS1 (Self Test Overlay 1) - Overlay section of self tests (tests 10 through 13)

SODTM (Serial ODT Monitor) - Controls serial ODT

SPR5SU (Stack Pointer/R5 Setup) - Initializes stack pointer and R5 for particular memory map selection

TNNERR (Test N Error Decoder) - Decodes error from given test

TRAP10 (Trap 10 Emulator) - Emulates trap to 10

TRAP24 (Trap 24 Emulator) - Emulates trap to 24

TRAP4 (Trap 4 Emulator) - Emulates trap to 4

TRAPX (Trap X Emulator) - Emulates trap to address passed in two-port RAM

TU58BT (TU58 Boot) - TU58 primary loader

VERS (Version) - Version code definition

WAITST (Wait State) - Wait for Q-Bus command loop

FIRMWARE MEMORY ORGANIZATION

The native firmware is contained in a 4 K word set of ROMs which are mapped in a 3 K word space with 1 K of overlay. It is organized into five sections in the native ROM. Table 1 describes these sections.

Table 1 Memory Organization

KXT Memory Address	ROM Address	Description
160000 - 163777	0000 - 3777	Normally mapped section of overlay memory (OVL0)
164000 - 172777	4000 - 12777	Permanently mapped section 1 (PERM1)
173000 - 173007	14000 - 14007	Interrupt entries (VECT)
173010 - 173777	14010 - 14777	Permanently mapped section 2 (PERM2)
160000 - 163777	15000 - 17777	Overlay section 1 (OVL1)

IMPORTANT ADDRESSES

The addresses listed in Table 2 are of special interest. Any address specified by symbolic name can be determined by looking that name up in the symbol list at the end of the native firmware listing.

Table 2 Special Addresses

Name	Module	Description
PERV(173000)	ENTRY	Power-up interrupt address
RSTV(173004)	ENTRY	Restart interrupt address
FARSTR	PUTEST	Fatal self test error loop
FAS	BSUPV	Fatal error loop when invalid Boot/Self Test switch position set
FAR	RESTRT	Loop address after Restart interrupt when a fatal error condition exists

REGISTER USE

The only CPU register that has a fixed use throughout the native firmware is R5. This register is used to point to the top of the native RAM (the highest word address) and is referenced throughout the firmware. This register should never be altered while the firmware is executing.

```
                                .SBTTL  KXTDEF.MAC - Dual Port RAM definitions on KXT11CA
                                .ident  /V1.5/
                        ;
                        ;  FUNCTIONAL DESCRIPTION:
                        ;
                        ;  This module defines addresses and bit definitions for various registers
                        ;  on the KXT11-CA.  The following naming conventions are used:
                        ;
                        ;    KW.xxx = Word addressable register on the KXT11-C
                        ;    KL.xxx = Low byte addressable register on the KXT11-C
                        ;    KH.xxx = High byte addressable register on the KXT11-C
                        ;    KX$xxx = Read/write bit definition for a KXT11-C register
                        ;    KW$xxx = Write only bit definition for a KXT11-C register
                        ;    KR$xxx = Read only bit definition for the KXT11-C register
                        ;    KM$xxx = Multi-bit mask for a KXT11-C register
                        ;    KP$xxx = Bit Pattern for Multi-bit fields of KXT11-C registers
                        ;
                        ;  These definitions can be used in one of two ways.  They can be accessed
                        ;  by using the macro KXTDF$ which will define each symbol locally in the
                        ;  module using the macro.  Secondly, this module can be assembled which will
                        ;  define each symbol globally in an object that other modules can link to.
                        ;  A KXT kernel includes the global definitions so any application merging
                        ;  with a KXT kernel need not use the local macro.
                        ;
                        ;-
                        ;
                        ;IF THIS MODULE IS ASSEMBLED, THEN GENERATE THE SYMBOLS IN A GLOBAL MODE
                        ;>  and show the macro expansions
                                .list   me
    000000                      KXTD$$  <==:>
                        ;
                        ;  KXT11-C CSR A  definitions

            177520              KW.CSA  ==:     177520          ; KXTCSRA word location
            177520              KL.CSA  ==:     177520          ; KXTCSRA low byte location

            000001                KX$ODT ==:    1               ; Sync Mode on channel B enable
            000002                KX$2EN ==:    2               ; SLU 2 read enable
            000004                KX$SMA ==:    4               ; Sync Mode on channel A enable
            000010                KX$TTC ==:    10              ; Terminal connect
            000020                KX$TIS ==:    20              ; Terminal in service
            000040                KX$DPE ==:    40              ; Diagnostic PROM enable
            000100                KX$RTE ==:    100             ; Real-time clock interrupt enable
            000200                KX$CTE ==:    200             ; Counter interrupt enable

                        ;  KXT11 CSR B definitions

            177522              KW.CSB  ==:     177522          ; Word address for KXTCSRB
            177522              KL.CSB  ==:     177522          ; Low byte address for KXTCSRB

            000001                KR$TNS ==:    1               ; Terminal in normal service mode
            000016                KM$MAP ==:    16              ; Memory map configuration
            000002                KX$MP0 ==:    2               ; Memory map bit 0
            000004                KX$MP1 ==:    4               ;   "     "    "  1
            000010                KX$MP2 ==:    10              ;   "     "    "  2
            000360                KM$SWS ==:    360             ; Boot/Self Test switch postion
```

```
                                     ;  KXT11 CSR C definitions

        177524                          KW.CSC  ==:     177524          ; Word address for KXTCSRC
        177524                          KL.CSC  ==:     177524          ; Low byte address for KXTCSRC

        000001                            KX$LD0 ==:    1               ; LED 0
        000002                            KX$LD1 ==:    2               ; LED 1
        000004                            KX$LD2 ==:    4               ; LED 2
        000010                            KX$LD3 ==:    10              ; LED 3
        000360                            KM$IDS ==:    360             ; KXT11-C ID switch setting

                                     ;  KXT11 CSR D definitions

        177530                          KW.CSD  ==:     177530          ; Word address for KXTCSRD
        177530                          KL.CSD  ==:     177530          ; Low byte address for KXTCSRD
        177531                          KH.CSD  ==:     177531          ; High byte address for KXTCSRD

        000001                            KX$R20 ==:    1               ; Dualport RAM interrupt 120 request
        000002                            KX$R24 ==:    2               ; Dualport RAM interrupt 124 request
        000004                            KX$R34 ==:    4               ; Dualport RAM interrupt 134 request
        000010                            KX$I20 ==:    10              ; Dualport RAM interrupt 120 enable
        000020                            KX$I24 ==:    20              ; Dualport RAM interrupt 124 enable
        000040                            KX$I34 ==:    40              ; Dualport RAM interrupt 134 enable
        000100                            KX$DEN ==:    100             ; Dualport RAM enable
        000200                            KX$DRT ==:    200             ; Dualport RAM non-maskable trap
        000400                            KX$BHE ==:    400             ; B-halt enable
        001000                            KX$BHF ==:    1000            ; B-halt flag
        002000                            KX$QRE ==:    2000            ; Q-bus reset enable
        004000                            KX$QRF ==:    4000            ; Q-bus reset
        010000                            KX$BQI ==:    10000           ; Block Q bus interrupt register
        020000                            KX$QIE ==:    20000           ; QIR interrupt enable
        040000                            KR$QRP ==:    40000           ; QIR request pending
        100000                            KX$NXM ==:    100000          ; Non-existent memory flag

                                     ;  KXT11 I/O Buffer Control definitions

        177540                          KW.IOC  ==:     177540          ; Word address for IOBFRC
        177540                          KL.IOC  ==:     177540          ; Low byte address for IOBFRC
        177541                          KH.IOC  ==:     177541          ; High byte address for IOBFRC


        000001                            KW$PB0 ==:    1               ; PIO port B byte 0 direction
        000002                            KW$PB1 ==:    2               ; PIO port B byte 1 direction
        000004                            KW$PB2 ==:    4               ; PIO port B byte 2 direction
        000010                            KW$PB3 ==:    10              ; PIO port B byte 3 direction
        000020                            KW$PB4 ==:    20              ; PIO port B byte 4 direction
        000040                            KW$PB5 ==:    40              ; PIO port B byte 5 direction
        000100                            KW$PB6 ==:    100             ; PIO port B byte 6 direction
        000200                            KW$PB7 ==:    200             ; PIO port B byte 7 direction
        000400                            KW$PLD ==:    400             ; PIO port A low-byte direction
        001000                            KW$PHD ==:    1000            ; PIO port A high-byte direction
        002000                            KW$P0D ==:    2000            ; PIO port C word 0 direction
        004000                            KW$P1D ==:    4000            ; PIO port C word 1 direction
        010000                            KW$P2D ==:    10000           ; PIO port C word 2 direction
        020000                            KW$P3D ==:    20000           ; PIO port C word 3 direction
        040000                            KW$PAB ==:    40000           ; PIO ports A and B active pull-up
        100000                          KW$PCO ==:     100000          ; PIO port C active pull-up

                                ;   DUAL PORT RAM IOP COMMAND REGISTER DEFINITIONS

        175000                       KW.CMD  ==:      175000          ;KXT DPR REGISTER 0 WORD LOC.
        175000                       KL.CMD  ==:      175000          ; "    "      "     " LOW BYTE LOC.
        175001                       KH.CMD  ==:      175001          ; "    "      "     " HIGH BYTE LOC.

                                ;
                                ;   SYSTEM COMMAND BIT DEFINITIONS
                                ;
        000001                          KX$TRC ==:     1              ;TRAP COMMAND
        000002                          KX$DMC ==:     2              ;DMA LOAD COMMAND
        000004                          KX$INC ==:     4              ;RE-INITIALIZE COMMAND
        000010                          KX$ODC ==:     10             ;ENTER Q BUS ODT MODE COMMAND
        000020                          KX$SHC ==:     20             ;SHOW CONFIGURATION COMMAND
        000040                          KX$NOP ==:     40             ;NO OPERATION COMMAND
        100001                          KX$T01 ==:     100001         ;CSP TEST COMMAND
        100002                          KX$T02 ==:     100002         ;RAM   "    "
        100004                          KX$T03 ==:     100004         ;ROM   "    "
        100010                          KX$T04 ==:     100010         ;CPU   "    "
        100020                          KX$T05 ==:     100020         ;BEVENT "    "
        100040                          KX$T06 ==:     100040         ;SLU1  "    "
        100100                          KX$T07 ==:     100100         ;SLU2  "    "
        100200                          KX$T10 ==:     100200         ;PARALLEL I/O TEST COMMAND
        100400                          KX$T11 ==:     100400         ;DMA TEST COMMAND
        101000                          KX$T12 ==:     101000         ;QIR   "    "
        102000                          KX$T13 ==:     102000         ;DPR   "    "
                                ;
                                ;   Q BUS ODT COMMAND DEFINITIONS
                                ;
        000001                          KX$OMO ==:     1              ;OPEN AND EXAMINE MEMORY COMMAND
        000002                          KX$ORO ==:     2              ;OPEN AND EXAMINE REGISTER COMMAND
        000004                          KX$DEO ==:     4              ;DEPOSIT COMMAND
        000010                          KX$GOO ==:     10             ;GO COMMAND
        000020                          KX$PRO ==:     20             ;PROCEED COMMAND
        100000                          KX$EXO ==:     100000         ;EXIT ODT COMMAND

                                ;
                                ;   DUAL PORT RAM IOP STATUS REGISTER DEFINITIONS
                                ;
        175002                       KW.STA  ==:      175002          ;KXT DPR REGISTER 1 WORD LOC.
        175002                       KL.STA  ==:      175002          ; "    "      "     " LOW BYTE LOC.
        175003                       KH.STA  ==:      175003          ; "    "      "     " HIGH BYTE LOC.

        000007                          KM$STF ==:     7              ;STATE FIELD

        000000                            KP$INI ==:  0               ;INITIALIZATION STATE
        000001                            KP$STS ==: 1                ;POWER UP SELF TEST STATE
        000002                            KP$QTS ==:  2               ;Q BUS CONTROLLED TEST MODE
        000003                            KP$QOD ==:  3               ;Q BUS ODT MODE
        000004                            KP$WST ==:  4               ;WAITING FOR COMMAND STATE
        000005                            KP$PBS ==:  5               ;PRIMARY BOOTSTRAP STATE
        000007                            KP$NNC ==:  7               ;EXECUTING NON NATIVE CODE

        000010                          KX$SEF ==:     10             ;STACK ERROR FLAG
        000020                          KX$OHL ==:    20             ;ENTER ODT ON HALT FLAG
        000040                          KX$FEF ==:    40             ;FATAL ERROR FLAG
        000100                          KX$SOF ==:    100            ;SERIAL ODT FLAG
        000200                          KX$QOF ==:    200            ;Q BUS ODT FLAG
        001000                          KX$PWF ==:    1000           ;POWER UP NO BATTERY BACKUP FLAG
        002000                          KX$BAT ==:    2000           ;BATTERY BACKUP POWER UP FLAG
        004000                          KX$SXT ==:    4000           ;STACK POINTER NXM TEST FLAG
        010000                          KX$NXF ==:    10000          ;NXM HANDLING FLAG
        020000                          KX$BDF ==:    20000          ;BREAK DISABLE FLAG
        040000                          KX$DME ==:    40000          ;DMA LOAD ERROR FLAG
        100000                          KX$CME ==:    100000         ;COMMAND ERROR FLAG
                                ;
                                ;   DUAL PORT RAM SYSTEM CONTROL REGISTER 2
                                ;
        175004                         KW.SC2  ==:    175004         ;KXT DPR REGISTER 2 WORD LOC.
        175004                         KL.SC2  ==:    175004         ; "   "    "     " LOW BYTE LOC.
        175005                         KH.SC2  ==:    175005         ; "   "    "     " HIGH BYTE LOC.
                                ;
                                ;   ERROR REPORT BIT DEFINITION OF REGISTER 2
                                ;
        000001                          KX$F01 ==:    1              ;CSR TEST FAILED
        000002                          KX$F02 ==:    2              ;RAM   "   "
        000004                          KX$F03 ==:    4              ;ROM   "   "
        000010                          KX$F04 ==:    10             ;CPU   "   "
        000020                          KX$F05 ==:    20             ;BEVENT "  "
        000040                          KX$F06 ==:    40             ;SLU1  "   "
        000100                          KX$F07 ==:    100            ;SLU2  "   "
        000200                          KX$F10 ==:    200            ;PARALLEL I/O TEST FAILED
        000400                          KX$F11 ==:    400            ;DMA TEST FAILED
        001000                          KX$F12 ==:    1000           ;QIR  "    "
        002000                          KX$F13 ==:    2000           ;DPR  "    "
                                ;
                                ;   DUAL PORT RAM SYSTEM CONTROL REGISTER 3
                                ;
        175006                         KW.SC3  ==:    175006         ;KXT DPR REGISTER 3 WORD LOC.
        175006                         KL.SC3  ==:    175006         ; "   "    "     " LOW BYTE LOC.
        175007                         KH.SC3  ==:    175007         ; "   "    "     " HIGH BYTE LOC.
                                ;
                                ;
                                ; Data Channel Base addresses (on the KXT side)
        175010                         KW.DC0  ==:    175010          ; CHANNEL 0
        175020                         KW.DC1  ==:    175020          ; CHANNEL 1
                                ;
                                ;
                                ; Offsets to Data Channel Base address
        000000                         KW.DCO  ==:    0               ; Command word
        000002                         KW.DST  ==:    2               ; Status word
        000004                         KW.DAT  ==:    4               ; beginning of data
                                ;
                                ; Bit definitions within Data Channel Command
                                ;
        000036                         KC.COM  ==:    36              ; Command field
        000040                         KC.ICC  ==:    40              ; Interrupt on command complete
        000100                         KC.IDA  ==:    100             ; Interrupt on DA
        000200                         KC.IDR  ==:    200             ; Interrupt on DR
        003400                         KC.LEN  ==:    3400            ; Length field
        004000                         KC.EOM  ==:    4000            ; End of message
```

```
              177400                     KC.VEC  ==:     177400          ; Vector Number (high byte)
                                 ;
                                 ; Bit definitions within Data Channel Status
              000036                     KS.ERC  ==:     36              ; Error code
              000040                     KS.DR   ==:     40              ; Data requested (Write will succeed)
              000100                     KS.EOM  ==:     100             ; End of message
              000200                     KS.DA   ==:     200             ; Data available (Read will succeed)
              003400                     KS.ALN  ==:     3400            ; Actual length of transfer
              020000                     KS.DBG  ==:     20000           ; Debug available
              040000                     KS.ON   ==:     40000           ; Interface ready (on)
              100000                     KS.ERR  ==:     100000          ; Error (Cumulative)
                                 ;
                                 ; Data Channel Command Codes
                                 ;
              000000                     KC$NOP  ==:     0.              ; NOP command
              000002                     KC$RES  ==:     2.              ; Reset command
              000004                     KC$EI   ==:     4.              ; Enable Interupts command
              000006                     KC$DI   ==:     6.              ; Disable Interupts command
              000010                     KC$GS   ==:     8.              ; Get Status command
              000012                     KC$SS   ==:     10.             ; Set Status command
              000014                     KC$RD   ==:     12.             ; Read data command
              000016                     KC$WD   ==:     14.             ; Write data command
              000020                     KC$ED   ==:     16.             ; Enable Debug command
              000022                     KC$DD   ==:     18.             ; Disable Debug command
              000022                     KC$MAX  ==:     KC$DD           ; Maximum command code
                                 ;
                                 ;  Data Channel Error Codes
                                 ;
              000000                     KE$OK   ==:     0.              ; Success (no)error code
              000002                     KE$NDA  ==:     2.              ; No data available (read rejected)
              000004                     KE$NDR  ==:     4.              ; No data requested (write rejected)
              000006                     KE$ILC  ==:     6.              ; Illegal command field
              000010                     KE$ILL  ==:     8.              ; Illegal length field
              000012                     KE$ILV  ==:     10.             ; Illegal vector
              000014                     KE$DNA  ==:     12.             ; Debug not available
              000014                     KE$MAX  ==:     KE$DNA          ; Maximum error code
                                 ;
    286                                  .nlist  me
    287                          ;
```

```
                        .SBTTL  NFDEF - Native Firmware definitions
                        .ENABLE LC,GBL

                ;
                ; Module name: DEF - DEFINITIONS
                ;
                ; System: KXT11-CA Native Firmware
                ;
                ;
                ;
                ;
                ;
                ;
                ; Functional Description:
                ;
                ;
                ;  This module defines addresses and bit definitions for various I/O registers
                ;  on the KXT11-CA.
                ;
```

```
                                        ;
                                        ;   BIT SYMBOL DEFINITIONS
                                        ;
        000001                                  BIT0    ==      1
        000002                                  BIT1    ==      2
        000004                                  BIT2    ==      4
        000010                                  BIT3    ==      10
        000020                                  BIT4    ==      20
        000040                                  BIT5    ==      40
        000100                                  BIT6    ==      100
        000200                                  BIT7    ==      200
        000400                                  BIT8    ==      400
        001000                                  BIT9    ==      1000
        002000                                  BIT10   ==      2000
        004000                                  BIT11   ==      4000
        010000                                  BIT12   ==      10000
        020000                                  BIT13   ==      20000
        040000                                  BIT14   ==      40000
        100000                                  BIT15   ==      100000
                                        ;
                                        ;    SLU1 ADDRESS DEFINITIONS
                                        ;
        177560                              RCSR1A  ==          177560

        004000                                  RCVACB ==       BIT11
        000200                                  RCVDNB ==       BIT7
        000100                                  RCVIEB ==       BIT6

        177562                              RBUF1A  ==          177562

        100000                                  ERRB    ==      BIT15
        040000                                  ORERRB   ==     BIT14
        020000                                  FRERRB   ==     BIT13
        004000                                  RCVBKB   ==     BIT11

        177564                              XCSR1A  ==          177564

        000200                                  XRDYB    ==     BIT7
        000100                                  XIEB     ==     BIT6
        000070                                  PBRF     ==     BIT5!BIT4!BIT3
        000004                                  MAINTB   ==     BIT2
        000002                                  PBREB    ==     BIT1
        000001                                  XBRKB    ==     BIT0

        177566                              XBUF1A  ==          177566


                                        ;
                                        ;   SLU2 Definitions
                                        ;
        175700                              SL2SAA  ==          175700          ;SLU 2 channel A Status register addres
        175710                              SL2SBA  ==          175710          ; "  "     "    B   "         "       "
        175704                              SL2CAA  ==          175704          ; "  "     "    A Command register    "
        175714                              SL2CBA  ==          175714          ; "  "     "    B   "         "       "
                                        ;
                                        ;   PIO Definitions
                                        ;
        177000                              PIOICA  ==          177000          ;PIO interrupt control register address
        177004                         PIOAVA  ==     177004        ;PIO Port A interrupt vector register
        177006                         PIOBVA  ==     177006        ; "    " B     "        "        "
        177010                         PIOTVA  ==     177010        ; "  Counter Timer  "   "        "
                                 ;
                                 ;   DMA Definitions
                                 ;
        174470                         DMAMMA  ==     174470        ;DMA Master Mode Register
        174532                         DMA0VA  ==     174532        ;DMA Channel 0 Interrupt Vector Register
        174530                         DMA1VA  ==     174530        ; "     "    1     "        "       "
        174442                         DMA0OA  ==     174442        ;DMA Channel 0 Chain Offset Field
        174446                         DMA0SA  ==     174446        ; "     "    "  "   Segment/Tag Field
        174454                         DMACDA  ==     174454        ; "  Command Register

                                 ;
                                 ;   8255 Control Register
                                 ;
        177526                         C8255   ==     177526

```

```
                                     .SBTTL  SELF TEST OVERLAY 0
                                     .LIST   ME,MEB,SEQ,LOC,BIN      ; NORMAL LISTING MODE
                                     .NLIST  MC,CND,BEX              ;       DITTO
                                     .ENABL  AMA,GBL
        160000                       . = 160000
                               ;
                               ; Module name: SELF TEST OVERLAY 0
                               ;
                               ; System: KXT11-CA Native Firmware
                               ;
                               ;
                               ;
                               ;
                               ;
                               ;
                               ; Functional Description:
                               ;
                               ;       This module contains overlay 0 of the self test code. The self test
                               ;       modules in this overlay are:
                               ;
                               ;               I/O register check,
                               ;               Native (and user) RAM,
                               ;               Native (and user) ROM,
                               ;               CPU (a null test),
                               ;               Line clock (BEVNT) interrupt,
                               ;               Console serial port (DC319), and
                               ;               Second serial port (NEC7201).
                               ;
                               ;       Overlay 0 physically occupies the first 2 KB of the boot ROM. It
                               ;       executes starting at 160000 when the DIAG PROM EN bit, bit 5, in
                               ;       KXTCSPA is 0.
                               ;
                               ;       RETURNED ERRORS ARE BIT ENCODED IN R0 (TTEEEE)
                               ;       WHERE T = TEST NUMBER
                               ;       . AND E = 12 DISCRETE ERROR FLAGS.
                               ;       . I.E.  010004 = TEST 01, ERROR(BIT) 2.
```

```
                        ; SCRATCH RAM ALLOCATION.
                        ;
                        ; R5 IS ODT'S "TOP-OF-MEM" POINTER, AND IS PRFSERVED AS SUCH.
                        ; ALL SCRATCH REFERENCES ARE INDEXED FROM R5 AS FOLLOWS:
                        ;
                        ;                               LO      MID     HI
                        ;                               ------  ------  ------
        000005          $TOP=   %5              ; 077776  137776  157776
                        ;
        150002          BUFR1=  -27776          ; 050000  110000  130000  TOP-6KW
        160002          BUFR2=  -17776          ; 060000  120000  140000  TOP-4KW
        170002          TMP1=   -07776          ; 070000  130000  150000  TOP-2KW
        170004          TMP2=   TMP1+2          ;                          \
        170006          TMP3=   TMP1+4          ;                           \
        170010          TMP4=   TMP1+6          ;                            \
        170012          TMP5=   TMP1+10         ;                             > SCRATCH REGISTERS...
        170014          TMP6=   TMP1+12         ;                            /  ...AND STACK SPACE.
        170016          TMP7=   TMP1+14         ;                           /
        170020          TMP8=   TMP1+16         ;                          /
        177776          $EF1=   -2              ; LAST 2 WORDS OF NATIVE RAM (BATTERY-BACK-FLAGS)...
        000005          $EF=    $TOP            ;...ARE UTILIZED AS OUR ERROR FLAGS.
                        ;
                        ; MISCELLANEOUS STUFF,
                        ;
        000000          $TN=    0               ; INIT TEST NUMBER SEQUENCE.
                        ;
                        ; A FEW MACROS.
                        ;
                        .MACRO  BEGIN NAME,TAG,V1,V2,V3
                                .NLIST
                                $TN=$TN+1               ;; SET TEST NUMBER...
                                $EN=0                   ;;...AND RESET ERROR NUMBER.
                                .LIST
                                .IRP    N,\$TN
                        .SBTTL  - T'N   NAME
                        ;;**********************************************************************
                        ;;* TEST N -- NAME
                        ;;**********************************************************************
                        TST'N:
                        TAG:
                                .ENDR
                        .IF NB <V1>
                                MOV     V1,-(SP)        ;; SAVE 'V1'.
                                MOV     V1+2,-(SP)
                         .IF NB <V2>
                                MOV     V2,-(SP)        ;; SAVE 'V2'.
                                MOV     V2+2,-(SP)
                          .IF NB <V3>
                                MOV     V3,-(SP)        ;; SAVE 'V3'.
                                MOV     V3+2,-(SP)
                          .ENDC
                         .ENDC
                        .ENDC
                                CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                .ENDM   BEGIN

                        .MACRO  EXIT    V1,V2,V3,?TAG
```

```
.IF NB <V1>
        MOV     (SP)+,V1+2      ;; RESTORE 'V1'.
        MOV     (SP)+,V1
 .IF NB <V2>
        MOV     (SP)+,V2+2      ;; RESTORE 'V2'.
        MOV     (SP)+,V2
  .IF NB <V3>
        MOV     (SP)+,V3+2      ;; RESTORE 'V3'.
        MOV     (SP)+,V3
  .ENDC
 .ENDC
.ENDC
        MOV     ($EF),R0        ;; ERROR BITS => R0<11:00>...
        BEQ     TAG             ;;...AND SKIP IF NONE.
        .IRP    TN,\$TN
        BIS     #<TN*BIT12>,R0  ;; ELSE, ADD TEST NUM => R0<15:12>...
        .ENDR
TAG:    RETURN                  ;;...AND RETURN (NZ).
        .ENDM   EXIT

.MACRO  ERROR   TEXT
        .NLIST  ME
        .RADIX  10
        .IRP    N,\$EN
        BIS     #BIT'N,($EF)    ;; E'N = 'TEXT
        .ENDR
        $EN=$EN+1
        .RADIX  8
        .NLIST
        .LIST   MF
        .LIST
        .ENDM   ERROR

.MACRO  LRESET
        SOB     R0,.
         RESET                  ;; LOCAL RESET.
          SOB   R0,.
        .ENDM   LRESET

        .NLIST MD
```

```
                            ; A FEW DEFINITIONS
                            ;
                            ; STANDARD STUFF.
                            ;
        000000              PR0=    000             ; THE PRIORITIES.
        000040              PR1=    040
        000100              PR2=    100
        000140              PR3=    140
        000200              PR4=    200
        000240              PR5=    240
        000300              PR6=    300
        000340              PR7=    340
                            ;
                            ; PRIMARY CONTROL REGISTERS.
                            ;
        177560              $SL1=   177560          ; DEC DC319 (DLART) ASYNC SERIAL PORT.
        175700              $SL2=   175700          ; NEC7201 SYNC/ASYNC SERIAL PORT.
        175720              $I8254= 175720          ; I8254 PROGRAMMABLE BAUD GEN FOR ABOVE.
        177000              $PIO=   177000          ; Z8036 PIO/PIT.
        174400              $DMA=   174400          ; AMZ8016 DMA CONTROLLER.
        177520              $CSRA=  177520          ; \
        177522              $CSRB=  177522          ;  > SECONDARY CSR'S...
        177524              $CSRC=  177524          ; /  ...EMBEDDED IN I8255 CHIP.
        177526              $CSRCON= 177526         ;
        177530              $CSR=   177530          ; PRIMARY CSR.
        177532              $QIR=   177532          ; Q-BUS INTERRUPT REGISTER.
        175000              $DPR=   175000          ; DUAL-PORT RAM (LOCAL).
        160000              $QDPR1= 160000          ; DUAL-PORT RAM (GLOBAL BASE ID 0-7...
        175400              $QDPR2= 175400          ;                         ...AND ID 8-F).
                            ;
        175000              $IPV=   $DPR            ; IPV INTERRUPTS (VIA DPR WORDS 0, 4, 8, AND 12).
        177524              $LEDS=  $CSRC           ; CSRC<3:0> DRIVE THE LEDS.
        177524              $UID=   $CSRC           ; CSRC<7:4> SHOW THE ID SWITCH SETTING.
```

```
                                ; DEFAULT VECTORS.
                                ;
        000024                  PWRV=   024             ; PWR-UP (BRESET)       (NON-MASKABLE).
        000060                  SL1RV=  060             ; CONSOLE (SLU1) RCVR   (PRI 4).
        000064                  SL1XV=  064             ; CONSOLE (SLU1) XMTR   (PRI 4).
        000070                  NECV=   070             ; SYNC/ASYNC BOTH CHANS (PRI 4).
        000100                  BEVNT=  100             ; LINE CLOCK            (PRI 6).
        000104                  PEVNT=  104             ; PROGRAMMABLE CLOCK(S) (PRI 6).
                                ;       110             ;       OPEN
        000114                  PRTYV=  114             ; MEMORY PARITY         (PRI 7).
        000120                  DPRV4=  120             ; DUAL-PORT RAM WORD 4  (PRI 5).
        000124                  DPRV8=  124             ; DUAL-PORT RAM WORD 8  (PRI 5).
        000130                  BIACKV= 130             ; Q-BUS "IACK"          (PRI 5).
        000134                  DPRV12= 134             ; DUAL-PORT-RAM WORD 12 (PRI 5).
                                ;       140             ;       RESERVED (FALCON).
        000144                  QIRV=   144             ; Q-BUS REQUEST         (ARBITER EXECUTES RTI).
        000150                  QIRV1=  150             ;   DITTO               (ARBITER EXECUTES RESET,RTI)
        000154                  PIOAV=  154             ; PIO PORT A            (PRI 4).
        000160                  PIOBV=  160             ; PIO PORT B            (PRI 4).
        000164                  DMAV=   164             ; DMA (BOTH CHANNELS)   (PRI 4).
```

```
                                        ; LOW SEGMENT ENTRY BLOCK
                                        ;
                                        ; CALL: BIC  #BIT5,CSRA ; ENABLE LO SEGMENT.
                                        ;       CALL TN         ; CALL TEST 0-7.
                                        ;       BEQ  OK         ; OR BNE ERROR.
                                        ;
        160000                          T0==.
160000  005000                                  CLR     R0      ;SINCE THERE IS NO TEST BY THIS NAME, SHOW NO ERRORS.
160002  000207                                  RETURN
        160004                          T1==.
160004  000137  160040                          JMP     TST1            ;; EXECUTE TEST 1.
        160010                          T2==.
160010  000137  160272                          JMP     TST2            ;; EXECUTE TEST 2.
        160014                          T3==.
160014  000137  160700                          JMP     TST3            ;; EXECUTE TEST 3.
        160020                          T4==.
160020  000137  161150                          JMP     TST4            ;; EXECUTE TEST 4.
        160024                          T5==.
160024  000137  161164                          JMP     TST5            ;; EXECUTE TEST 5.
        160030                          T6==.
160030  000137  161344                          JMP     TST6            ;; EXECUTE TEST 6.
        160034                          T7==.
160034  000137  161730                          JMP     TST7            ;; EXECUTE TEST 7.
```

```
                                .SBTTL  - T1    I/O REGISTER CHECK
                                ;;*********************************************************************
                                ;;* TEST 1 -- I/O REGISTER CHECK
                                ;;*********************************************************************
 160040                         TST1:
 160040                         TIOR:
 160040  005015                         CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                ;
                                ; INPUT:  NONE.
                                ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                ;
                                ; VERIFY THAT ALL ASSIGNED I/O ADDRESSES ARE VALID.
                                ;
                                ;       T1      I/O REGISTER CHECK
                                ;       .         E0 = BUS-ERROR AT CSR ADDRESS
                                ;       .         E1 = BUS-ERROR AT QIR ADDRESS
                                ;       .         E2 = BUS-ERROR AT DPR ADDRESS
                                ;       .         E3 = BUS-ERROR AT DC319 ADDRESS
                                ;       .         E4 = BUS-ERROR AT NEC7201 ADDRESS
                                ;       .         E5 = BUS-ERROR AT I8254 ADDRESS
                                ;       .         E6 = BUS-ERROR AT Z8036 ADDRESS
                                ;       .         E7 = BUS-ERROR AT Z8016 ADDRESS

 160042  012700  177520         1$:     MOV     #$CSRA,R0       ; THE CSR SET...
 160046  012701  000005                 MOV     #5.,R1          ;...5 REGISTERS.
 160052  004737  160254                 CALL    10$
 160056  052715  000001                 BIS     #BIT0,($EF)     ;; E0 = BUS-ERROR AT CSR ADDRESS

 160062  012700  177532         2$:     MOV     #$QIR,R0        ; THE QIR...
 160066  012701  000001                 MOV     #1,R1           ;...ONE-OF-A-KIND.
 160072  004737  160254                 CALL    10$
 160076  052715  000002                 BIS     #BIT1,($EF)     ;; E1 = BUS-ERROR AT QIR ADDRESS

 160102  012700  175000         3$:     MOV     #$DPR,R0        ; THE 2-PORT-RAM...
 160106  012701  000020                 MOV     #16.,R1         ;...16 REGISTERS.
 160112  004737  160254                 CALL    10$
 160116  052715  000004                 BIS     #BIT2,($EF)     ;; E2 = BUS-ERROR AT DPR ADDRESS

 160122  012700  177560         4$:     MOV     #$SL1,R0        ; THE CONSOLE PORT...
 160126  012701  000004                 MOV     #4.,R1          ;...4 REGISTERS.
 160132  004737  160254                 CALL    10$
 160136  052715  000010                 BIS     #BIT3,($EF)     ;; E3 = BUS-ERROR AT DC319 ADDRESS

 160142  012700  175700         5$:     MOV     #$SL2,R0        ; THE SYNC/ASYNC PORT...
 160146  012701  000010                 MOV     #8.,R1          ;...8 REGISTERS...
 160152  004737  160254                 CALL    10$
 160156  052715  000020                 BIS     #BIT4,($EF)     ;; E4 = BUS-ERROR AT NEC7201 ADDRESS

 160162  012700  175720         6$:     MOV     #$I8254,R0      ;...AND ITS CLOCK GENERATOR...
 160166  012701  000010                 MOV     #8.,R1          ;...HAS 8 MORE.
 160172  004737  160254                 CALL    10$
 160176  052715  000040                 BIS     #BIT5,($EF)     ;; E5 = BUS-ERROR AT I8254 ADDRESS

 160202  012700  177000         7$:     MOV     #$PIO,R0        ; THE PARALLEL I/O PORT...
 160206  012701  000060                 MOV     #48.,R1         ;...A WHOPPING 48 REGISTERS.
 160212  004737  160254                 CALL    10$
 160216  052715  000100                 BIS     #BIT6,($EF)     ;; E6 = BUS-ERROR AT Z8036 ADDRESS
```

```
     52
     53 160222  012700  174400          8$:     MOV     #$DMA,R0            ; THE DMA ENGINE...
     54 160226  012701  000056                  MOV     #46.,R1             ;...A CLOSE SECOND WITH 46.
     55 160232  004737  160254                  CALL    10$
     56 160236  052715  000200                  BIS     #BIT7,($EF)         ;; E7 = BUS-ERROR AT Z8016 ADDRESS
     57 160242                          9$:
        160242  011500                          MOV     ($EF),R0            ;; ERROR BITS => R0<11:00>...
        160244  001402                          BEQ     30000$              ;;...AND SKIP IF NONE.
        160246  052700  010000                  BIS     #<1*BIT12>,R0       ;; ELSE, ADD TEST NUM => R0<15:12>...
        160252  000207                  30000$: RETURN                      ;;...AND RETURN (NZ).
     58
     59 160254  005720                  10$:    TST     (R0)+               ; TEST AN ADDRESS.
     60 160256  000240                          NOP
     61 160260  103403                          BCS     11$                 ; TAKE ERROR RETURN IF IT TRAPPED.
     62 160262  077104                          SOB     R1,10$
     63 160264  062716  000004                  ADD     #4,(SP)             ; ALL OK, TAKE SKIP RETURN.
     64 160270  000207                  11$:    RETURN
```

```
                                .SbTTL  - T2    NATIVE (AND USER) RAM
                                ;;*******************************************************************
                                ;;* TEST 2 -- NATIVE (AND USER) RAM
                                ;;*******************************************************************
160272                          TST2:
160272                          TRAM:
160272  005015                          CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                ;
                                ; INPUT:  R1 = NONE.
                                ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                ;
                                ; TEST LOCAL RAM FROM BOTTOM UP (NON-DESTRUCTIVE).
                                ; IF SOCKET "A" CONTAINS USER RAM, TEST IT AS WELL.
                                ;
                                ; 1. WRITE (DATO) EACH LOCATION WITH IT'S OWN ADDRESS.
                                ; 2. READ (DATI) AND VERIFY CORRECT DATA.
                                ; 3. COMPLIMENT WORD (DATIO) AND VERIFY.
                                ; 4. COMPLIMENT LO AND HI BYTES (DATIO(B)) AND VERIFY.
                                ;
                                ;       T2      NATIVE (AND USER) RAM
                                ;       .         E0 = BUS-ERROR AT RAM ADDRESS
                                ;       .         E1 = WRITE-READ ERROR
                                ;       .         E2 = READ-MOD-WRITE ERROR
                                ;       .         E3 = READ-MOD-WRITE(LB) ERROR
                                ;       .         E4 = READ-MOD-WRITE(HB) ERROR
                                ;       .         E5 TO E9 = SAME AS E0 TO E4 IN USER RAM SPACE

160274  020527  137776                  CMP     $TOP,#137776
160300  101007                          BHI     MEM3            ; BR IF RAM IS IN HIGH...
160302  001403                          BEQ     MEM2            ;...OR MID RANGE.
160304  012702  000000          MEM1:   MOV     #0,R2           ; SET BOTTOM AT 0.
160310  000405                          BR      MEM3A
160312  012702  040000          MEM2:   MOV     #40000,R2       ; SET BOTTOM AT 8K.
160316  000402                          BR      MEM3A
160320  012702  100000          MEM3:   MOV     #100000,R2      ; SET BOTTOM AT 16K.
160324  010504                  MEM3A:  MOV     $TOP,R4         ; SET TOP.
160326  005046                          CLR     -(SP)           ; CLEAR NATIVE/USER SWITCH.
160330  000241                  1$:     CLC
160332  011200                          MOV     (R2),R0         ; SAVE TARGET DATA.
160334  000240                          NOP
160336  103003                          BCC     2$
160340  052715  000001                  BIS     #BIT0,($EF)     ;; E0 = BUS-ERROR AT RAM ADDRESS
160344  000444                          BR      7$

160346  010201                  2$:     MOV     R2,R1           ; TARGET ADDRESS => R1.
160350  010211                          MOV     R2,(R1)         ; WRITE (DATO)...
160352  011103                          MOV     (R1),R3         ;...READ (DATI)...
160354  020203                          CMP     R2,R3           ;...AND CHECK.
160356  001402                          BEQ     3$
160360  052715  000002                  BIS     #BIT1,($EF)     ;; E1 = WRITE-READ ERROR
160364  005102                  3$:     COM     R2
160366  005111                          COM     (R1)            ; READ-MOD-WRITE (DATIO)...
160370  021102                          CMP     (R1),R2         ;...READ AND CHECK.
160372  001402                          BEQ     4$
160374  052715  000004                  BIS     #BIT2,($EF)     ;; E2 = READ-MOD-WRITE ERROR
160400  005102                  4$:     COM     R2
160402  105111                          COMB    (R1)            ; READ-MOD-WRITE-LB (DATIO(LB))...
```

```
   160404  121102                              CMPB    (R1),R2          ;...READ AND CHECK.
   160406  001402                              BEQ     5$
   160410  052715  000010                      BIS     #BIT3,($EF)      ;; E3 = READ-MOD-WRITE(LB) ERROR
   160414  000302                    5$:       SWAB    R2
   160416  105161  000001                      COMB    1(R1)            ; READ-MOD-WRITE-HB (DATIO(HB))...
   160422  126102  000001                      CMPB    1(R1),R2         ;...READ AND CHECK.
   160426  001402                              BEQ     6$
   160430  052715  000020                      BIS     #BIT4,($EF)      ;; E4 = READ-MOD-WRITE(HB) ERROR

   160434  000302                    6$:       SWAB    R2
   160436  010022                              MOV     R0,(R2)+         ; RESTORE AND BUMP TARGET ADDRESS.
   160440  020227  160330                        CMP   R2,#1$           ; **********************************************
   160444  001002                                BNE   65$              ; *** IF RUNNING IN RAM, DON'T TEST THE "TEST".
   160446  012702  160456                        MOV   #7$,R2           ; **********************************************
   160452  020204                    65$:      CMP     R2,R4            ; REACHED TOP ??
   160454  101725                              BLOS    1$               ; LOOP IF NOT.

   160456  005116                    7$:       COM     (SP)             ; TOGGLE NATIVE/USER SWITCH...
   160460  001416                              BEQ     8$               ;...AND EXIT IF BOTH DONE.
   160462  011565  177776                      MOV     ($EF),-2($EF)    ; SAVE NATIVE ERRORS...
   160466  005015                              CLR     ($EF)            ;...AND CLEAR FOR 2ND PASS.
   160470  004737  160546                      CALL    SKTA             ; GET SOCKET SPECS.
   160474  020227  000003                      CMP     R2,#3            ; CHECK WHAT'S THERE...
   160500  001406                              BEQ     8$               ;...AND BR IF ROM (OR EMPTY).
   160502  010002                              MOV     R0,R2            ; IT'S RAM, 1ST ADDRESS => R2...
   160504  010104                              MOV     R1,R4            ;...AND SIZE => R4.
   160506  005304                              DEC     R4               ; \
   160510  006304                                ASL   R4               ;  > CHANGE SIZE TO LAST ADDRESS...
   160512  060204                                  ADD R2,R4            ; /
   160514  000705                              BR      1$               ;...AND GO 'ROUND ONCE MORE.
                                               ;
                                     ;                          E5 TO E9 = SAME AS E0 TO E4 IN USER RAM SPACE
                                               ;
   160516  005726                    8$:       TST     (SP)+            ; DONE, POP THE STACK.
   160520  012700  000005                      MOV     #5,R0
   160524  006315                    9$:       ASL     ($EF)            ; SHIFT USER ERRORS TO BITS<9:5>...
   160526  077002                              SOB     R0,9$
   160530  056515  177776                      BIS     -2($EF),($EF)    ;...AND ADD NATIVE ERRORS IN BITS<4:0>.
   160534  011500                              MOV     ($EF),R0         ;; ERROR BITS => R0<11:00>...
   160536  001402                              BEQ     30001$           ;;...AND SKIP IF NONE.
   160540  052700  020000                      BIS     #<2*BIT12>,R0    ;; ELSE, ADD TEST NUM => R0<15:12>...
   160544  000207                    30001$: RETURN                     ;;...AND RETURN (NZ).
```

```
                                        ;
                                        ; SUBROUTINE TO DETERMINE USER (SOCKET A) ADDRESS AND SIZE.
                                        ;
                                        ; RETURN R0 = 1ST ADDRESS
                                        ;        R1 = SIZE (WORDS)
                                        ;        R2 = 3 IF ROM (OR EMPTY).
                                        ;
 160546  012700  100000                 SKTA:   MOV     #100000,R0      ; ASSUME ROM OVER RAM.
 160552  005705                                 TST     $TOP
 160554  100001                                 BPL     1$              ; BR IF SO...
 160556  005000                                 CLR     R0              ;...ELSE, SET ROM AT 0.
 160560  012701  004000                 1$:     MOV     #2048.,R1       ; START SIZING AT 2KW.
 160564  032737  000006  177522                  BIT    #<3*BIT1>,$CSRB
 160572  001417                                   BEQ   2$              ; BR IF 2KW SELECTED    (MAP 0 OR 4).
 160574  006301                                 ASL     R1              ; RAISE TO 4KW.
 160576  032737  000004  177522                  BIT    #<2*BIT1>,$CSRB
 160604  001412                                   BEQ   2$              ; BR IF 4KW SELECTED    (MAP 1 OR 5).
 160606  006301                                 ASL     R1              ; RAISE TO 8KW.
 160610  032737  000010  177522                  BIT    #<4*BIT1>,$CSRB
 160616  001405                                   BEQ   2$              ; BR IF 8KW SELECTED    (MAP 2 OR 3).
 160620  032737  000002  177522                   BIT   #<1*BIT1>,$CSRB
 160626  001401                                   BEQ   2$              ;       DITTO           (MAP 6).
 160630  006301                                 ASL     R1              ; RAISE TO 16KW         (MAP 7).
 160632  000401                         2$:     BR      RAMLOA          ; EXIT THRU ROM/RAM CHECKER.
                                        ;
                                        ; SUBROUTINE TO ESTABLISH ROM OR RAM IN VECTOR SPACE.
                                        ;
                                        ; RETURN R2<00> = 1 IF LO BYTE NON-WRITABLE.
                                        ;        R2<01> = 1 IF HI BYTE NON-WRITABLE.
                                        ;
 160634  005000                         RAMLO:  CLR     R0
 160636  005002                         RAMLOA: CLR     R2
 160640  011046                                 MOV     (R0),-(SP)      ; GET TARGET LOCATION...
 160642  005116                                 COM     (SP)            ;...AND COMPLIMENT IT.
 160644  005110                                 COM     (R0)            ; COMPLIMENT TARGET LOCATION.
 160646  126066  000001  000001                 CMPB    1(R0),1(SP)     ; HI BYTE WRITABLE ??
 160654  001402                                 BEQ     1$              ;       YES
 160656  052702  000002                         BIS     #BIT1,R2        ;       NO
 160662  121026                         1$:     CMPB    (R0),(SP)+      ; LO BYTE WRITABLE ??
 160664  001402                                 BEQ     2$              ;       YES
 160666  052702  000001                         BIS     #BIT0,R2        ;       NO
 160672  005110                         2$:     COM     (R0)            ; RESTORE TARGET LOCATION.
 160674  005702                                 TST     R2              ; SET CONDITION FOR CALLER.
 160676  000207                                 RETURN
```

```
                            .SBTTL  - T3    NATIVE (AND USER) ROM
                            ;;**********************************************************************
                            ;;* TEST 3 -- NATIVE (AND USER) ROM
                            ;;**********************************************************************
160700                      TST3:
160700                      TROM:
160700  005015                      CLR     ($EF)           ;; CLEAR ERROR FLAG.
                            ;
                            ; INPUT:  R1<BIT0> = INCLUDE USER ROM.
                            ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                            ;
                            ; CHECK SUM BOOT/SELF-TEST ROM (SOCKET B).
                            ; IF SELECTED, DO THE USER ROM (SOCKET A) AS WELL.
                            ;
                            ;       T3      NATIVE (AND USER) ROM
                            ;       .         E0 = LO BYTE CHECK SUM ERROR (NATIVE)
                            ;       .         E1 = HI BYTE CHECK SUM ERROR (NATIVE)
                            ;       .         E2 = LO BYTE CHECK SUM ERROR (USER)
                            ;       .         E3 = HI BYTE CHECK SUM ERROR (USER)

160702  010104                      MOV     R1,R4           ; SAVE OPTION FLAG.
160704  012700  161150              MOV     #ACC+2,R0
160710  012701  000043              MOV     #<ACC+2-CKSUM>/2,R1
160714  014046              1$:     MOV     -(R0),-(SP)     ; PUSH CHECK-SUM CODE ONTO THE STACK.
160716  077102                      SOB     R1,1$

160720  012700  160000              MOV     #160000,R0
160724  012701  010000              MOV     #4096.,R1
160730  004766  000000              CALL    0(SP)           ; CHECK SUM (LO BYTE).
160734  052715  000001              BIS     #BIT0,($EF)     ;; E0 = LO BYTE CHECK SUM ERROR (NATIVE)

160740  012700  160001      2$:     MOV     #160001,R0
160744  012701  010000              MOV     #4096.,R1
160750  004766  000000              CALL    0(SP)           ; CHECK SUM (HI BYTE).
160754  052715  000002              BIS     #BIT1,($EF)     ;; E1 = HI BYTE CHECK SUM ERROR (NATIVE)
160760                      3$:     ;
                                    ; IF ERROR, CHECK FOR OVERLAY PROBLEM ??
                                    ;
160760  032704  000001      4$:     BIT     #BIT0,R4        ; USER ROM SELECTED ??
160764  001417                      BEQ     6$              ; EXIT IF NOT.
160766  004737  160546              CALL    SKTA            ; YES, GET SOCKET SPECS.
160772  010046                      MOV     R0,-(SP)        ; PUSH START ADDRESS...
160774  010146                      MOV     R1,-(SP)        ;...AND SIZE.
160776  004766  000004              CALL    4(SP)           ; CHECK SUM (LO BYTE).
161002  052715  000004              BIS     #BIT2,($EF)     ;; E2 = LO BYTE CHECK SUM ERROR (USER)

161006  012601              5$:     MOV     (SP)+,R1        ; POP SIZE...
161010  012600                      MOV     (SP)+,R0        ;...AND ADDRESS.
161012  005200                      INC     R0
161014  004766  000000              CALL    0(SP)           ; CHECK SUM (HI BYTE).
161020  052715  000010              BIS     #BIT3,($EF)     ;; E3 = HI BYTE CHECK SUM ERROR (USER)

161024  062706  000106      6$:     ADD     #ACC+2-CKSUM,SP ; FIX STACK.
161030  011500                      MOV     ($EF),R0        ;; ERROR BITS => R0<11:00>...
161032  001402                      BEQ     30002$          ;;...AND SKIP IF NONE.
161034  052700  030000              BIS     #<3*BIT12>,R0   ;; ELSE, ADD TEST NUM => R0<15:12>...
161040  000207              30002$: RETURN                  ;;...AND RETURN (NZ).
```

```
                                      ;**********************************************************************
                                      ; ROM CHECK SUM ROUTINE.
                                      ; THIS STUFF IS COPIED TO AND EXECUTED IN STACK SPACE.
                                      ;
                                      .DSABL  AMA      ; RELATIVE PIC.
                                      ;
                                      ; ON ENTRY,  R0 = START ADDRESS AND R1 = BYTE COUNT.
                                      ; ON RETURN, R2 = ACCUMULATED SUM AND R3 = CHECK-SUM.
                                      ;
161042  005067  000100                CKSUM:  CLR     ACC               ; CLEAR ACCUMULATOR...
161046  005301                                DEC     R1                ;...AND DECR THE BYTE COUNT.
161050  111002                        1$:     MOVB    (R0),R2           ; 1ST/NEXT ROM LOCATION.
161052  042702  177400                        BIC     #^C377,R2
161056  060267  000064                        ADD     R2,ACC            ; \
161062  106367  000060                         ASLB   ACC               ;  > ACCUMULATE BYTE SUM.
161066  005567  000054                          ADC   ACC               ; /
161072  062700  000002                        ADD     #2,R0
161076  020027  174000                        CMP     R0,#174000        ; END OF NATIVE ROM ??
161102  103405                                BLO     2$                ; BR IF NOT.
161104  052737  000040  177520                BIS     #BIT5,@#$CSRA     ; YES, MAP IN THE OVERLAY...
161112  042700  017776                        BIC     #^C160001,R0      ;...AND ADJUST SRC POINTER.
161116  077124                        2$:     SOB     R1,1$             ; LOOP 'TIL DONE.
161120  116702  000022                        MOVB    ACC,R2            ; THEN PUT ACCUMULATED SUM IN R2...
161124  111003                                MOVB    (R0),R3           ;...AND THE CHECK-SUM IN R3.
161126  042737  000040  177520                BIC     #BIT5,@#$CSRA     ; UNMAP OVERLAY.
161134  120203                                CMPB    R2,R3             ; ACCUMULATED SUM SHOULD EQUAL CHECK-SUM.
161136  001002                                BNE     3$                ; ERROR RETURN IF NOT.
161140  062716  000004                        ADD     #4,(SP)           ; SKIP RETURN IF SO.
161144  000207                        3$:     RETURN
161146  000000                        ACC:    0
                                              .ENABL  AMA
                                      ;**********************************************************************
```

```
                                 .SBTTL  - T4    T11 CPU INSTRUCTIONS AND TRAPS
                                 ;;**************************************************************************
                                 ;;* TEST 4 -- T11 CPU INSTRUCTIONS AND TRAPS
                                 ;;**************************************************************************
161150                           TST4:
161150                           TCPU:
161150  005015                           CLR     ($EF)            ;; CLEAR ERROR FLAG.
                                 ;
                                 ; INPUT:  NONE.
                                 ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                 ;
                                 ; TEST THE CPU (WHATEVER THAT MEANS) !!
                                 ;
                                 ;       T4      T11 CPU INSTRUCTIONS AND TRAPS
                                 ;       .         NOT INSTALLED.

161152  011500                           MOV     ($EF),R0         ;; ERROR BITS => R0<11:00>...
161154  001402                           BEQ     30003$           ;;...AND SKIP IF NONE.
161156  052700  040000                   BIS     #<4*BIT12>,R0    ;; ELSE, ADD TEST NUM => R0<15:12>...
161162  000207                   30003$: RETURN                   ;;...AND RETURN (NZ).
```

```
                                        .SBTTL  - T5    LINE CLOCK (BEVNT) INTERRUPT
                                        ;;*********************************************************************
                                        ;;* TEST 5 -- LINE CLOCK (BEVNT) INTERRUPT
                                        ;;*********************************************************************
  161164                                TST5:
  161164                                TCLK:
  161164  013746  000100                        MOV     BEVNT,-(SP)     ;; SAVE BEVNT.
  161170  013746  000102                        MOV     BEVNT+2,-(SP)
  161174  005015                                CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                        ;
                                        ; INPUT:  NONE.
                                        ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                        ;
                                        ; LINE CLOCK IS JUMPERED FROM CONSOLE DC319 OR Q-BUS BEVNT.
                                        ; VERIFY THAT WE CAN ENABLE/DISABLE THE CLOCK VIA CSRA<6>,
                                        ; AND THAT IT INTERRUPTS THRU VECTOR 100 AT PRI 6.
                                        ;
                                        ;       T5      LINE CLOCK (BEVNT) INTERRUPT
                                        ;       .         E0 = ROM IN VECTOR SPACE -- CAN'T RUN
                                        ;       .         E1 = CLOCK INTERRUPT NOT MASKED AT LEVEL 6
                                        ;       .         E2 = CLOCK DOESN'T INTERRUPT
                                        ;       .         E3 = CAN'T SHUT IT OFF

  161176  004737  160634                        CALL    RAMLO ,         ; VECTOR SPACE USABLE ??
  161202  001405                                BEQ     2$              ; PROCEED IF SO.
  161204  052715  000001                        BIS     #BIT0,($EF)     ;; E0 = ROM IN VECTOR SPACE -- CAN'T RUN
  161210  000444                                BR      TCXIT
                                                ;
  161212  005201                        1$:     INC     R1              ; ON "BEVNT", TICK AND RETURN.
  161214  000002                                RTI
                                        ;
  161216  106427  000300                2$:     MTPS    #PR6            ; RAISE CPU.
  161222  012737  161212  000100                MOV     #1$,BEVNT       ; SET CLOCK VECTOR.
  161230  012737  000300  000102                MOV     #PR6,BEVNT+2
  161236  005000                                CLR     R0              ; CLEAR LOOP CONTROL...
  161240  005001                                CLR     R1              ;...AND TICKER.
  161242  052737  000100  177520                BIS     #BIT6,$CSRA     ; TURN ON CLOCK INTERRUPT.
  161250  077001                                SOB     R0,.            ; DELAY, INTERRUPT SHOULD...
  161252  005701                                TST     R1              ;...BE MASKED AT THIS LEVEL.
  161254  001402                                BEQ     3$              ; BR IF SO.
  161256  052715  000002                        BIS     #BIT1,($EF)     ;; E1 = CLOCK INTERRUPT NOT MASKED AT LEVEL 6
  161262  106427  000240                3$:     MTPS    #PR5            ; NOW, LOWER CPU TO PRI 5.
  161266  077001                                SOB     R0,.            ; DELAY AGAIN, CLOCK SHOULD COME IN...
  161270  005301                                DEC     R1              ;...AND GIVE US A FEW TICKS (MORE THEN 1).
  161272  003002                                BGT     4$              ; BR IF SO.
  161274  052715  000004                        BIS     #BIT2,($EF)     ;; E2 = CLOCK DOESN'T INTERRUPT

  161300  005037  177520                4$:     CLR     $CSRA           ; TURN IT OFF.
  161304  000240                                NOP                     ;...FOR HARDWARE LOGIC PUPOSE...
  161306  005001                                CLR     R1
  161310  077001                                SOB     R0,.            ; DELAY ONCE MORE...
  161312  005701                                TST     R1              ;...CLOCK SHOULD BE SHUT OFF.
  161314  001402                                BEQ     6$              ; BR IF SO.
  161316  052715  000010                        BIS     #BIT3,($EF)     ;; E3 = CAN'T SHUT IT OFF
  161322                                6$:TCXIT:
  161322  012637  000102                        MOV     (SP)+,BEVNT+2   ;; RESTORE BEVNT.
  161326  012637  000100                        MOV     (SP)+,BEVNT
```

```
        161332  011500                             MOV     ($EF),R0        ;; ERROR BITS => R0<11:00>...
        161334  001402                             BEQ     30004$          ;;...AND SKIP IF NONE.
        161336  052700  050000                     BIS     #<5*BIT12>,R0   ;; ELSE, ADD TEST NUM => R0<15:12>...
        161342  000207                     30004$: RETURN                  ;;...AND RETURN (NZ).
```

```
                                        .SBTTL  - T6    CONSOLE SERIAL PORT DC319
                                        ;;*******************************************************************
                                        ;;* TEST 6 -- CONSOLE SERIAL PORT DC319
                                        ;;*******************************************************************
 161344                                 TST6:
 161344                                 TDC:
 161344  013746  000064                         MOV     SL1XV,-(SP)     ;; SAVE SL1XV.
 161350  013746  000066                         MOV     SL1XV+2,-(SP)
 161354  013746  000060                         MOV     SL1RV,-(SP)     ;; SAVE SL1RV.
 161360  013746  000062                         MOV     SL1RV+2,-(SP)
 161364  005015                                 CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                        ;
                                        ; INPUT:  NONE.
                                        ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                        ;
                                        ; *** REQUIRES EXTERNAL LOOP-BACK ***
                                        ;
                                        ; TEST THAT THE CONSOLE PORT WORKS AS ADVERTISED.
                                        ;
                                        ;       T6      CONSOLE SERIAL PORT DC319
                                        ;       .         E0 = ROM IN VECTOR SPACE -- INTERRUPTS NOT TESTED
                                        ;       .         E1 = XMTR INTERRUPT NOT MASKED AT LEVEL 4
                                        ;       .         E2 = XMTR INTERRUPT NOT RECEIVED
                                        ;       .         E3 = RCVR INTERRUPT NOT MASKED AT LEVEL 4
                                        ;       .         E4 = RCVR INTERRUPT NOT RECEIVED
                                        ;       .         E5 = RECEIVED DATA INCORRECT
                                        ;       .         E6 = NO RCVR DONE, LOOP-BACK OPEN

 161366  012704  177560                         MOV     #$SL1,R4        ; POINTER => R4.
 161372  004737  160634                         CALL    RAMLO           ; VECTOR SPACE USABLE ??
 161376  001403                                 BEQ     DC.XMT          ; PROCEED IF SO.
 161400  052715  000001                         BIS     #BIT0,($EF)     ;; E0 = ROM IN VECTOR SPACE -- INTERRUPTS NOT TESTED
 161404  000503                                 BR      DC.DAT

 161406  106427  000200                 DC.XMT: MTPS    #PR4            ; RAISE CPU TO XMTR LEVEL.
 161412  012737  161440  000064                 MOV     #2$,SL1XV       ; SET VECTOR.
 161420  012737  000200  000066                 MOV     #PR4,SL1XV+2
 161426  052764  000100  000004                 BIS     #100,4(R4)      ; SET INTERRUPT ENABLE.
 161434  000240                                 NOP                     ; INTERRUPT SHOULD BE HELD OFF...
 161436  000403                                 BR      3$              ;...BR IF SO.
 161440                                 2$:
 161440  052715  000002                         BIS     #BIT1,($EF)     ;; E1 = XMTR INTERRUPT NOT MASKED AT LEVEL 4
 161444  000411                                 BR      4$
 161446  012737  161470  000064         3$:     MOV     #4$,SL1XV       ;CHANGE THE VECTOR.
 161454  106427  000000                         MTPS    #PR0            ; LOWER CPU PRIORITY.
 161460  000240                                 NOP                     ; INTERRUPT SHOULD HAVE COME IN.
 161462  052715  000004                         BIS     #BIT2,($EF)     ;; E2 = XMTR INTERRUPT NOT RECEIVED
 161466  000401                                 BR      5$
 161470  022626                         4$:     CMP     (SP)+,(SP)+     ; ON INTERRUPT, FIX STACK.
 161472  042764  000100  000004         5$:     BIC     #100,4(R4)      ; CLEAR THE INT ENABLE.

 161500  016400  000002                 DC.RCV: MOV     2(R4),R0        ; ENSURE RCVR DONE IS CLEAR.
 161504  106427  000200                         MTPS    #PR4            ; RAISE CPU LEVEL.
 161510  005000                                 CLR     R0
 161512  110064  000006                         MOVB    R0,6(R4)        ;...TRANSMIT A NULL CHARACTER...
 161516  105714                         1$:     TSTB    (P4)            ;...WAIT 'TIL RCVR GETS IT...
 161520  100402                                 BMI     2$
```

```
161522  077003                           SOB     R0,1$             ;...(BUT DON'T WAIT FOREVER).
161524  000462                           BR      DC.LBO

161526  012737  161552  000060   2$:     MOV     #3$,SL1RV         ; OK, SET RCVR VECTOR.
161534  012737  000200  000062           MOV     #PR4,SL1RV+2
161542  052714  000100                   BIS     #100,(R4)         ; SET RCVR INTERRUPT ENABLE.
161546  000240                           NOP                       ; INTERRUPT SHOULD BE HELD OFF...
161550  000403                           BR      4$                ;...BR IF SO.
161552                           3$:
161552  052715  000010                   BIS     #BIT3,($EF)       ;; E3 = RCVR INTERRUPT NOT MASKED AT LEVEL 4
161556  000411                           BR      5$
161560  012737  161602  000060   4$:     MOV     #5$,SL1PV         ; CHANGE VECTOR...
161566  106427  000000                   MTPS    #PR0              ;...AND LOWER CPU PRIORITY.
161572  000240                           NOP                       ; INTERRUPT SHOULD COME IN.
161574  052715  000020                   BIS     #BIT4,($EF)       ;; E4 = RCVR INTERRUPT NOT RECEIVED
161600  000401                           BR      6$
161602  022626                   5$:     CMP     (SP)+,(SP)+       ; ON INTERRUPT, FIX STACK.
161604  042714  000100           6$:     BIC     #100,(R4)         ; CLEAR THE INT ENABLE...
161610  016400  000002                   MOV     2(R4),R0          ;...AND LOWER RCVR DONE FLAG.

161614  012700  163666           DC.DAT: MOV     #FLT10,R0         ; GET DATA TABLE POINTER.
161620  005001                           CLR     R1
161622  005002                           CLR     R2
161624  010164  000006                   MOV     R1,6(R4)          ; XMIT A NULL...
161630  105714                   10$:    TSTB    (R4)              ;...AND SEE IF LOOP IS INTACT.
161632  100410                           BMI     3$                ; IT IS, PROCEED AT 3$.
161634  077203                           SOB     R2,10$
161636  000415                           BR      DC.LBO            ; IT'S NOT, ABORT.

161640  112001                   1$:     MOVB    (R0)+,R1          ; NOW FLOAT DATA THRU THE LOOP...
161642  001415                           BEQ     DC.XIT            ;...UNTIL DONE.
161644  010164  000006                   MOV     R1,6(R4)          ; XMIT A BYTE...
161650  105714                   2$:     TSTB    (R4)              ;...AND WAIT.
161652  100376                           BPL     2$
161654  016402  000002           3$:     MOV     2(R4),R2          ; READ...
161660  120102                           CMPB    R1,R2             ;...AND VERIFY.
161662  001766                           BEQ     1$                ; LOOP IF OK.
161664  052715  000040                   BIS     #BIT5,($EF)       ;; E5 = RECEIVED DATA INCORRECT
161670  000402                           BR      DC.XIT
161672                           DC.LBO:
161672  052715  000100                   BIS     #BIT6,($EF)       ;; E6 = NO RCVR DONE, LOOP-BACK OPEN
161676                           DC.XIT:
161676  012637  000062                   MOV     (SP)+,SL1RV+2     ;; RESTORE SL1RV.
161702  012637  000060                   MOV     (SP)+,SL1RV
161706  012637  000066                   MOV     (SF)+,SL1XV+2     ;; RESTORE SL1XV.
161712  012637  000064                   MOV     (SP)+,SL1XV
161716  011500                           MOV     ($EF),R0          ;; ERROR BITS => R0<11:00>...
161720  001402                           BEQ     30005$            ;;...AND SKIP IF NONE.
161722  052700  060000                   BIS     #<6*BIT12>,R0     ;; ELSE, ADD TEST NUM => R0<15:12>...
161726  000207                   30005$: RETURN                    ;;...AND RETURN (NZ).
```

```
                                        .SBTTL  - T7    SECOND SERIAL PORT NEC7201
                                        ;;*****************************************************************
                                        ;;* TEST 7 -- SECOND SERIAL PORT NEC7201
                                        ;;*****************************************************************
 161730                                 TST7:
 161730                                 TNEC:
 161730  013746  000104                         MOV     PEVNT,-(SP)     ;; SAVE PEVNT.
 161734  013746  000106                         MOV     PEVNT+2,-(SP)
 161740  013746  000070                         MOV     NECV,-(SP)      ;; SAVE NECV.
 161744  013746  000072                         MOV     NECV+2,-(SP)
 161750  005015                                 CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                        ;
                                        ; INPUT:  R1<BIT0> = RUN CHAN A ONLY (INCLUDES DMA AND MODEM CONTROL).
                                        ;         R1<BIT1> = RUN CHAN B ONLY.
                                        ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                        ;
                                        ; NEC7201 SYNC/ASYNC (MULTI-PROTOCOL) SERIAL INTERFACE.
                                        ; INCLUDES AN INTEL 8254 TRIPLE PIT FOR BAUD RATE GENERATION.
                                        ;
                                        ; FIRST, SET UP THE BAUD GENERATOR, AND VERIFY THAT
                                        ; GENERAL PURPOSE TIMER 2 (800HZ) WORKS AS ADVERTISED.
                                        ;
                                        ;       T7      SECOND SERIAL PORT NEC7201
                                        ;       .         E0 = ROM IN VECTOR SPACE -- CAN'T RUN
                                        ;       .         E1 = I8254 TIMER 2 (800HZ) DOESN'T INTERRUPT
                                        ;       .         E2 = ASYNC MODE, DATA XFER INCOMPLETE
                                        ;       .         E3 = SYNC MODE, EOF-SDLC NOT RECEIVED
                                        ;       .         E4 = SYNC MODE, DATA XFER INCOMPLETE
                                        ;       .         E5 = SYNC/ASYNC MODES, RECEIVED DATA INCORRECT

 161752  004737  160634                         CALL    RAMLO           ; VECTOR SPACE USABLE ??
 161756  001404                                 BEQ     1$              ; PROCEED IF SO.
 161760  052715  000001                         BIS     #BIT0,($EF)     ;; E0 = ROM IN VECTOR SPACE -- CAN'T RUN
 161764  000137  163470                         JMP     NECXIT

 161770  012704  175736                 1$:     MOV     #$SL2+36,R4     ; GET I8254 CONTROL REG ADDRESS.
 161774  106427  000300                         MTPS    #PR6            ; RAISE CPU.
 162000  112714  000066                         MOVB    #066,(R4)       ; CHAN A <TIMER-0!2-BYTES!MODE-3!BIN>.
 162004  112714  000166                         MOVB    #166,(R4)       ; CHAN B <TIMER-1!      SAME         >.
 162010  112714  000264                         MOVB    #264,(R4)       ; 800HZ  <TIMER-2! SAME EXCEPT MODE-2>.
 162014  013744  162124                         MOV     HZ800,-(R4)     ; SET 800HZ DIVIDER == 50HZ
 162020  113714  162125                         MOVB    HZ800+1,(R4)
 162024  013744  162126                         MOV     A4800,-(R4)     ; SET CHAN B...
 162030  113714  162127                         MOVB    A4800+1,(R4)
 162034  013744  162126                         MOV     A4800,-(R4)     ;...AND CHAN A FOR ASYNC 4800 (SYNC 76.8K).
 162040  113714  162127                         MOVB    A4800+1,(R4)

 162044  012737  162120  000104                 MOV     #3$,PEVNT       ; SET PEVNT VECTOR...
 162052  012737  000300  000106                 MOV     #PR6,PEVNT+2
 162060  052737  000200  177520                 BIS     #BIT7,$CSRA     ;...AND ENABLE IT.
 162066  005002                                 CLR     R2
 162070  005000                                 CLR     R0
 162072  106400                                 MTPS    R0              ; LOWER CPU.
 162074  077001                                 SOB     R0,.            ; DELAY, PEVNT SHOULD COME IN...
 162076  005302                                 DEC     R2              ;...AND GIVE A FEW TICKS (MORE THAN 1).
 162100  003002                                 BGT     2$
 162102  052715  000002                         BIS     #BIT1,($EF)     ;; E1 = I8254 TIMER 2 (800HZ) DOESN'T INTERRUPT
```

```
162106  005037  177520          2$:     CLR     $CSPA           ; TURN OFF PEVNT...
162112  012714  000260                  MOV     #260,(R4)       ;...AND TIMER 2 (MODE-0).
162116  000434                          BR      NEC.DATA        ;...AND MOVE ON.

162120  005202                  3$:     INC     R2              ; ON "PEVNT", TICK AND RETURN.
162122  000002                          RTI

162124  000020                  HZ800:  16.             ; DIVIDER YIELDS == 50HZ
162126  000200                  A4800:  128.            ; DIVIDER YIELDS ASYNC 4800 AND/OR SYNC 76.8K.
```

```
                                         ;
                                         ; FIVE PASS SYNC/ASYNC DATA TEST.
                                         ;
                                         ; *** REQUIRES EXTERNAL LOOP-BACKS (2) ***
                                         ;
                                         ;       1. CHAN B, ASYNC (4800 BAUD).
                                         ;       2. CHAN A, ASYNC (4800 BAUD).
                                         ;       3. CHAN B, SYNC (76.8K BAUD).
                                         ;       4. CHAN A, SYNC (76.8K BAUD).
                                         ;       5. CHAN A, SYNC, DMA DRIVEN.
                                         ;
          000003                         CHA=    %3               ; DEDICATE R3 AS "CHAN-A" AND...
          000004                         CHX=    %4               ;...R4 AS "CHAN-IN-TEST" PORT POINTERS.
          177774                         STAT=   -4               ; INDICES TO CHAN A/B STATUS...
          177776                         RDB=    -2               ;...RCVR DATA...
          000000                         CTRL=    0               ;...CONTROL...
          000002                         XDB=     2               ;...AND XMTR DATA PORTS.
          170002                         KAT=    TMP1             ; EQUATE A KEEP-ALIVE-TIMER...
          170004                         MODE=   TMP2             ;...SET-UP/MODE FLAG...
          170006                         STATS=  TMP3             ;...STATUS BYTES...
          170010                         INHIB=  TMP4             ;...AND INHIBIT (A/B) FLAG IN RAM.
                                         ;
                                         ; ASYNC SET-UP   R  CODE          FUNCTION
                                         ;                -  ----          --------
 162130      000       030               A.SET:  .BYTE   0, 030  ; COMMAND        <CHANNEL RESET>.
 162132      002       024                       .BYTE   2, 024  ; BUS INTERFACE <NON-VECTORED!PRI1!NO-DMA>.
 162134      004       104                       .BYTE   4, 104  ; PROTOCOL       <X16!ASYNC-1-STOP!NO-PARITY>.
 162136      003       301                       .BYTE   3, 301  ; RCVR CONTROL   <8-BPC!RX-ENAB>.
 162140      005       152                       .BYTE   5, 152  ; XMTR CONTROL   <8-BPC!TX-ENAB!PRTYX-B>.
 162142      000       020                       .BYTE   0, 020  ; COMMAND        <RESET-EXT-INTS>.
 162144      001       036                       .BYTE   1, 036  ; INT CONTROL    <RXI-ON-ALL!CAV!TX-IE!NO-EXT-IE>.
 162146   177777                                 -1
                                         ;
                                         ; SYNC SET-UP    R  CODE          FUNCTION
                                         ;                -  ----          --------
 162150      000       030               S.SET:  .BYTE   0, 030  ; COMAND         <CHANNEL RESET>.
 162152      002       024                       .BYTE   2, 024  ; BUS INTERFACE <NON-VECTORED!PRI1!NO-DMA>.
 162154      004       040                       .BYTE   4, 040  ; PROTOCOL       <X1!SDLC!SYNC!NO-PARITY>.
 162156      006       000                       .BYTE   6, 000  ; SYNC 1         <SEARCH ADDRESS -- UNUSED>.
 162160      007       176                       .BYTE   7, 176  ; SYNC 2         <FLAG CODE>.
 162162      003       311                       .BYTE   3, 311  ; RCVP CONTROL   <8-BPC!CRC-ENAB!RX-ENAB>.
 162164      005       153                       .BYTE   5, 153  ; XMTR CONTROL   <8-BPC!TX-ENAB!CCITT!PRTYX-B!CRC-ENAB>.
 162166      000       200                       .BYTE   0, 200  ; COMMAND        <RESET TX-CRC-GEN>.
 162170      000       100                       .BYTE   0, 100  ; COMMAND        <RESET RX-CRC-GEN>.
 162172      000       020                       .BYTE   0, 020  ; COMMAND        <RESET EXT-INT>.
 162174      001       036                       .BYTE   1, 036  ; INT CONTROL    <RXI-ON-ALL!CAV!TX-IE!NO-EXT-IE>.
 162176   177777                                 -1
 162200      001       004               C.CLR:  .BYTE   1, 004  ; INT CONTROL    <KEEP-CAV!DISABLE-ALL-ELSF>
 162202      003       300                       .BYTE   3, 300  ; RCVR           <RX-DISABLE>.
 162204      005       142                       .BYTE   5, 142  ; XMTP           <TX-DISABLE>.
 162206   177777                                 -1
```

```
                                   ;
                                   ; FIRST, SEE IF EITHER CHAN IS INHIBITED.
                                   ;
162210  005000                     NEC.DATA:  CLR    R0
162212  032701   000003                       BIT    #BIT1!BIT0,R1     ; SINGLE CHAN SELECTED ??
162216  001407                                BEQ    1$                ; PROCEED IF NOT (RUN BOTH).
162220  012700   175714                       MOV    #$SSL2+14,R0      ; YES, ASSUME RUN A, INHIBIT B...
162224  032701   000001                       BIT    #BIT0,R1
162230  001002                                BNE    1$                ;...AND SKIP IF WE GUESSED RIGHT.
162232  012700   175704                       MOV    #$SSL2+4,R0       ; ELSE, SET TO INHIBIT A.
162236  010065   170010            1$:        MOV    R0,INHIB($TOP)    ; SET/CLEAR CHAN INHIBIT FLAG.
                                   ;
                                   ; 1ST 4 PASSES.
                                   ;
162242  012700   162130            NEC.ASYNC: MOV    #A.SET,R0         ; ASYNC MODE FIRST...
162246  000405                                BR     NEC.AA
162250  012700   162150            NEC.SYNC:  MOV    #S.SET,R0         ;...THEN, SYNC (SDLC) MODE...
162254  052737   000004   177520              BIS    #4,SCSRA          ;...REQUIRES <SYNCMA!-SYNCMB> IN CSRA.
162262  010065   170004            NEC.AA:    MOV    R0,MODE($TOP)     ; SAVE POINTER AS SYNC/ASYNC FLAG.
162266  012703   175704                       MOV    #$SSL2+4,CHA      ; SET CHAN-A CONTROL PORT POINTER.
162272  012704   175714                       MOV    #$SSL2+14,CHX     ; SET CHAN-B AS CURRENT CHAN-IN-TEST.
162276  111013                                MOVB   (R0),(CHA)
162300  116013   000001                       MOVB   1(R0),(CHA)       ; COMMAND <CHAN-A-RESET (INCLUDES INT LOGIC)>.
162304  000240                                NOP                      ;       WAIT A MOMENT.
162306  112014                                MOVB   (R0)+,(CHX)
162310  112014                                MOVB   (R0)+,(CHX)       ; COMMAND <CHAN-B-RESET>.
162312  000240                                NOP                      ;       WAIT AGAIN.
162314  106427   000200            1$:        MTPS   #PR4              ; RAISE CPU.
162320  026504   170010                       CMP    INHIB($TOP),CHX ; THIS CHANNEL INHIBITED ??
162324  001511                                BEQ    5$                ; BR IF SO.
162326  112013                                MOVB   (R0)+,(CHA)       ; ELSE, POINT TO...
162330  112013                                MOVB   (R0)+,(CHA)       ;...AND SET BUS-INTERFACE (CR2A ALWAYS).
162332  112014                     2$:        MOVB   (R0)+,(CHX)       ; POINT TO...
162334  112014                                MOVB   (R0)+,(CHX)       ;...AND SET-UP THE REST.
162336  105710                                TSTB   (R0)
162340  100374                                BPL    2$
162342  012701   163666                       MOV    #FLT10,R1         ; XMT BUFFER POINTER => R1.
162346  010502                                MOV    $TOP,R2
162350  062702   150002                       ADD    #BUFR1,R2         ; RCV BUFFER POINTER => R2.
162354  010246                                MOV    R2,-(SP)          ; SAVE A COPY OF R2.
162356  012700   000020                       MOV    #16.,R0
162362  005022                     3$:        CLR    (R2)+             ; CLEAR IT...
162364  077002                                SOB    R0,3$
162366  011602                                MOV    (SP),R2           ;...AND RESET POINTER.
162370  005065   170006                       CLR    STATS($TOP)       ; CLEAR SAVED STATUS...
162374  005065   170002                       CLR    KAT($TOP)         ;...AND KEEP-ALIVE-TIMER.
162400  012737   163522   000070              MOV    #NEC.I,NECV       ; SET VECTOR...
162406  012737   000200   000072              MOV    #PR4,NECV+2
162414  106427   000000                       MTPS   #PR0              ;...AND LOWER CPU.
162420  012700   000001                       MOV    #1,R0             ; INIT BYTE COUNTS (RCV=0, XMT=1)...
162424  112164   000002                       MOVB   (R1)+,XDB(CHX)    ;...XMIT 1ST BYTE...
162430  112714   000300                       MOVB   #<3*BIT6>,(CHX) ;...AND COMMAND <RESET-IDLE/CRC-LATCH>.
162434  026527   170004   162150              CMP    MODE($TOP),#S.SET ; SYNC MODE ??
162442  001411                                BEQ    41$               ; BR IF SO.

162444  020027   011022            4$:        CMP    R0,#<18.*BIT8>!18. ; ASYNC -- 18 BYTES TRANSFERRED ??
162450  001423                                BEQ    43$               ; BR IF SO.
```

```
162452  005365  170002                  DEC     KAT($TOP)
162456  001372                          BNE     4$                 ; ELSE, HANG-AROUND.
162460  052715  000004                  BIS     #BIT2,($EF)        ;; E2 = ASYNC MODE, DATA XFER INCOMPLETE
162464  000415                          BR      43$

162466  005765  170006          41$:    TST     STATS($TOP)        ; SYNC -- <EOF> RECEIVED ??
162472  100405                          BMI     42$                ; BR IF SO.
162474  005365  170002                  DEC     KAT($TOP)
162500  001372                          BNE     41$                ; ELSE, SAME-OLE-CRAP.
162502  052715  000010                  BIS     #BIT3,($EF)        ;; E3 = SYNC MODE, EOF-SDLC NOT RECEIVED

162506  020027  012022          42$:    CMP     R0,#<20.*BIT8!18.> ; BYTE COUNTS RIGHT ??
162512  001402                          BEQ     43$                ; PROCEED IF SO.
162514  052715  000020                  BIS     #BIT4,($EF)        ;; E4 = SYNC MODE, DATA XFER INCOMPLETE

162520  010065  170002          43$:    MOV     R0,KAT($TOP)       ; *** DEBUG SAVE FINAL BYTE COUNTS.
162524  012602                          MOV     (SP)+,R2           ; GET RCV'D...
162526  012701  163666                  MOV     #FLT10,R1          ;...AND XMT'D POINTERS.
162532  012700  000022                  MOV     #18.,R0
162536  122221                  44$:    CMPB    (R2)+,(R1)+        ; RCV'D SAME AS THAT XMT'D ??
162540  001402                          BEQ     45$                ; BR IF SO.
162542  052715  000040                  BIS     #BIT5,($EF)        ;; E5 = SYNC/ASYNC MODES, RECEIVED DATA INCORRECT
162546  077005                  45$:    SOB     R0,44$

162550  012700  162200          5$:     MOV     #C.CLR,R0          ; DONE, CLEAN-UP TABLE POINTER => R0.
162554  112014                  6$:     MOVB    (R0)+,(CHX)        ; POINT...
162556  112014                          MOVB    (R0)+,(CHX)        ;...AND CLEAR (RESET) AS NECESSARY.
162560  105710                          TSTB    (R0)
162562  100374                          BPL     6$
162564  020403                          CMP     CHX,CHA            ; BOTH CHANNELS DONE ??
162566  001405                          BEQ     7$                 ; BR IF SO.
162570  010304                          MOV     CHA,CHX            ; ELSE, POINT TO CHAN A...
162572  016500  170004                  MOV     MODE($TOP),R0      ;...GET SET-UP POINTER => R0...
162576  005720                          TST     (R0)+              ;...BUMP PAST THE CHAN-RESET...
162600  000645                          BR      1$                 ;...AND GO 'ROUND ONCE.

162602  026527  170004  162150  7$:     CMP     MODE($TOP),#S.SET  ; SYNC MODE DONE ??
162610  001217                          BNE     NEC.SYNC           ; NO, DO IT NOW.

162612  026503  170010                  CMP     INHIB($TOP),CHA    ; CHAN A INHIBITED ??
162616  001002                          BNE     NEC.DMA            ; PROCEED TO DMA AND MODEM TESTS IF NOT...
162620  000137  163470                  JMP     NECXIT             ;...ELSE, QUIT HERE.
```

```
                                ;
                                ; 5TH PASS USES DMA (ASSUMING THAT 8016 IS ALIVE AND WELL).
                                ; THE 7201 CHIP IS STILL SET-UP FOR "CHAN A SYNC" MODE.
                                ;
162624  106427  000200          NEC.DMA: MTPS   #PR4
162630  012701  162650                  MOV     #2$,R1          ; ADJUST CHAN A SET-UP.
162634  012702  000004                  MOV     #4,R2
162640  112113                  1$:     MOVB    (R1)+,(CHA)     ; POINT TO...
162642  112113                          MOVB    (R1)+,(CHA)     ;...AND CHANGE...
162644  077203                          SOB     R2,1$           ;...THE FOLLOWING 4 REGISTERS.
162646  000404                          BR      3$
162650     002     025          2$:     .BYTE   2, 025          ; CHANGE BUS-INTERFACE TO <CHAN-A-DMA>
162652     003     311                  .BYTE   3, 311          ; ENABLE RECVR...
162654     005     151                  .BYTE   5, 151          ;...AND XMITTR.
162656     001     016                  .BYTE   1, 016          ; CHANGE INT CONTROL TO <RXI-ON-1ST-CHAR>

                                ;DEFINE THE DMA CHIP REGISTERS THAT ARE USED BELOW.
        000070                  MMR=    70              ; MASTER MODE REGISTER.
        000056                  STAT1=  56              ; CHANNEL 1 STATUS.
        000054                  CMDR=   54              ;...AND COMMAND REGISTER (FOR BOTH).
        000046                  CHA1H=  46              ; CHAIN ADDRESS REGISTERS HIGH (SEG/TAG)...
        000044                  CHA2H=  44
        000042                  CHA1L=  42              ;...AND THE LOW (OFFSET) HALFS.
        000040                  CHA2L=  40
        000001                  TC=     1               ; TERMINAL COUNT STATUS BIT (BIT0).
        020000                  IP=     20000           ; INTERRUPT POSTED (DONE) STATUS BIT (BIT13).

162660  010500                  3$:     MOV     $TOP,R0
162662  062700  150002                  ADD     #BUFR1,R0       ; GET BUFFER POINTER.
162666  012701  000020                  MOV     #16.,R1
162672  005020                  35$:    CLR     (R0)+           ; CLEAR RECEIVING BUFFER.
162674  077102                          SOB     R1,35$
162676  012737  163632  000070          MOV     #IXIT+2,NECV    ; CHANGE THE VECTOR (INTS DISMISSED).
162704  012704  174400                  MOV     #$DMA,R4        ; SET-UP THE DMA ENGINE.
162710  005064  000054                  CLR     CMDR(R4)        ; CHIP RESET.
162714  012764  000115  000070          MOV     #115,MMR(R4)    ; SET MASTER MODE <VI!WAIT!CPINTLV!ENAB>.
162722  012764  000074  000054          MOV     #40!34,CMDR(R4) ; CLEAR IE'S...
162730  012764  000075  000054          MOV     #41!34,CMDR(R4) ;...BOTH CHANNELS.
162736  005064  000046                  CLR     CHA1H(R4)       ; CLEAR HI CHAIN ADDRESS...
162742  005064  000044                  CLR     CHA2H(R4)       ;...BOTH CHANNELS.
162746  012764  163004  000040          MOV     #4$,CHA2L(R4)   ; SET CHAN 2 (XMT)...
162754  012764  163024  000042          MOV     #5$,CHA1L(R4)   ;...AND CHAN 1 (RCV) CHAIN ADDRESSES.
162762  012764  000240  000054          MOV     #240,CMDR(R4)   ; CHAIN-LOAD CHAN 1 (RCVR)...
162770  060564  000002                  ADD     $TOP,2(R4)      ;...AND ADJUST BUFR1 ADDRESS.
162774  012764  000241  000054          MOV     #241,CMDR(R4)   ; CHAIN-LOAD AND START CHAN 2 (XMTR).
163002  000420                          BR      6$
163004  001602                  4$:     1602                    ; CHAIN-LOAD ARA, ARB, OPK, AND CH-MODE.
163006  000000  163666                  0, FLT10                ; FROM -- DATA TABLE.
163012  000020  175707                  20, 175706+1            ; TO ---- 7201 XDB(LB), NO AUTO-INCR.
163016  000022                          18.                     ; OPK --- 18 BYTES.
163020  000020  001201                  20, 1201                ; MODE -- SFT-REQ,IP-ON-TC!EOP,SNGL,BYTE-BYTE.
163024  001602                  5$:     1602
163026  000020  175703                  20, 175702+1            ; FROM -- 7201 RDB(LB), NO AUTO-INCR.
163032  000000  150002                  0, BUFR1                ; TO ---- LOCAL BUFR1.
163036  000024                          20.                     ; OPK --- 20 BYTES (18 DATA + 2 CRC).
163040  000000  001201                  0, 1201                 ; MODE -- SAME AS ABOVE (EXCEPT NO SFT-REQ).

```

```
 163044  112713  000300          6$:      MOVB    #<3*BIT6>,(CHA) ; COMMAND <RESET-IDLE/CRC-LATCH>...
 163050  005000                           CLR     R0
 163052  106400                           MTPS    R0              ;...LOWER CPU.
 163054  032764  020000  000056  7$:      BIT     #IP,STAT1(R4)   ;...AND WAIT FOR CHAN1 (RCVR) DONE.
 163062  001002                           BNE     8$
 163064  077005                           SOB     R0,7$
 163066  000404                           BR      9$
 163070  032764  000001  000056  8$:      BIT     #TC,STAT1(R4)   ; GOT <IP>, SHOULD ALSO HAVE <TC>.
 163076  001002                           BNE     11$             ; BR IF SO (ALL RECEIVED).
 163100                          9$:
 163100  052715  000100                   BIS     #BIT6,($EF)     ;; E6 = DMA MODE, DATA XFER INCOMPLETE

 163104  012700  163666          11$:     MOV     #FLT10,R0
 163110  010501                           MOV     $TOP,R1
 163112  062701  150002                   ADD     #BUFR1,R1
 163116  012702  000022                   MOV     #18.,R2
 163122  122021                  12$:     CMPB    (R0)+,(R1)+     ; CHECK DATA.
 163124  001402                           BEQ     13$             ; BR IF OK.
 163126  052715  000200                   BIS     #BIT7,($EF)     ;; E7 = DMA MODE RECEIVED DATA INCORRECT
 163132  077205                  13$:     SOB     R2,12$          ; LOOP.

 163134  005037  177520                   CLR     $CSRA           ; ALL DONE, CLEAR CSRA...
 163140  005064  000054                   CLR     CMDR(R4)        ;...RESET DMA CHIP AND FALL THRU.
```

- T7   SECOND SERIAL PORT NEC7201

```
 1                                   ;
 2                                   ; FINALLY, TEST ALL THE MODEM CONTROL FUNCTIONS.
 3                                   ;
 4         000004                    CHB=     %4              ; REDEFINE R4 AS CHAN B POINTER.
 9                                   ;
10 163144  106427  000200            NEC.MDM: MTPS    #PR4             ; RAISE CPU.
11 163150  010304                             MOV     CHA,CHB
12 163152  062704  000010                     ADD     #10,CHB          ; SET CHAN B POINTER.
13 163156  112713  000030            1$:      MOVB    #<3*BIT3>,(CHA) ; <CHAN-RESET>.
14 163162  112714  000030                     MOVB    #<3*BIT3>,(CHB)
15 163166  012700  000031                     MOV     #100*2/5,R0      ;;DELAY 100 USEC.
   163172  077001                             SOB     R0,.
16 163174  112713  000021                     MOVB    #<2*BIT3>!1,(CHA) ; <RESET-EXT>, POINT TO CR1...
17 163200  112714  000021                     MOVB    #<2*BIT3>!1,(CHB)
18 163204  112713  000005                     MOVB    #5,(CHA)         ;...AND SET <CAV!INT-ON-EXT>.
19 163210  112714  000005                     MOVB    #5,(CHB)
20 163214  012737  163640  000070             MOV     #NEC.EX,NECV     ; SET THE VECTOR.
21 163222  005001                             CLR     R1               ; CLEAR A SPACE FOR SRO(A)...
22 163224  005002                             CLR     R2               ;...AND SRO(B).
23                                   ;
24                                   ; TEST REQ-TO-SEND, CLEAR-TO-SEND, AND RCVR-READY BITS.
25                                   ;
26 163226  005000                    3$:      CLR     R0    ,          ; KEEP-ALIVE TIMER.
27 163230  106400                             MTPS    R0               ; LOWER CPU.
28 163232  112713  000005                     MOVB    #5,(CHA)         ; POINT TO CR5(A)...
29 163236  112713  000002                     MOVB    #2,(CHA)         ;...AND SET <REQ-TO-SEND>.
30 163242  105701                    4$:      TSTB    R1               ; SHOULD GET SRO(A) STATUS INTERRUPT.
31 163244  001002                             BNE     5$               ; BR IF SO.
32 163246  077003                             SOB     R0,4$
33 163250  000406                             BR      6$               ; INTERRUPT NOT RECEIVED.
34 163252  032701  000040            5$:      BIT     #BIT5,R1         ; SHOULD HAVE <CLR-TO-SEND>...
35 163256  001403                             BEQ     6$
36 163260  032701  000010                     BIT     #BIT3,R1         ;...AND <RCVR-READY>.
37 163264  001002                             BNE     7$
38 163266                            6$:
   163266  052715  000400                     BIS     #BIT8,($EF)      ;; E8 = STATUS WRONG OR NO INTERRUPT WITH REQ-TO-SEND SET
39                                   ;
40                                   ; NEXT, TEST TT108/2 (IN CSRA) AND DATA-MODE BITS.
41                                   ;
42 163272  005002                    7$:      CLR     R2
43 163274  112737  000010  177520             MOVB    #BIT3,$CSRA      ; SET <TT108/2>.
44 163302  105702                    8$:      TSTB    R2               ; SHOULD GET SRO(B) STATUS INTERRUPT.
45 163304  001002                             BNE     9$               ; BR IF SO.
46 163306  077003                             SOB     R0,8$
47 163310  000406                             BR      10$              ; INTERRUPT NOT RECEIVED.
48 163312  032702  000010            9$:      BIT     #BIT3,R2         ; SHOULD HAVE <DATA-MODE>...
49 163316  001403                             BEQ     10$
50 163320  032702  000040                     BIT     #BIT5,R2         ;...AND NOT <INCOMING-CALL>.
51 163324  001402                             BEQ     11$
52 163326                            10$:
   163326  052715  001000                     BIS     #BIT9,($EF)      ;; E9 = STATUS WRONG OR NO INTERRUPT WITH TT108/2 SET
53                                   ;
54                                   ; NEXT, TEST TERM-IN-SERV (CSRA), TT142 (CSRB), AND INCOMING-CALL BITS.
55                                   ;
56 163332  005002                    11$:     CLR     R2
57 163334  052737  000020  177520             BIS     #BIT4,$CSRA      ; SET <TERM-IN-SERV>.
58 163342  105702                    12$:     TSTB    R2               ; SHOULD GET ANOTHER SRO(B) CHANGE.
```

```
   163344  001002                           BNE     13$
   163346  077003                           SOB     R0,12$
   163350  000407                           BR      14$                ; INTERRUPT NOT RECEIVED.
   163352  032737  000001  177522  13$:     BIT     #BIT0,$CSRB        ; SHOULD HAVE <TT142> IN CSRB...
   163360  001403                           BEQ     14$
   163362  032702  000040                   BIT     #BIT5,R2           ;...AND <INCOMING-CALL>.
   163366  001002                           BNE     15$
   163370                          14$:
   163370  052715  002000                   BIS     #BIT10,($EF)       ;; E10 = STATUS WRONG OR NO INTERRUPT WITH TERM-IN-SERV SET
                                   ;
                                   ; FINALLY, TURN ALL THAT STUFF OFF !!!!!
                                   ;
   163374  042737  000010  177520  15$:     BIC     #BIT3,$CSRA        ; CLEAR <TT108/2>
   163402  012700  000031                   MOV     #100*2/5,R0        ;;DELAY 100 USEC.
   163406  077001                           SOB     R0,.
   163410  005037  177520                   CLR     $CSRA              ; CLEAR <TERM-IN-SERV>.
   163414  012700  000031                   MOV     #100*2/5,R0        ;;DELAY 100 USEC.
   163420  077001                           SOB     R0,.
   163422  112713  000005                   MOVB    #5,(CHA)           ; POINT TO CR5(A)...
   163426  112713  000000                   MOVB    #0,(CHA)           ;...AND CLEAR <REQ-TO-SEND>.
   163432  012700  000031                   MOV     #100*2/5,R0        ;;DELAY 100 USEC.
   163436  077001                           SOB     R0,.
   163440  032701  000050                   BIT     #BIT5!BIT3,R1      ; <CS!RR> IN SR0(A) SHOULD BE CLEAR...
   163444  001003                           BNE     16$
   163446  032702  000050                   BIT     #BIT5!BIT3,R2      ;...AND <IC!DM> IN SR0(B) AS WELL.
   163452  001402                           BEQ     17$
   163454                          16$:
   163454  052715  004000                   BIS     #BIT11,($EF)       ;; E11 = STATUS WRONG WITH EVERYTHING OFF

   163460  112713  000030          17$:     MOVB    #<3*BIT3>,(CHA) ; ALL DONE, RESET THE CHIP.
   163464  112714  000030                   MOVB    #<3*BIT3>,(CHB)
   163470                          NECXIT:
   163470  012637  000072                   MOV     (SP)+,NECV+2       ;; RESTORE NECV.
   163474  012637  000070                   MOV     (SP)+,NECV
   163500  012637  000106                   MOV     (SP)+,PEVNT+2      ;; RESTORE PEVNT.
   163504  012637  000104                   MOV     (SP)+,PEVNT
   163510  011500                           MOV     ($EF),R0           ;; ERROR BITS => R0<11:00>...
   163512  001402                           BEQ     30006$             ;;...AND SKIP IF NONE.
   163514  052700  070000                   BIS     #<7*BIT12>,R0      ;; ELSE, ADD TEST NUM => R0<15:12>...
   163520  000207                  30006$:  RETURN                     ;;...AND RETURN (NZ).
```

```
                                ;
                                ; SYNC/ASYNC DATA TEST INTERRUPT HANDLER.
                                ; BYTE COUNTS ARE MAINTAINED IN RO (LO=XMT, HI=RCV).
                                ;
163522  112763  000002  000010  NEC.I:  MOVB    #2,10(CHA)      ; POINT TO SR2B...
163530  116346  000004                  MOVB    STAT+10(CHA),-(SP) ;...AND PUSH VECT/COND.
163534  006016                          ROR     (SP)            ; BIT 0 => "C"...
163536  103420                          BCS     ISEXT           ;...AND BR IF EXT/SPECIAL (1, 3, 5, OR 7)
163540  006016                          ROR     (SP)            ; BIT 1 => "C"...
163542  103411                          BCS     IRX             ;...AND BR IF RCVR (2 OR 6)...
                                                                ;...ELSE FALL THRU (0 OR 4).

163544  105711                  ITX:    TSTB    (R1)            ; XMIT -- ANYTHING LEFT ??
163546  001404                          BEQ     1$              ; BR IF NOT.
163550  112164  000002                  MOVB    (R1)+,XDB(CHX)  ; SEND NEXT BYTE...
163554  005200                          INC     R0              ;...AND COUNT IT.
163556  000424                          BR      IXIT
163560  112714  000050          1$:     MOVB    #<5*BIT3>,(CHX) ; XMIT DONE, COMMAND <RESET-TXIP>...
163564  000421                          BR      IXIT            ;...AND RETURN

163566  116422  177776          IRX:    MOVB    RDB(CHX),(R2)+  ; RCVR -- STUFF THE CHAR...
163572  062700  000400                  ADD     #<1*BIT8>,R0    ;...AND COUNT IT.
163576  000414                          BR      IXIT

163600  116465  177774  170006  ISEXT:  MOVB    STAT(CHX),STATS($TOP) ; EXTERNAL/SPECIAL, GET SR0...
163606  112714  000001                  MOVB    #1,(CHX)
163612  116465  177774  170007          MOVB    STAT(CHX),STATS+1($TOP) ;...AND SR1.
163620  112714  000060          1$:     MOVB    #<6*BIT3>,(CHX) ; COMMAND <RESET-ERROR>...
163624  112714  000020                  MOVB    #<2*BIT3>,(CHX) ;...AND <RESET-EXTERNAL>.

163630  005726                  IXIT:   TST     (SP)+           ; FIX STACK.
163632  112713  000070                  MOVB    #<7*BIT3>,(CHA) ; SIGNAL <END-OF-INTERRUPT>.
163636  000002                          RTI                     ;...AND RETURN.
                                ;
                                ; MODEM CONTROL TEST INTERRUPT HANDLER.
                                ;
163640  112713  000020          NEC.EX: MOVB    #<2*BIT3>,(CHA) ; COMMAND <RESET-EXTERNAL>.
163644  112714  000020                  MOVB    #<2*BIT3>,(CHB) ;       DITTO.
163650  116301  177774                  MOVB    STAT(CHA),R1    ; GET SR0(A)...
163654  116402  177774                  MOVB    STAT(CHB),R2    ;...AND SR0(B).
163660  112713  000070                  MOVB    #<7*BIT3>,(CHA) ; SIGNAL <END-OF-INTERRUPT>.
163664  000002                          RTI
                                ;
                                ; FLOATING 1 AND 0 DATA TABLE.
                                ;
163666     015     012          FLT10:  .BYTE   15, 12          ; SYNC PATR.
163670     001     002     004  FLT1:   .BYTE   001, 002, 004, 010, 020, 040, 100, 200  ; FLOATING 1.
163700     376     375     373  FLT0:   .BYTE   376, 375, 373, 367, 357, 337, 277, 177  ; FLOATING 0.
163710  000000                          .WORD   0               ; TERMINATOR.

                                        .DSABL  AMA
```

```
        .SBTTL  LBTEST - LOOP BACK TESTS

        .ENABLE GBL,LC
;
; Module name: LOOP BACK TEST - LBTEST
;
; System: KXT11-CA Native Firmware
;
;
;
;
; Functional Description:
;
; This module controls the execution of the loop back tests.
; The tests are:
;
;       SLU1
;       SLU2A
;       SLU2B
;       PIO
;       Power Up Tests
;
;
;
; Input Parameters:
;
;       None
;
; Output Parameters:
;
;       None
;
;
; Routines Used:
;
;       1) Execute Test (EXTEST)
;       2) Power Up Tests (PUTEST)


```

```
                                                                  ;* PROCEDURE CONTINUOUS SELF TEST
163712                          LBTEST::
                                                                  ;* * DO UNTIL POWER DOWN
163712                          LB100:
                                                                  ;* * * SET LEDS TO LOOP BACK TEST DISPLAY
163712  012700  000006                  MOV     #CSTMD,R0
163716  004767  003256                  CALL    SETLED
                                                                  ;* * * TEST SLU1 LOOP BACK
163722  012702  000006                  MOV     #6,R2
163726  004767  006636                  CALL    EXTEST
                                                                  ;* * * TEST SLU2 CHANNEL A
163732  012701  000001                  MOV     #BIT0,R1
163736  012702  000007                  MOV     #7,R2
163742  004767  006622                  CALL    EXTEST
                                                                  ;* * * TEST SLU2 CHANNEL B
163746  012701  000002                  MOV     #BIT1,R1
163752  012702  000007                  MOV     #7,R2
163756  004767  006606                  CALL    EXTEST
                                                                  ;* * * TEST PIO
163762  012702  000010                  MOV     #10,R2
163766  004767  006576                  CALL    EXTEST
                                                                  ;* * * PERFORM POWER UP SELF TESTS
163772  004767  006326                  CALL    PUTEST
                                                                  ;* * END UNTIL
163776  000745                          BR      LB100
                                                                  ;* END CONTINUOUS SELF TEST
```

```
                                        .SBTTL  RESTRT - RESTART HANDLER
                                        .ENABLE LC,GBL
        164000                          . = 164000

                                ;
                                ; Module name: RESTRT - RESTART
                                ;
                                ; System: KXT11-CA Native Firmware
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ; Functional Description:
                                ;
                                ;This module handles the restart interrupt at location 173004.
                                ;
                                ;Input Parameters:
                                ;
                                ;       NONE
                                ;
                                ;Output Parameters
                                ;
                                ;       NONE
                                ;
                                ;Data Structures Used
                                ;
                                ;Routines Used
                                ;
                                ;     1.  SODTM (jump to)
                                ;
                                ;     2.  QCOMIN (jump to)
                                ;
                                ;     3.  TRAP4 (jump to)
                                ;
                                ;     4.  TRAP10 (jump to)
                                ;
                                ;     5.  TRAP24 (jump to)
```

```
                                                                ;* PROCEDURE RESTART HANDLER
164000                          RESTRT::
                                                                ;* * IF BREAK CAUSED RESTART
164000  132737  000020  177522          BITB    #BIT4,@#KL.CSB
164006  001437                          BEQ     RE400
164010  132737  000040  177522          BITB    #BIT5,@#KL.CSB
164016  001433                          BEQ     RE400
164020  132737  000100  177522          BITB    #BIT6,@#KL.CSB
164026  001427                          BEQ     RE400
                                                                ;* * THEN
                                                                ;* * * CLEAR NXM AND DPR INDICATORS
164030  042737  100200  177530          BIC     #<KX$DRT ! KX$NXM>,@#KW.CSD
                                                                ;* * * CLEAR BREAK
164036  005737  177562                  TST     @#RBUF1A
                                                                ;* * * IF BREAK DISABLE BIT = 0
164042  032737  020000  175002          BIT     #KX$BDF,@#KW.STA
164050  001015                          BNE     RE250
                                                                ;* * * THEN
                                                                ;* * * * IF STATUS REGISTER SERIAL ODT FLAG = 0
164052  032737  000100  175002          BIT     #KX$SOF,@#KW.STA
164060  001005                          BNE     RE200
                                                                ;* * * * THEN
                                                                ;* * * * * DISABLE BREAKS
164062  052737  020000  175002          BIS     #KX$BDF,@#KW.STA
                                                                ;* * * * * PERFORM SERIAL ODT
164070  000167  000532                  JMP     SODTM   ;(NON STRUCTURED EXIT)
                                                                ;* * * * ELSE
164074                          RE200:
                                                                ;* * * * * REMOVE RETURN ADDR. AND PSW FROM STACK
164074  062706  000004                  ADD     #4,SP
                                                                ;* * * * * RETURN TO SERIAL ODT VIA BREAK ENTRY POINT
164100  000167  000570                  JMP     BRKENT  ;(NON STRUCTURED EXIT)
                                                                ;* * * * END IF
                                                                ;* * * ELSE
164104                          RE250:
                                                                ;* * * * RETURN FROM INTERRUPT
164104  000002                          RTI             ;(NON STRUCTURED EXIT)
                                                                ;* * * END IF
                                                                ;* * END IF
164106                          RE400:
                                                                ;* * IF FATAL ERROR FLAG OF STATUS REGISTER = 1
164106  032737  000040  175002          BIT     #KX$FEF,@#KW.STA
164114  001401                          BEQ     RE450
                                                                ;* * THEN
                                                                ;* * * STOP
164116  000777                  FAR::   BR
                                                                ;* * ENF IF
164120                          RE450:
                                                                ;* * CASE CAUSE OF RESTART
                                                                ;* * WHEN Q BUS COMMAND RECEIVED
164120  032737  000200  177530          BIT     #KX$DRT,@#KW.CSD
164126  001405                          BEQ     RE500
                                                                ;* * * CLEAR NXM AND DPR INDICATORS
164130  042737  100200  177530          BIC     #<KX$DRT ! KX$NXM>,@#KW.CSD
                                                                ;* * * DECODE COMMAND
164136  000167  003316                  JMP     QCOMIN  ;(NON STRUCTURED EXIT)
                                                                ;* * WHEN NXM
```

```
164142  032737  100000  177530  RE500:  BIT     #KX$NXM,@#KW.CSD
164150  001500                          BEQ     RE1040
                                                                        ;* * * CLEAR NXM AND DPR INDICATORS
164152  042737  100200  177530          BIC     #<KX$DRT ! KX$NXM>,@#KW.CSD
                                                                        ;* * * IF STACK NXM TEST FLAG = 1
164160  032737  004000  175002          BIT     #KX$SXT,@#KW.STA
164166  001442                          BEQ     RE900
                                                                        ;* * * THEN
                                                                        ;* * * * CLEAR STACK NXM TEST FLAG
164170  042737  004000  175002          BIC     #KX$SXT,@#KW.STA
                                                                        ;* * * * SET STACK ERROR FLAG OF STATUS REG.
164176  052737  000010  175002          BIS     #KX$SEF,@#KW.STA
                                                                        ;* * * * CASE MAP SELECTION
                                                                        ;* * * * WHEN MAP BIT 2 = 0
164204  132737  000010  177522          BITB    #KX$MP2,@#KW.CSB
164212  001005                          BNE     RE600
                                                                        ;* * * * * SAVE STACK POINTER VALUE AT TOR 1 - APPLICATION
                                                                        ;                   DEFAULT STACK OFFSET
164214  010637  077676                  MOV     SP,@#RTOP1+ADSTKO
                                                                        ;* * * * * SET STACK POINTER TO THAT ADDRESS
164220  012706  077676                  MOV     #RTOP1+ADSTKO,SP
164224  000421                          BR      RE800
                                                                        ;* * * * WHEN MAP BITS 0 AND 1 = 11
164226  132737  000002  177524  RE600:  BITB    #KX$MP0,@#KW.CSC
164234  001411                          BEQ     RE700
164236  132737  000004  177524          BITB    #KX$MP1,@#KW.CSC
164244  001405                          BEQ     RE700
                                                                        ;* * * * * SAVE STACK POINTER VALUE AT TOR 3 - APPLICATION
                                                                        ;                   DEFAULT STACK OFFSET
164246  010637  157676                  MOV     SP,@#RTOP3+ADSTKO
                                                                        ;* * * * * SET STACK POINTER TO THAT ADDRESS
164252  012706  157676                  MOV     #RTOP3+ADSTKO,SP
164256  000404                          BR      RE800
                                                                        ;* * * * ELSE
164260                          RE700:
                                                                        ;* * * * * SAVE STACK POINTER VALUE AT TOR - APPLICATION
                                                                        ;                   DEFAULT STACK OFFSET
164260  010637  137676                  MOV     SP,@#RTOP2+ADSTKO
                                                                        ;* * * * * SET STACK POINTER TO THAT ADDRESS
164264  012706  137676                  MOV     #RTOP2+ADSTKO,SP
                                                                        ;* * * * END CASE
164270                          RE800:
                                                                        ;* * * * EMULATE TRAP TO 4 (NON STRUCTURED EXIT)
164270  000167  002756                  JMP     TRAP4
                                                                        ;* * * ELSE
164274                          RE900:
                                                                        ;* * * * IF NXM FLAG = 1
164274  032737  010000  175002          BIT     #KX$NXF,@#KW.STA
164302  001404                          BEQ     RE1000
                                                                        ;* * * * THEN
                                                                        ;* * * * * SET THE CARRY FLAG OF RETURN PSW
164304  052766  000001  000002          BIS     #1,2(SP)
164312  000417                          BR      RE1010
                                                                        ;* * * * ELSE
164314                          RE1000:
                                                                        ;* * * * * CLEAR STACK ERROR FLAG
164314  042737  000010  175002          BIC     #KX$SEF,@#KW.STA
```

```
                                                                            ;* * * * * SET TEST STACK POINTER NXM FLAG = 1
164322  052737  004000  175002          BIS     #KX$SXT,@#KW.STA
                                                                            ;* * * * * TEST STACK BEFORE EMULATING TRAP
164330  005766  177776                  TST     -2(SP)  ;(READ NEXT TWO STACK SLOTS, WILL GET RESTART INT. IF STACK BAD)
164334  005766  177774                  TST     -4(SP)
                                                                            ;* * * * * SET TEST STACK POINTER NXM FLAG = 0
164340  042737  004000  175002          BIC     #KX$SXT,@#KW.STA
                                                                            ;* * * * * EMULATE TRAP TO 4 (NON STRUCTURED EXIT)
164346  000167  002700                  JMP     TRAP4
                                                                            ;* * * * END IF
164352                          RE1010:
                                                                            ;* * * ENDIF
164352                          RE1020:
                                                                            ;* * WHEN HALT (TEST FOR HALT INSTRUCTION AT
                                                                            ;               RETURN PC -2)
164352  162716  000002          RE1040: SUB     #2,(SP)
164356  005776  000000                  TST     @(SP)
164362  001403                          BEQ     RE1045
164364  062716  000002                  ADD     #2,(SP)
164370  000412                          BR      RE1070
164372  062716  000002          RE1045: ADD     #2,(SP)
                                                                            ;* * * IF ENTER ODT ON HALT FLAG = 1
164376  032737  000020  175002          BIT     #KX$OHL,@#KW.STA
164404  001402                          BEQ     RE1050
                                                                            ;* * * THEN
                                                                            ;* * * * ENTER SERIAL ODT
164406  000167  000214                  JMP     SODTM   ;(NON STRUCTURED EXIT)
                                                                            ;* * * ELSE
164412                          RE1050:
                                                                            ;* * * * EMULATE TRAP TO 10
164412  000167  002676                  JMP     TRAP10  ;(NON STRUCTURED EXIT)
                                                                            ;* * * END IF
164416                          RE1060:
                                                                            ;* * WHEN BHALT
164416                          RE1070:
164416  032737  001000  177530          BIT     #KX$BHF,@#KW.CSD
164424  001411                          BEQ     RE1080
                                                                            ;* * * IF SODT OR QODT FLAG NOT SET
164426  032737  000300  175002          BIT     #<KX$SOF ! KX$QOF>,@#KW.STA
164434  001002                          BNE     RE1072
                                                                            ;* * * THEN
                                                                            ;* * * * EMULATE TRAP TO 24
164436  000167  002714                  JMP     TRAP24
                                                                            ;* * * ELSE
164442                          RE1072:
                                                                            ;* * * * CLEAR BHALT INDICATOR
164442  042737  001000  177530          BIC     #KX$BHF,@#KW.CSD
                                                                            ;* * * END IF
164450                          RE1075:
                                                                            ;* * END CASE
164450                          RE1080:
                                                                            ;* * RETURN FROM INTERRUPT
164450  000002                          RTI
                                                                            ;* END RESTART HANDLER
```

```
                .SBTTL  SAVERG - SAVE REGISTERS
                .ENABLE LC,GBL

        ;
        ; Module name: SAVERG - SAVE REGISTERS
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module is used by both the serial and Q bus ODT monitors. It
        ;       saves the contents of the users CPU registers. This module exits
        ;       by jumping back to the ODT monitor which called it. It determines
        ;       which ODT monitor it is to return to by testing the state field of
        ;       the IOP status register. The following table discribes the use of the
        ;       native firmware reserved RAM for this purpose.
        ;
        ;                       NATIVE FIRMWARE RESERVED RAM USE
        ;
        ;
        ;                       USE                             ADDRESS
        ;               ---------------------------     -----------------------
        ;               BATTERY BACKUP FLAG 1           TOP OF NATIVE RAM
        ;                  "       "     " 2            TOR - 2
        ;               BOOT TYPE FLAG WORD             TOR - 4
        ;               NATIVE FIRMWARE FLAG WORD       TOR - 6
        ;               APPLICATION STACK POINTER HOLD  TOR - 10
        ;                    "      PS SAVE             TOR - 12
        ;                    "      PC  "               TOR - 14
        ;                    "      SP  "               TOR - 16
        ;                    "      R5  "               TOR - 20
        ;                    "      R4  "               TOR - 22
        ;                    "      R3  "               TOR - 24
        ;                    "      R2  "               TOR - 26
        ;                    "      R1  "               TOR - 30
        ;                    "      R0  "               TOR - 32
        ;               ODT TOP OF WORKING STACK        TOR - 34
        ;               ODT STACK GUARD WORD            TOR - 76
        ;               APPLICATION DEFAULT STACK       TOR - 100
        ;
        ;       This routine exits by jumping to return address of QODTM
        ;       or SODTM depending on which mode the KXT is in.
        ;
        ;Input Parameters:
        ;
        ;       ALL CPU REGISTERS
        ;
        ;Output Parameters
        ;
        ;       STACK POINTER
        ;
        ;Routines Used
        ;
        ;       NONE
        ;
```

```
                              ;NATIVE FIRMWARE RESERVED RAM DEFINTIONS
                              ; THESE ARE ADDRESS OFFSETS FROM THE TOP OF THE NATIVE RAM

        000000                BBF10   ==:     0                ;BATTERY BACKUP FLAG 1 ADDRESS OFFSET
        177776                BBF20   ==:     BBF10 - 2        ;  "        "    "  2    "       "
        177774                BOOTFO  ==:     BBF20 - 2        ;BOOT TYPE FLAG, SPECIFIES BOOT DEVICE AND TEST
        177772                NFFWDO  ==:     BOOTFO - 2       ;NATIVE FIRMWARE FLAG WORD ADDRESS OFFSET

        177770                APLPSO  ==:     NFFWDO - 2       ;APPLICATION PS SAVE ADDRESS OFFSET
        177766                APLPCO  ==:     APLPSO - 2       ;      "        PC  "     "       "
        177764                APLSPO  ==:     APLPCO - 2       ;      "        SP  "     "       "
        177762                APLR5O  ==:     APLSPO - 2       ;      "        R5  "     "       "
        177760                APLR4O  ==:     APLR5O - 2       ;      "        R4  "     "       "
        177756                APLR3O  ==:     APLR4O - 2       ;      "        R3  "     "       "
        177754                APLR2O  ==:     APLR3O - 2       ;      "        R2  "     "       "
        177752                APLR1O  ==:     APLR2O - 2       ;      "        R1  "     "       "
        177750                APLR0O  ==:     APLR1O - 2       ;      "        R0  "     "       "
        177746                ODTSTK  ==:     APLR0O - 2       ;ODT WORKING TOP OF STACK STACK ADDRESS OFFSET
        177702                ODTSTG  ==:     -76              ;ODT STACK GUARD WORD ADDRESS OFFSET
        177700                ADSTKO  ==:     -100             ;APPLICATION DEFAULT TOP OF STACK OFFSET ADDRESS
                              ;
        077776                RTOP1   ==:     77776            ;TOP OF RAM 1
        137776                RTOP2   ==:     137776           ; "  "   "  2
        157776                RTOP3   ==:     157776           ; "  "   "  3
        000000                RBOT1   ==:     0                ;BOTTOM OF RAM 1
        040000                RBOT2   ==:     40000            ;   "    "   "  2
        100000                RBOT3   ==:     100000           ;   "    "   "  3
        077762                SP1     ==:     RTOP1 + APLSPO   ;INITIAL ODT STACK 1 ABSOLUTE ADDRESS
        137762                SP2     ==:     RTOP2 + APLSPO   ;   "     "    "   2    "        "
        157762                SP3     ==:     RTOP3 + APLSPO   ;   "     "    "   3    "        "
        000014                SPOFF   ==:     -APLSPO          ;OFFSET OF STACK POINTER FROM TOP OF RAM
```

```
                                                             ;* PROCEDURE SAVE REGISTERS
164452                          SAVERG::
                                                             ;* * CASE MAP BITS
                                                             ;* * WHEN MAP BIT 2 = 0
164452  032737  000010  177522          BIT     #KX$MP2,@#KW.CSB
164460  001005                          BNE     SR100
                                                             ;* * * SAVE USERS STACK POINTER AT ODT STACK 1
164462  010637  077762                  MOV     SP,@#SP1
                                                             ;* * * SET STACK POINTER = ODT STACK 1
164466  012706  077762                  MOV     #SP1,SP
164472  000421                          BR      SR300
                                                             ;* * WHEN MAP BITS 0 AND 1 = 11
164474  032737  000002  177522  SR100:  BIT     #KX$MP0,@#KW.CSB
164502  001411                          BEQ     SR200
164504  032737  000004  177522          BIT     #KX$MP1,@#KW.CSB
164512  001405                          BEQ     SR200
                                                             ;* * * SAVE USERS STACK POINTER AT ODT STACK 3
164514  010637  157762                  MOV     SP,@#SP3
                                                             ;* * * SET STACK POINTER TO ODT STACK 3
164520  012706  157762                  MOV     #SP3,SP
164524  000404                          BR      SR300
                                                             ;* * ELSE
164526                          SR200:
                                                             ;* * * SAVE USERS STACK POINTER AT ODT STACK 2
164526  010637  137762                  MOV     SP,@#SP2
                                                             ;* * * SET STACK POINTER TO ODT STACK 2
164532  012706  137762                  MOV     #SP2,SP
                                                             ;* * END CASE
164536                          SR300:
                                                             ;* * SAVE USER REGISTERS 5 THROUGH 0
164536  010546                          MOV     R5,-(SP)
164540  010446                          MOV     R4,-(SP)
164542  010346                          MOV     R3,-(SP)
164544  010246                          MOV     R2,-(SP)
164546  010146                          MOV     R1,-(SP)
164550  010046                          MOV     R0,-(SP)
                                                             ;* * SET R5 = TOP OF RAM ADDRESS
164552  010605                          MOV     SP,R5
164554  162705  177750                  SUB     #APLR00,R5
                                                             ;* * SAVE USER PC AND PS
164560  016500  177764                  MOV     APLSP0(R5),R0
164564  012065  177766                  MOV     (R0)+,APLPC0(R5)
164570  012065  177770                  MOV     (R0)+,APLPS0(R5)
                                                             ;* * REMOVE USER PS AND PC FROM USER STACK
164574  010065  177764                  MOV     R0,APLSP0(R5)
                                                             ;* * WRITE GUARD WORD = 123456
164600  012765  123456  177702          MOV     #123456,ODTSTG(R5)
                                                             ;* * IF STATUS REGISTER Q BUS ODT MODE FLAG SET
164606  032767  000200  010166          BIT     #KX$QOF,KW.STA
164614  001402                          BEQ     SR400
                                                             ;* * THEN
                                                             ;* * * RETURN TO Q BUS ODT MONITOR (NON STRUCTURED JUMP TO Q
164616  000167  004622                  JMP     QSRET
                                                             ;* * ELSE
164622                          SR400:
                                                             ;* * * RETURN TO SERIAL ODT MONITOR (NON STRUCTURED JUMP TO
164622  000167  000026                  JMP     SSRET
```

```
;* * END IF
;* END SAVE REGISTERS
```

```
                .SBTTL  SODTM - SERIAL ODT MONITOR

                .ENABL  LC,GBL
        ;
        ; Module name: SODTM - SERIAL ODT MONITOR
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This is the main controlling module for the serial ODT mode.
        ;       It supervises the ODT function and on completion returns
        ;       control back to the application software.
        ;       There is a secondary entry point to this module. This
        ;       entry point is used by the RESTRT module when it determines
        ;       a BREAK was received on the console serial line and the IOP
        ;       was executing serial ODT. This entry causes the ODT to
        ;       start with a fresh command prompt without effecting the
        ;       contents of the application register storage area.
        ;
        ;       The valid ODT commands are as follows:
        ;
        ;       Rn/     -  Open register n (n = 0 through 7 and S)
        ;       n/      -  Open location n
        ;       /       -  Re-open previous location or register
        ;       m,n/    -  Examine locations m through n, leaving location n open
        ;       Rm,n/   -  Examine registers m through n, leaving location n open
        ;       n       -  Deposit n in open register or location
        ;       <LF>    -  Close last location or register, open next +
        ;       ^       -  Close last location or register, open next -
        ;       <CR>    -  Close location or register
        ;       P       -  Proceed with execution of application
        ;       Gn      -  Execute application at location n
        ;
        ;               NOTE: A range examine can be aborted by pressing BREAK key.
        ;
        ;Input Parameters:
        ;
        ;       NONE
        ;
        ;Output Parameters:
        ;
        ;       NONE
        ;
        ;Routines Used:
        ;
        ;       SAVERG
        ;       SETLED
        ;       ADDREG
        ;       EXAMIN
        ;       DISPST
        ;       DATCOM
```

```
                        ;       DEPOS
                        ;       GOPROC

```

```
                        ;REGISTER 3 IS USED AS THE FLAG REGISTER FOR ALL ODT ROUTINES
                        ;THE FOLLOWING ARE THE BIT DEFINITIONS FOR THIS REGISTER

        000001          DCDFGB  ==:     1               ;DATA/COMMAND ROUTINE - DONE FLAG
        000002          DARFGB  ==:     2               ;DATA/COMMAND ROUTINE - DATA RECV'D WITH COMM. FLAG
        000004          APDFGB  ==:     4               ;ADDRESS/REGISTER ROUTINE - DONE FLAG
        000010          FCHFGB  ==:     10              ;ADDRESS/REGISTER ROUTINE - FIRST CHAR. RECV'D FLAG
        000020          CNDFGB  ==:     20              ;SERIAL ODT MONITOR ROUTINE - CONDITION FLAG
        000040          XOFFGB  ==:     40              ;GENERAL - XOFF FLAG
        000200          REGFGB  ==:     200             ;GENERAL - REGISTER FLAG
        100000          ERRFGB  ==:     100000          ;GENERAL - ERROR FLAG

```

```
                                                              ;* PROCEDURE SERIAL ODT MONITOR
164626                             SODTM::
                                                              ;* * DISABLE DUAL PORT RAM
164626  042737  000100  177530             BIC     #KX$DEN,@#KW.CSD
                                                              ;* * DISABLE BREAKS
                                                              ;* * DISABLE NXM TRAP TO 4
                                                              ;* * SET SERIAL ODT FLAG
164634  052737  030100  175002             BIS     #<KX$BDF ! KX$NXF ! KX$SOF>,@#KW.STA
                                                              ;* * CLEAR Q BUS ODT FLAG
164642  042737  000200  175002             BIC     #KX$QOF,@#KW.STA
                                                              ;* * SET SERIAL ODT FLAG
                                                              ;* * SAVE REGISTERS (JUMP TO AND BACK FROM
                                                              ;                         SAVE REGISTERS MODULE)
164650  000167  177576                     JMP     SAVERG
164654                             SSRET::
                                                              ;* * SET LEDS TO SERIAL ODT MODE DISPLAY
164654  012700  000004                     MOV     #SODTMD,R0
164660  004767  002314                     CALL    SETLED
                                                              ;* * READ RBUF
164664  105737  177562                     TSTB    @#RBUF1A
                                                              ;* * SET BAUD RATE
164670  004767  001762                     CALL    ABAUD
                                                              ;* * CLEAR RECEIVE BUFFER
164674                             BRKENT::
                                                              ;* * RE-INITIALIZE ODT STACK POINTER
164674  012700  177750                     MOV     #ODTSTK+2,R0
164700  060500                             ADD     R5,R0
164702  010006                             MOV     R0,SP
                                                              ;* * ENABLE BREAK
164704  042737  020000  175002             BIC     #KX$BDF,@#KW.STA
                                                              ;* * CLEAR ALL ODT FLAG BITS
164712  005003                             CLR     R3
                                                              ;* * CLEAR TARGET BUFFER
164714  005002                             CLR     R2
                                                              ;* * DISABLE TRAP TO 4
164716  052737  010000  175002             BIS     #KX$NXF,@#KW.STA
                                                              ;* * PRINT CR LF
164724  004767  000560                     CALL    PCRLF   ;(LOCAL CALL)
                                                              ;* * DISPLAY PC VALUE
164730  016501  177766                     MOV     APLPCO(R5),R1
164734  004767  001572                     CALL    DISPWD
                                                              ;* * ENABLE BREAKS
164740  042737  020000  175002             BIC     #KX$BDF,@#KW.STA
                                                              ;* * DO UNTIL PROCEED COMMAND
164746                             S0100:
                                                              ;* * OR GO COMMAND
164746                             S0200:
                                                              ;* * * DISPLAY ODT PROMPT
164746  012701  166266                     MOV     #PROMPT,R1
164752  004767  001370                     CALL    DISPST
                                                              ;* * * CLEAR FIRST CHARACTER FLAG
164756  042703  000010                     BIC     #FCHFGB,R3
                                                              ;* * * GET ADDRESS OR REGISTER NUMBER
164762  004767  000534                     CALL    ADDREG
                                                              ;* * * IF ERROR FLAG = 0
164766  005703                             TST     R3
164770  100002                             BPL     S0300
```

```
164772  000167  000414            JMP     S01850
164776                    S0300:
                                                   ;* * * THEN
                                                   ;* * * * IF ADDRESS OR REGISTER NO. ENTERED
164776  032703  000002            BIT     #DARFGB,R3
165002  001401                    BEQ     S0400
                                                   ;* * * * THEN
                                                   ;* * * * * SAVE FIRST TARGET
165004  010402                    MOV     R4,R2
                                                   ;* * * * END IF
165006                    S0400:
                                                   ;* * * * CASE COMMAND FOLLOWING ADDRESS OR REGISTER NUMBER
                                                   ;* * * * WHEN ","
165006  122700  000054            CMPB    #',,R0
165012  001023                    BNE     S0900
                                                   ;* * * * * GET ANOTHER ADDRESS OR REGISTER NUMBER
165014  004767  000502            CALL    ADDREG
                                                   ;* * * * * IF VALID ADDRESS OR REGISTER NUMBER
165020  005703                    TST     R3
165022  100414                    BMI     S0700
                                                   ;* * * * * AND COMMAND FOLLOWING ADDRESS OR
                                                   ;             REGISTER NUMBER IS "/"
165024  122700  000057            CMPB    #'/,R0
165030  001011                    BNE     S0700
                                                   ;* * * * * AND ADDRESS OR REGISTER NUMBER GREATER
                                                   ;                   THEN FIRST TARGET
165032  020402                    CMP     R4,R2
165034  101407                    BLOS    S0700
                                                   ;* * * * * THEN
                                                   ;* * * * * * DISPLAY DATA OF RANGE
                                  ;R2=TARGET 1     ;(PASS TARGET 1)
                                  ;R4=TARGET 2     ;(PASS TARGET 2)
165036  012700  000002            MOV     #RNGFGB,R0  ;(PASS SERIAL ODT FLAG = 1, RANGE FLAG = 1)
165042  004767  000674            CALL    EXAMIN
                                                   ;* * * * * * SET CONDITION FLAG = OPEN
165046  052703  000020            BIS     #CNDFGB,R3
165052  000402                    BR      S0800
                                                   ;* * * * * ELSE
165054                    S0700:
                                                   ;* * * * * * SET ERROR FLAG
165054  052703  100000            BIS     #ERRFGB,R3
                                                   ;* * * * * END IF
165060  000445            S0800:  BR      S01450
                                                   ;* * * * WHEN "/"
165062                    S0900:
165062  122700  000057            CMPB    #'/,R0
165066  001006                    BNE     S01000
                                                   ;* * * * * DISPLAY DATA
165070  005000                    CLR     R0       ;(PASS SERIAL ODT FLAG = 1, RANGE FLAG = 0)
                                                   ;(PASS TARGET 1)
165072  004767  000644            CALL    EXAMIN
                                                   ;* * * * * SET CONDITION FLAG = OPEN
165076  052703  000020            BIS     #CNDFGB,R3
165102  000434                    BR      S01450
                                                   ;* * * * WHEN "G"
165104                    S01000:
165104  122700  000107            CMPB    #'G,R0
```

```
165110  001014                      BNE     S01300
                                                    ;* * * * AND TARGET IS AN ADDRESS
165112  105703                      TSTB    R3
165114  100412                      BMI     S01300
                                                    ;* * * * * SET LEDS TO NON NATIVE CODE DISPLAY
165116  012700  000000              MOV     #NNCED,R0
165122  004767  002052              CALL    SETLED
                                                    ;* * * * * EXECUTE AT ADDRESS ENTERED (WILL NOT
                                                    ;               RETURN IF NO ERROR)
165126  010400                      MOV     R4,R0
165130  004767  001636              CALL    GO
                                                    ;* * * * * SET ERROR FLAG = 1
165134  052703  100000              BIS     #ERRFGB,R3
165140  000415                      BR      S01450
                                                    ;* * * * WHEN "P"
165142                      S01300:
165142  122700  000120              CMPB    #'P,R0
165146  001010                      BNE     S01400
                                                    ;* * * * AND NO DATA PRECEDED COMMAND
165150  032703  000002              BIT     #DARFGB,R3
165154  001005                      BNE     S01400
                                                    ;* * * * * EXECUTE PROCEED (WILL NOT RETURN IF
                                                    ;                       NO ERROR)
165156  004767  001626              CALL    PROC
                                                    ;* * * * * SET ERROR FLAG = 1
165162  052703  100000              BIS     #ERRFGB,R3
165166  000402                      BR      S01450
                                                    ;* * * * ELSE (INVALID COMMAND)
165170                      S01400:
                                                    ;* * * * * SET ERROR FLAG
165170  052703  100000              BIS     #ERRFGB,R3
                                                    ;* * * * END CASE
165174                      S01450:
                                                    ;* * * * DO WHILE CONDITION FLAG = OPEN
165174                      S01470:
165174  032703  000020              BIT     #CNDFGB,R3
165200  001504                      BEQ     S01800
                                                    ;* * * * AND ERROR FLAG = 0
165202  005703                      TST     R3
165204  100502                      BMI     S01800
                                                    ;* * * * * PRINT SPACE
165206  012701  166272              MOV     #SPACE,R1
165212  004767  001130              CALL    DISPST
                                                    ;* * * * * GET INPUT DATA
165216  004767  001162              CALL    DATCOM
                                                    ;* * * * * IF VALID DATA AND COMMAND
165222  005703                      TST     R3
165224  100467                      BMI     S01600
                                                    ;* * * * * THEN
                                                    ;* * * * * * IF DATA PROCEEDED COMMAND
165226  032703  000002              BIT     #DARFGB,R3
165232  001402                      BEQ     S01500
                                                    ;* * * * * * THEN
                                                    ;* * * * * * * DEPOSIT DATA INTO ADDRESS OR REGISTER
                                                    ;               (R2=ADDR.)
                                                    ;               (R4=DATA)
165234  004767  001234              CALL    DEPOS
```

```
                                                                        ;* * * * * * END IF
165240                              S01500:
                                                                        ;* * * * * * CASE COMMAND
                                                                        ;* * * * * * WHEN CR
165240  122700  000015                      CMPR    #CR,R0
165244  001003                              BNE     S01550
                                                                        ;* * * * * * * SET CONDITION FLAG = CLOSED
165246  042703  000020                      BIC     #CNDFGB,R3
165252  000453                              BR      S01570
                                                                        ;* * * * * * WHEN LF
165254                              S01550:
165254  122700  000012                      CMPB    #LF,R0
165260  001023                              BNE     S01560
                                                                        ;* * * * * * * PRINT CR LF
165262  004767  000222                      CALL    PCRLF   ;(LOCAL CALL)
                                                                        ;* * * * * * * INCREMENT TARGET
165266  005202                              INC     R2
                                                                        ;* * * * * * * IF TYPE IS REGISTER
165270  105703                              TSTB    R3
165272  100007                              BPL     S01555
                                                                        ;* * * * * * * THEN
                                                                        ;* * * * * * * * SET TARGET MOD 10
165274  022702  000011                      CMP     #11,R2
165300  101001                              BHI     S01552
165302  005002                              CLR     R2
165304                              S01552:
                                                                        ;* * * * * * * * DISPLAY REGISTER NUMBER
165304  004767  000140                      CALL    DISPRG  ;(LOCAL CALL)
165310  000404                              BR      S01557
                                                                        ;* * * * * * * ELSE
165312                              S01555:
                                                                        ;* * * * * * * * INCREMENT TARGET AGAIN
165312  005202                              INC     R2
                                                                        ;* * * * * * * * DISPLAY TARGET
165314  010201                              MOV     R2,P1
165316  004767  001210                      CALL    DISPWD
                                                                        ;* * * * * * * END IF
165322                              S01557:
                                                                        ;* * * * * * * EXAMINE
165322  004767  000142                      CALL    EXMDIS  ;(LOCAL CALL)
165326  000425                              BR      S01570
                                                                        ;* * * * * * WHEN ^
165330                              S01560:
165330  122700  000136                      CMPB    #'^,R0
165334  001022                              BNE     S01570
                                                                        ;* * * * * * * PRINT CR LF
165336  004767  000146                      CALL    PCRLF   ;(LOCAL CALL)
                                                                        ;* * * * * * * IF TYPE IS REGISTER
165342  105703                              TSTB    R3
165344  100007                              BPL     S01567
                                                                        ;* * * * * * * THEN
                                                                        ;* * * * * * * * DECREMENT TARGET
165346  005302                              DEC     R2
                                                                        ;* * * * * * * * SET REGISTER NUMBER MOD 10
165350  100002                              BPL     S01565
165352  012702  000010                      MOV     #10,R2
165356                              S01565:
```

```
                                                            ;* * * * * * * * DISPLAY REGISTER NUMBER
165356  004767  000066                  CALL    DISPRG  ;(LOCAL CALL)
165362  000405                          BR      S01569
                                                            ;* * * * * * * ELSE
165364                          S01567:
                                                            ;* * * * * * * * DECREMENT TARGET BY 2
165364  005302                          DEC     R2
165366  005302                          DEC     R2
                                                            ;* * * * * * * * DISPLAY TARGET
165370  010201                          MOV     R2,R1
165372  004767  001134                  CALL    DISPWD
                                                            ;* * * * * * * END IF
165376                          S01569:
                                                            ;* * * * * * * EXAMINE
165376  004767  000066                  CALL    EXMDIS  ;(LOCAL CALL)
                                                            ;* * * * * * END CASE
165402                          S01570:
165402  000402                          BR      S01700
                                                            ;* * * * * ELSE
165404                          S01600:
                                                            ;* * * * * * SET ERROR FLAG
165404  052703  100000                  BIS     #ERRFGB,R3
                                                            ;* * * * * END IF
165410                          S01700:
                                                            ;* * * * END WHILE
165410  000671                          BR      S01470
165412                          S01800:
                                                            ;* * * END IF
165412                          S01850:
                                                            ;* * * IF ERROR FLAG = 1
165412  005703                          TST     R3
165414  100011                          BPL     S01900
                                                            ;* * * THEN
                                                            ;* * * * SEND ERROR PROMPT
165416  012701  166264                  MOV     #ERROR,R1
165422  004767  000720                  CALL    DISPST
                                                            ;* * * * CLEAR REGISTER FLAG
165426  042703  000200                  BIC     #REGFGB,R3
                                                            ;* * * * SET TARGET2 TO NXM ADDRESS
                                                            ;          (TO CAUSE AN ERROR IF RETURN / ENTERED)
165432  012702  177777                  MOV     #177777,R2
                                                            ;* * * * CLEAR TARGET1
165436  005004                          CLR     R4
                                                            ;* * * END IF
165440                          S01900:
                                                            ;* * * CLEAR ALL ODT FLAGS EXCEPT
                                                            ;        REGISTER FLAG
165440  042703  177577                  BIC     #^C<REGFGB>,R3
                                                            ;* * END UNTIL
165444  000167  177276                  JMP     S0100
                                                            ;* END SERIAL ODT MONITOR
```

```
                                ; The following subroutines were created to conserve memory and are only used
                                ; by this module.

                                                                    ;* PROCEDURE DISPLAY REGISTER NUMBER
165450                          DISPRG:
                                                                    ;* * CALCULATE DISPLAY POINTER
165450  010201                          MOV     R2,R1
165452  006301                          ASL     R1
165454  006301                          ASL     R1
165456  062701  166303                  ADD     #REGDIS,R1
                                                                    ;* * DISPLAY STRING
165462  004767  000660                  CALL    DISPST
                                                                    ;* END DISPLAY REGISTER NUMBER
165466  000207                          RETURN


                                                                    ;* PROCEDURE EXAMINE AND DISPLAY
165470                          EXMDIS:
                                                                    ;* * DISPLAY "/"
165470  012701  166301                  MOV     #OPENDS,R1
165474  004767  000646                  CALL    DISPST
                                                                    ;* * EXAMINE DATA @ TARGET
165500  005000                          CLR     R0
165502  004767  000234                  CALL    EXAMIN
                                                                    ;* END EXAMINE AND DISPLAY
165506  000207                          RETURN


                                                                    ;* PROCEDURE PRINT CRLF
165510                          PCRLF:
                                                                    ;* * POINT TO STRING
165510  012701  166274                  MOV     #CRLF,R1
                                                                    ;* * DISPLAY STRING
165514  004767  000626                  CALL    DISPST
                                                                    ;* END PRINT CRLF
165520  000207                          RETURN

```

```
        .SBTTL  ADDREG - GET ADDRESS/PEGISTER
        .ENABLE LC,GBL

;
; Module name: ADDREG - GET ADDRESS/REGISTER
;
; System: KXT11-CA Native Firmware
;
;
;
;
; Functional Description:
;
;       This module is used by the serial ODT monitor. It receives data
;       from the console serial line, accumulates a word target delimited
;       by a valid command or an error. The valid commands are: ",", CR, LF,
;       "/", "G" or "P". It returns the target, the command
;       the target type (register or address) and the error status.

;Input Parameters:

;       NONE

;Output Parameters

;       1) R0 = COMMAND RECEIVED
;       2) R4 = TARGET (ADDRESS OR REGISTER NO.)
;       3) R3, BIT REGFGB - 1 = REGISTER NO., 0 = ADDRESS
;       4) R3, BIT DARFGB - ADDRESS/REGISTER RECEIVED WITH COMMAND FLAG,
;                                                       1 = YES. 0 = NO
;       5) R3, BIT ERRFGB - 1 = ERROR, NOT ALTERED IF NO ERROR

;DATA STRUCTURES USED

;       NONE

;Routines Used

;       1) CHKCHR
;       2) ACCUM

```

```
                                                                   ;* PROCEDURE GET ADDRESS/REGISTER
165522                            ADDREG::
                                                                   ;* * CLEAR ADDRESS/REGISTER DONE FLAG
                                                                   ;* * CLEAR DATA RECEIVED FLAG
165522  042703  000006                    BIC     #<ARDFGB ! DARFGB>,R3
                                                                   ;* * CLEAR ACCUMULATED NUMBER
165526  005004                            CLR     R4
                                                                   ;* * DO UNTIL DONE FLAG SET
165530                            AD100:
                                                                   ;* * * DO UNTIL NON NUMBER CHARACTER RECEIVED
165530                            AD200:
                                                                   ;* * * * GET INPUT
165530  004767  006016                    CALL    GETIN
                                                                   ;* * * * IF INPUT = 0 THRU 7
165534  103417                            BCS     AD253
                                                                   ;* * * * * IF FIRST CHARACTER
165536  032703  000010                    BIT     #FCHFGB,R3
165542  001002                            BNE     AD220
                                                                   ;* * * * * THEN
                                                                   ;* * * * * * CLEAR REGISTER FLAG
165544  042703  000200                    BIC     #REGFGB,R3
                                                                   ;* * * * * END IF
165550                            AD220:
                                                                   ;* * * * * IF REGISTER FLAG = 1
165550  105703                            TSTB    R3
165552  100002                            BPL     AD240
                                                                   ;* * * * * THEN
                                                                   ;* * * * * * REPLACE ACCUM VALUE WITH INPUT
165554  010004                            MOV     R0,R4
165556  000402                            BR      AD250
                                                                   ;* * * * * ELSE
165560                            AD240:
                                                                   ;* * * * * * ACCUMULATE DATA
165560  004767  005754                    CALL    ACCUM
                                                                   ;* * * * * END IF
165564                            AD250:
                                                                   ;* * * * * SET FIRST CHARACTER RECEIVED FLAG = 1
165564  052703  000010                    BIS     #FCHFGB,R3
                                                                   ;* * * * * CLEAR NON NUMERIC INPUT FLAG
165570  000241                            CLC
165572  000424                            BR      AD290
                                                                   ;* * * * ELSE
165574                            AD253:
                                                                   ;* * * * * IF INPUT => "a"
165574  120027  000141                    CMPB    R0,#'a
165600  103402                            BLO     AD255
                                                                   ;* * * * * THEN
                                                                   ;* * * * * * CONVERT TO UPPER CASE
165602  042700  000040                    BIC     #40,R0
                                                                   ;* * * * * END IF
165606                            AD255:
                                                                   ;* * * * * IF INPUT = "S" OR "s"
165606  122700  000123                    CMPB    #'S,R0
165612  001013                            BNE     AD280
165614                            AD270:
                                                                   ;* * * * * AND REGISTER FLAG = 1
165614  105703                            TSTB    R3
```

```
165616  100011                    BPL     AD280
                                                  ;* * * * * AND NOT FIRST CHARACTER RECEIVED
165620  032703  000010            BIT     #FCHFGB,R3
165624  001406                    BEQ     AD280
                                                  ;* * * * * * SET ACCUMULATED VALUE = 10
165626  012704  000010            MOV     #10,R4
                                                  ;* * * * * * SET DATA RECIEVED FLAG = 1
                                                  ;* * * * * * SET FIRST CHARACTER RECEIVED FLAG = 1
165632  052703  000012            BIS     #<DARFGB ! FCHFGB>,R3
                                                  ;* * * * * * SET NON NUMERIC FLAG = 0
165636  000241                    CLC
165640  000401                    BR      AD290
                                                  ;* * * * * ELSE
165642                    AD280:
                                                  ;* * * * * * SET NON NUMERIC INPUT FLAG = 1
165642  000261                    SEC
                                                  ;* * * * * END IF
165644                    AD290:
                                                  ;* * * * END IF
                                                  ;* * * END UNTIL
165644  103331                    BCC     AD200
165646                    AD300:
                                                  ;* * * CASE COMMAND RECEIVED
                                                  ;* * * WHEN ","
                                                  ;* * * OR CR
                                                  ;* * * OR LF
                                                  ;* * * OR "/"
                                                  ;* * * OR "G"
                                                  ;* * * OR "P"
165646  012701  165734            MOV     #CHTAB,R1
165652  122100            AD400:  CMPB    (R1)+,R0
165654  001404                    BEQ     AD500
165656  022701  165742            CMP     #CHEND,R1
165662  001404                    BEQ     AD600
165664  000772                    BR      AD400
165666                    AD500:
                                                  ;* * * SET FIRST CHARACTER FLAG = 1
                                                  ;* * * SET DONE FLAG = 1
165666  052703  000014            BIS     #<FCHFGB ! ARDFGB>,R3
165672  000414                    BR      AD800
                                                  ;* * * WHEN "R"
165674                    AD600:
165674  122700  000122            CMPB    #'R,R0
165700  001007                    BNE     AD700
                                                  ;* * * AND FIRST CHARACTER RECEIVED FLAG = 0
165702  032703  000010            BIT     #FCHFGB,R3
165706  001004                    BNE     AD700
                                                  ;* * * * SET REGISTER FLAG = 1
                                                  ;* * * * SET FIRST CHARACTER FLAG = 1
                                                  ;* * * * SET DATA RECEIVED FLAG = 1
165710  052703  000212            BIS     #<DARFGB ! REGFGB ! FCHFGB>,R3
                                                  ;* * * * CLEAR TARGET
165714  005004                    CLR     R4
165716  000402                    BR      AD800
                                                  ;* * * ELSE
165720                    AD700:
                                                  ;* * * * SET ERROR FLAG
```

```
                                                                   ;* * * * SET DONE FLAG
165720  052703  100004                      BIS     #<ERRFGB ! ARDFGB>,R3
                                                                   ;* * * END CASE
165724                              AD800:
                                                                   ;* * END UNTIL
165724  032703  000004                      BIT     #ARDFGB,R3
165730  001677                              BEQ     AD100
                                                                   ;* * RETURN
165732  000207                              RETURN
                                                                   ;* END GET ADDRESS/REGISTER


                                    ;CHARACTER TABLE

165734     054     015     012      CHTAB:  .ASCII  ',"<15><12>"/GP'
165742                              CHEND:
                                            .EVEN
```

```
        .SBTTL  EXAMIN - EXAMINE
        .ENABLE LC,GBL

;
; Module name: EXAMIN - EXAMINE
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module performs the memory or register examine function for
;       both the Q bus and serial ODT monitors. The output of this module
;       is directly to the user (the serial line or the dual port RAM).

;Input Parameters:

;       R0 - FLAGS,
;          BIT (RNGFGB) = 1 (RANGE EXAMIN), BIT = 0 (SINGLE EXAMIN)
;       R2 - TARGET 1
;          BIT 0 - TYPE, BIT = 0 (REGISTER), BIT = 1 (ADDRESS)
;       R4 - TARGET 2
;          BIT 0 - TYPE, BIT = 0 (REGISTER), BIT = 1 (ADDRESS)

;Output Parameters

;       R3, BIT (ERRFGB) - ERROR FLAG, BIT = 1 (ERROR), NO ERROR BIT NOT ALTERED

;DATA STRUCTURES

;       1) PRINT STRINGS
;       2) DUAL PORT RAM REGISTER 2
;       3) IOP STATUS REGISTER
;       4) IOP COMMAND REGISTER

;Routines Used

;       1) DISPLAY STRING (DISPST)
;       2) DISPLAY WORD (DISPWD)
;       3) CHECK CHARACTER (CHKCHR)
;
```

```
        000002                          RNGFGB  ==:     2



                                                                        ;* PROCEDURE EXAMINE
165742                                  EXAMIN::
                                                                        ;* * IF SERIAL ODT MODE
165742  032737  000100  175002                  BIT     #KX$SOF,@#KW.STA
165750  001525                                  BEQ     EX1000
                                                                        ;* * THEN
                                                                        ;* * * IF THE TARGET IS A REGISTER
165752  105703                                  TSTB    R3
165754  100021                                  BPL     EX300
                                                                        ;* * * THEN
                                                                        ;* * * * SET TARGET 1 TO MOD 10
165756  022702  000011                          CMP     #11,R2
165762  101001                                  BHI     EX10
165764  005002                                  CLR     R2
165766                                  EX10:
                                                                        ;* * * * SET TARGET 2 TO  MOD 10
165766  022704  000011                          CMP     #11,R4
165772  101001                                  BHI     EX20
165774  005004                                  CLR     R4
165776                                  EX20:
                                                                        ;* * * * SET TARGET 1 =
                                                                        ;       (TARGET * 2) 1 + REG 0 SAVE ADDRESS OFFSET
165776  006302                                  ASL     R2
166000  062702  177750                          ADD     #APLR00,R2
166004  060502                                  ADD     R5,R2
                                                                        ;* * * * SET TARGET 2 =
                                                                        ;       (TARGET 2 * 2) + REG 0 SAVE ADDRESS OFFSET
166006  006304                                  ASL     R4
166010  062704  177750                          ADD     #APLR00,R4
166014  060504                                  ADD     R5,R4
166016  000404                                  BR      EX400
                                                                        ;* * * ELSE
166020                                  EX300:
                                                                        ;* * * * CLEAR BIT 0 OF TARGET 1 AND TARGET 2
166020  042702  000001                          BIC     #BIT0,R2
166024  042704  000001                          BIC     #BIT0,R4
                                                                        ;* * * END IF
166030                                  EX400:
                                                                        ;* * * IF NOT RANGE EXAMINE
166030  032700  000002                          BIT     #RNGFGB,R0
166034  001001                                  BNE     EX500
                                                                        ;* * * THEN
                                                                        ;* * * * SET TARGET 2 = TARGET 1
166036  010204                                  MOV     R2,R4
                                                                        ;* * * END IF
166040                                  EX500:
                                                                        ;* * * DO UNTIL TARGET 1 = TARGET 2
166040                                  EX600:
                                                                        ;* * * OR ERROR FLAG = 1
                                                                        ;* * * OR INCREMENTED TARGET 1 = 0
                                                                        ;* * * * GET DATA PER TARGET 1 AND INCREMENT
                                                                        ;               TARGET 1 POINTER
```

```
166040  000241                            CLC
166042  012201                            MOV     (R2)+,R1
                                                                  ;* * * * IF NXM
166044  000240                            NOP
166046  103003                            BCC     EX700
                                                                  ;* * * * THEN
                                                                  ;* * * * * SET ERROR FLAG
166050  052703  100000                    BIS     #ERPFGB,R3
                                                                  ;* * * * * EXIT EXAMINE (NON STRUCTURED EXIT)
166054  000502                            BR      EX1300
                                                                  ;* * * * END IF
166056                          EX700:
                                                                  ;* * * * DISPLAY DATA
166056  004767  000450                    CALL    DISPWD
                                                                  ;* * * * IF TARGET 1 = TARGET 2
166062  020204                            CMP     R2,R4
166064  101402                            BLOS    EX750
                                                                  ;* * * * THEN
                                                                  ;* * * * * SET RANGE FLAG = 0
166066  042700  000002                    BIC     #RNGFGB,R0
                                                                  ;* * * * END IF
166072                          EX750:
                                                                  ;* * * * IF RANGE EXAMINE
166072  032700  000002                    BIT     #RNGFGB,R0
166076  001427                            BEQ     EX800
                                                                  ;* * * * THEN
                                                                  ;* * * * * DISPLAY CR LF
166100  012701  166274                    MOV     #CRLF,R1
166104  004767  000236                    CALL    DISPST
                                                                  ;* * * * * IF TYPE IS REGISTER
166110  105703                            TSTB    R3
166112  100012                            BPL     EX760
                                                                  ;* * * * * THEN
                                                                  ;* * * * * * DISPLAY REGISTER NUMBER
166114  010201                            MOV     R2,R1
166116  160501                            SUB     R5,R1
166120  162701  177750                    SUB     #APLR00,R1
166124  006301                            ASL     R1
166126  062701  166303                    ADD     #REGDIS,R1
166132  004767  000210                    CALL    DISPST
166136  000403                            BR      EX770
                                                                  ;* * * * * ELSE
166140                          EX760:
                                                                  ;* * * * * * DISPLAY ADDRESS
166140  010201                            MOV     R2,R1
166142  004767  000364                    CALL    DISPWD
                                                                  ;* * * * * END IF
166146                          EX770:
                                                                  ;* * * * * DISPLAY "/"
166146  012701  166301                    MOV     #OPENDS,R1
166152  004767  000170                    CALL    DISPST
                                                                  ;* * * * END IF
166156                          EX800:
                                                                  ;* * * END UNTIL
166156  005703                            TST     R3
166160  100404                            BMI     EX900
166162  005702                            TST     R2
```

```
166164  001402                          BEQ     EX900
166166  020204                          CMP     R2,R4
166170  101723                          BLOS    EX600
166172                          EX900:
                                                                ;* * * SET TARGET BACK 2
166172  162702  000002                  SUB     #2,R2
                                                                ;* * * IF TYPE IS REGISTER
166176  105703                          TSTB    R3
166200  100010                          BPL     EX970
                                                                ;* * * THEN
                                                                ;* * * * RESTORE TARGET 1 TO REGISTER NO.
166202  160502                          SUB     R5,R2
166204  162702  177750                  SUB     #APLR00,R2
166210  006202                          ASR     R2
                                                                ;* * * * SET TARGET TO MOD 10
166212  022702  000011                  CMP     #11,R2
166216  101001                          BHI     EX950
166220  005002                          CLR     R2
166222                          EX950:
                                                                ;* * * END IF
166222                          EX970:
166222  000417                          BR      EX1300
                                                                ;* * ELSE
166224                          EX1000:
                                                                ;* * * SAVE TARGET
166224  010246                          MOV     R2,-(SP)
                                                                ;* * * IF TARGET IS REGISTER
166226  105703                          TSTB    R3
166230  100004                          BPL     EX1050
                                                                ;* * * THEN
                                                                ;* * * * SET TARGET =
                                                                ;       (2*TARRGET) + REGISTER 0 SAVE ADDRESS OFFSET
166232  006302                          ASL     R2
166234  062702  177750                  ADD     #APLR00,R2
166240  060502                          ADD     R5,R2
                                                                ;* * * END IF
166242                          EX1050:
                                                                ;* * * GET DATA PER TARGET AND PLACE IN DPR 2
166242  000241                          CLC
166244  011237  175004                  MOV     (R2),@#KW.SC2
                                                                ;* * * IF NXM
166250  000240                          NOP
166252  103002                          BCC     EX1200
                                                                ;* * * THEN
                                                                ;* * * * SET ERROR FLAG
166254  052703  100000                  BIS     #ERRFGB,R3
                                                                ;* * * END IF
166260                          EX1200:
                                                                ;* * * RESTORE TARGET
166260  012602                          MOV     (SP)+,R2
                                                                ;* * END IF
166262                          EX1300:
                                                                ;* * RETURN
166262  000207                          RETURN
                                                                ;* END EXAMINE
```

```
                .SBTTL  DISPST - DISPLAY STRING
                .ENABLE LC,GBL

;
; Module name: DISPST - DISPLAY STRING
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module is a general use utility that sends an ASCII string
;       out the console serial line. The display string is specified by
;       passing the address of the first character to the routine. The
;       end of the string is marked by a null character.

;INPUT PARAMETERS:

;       1) R1 - Start of string address

;OUTPUT PARAMETERS:

;       NONE

;DATA STRUCTURES USED:

;       1) SLU1 CSRs
```

```

         000015                    CR       ==:     15
         000012                    LF       ==:     12


  166264    077     000            ERROR::  .ASCIZ '?'
  166266    015     012     100    PROMPT:: .ASCIZ <CR><LF>'@'
  166272    040     000            SPACE::  .ASCIZ ' '
  166274    015     012     000    CRLF::   .ASCIZ <CR><LF>
  166277    015     000            CRDS::   .ASCIZ <CR>
  166301    057     000            OPENDS:: .ASCIZ '/'
  166303    122     060     000    REGDIS:: .ASCIZ 'R0'
  166306                                    .BLKB  1
  166307    122     061     000             .ASCIZ 'R1'
  166312                                    .BLKB  1
  166313    122     062     000             .ASCIZ 'R2'
  166316                                    .BLKB  1
  166317    122     063     000             .ASCIZ 'R3'
  166322                                    .BLKB  1
  166323    122     064     000             .ASCIZ 'R4'
  166326                                    .BLKB  1
  166327    122     065     000             .ASCIZ 'R5'
  166332                                    .BLKB  1
  166333    122     066     000             .ASCIZ 'R6'
  166336                                    .BLKB  1
  166337    122     067     000             .ASCIZ 'R7'
  166342                                    .BLKB  1
  166343    122     123     000             .ASCIZ 'RS'

                                           .EVEN

```

```
                                                          ;* PROCEDURE DISPLAY STRING
166346                        DISPST::
                                                          ;* * DO UNTIL NULL CHARACTER POINTED TO
166346                        DS100:
                                                          ;* * * DO UNTIL TRANSMIT BUFFER EMPTY
166346                        DS200:
                                                          ;* * * OR XOFF FLAG = 1
                                                          ;* * * * TEST FOR INPUT
166346  010046                        MOV     R0,-(SP)
166350  004767  005240                CALL    CHKCHR
166354  012600                        MOV     (SP)+,R0

                                                          ;* * * END UNTIL
166356  105737  177564                TSTB    @#XCSR1A
166362  100371                        BPL     DS200
166364  032703  000040                BIT     #XOFFGB,R3
166370  001366                        BNE     DS200
                                                          ;* * * SEND CHARACTER PER POINTER
166372  112137  177566                MOVB    (R1)+,@#XBUF1A
                                                          ;* * END UNTIL
166376  105711                        TSTB    (R1)
166400  001362                        BNE     DS100
                                                          ;* * RETURN FROM CALL
166402  000207                        RETURN
                                                          ;* END DISPLAY STRING
```

```
                .SBTTL  DATCOM - GET DATA/COMMAND
                .ENABLE LC,GBL

        ;
        ; Module name: DATCOM - GET DATA/COMMAND
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module is used by the serial ODT monitor. It poles SLU1
        ;       for valid data and returns when a CR, LF OR ^ are received
        ;       or on an error.

        ;Input Parameters:

        ;       NONE

        ;Output Parameters

        ;       1) R0 - COMMAND RECEIVED
        ;       2) R4 - INPUT DATA
        ;       3) R3 - BIT ERRFGB, ERROR FLAG: 1 = ON ERROR, NOT ALTERED IF NO ERRORS
        ;       4) R3 - BIT DARFGB, DATA RECV'D WITH COMMAND FLAG: 1 = YES, 0 = NO

        ;DATA STRUCTURES USED

        ;       NONE

        ;Routines Used

        ;       1) ACCUM
        ;       2) CHKCHR

```

```
                                                            ;* PROCEDURE GET DATA/COMMAND
166404                      DATCOM::
                                                            ;* * CLEAR DATA RECEIVED FLAG
                                                            ;* * CLEAR DATA/COMMAND DONE FLAG
166404  042703  000003              BIC     #<DARFGB ! DCDFGB>,R3
                                                            ;* * CLEAR ACCUMULATED NUMBER
166410  005004                      CLR     R4
                                                            ;* * DO UNTIL DONE FLAG SET
166412                      DA100:
                                                            ;* * * * DO UNTIL NON NUMERIC INPUT
166412                      DA200:
                                                            ;* * * * * GET INPUT
166412  004767  005134              CALL    GETIN
                                                            ;* * * * * IF NUMERIC INPUT
166416  103403                      BCS     DA370
                                                            ;* * * * * * ACCUMULATE NUMBER
166420  004767  005114              CALL    ACCUM
                                                            ;* * * * * * SET NON NUMERIC FLAG = 0
166424  000241                      CLC
                                                            ;* * * * * END IF
166426                      DA370:
                                                            ;* * * * END UNTIL
166426  103371                      BCC     DA200
                                                            ;* * * CASE CHARACTER INPUT
                                                            ;* * * WHEN CR
166430  122700  000015              CMPB    #15,R0
166434  001406                      BEQ     DA400
                                                            ;* * * OR LF
166436  122700  000012              CMPB    #12,R0
166442  001403                      BEQ     DA400
                                                            ;* * * OR ^
166444  122700  000136              CMPB    #'^,R0
166450  001003                      BNE     DA500
166452                      DA400:
                                                            ;* * * * SET DONE FLAG
166452  052703  000001              BIS     #DCDFGB,R3
166456  000402                      BR      DA600
                                                            ;* * * ELSE
166460                      DA500:
                                                            ;* * * * SET ERROR FLAG
                                                            ;* * * * SET DONE FLAG
166460  052703  100001              BIS     #<ERRFGB ! DCDFGB>,R3
                                                            ;* * * END CASE
166464                      DA600:
                                                            ;* * END UNTIL
166464  032703  000001              BIT     #DCDFGB,R3
166470  001750                      BEQ     DA100
166472                      DA700:
                                                            ;* * RETURN
166472  000207                      RETURN
                                                            ;* END GET DATA/COMMAND
```

```
        .SBTTL  DEPOS - DEPOSITE
        .ENABLE GBL,LC

;
; Module name: DEPOS - DEPOSIT
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module is used by both the Q bus ODT and the serial ODT monitors.
;       It deposits the passed data in the register or memory location
;       specified.

;Input Parameters:

;       1) R2 - ADDRESS
;       2) R4 - DATA
;       3) R3 bit REGFGB - Register flag

;Output Parameters

;       1) R3, BIT (ERRFGB) - ERROR FLAG, BIT = 1 (ERROR), BIT NOT ALTERED IF NO ERROR

;DATA STRUCTURES USED

;       1) USER REGISTER SAVE AREA

;Routines Used

;       NONE

;Global Symbols Defined

;       NONE
;
```

```
                                                                           ;* PROCEDURE
 166474                         DEPOS::
 166474  010246                         MOV     R2,-(SP)
                                                                           ;* * IF TARGET IS A REGISTER
 166476  105703                         TSTB    R3
 166500  100004                         BPL     DE50
                                                                           ;* * THEN
                                                                           ;* * * CONVERT REGISTER NO. TO ADDRESS
 166502  006302                         ASL     R2
 166504  060502                         ADD     R5,R2
 166506  062702  177750                 ADD     #APLR00,R2
                                                                           ;* * END IF
 166512                         DE50:
                                                                           ;* * DEPOSIT DATA AT SUPPLIED ADDRESS
 166512  000241                         CLC
 166514  010412                         MOV     R4,(R2)
 166516  000240                         NOP
                                                                           ;* * IF NXM
 166520  103002                         BCC     DE200
                                                                           ;* * THEN
                                                                           ;* * * SET ERROR FLAG
 166522  052703  100000                 BIS     #ERRFGB,R3
                                                                           ;* * END IF
 166526                         DE200:
                                                                           ;* * RETURN FROM CALL
 166526  012602                         MOV     (SP)+,R2
 166530  000207                         RETURN
                                                                           ;* END DEPOSIT
```

```
                .SBTTL  DISPWD - DISPLAY WORD
                .ENABLE LC,GBL

        ;
        ; Module name: DISPWD - DISPLAY WORD
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module is a general use utility that converts an 16 bit
        ;       value to an ASCII string and sends it out the console serial
        ;       line.

        ;INPUT PARAMETERS:

        ;       1) R1 - DATA WORD TO BE DISPLAYED

        ;OUTPUT PARAMETERS:

        ;       NONE

        ;DATA STRUCTURES USED:

        ;       1) SLU1 CSRs
```

```

                                                            ;* PROCEDURE DISPLAY WORD
166532                          DISPWD::
                                                            ;* * SAVE REGISTERS USED
166532  010046                          MOV     R0,-(SP)
166534  010246                          MOV     R2,-(SP)
                                                            ;* * SET CHARACTER COUNT = 6
166536  012702  000006                  MOV     #6,R2
                                                            ;* * SHIFT HIGH BIT OF DISPLAY WORD INTO
                                                            ;    NEXT NUMBER BUFFER
166542  005000                          CLR     R0
166544  006301                          ASL     R1
166546  005500                          ADC     R0
                                                            ;* * DO UNTIL CHARACTER COUNT = 0
166550                          DI100:
                                                            ;* * * CONVERT NEXT NUMBER BUFFER TO ASCII
166550  062700  000060                  ADD     #60,R0
                                                            ;* * * DO UNTIL TRANSMIT BUFFER EMPTY
166554                          DI200:
                                                            ;* * * OR XOFF FLAG = 1
                                                            ;* * * * CHECK FOR INPUT
166554  010046                          MOV     R0,-(SP)
166556  004767  005032                  CALL    CHKCHR
166562  012600                          MOV     (SP)+,R0
                                                            ;* * * END UNTIL
166564  105737  177564                  TSTB    @#XCSR1A
166570  100371                          BPL     DI200
166572  032703  000040                  BIT     #XOFFGB,R3
166576  001366                          BNE     DI200
                                                            ;* * * DISPLAY NUMBER
166600  110037  177566                  MOVB    R0,@#XBUF1A
                                                            ;* * * CLEAR NEXT NUBBER BUFFER
166604  005000                          CLR     R0
                                                            ;* * * SHIFT NEXT THREE HIGH BITS OF DISPLAY
                                                            ;      WORD ONTO NEXT NUMBER BUFFER
166606  006301                          ASL     R1
166610  005500                          ADC     R0
166612  006300                          ASL     R0
166614  006301                          ASL     R1
166616  005500                          ADC     R0
166620  006300                          ASL     R0
166622  006301                          ASL     R1
166624  005500                          ADC     R0
                                                            ;* * * DECREMENT CHARACTER COUNT
166626  005302                          DEC     R2
                                                            ;* * END UNTIL
166630  001347                          BNE     DI100
                                                            ;* * RESTORE REGISTERS
166632  012602                          MOV     (SP)+,R2
166634  012600                          MOV     (SP)+,R0
                                                            ;* * RETURN FROM CALL
166636  000207                          RETURN
                                                            ;* END DISPLAY WORD
```

```
        .SBTTL  ABAUD - AUTO BAUD
        .ENABLE LC,GBL

;
; Module name: ABAUD - AUTO BAUD
;
; System: KXT11-CA Native Firmware
;
;
;
;
; Functional Description:
;
;       This module performs the auto baud function on the console serial line.
;       It is used by the serial ODT monitor. The decision whether to attempt
;       to auto baud is made in this module. If the programmable baud rate
;       enable bit of the DLART is set or writing to the baud rate bits
;       does not change there value, the auto baud function is not performed.
;       If the auto baud function is performed this mode will continue to
;       attempt to determine the baud rate until it succeeds.
;
;       The auto baud function is performed by setting the baud rate to 2400
;       and comparing, the input character data td that of a table. This
;       character must be a CR and be sent at a baud rate within the range of
;       300 to 9600 (in doubling increments).
;
;       The baud rate table has two enteries for 4800 baud to eliminate the
;       4800 baud detection problem that exists for the Falcon.
;
;Input Parameters:
;
;       NONE
;
;Output Parameters
;
;       NONE
;
;DATA STRUCTURES
;
;       SLU1 CSR (XCSR1A)
;
;Routines Used
;
;       NONE

```

```
        000002                  B300    =       2               ;300 BAUD DLART RATE CODE
        000012                  B600    =       12              ;600  "    "    "    "
        000022                  B1200   =       22              ;1200 "    "    "    "
        000032                  B2400   =       32              ;2400 "    "    "    "
        000042                  B4800   =       42              ;4800 "    "    "    "
        000052                  B9600   =       52              ;9600 "    "    "    "


 166640    200                  INDATA: .BYTE   200             ;RECEIVE DATA AT 300 BAUD/BAUD RATE CODE
 166641    002                          .BYTE   B300
 166642    170                          .BYTE   170             ;  "    "    " 600    "    "    "
 166643    012                          .BYTE   B600
 166644    346                          .BYTE   346             ;  "    "    " 1200   "    "    "
 166645    022                          .BYTE   B1200
 166646    015                          .BYTE   15              ;  "    "    " 2400   "    "    "
 166647    032                          .BYTE   B2400
 166650    362                          .BYTE   362             ;  "    "    " 4800   "    "    "
 166651    042                          .BYTE   B4800
 166652    363                          .BYTE   363             ;  "    "    " 4800   "    "    "
 166653    042                          .BYTE   B4800
 166654    377                          .BYTE   377             ;  "    "    " 9600   "    "    "
 166655    052                  ENDTAB: .BYTE   B9600
                                        .EVEN
```

```
                                                                      ;* PROCEDURE AUTO BAUD
 166656                              ABAUD::
                                                                      ;* * IF DLART XCSR PBRE BIT = 0
 166656  032737  000002  177564              BIT     #2,@#XCSR1A
 166664  001041                              BNE     AB600
                                                                      ;* * THEN
                                                                      ;* * * SET PBE BIT = 1
 166666  052737  000002  177564              BIS     #2,@#XCSR1A
                                                                      ;* * * IF PBE READS BACK AS 1
 166674  032737  000002  177564              BIT     #2,@#XCSR1A
 166702  001432                              BEQ     AB500
                                                                      ;* * * THEN
                                                                      ;* * * * SET BAUD RATE TO 2400
 166704  012737  000032  177564              MOV     #B2400,@#XCSR1A
                                                                      ;* * * * DO UNTIL BAUD RATE FOUND
 166712                              AB100:
                                                                      ;* * * * * DELAY .25 SECONDS
 166712  005000                              CLR     R0
 166714  077001                              SOB     R0,.
                                                                      ;* * * * * CLEAR RECEIVE BUFFER
 166716  105737  177562                      TSTB    @#RBUF1A
                                                                      ;* * * * * DO UNTIL DATA RECEIVED
 166722                              AB200:
                                                                      ;* * * * * END UNTIL
 166722  105737  177560                      TSTB    @#RCSR1A
 166726  100375                              BPL     AB200
                                                                      ;* * * * * GET INPUT DATA
 166730  113700  177562                      MOVB    @#RBUF1A,R0
                                                                      ;* * * * * SET POINTER TO TOP OF TABLE
 166734  012701  166640                      MOV     #INDATA,R1
                                                                      ;* * * * * DO WHILE DATA <> (POINTER) OR LAST VALUE
 166740  120021                      AB300:  CMPB    R0,(R1)+
 166742  001405                              BEQ     AB400
 166744  022701  166655                      CMP     #ENDTAB,R1
 166750  001760                              BEQ     AB100   ;(NON STRUCTURED BRANCH FOR CODE OPIMIZATION)
                                                                      ;* * * * * * INCREMENT POINTER
 166752  005201                              INC     R1
                                                                      ;* * * * * END WHILE
 166754  000771                              BR      AB300
                                                                      ;* * * * END UNTIL (CONDITION TESTED ABOVE TO SAVE CODE)
 166756                              AB400:
                                                                      ;* * * * SET BAUD RATE PER TABLE
 166756  111137  177564                      MOVB    (R1),@#XCSR1A
                                                                      ;* * * * DELAY .5 SECONDS
 166762  005000                              CLR     R0
 166764  077001                              SOB     R0,.
 166766  077001                              SOB     R0,.

                                                                      ;* * * END IF
 166770                              AB500:
                                                                      ;* * END IF
 166770                              AB600:
                                                                      ;* * RETURN
 166770  000207                              RETURN
                                                                      ;* END AUTO BAUD
```

```

        .SBTTL  GOPROC - GO/PROCEFD
        .ENABLE LC,GBL

;
; Module name: GOPROC - GO/PROCEED
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module performs the go and the proceed functions for both
;       the serial and Q bus ODT. It has seperate entry points for
;       each Go and PROCEFD. The difference being that a PC value
;       is passed to the module for the GO function. This module
;       tests the user stack area and the user PC value. If a NXM is
;       encountered at the user SP the module returns from the call.
;       If no error is encountered control is passed to the user application.

;Input Parameters:

;       1) R0 = USER PC VALUE FOR GO COMMAND

;Output Parameters

;       NONE

;Routines Used

;       NONF

```

```
                                                                        ;* PROCEDURE GO/PROCEED
                                                                        ;* * IF GO
166772                          GO::
                                                                        ;* * THEN
                                                                        ;* * * CLEAR ADDRESS BIT0 OF TARGET
166772  042700  000001                  BIC     #BIT0,R0
                                                                        ;* * * SET APPLICATION PC = ADDRESS PASSED
166776  010065  177766                  MOV     R0,APLPC0(R5)
                                                                        ;* * * SET APLICATION PSW = 0
167002  005065  177770                  CLR     APLPS0(R5)
                                                                        ;* * * RESET IOP
167006  000005                          RESET
                                                                        ;* * END IF
167010                          PROC::
                                                                        ;* * TEST USER SP FOR NXM AND WRITABLE
167010  052737  010000  175002          BIS     #KX$NXF,@#KW.STA
167016  016500  177764                  MOV     APLSP0(R5),R0
167022  012701  052525                  MOV     #52525,R1
167026  000241                          CLC
                                                                        ;* * IF TEST OF USER SP PASSES

167030  010160  177776                  MOV     R1,-2(R0)
167034  000240                          NOP
167036  103457                          BCS     GP300
167040  010160  177774                  MOV     R1,-4(R0)
167044  000240                          NOP
167046  103453                          BCS     GP300
167050  020160  177776                  CMP     R1,-2(R0)
167054  001050                          BNE     GP300
167056  020160  177774                  CMP     R1,-4(R0)
167062  001045                          BNE     GP300
                                                                        ;* * THEN
                                                                        ;* * * SET LEDS TO NON NATIVE CODE STATE
167064  012700  000000                  MOV     #NNCED,R0
167070  004767  000104                  CALL    SETLED
                                                                        ;* * * ENABLE NXM'S TO TRAP TO 4
167074  042737  010000  175002          BIC     #KX$NXF,@#KW.STA
                                                                        ;* * * PUT APPLICATION PC AND PS ON APPLICATION STACK
167102  016500  177764                  MOV     APLSP0(R5),R0
167106  016540  177770                  MOV     APLPS0(R5),-(R0)
167112  016540  177766                  MOV     APLPC0(R5),-(R0)
167116  010065  177764                  MOV     R0,APLSP0(R5)
                                                                        ;* * * SET STACK POINTER TO APPL. R0 SAVE ADDR.
167122  012706  177750                  MOV     #APLR00,SP
167126  060506                          ADD     R5,SP
                                                                        ;* * * RESORE USER REGISTERS R0 - R5
167130  012600                          MOV     (SP)+,R0
167132  012601                          MOV     (SP)+,R1
167134  012602                          MOV     (SP)+,R2
167136  012603                          MOV     (SP)+,R3
167140  012604                          MOV     (SP)+,R4
167142  012605                          MOV     (SP)+,R5
                                                                        ;* * * RESTORE USER SP
167144  011606                          MOV     (SP),SP
                                                                        ;* * * ENABLE DUAL PORT RAM
167146  052737  000100  177530          BIS     #KX$DEN,@#KW.CSD
                                                                        ;* * * CLEAR SERIAL ODT, QBUS ODT  AND NXM FLAGS OF
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 54-1
GOPROC - GO/PROCEED

                                                                 ;       STATUS REGISTER
 167154  042737  010307  175002          BIC     #<KX$SOF ! KX$QOF ! KX$NXF ! KM$STF>,@#KW.STA
                                                                 ;* * * SET STATUS REGISTER TO NON NATIVE CODE STATE
 167162  052737  000007  175002          BIS     #KP$NNC,@#KW.STA
                                                                 ;* * * CLEAR COMMAND REGISTER
 167170  005037  175000                  CLR     @#KW.CMD
                                                                 ;* * * RETURN TO APPLICATION (NON STRUCTURED EXIT)
 167174  000002                          RTI
                                                                 ;* * END IF
 167176                          GP300:
                                                                 ;* * RETURN FROM CALL
 167176  000207                          RETURN
                                                                 ;* END GO/PROCEED
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 55
SETLED - SET LEDS

                                         .SBTTL  SETLED - SET LEDS
                                         .ENABLE LC,GBL

                                 ;
                                 ; Module name: SETLED - SET LEDS
                                 ;
                                 ; System: KXT11-CA Native Firmware
                                 ;
                                 ;
                                 ;
                                 ;
                                 ;
                                 ;
                                 ; Functional Description:
                                 ;
                                 ;       This mode sets the LED display to the state pass to it as a parameter.
                                 ;       It only changes the state of the LEDs if they are not currently
                                 ;       displaying an error or if the new display is to be Fatal Error.
                                 ;       There are two entry points to this routine. SETLED is the entry
                                 ;       used if the return address is on the stack. SETLDJ is used if there
                                 ;       is not a usable stack and the return address is in R1.
                                 ;
                                 ;Input Parameters:
                                 ;
                                 ;     1.  New LED State (R0)
                                 ;     2.  Return address if SETLDJ entry is used (R1)
                                 ;
                                 ;Output Parameters
                                 ;
                                 ;       NONE
                                 ;
                                 ;Data structures used
                                 ;
                                 ;     1.  LED State (KXTCSR C, bits 3 - 0)
                                 ;
                                 ;Routines Used
                                 ;
                                 ;       NONE
                                 ;
```

```
        000017          LS0     ==:     17              ;LED STATE 0
        000016          LS1     ==:     16              ; "     "  1
        000015          LS2     ==:     15              ; "     "  2
        000014          LS3     ==:     14              ; "     "  3
        000013          LS4     ==:     13              ; "     "  4
        000012          LS5     ==:     12              ; "     "  5
        000011          LS6     ==:     11              ; "     "  6
        000010          LS7     ==:     10              ; "     "  7
        000007          LS8     ==:     7               ; "     "  8
        000006          LS9     ==:     6               ; "     "  9
        000005          LS10    ==:     5               ; "     "  10
        000004          LS11    ==:     4               ; "     "  11
        000003          LS12    ==:     3               ; "     "  12
        000002          LS13    ==:     2               ; "     "  13
        000001          LS14    ==:     1               ; "     "  14
        000000          LS15    ==:     0               ; "     "  15

        000017          LEDTSD  ==:     LS0             ;LED TEST DISPLAY
        000017          FERRD   ==:     LS0             ;FATAL ERROR DISPLAY
        000010          NFERRD  ==:     LS7             ;NON FATAL ERROR DISPLAY
        000011          UROM0D  ==:     LS6             ;USER ROM 0 FAILED DISPLAY
        000012          UROM1D  ==:     LS5             ;USER ROM 1 FAILED DISPLAY
        000013          S1LBD   ==:     LS4             ;SLU1 LOOP BACK TEST FAILED DISPLAY
        000014          S2ALBD  ==:     LS3             ;SLU2 CHAN A LOOP BACK TEST FAILED DISPLAY
        000015          S2BLBD  ==:     LS2             ;SLU2 CHAN B LOOP BACK TEST FAILED DISPLAY
        000016          PARLBD  ==:     LS1             ;PARALLEL LOOP BACK TEST FAILED DISPLAY

        000007          PUTSDS  ==:     LS8             ;POWER UP TESTS RUNNING
        000006          CSTMD   ==:     LS9             ;CONTINUOUS SELF TEST MODE DISPLAY
        000005          QODTMD  ==:     LS10            ;Q BUS ODT MODE DISPLAY
        000004          SODTMD  ==:     LS11            ;SERIAL ODT MODE DISPLAY
        000003          WAITD   ==:     LS12            ;WAIT MODE DISPLAY
        000002          DMAED   ==:     LS13            ;EXECUTING DMA LOAD DISPLAY
        000001          PRIBD   ==:     LS14            ;EXECUTING PRIMARY BOOTSTRAP DISPLAY
        000000          NNCED   ==:     LS15            ;EXECUTING NON NATIVE CODE DISPLAY
```

SETLED - SET LEDS

```

                                                                                ;* PROCEDURE SET LEDS
                                                                                ;* * IF CALLED
167200                          SETLED::
                                                                                ;* * THEN
                                                                                ;* * * SET RETURN ADDRESS/FLAG = 0
167200  005001                                  CLR     R1
                                                                                ;* * END IF
167202                          SETLDJ::
                                                                                ;* * IF NEW DISPLAY IS FATAL ERROR
167202  022700  000017                          CMP     #FERRD,R0
167206  001405                                  BEQ     SE100
                                                                                ;* * OR LEDS NOT DISPLAYING ERROR
167210  132737  000010  177524                  BITB    #KX$LD3,@#KW.CSC
167216  001401                                  BEQ     SE100
167220  000410                                  BR      SE200
                                                                                ;* * THEN
167222                          SE100:
                                                                                ;* * * SET LED TO STATE PASSED TO ROUTINE (R0)
167222  110037  177524                          MOVB    R0,@#KW.CSC
                                                                                ;* * * IF LEDS SET TO FATAL ERROR
167226  122700  000017                          CMPB    #FERRD,R0
167232  001003                                  BNE     SE150
                                                                                ;* * * THEN
                                                                                ;* * * * SET FATAL ERROR FLAG OF STATUS REGISTER
167234  052737  000040  175002                  BIS     #KX$FEF,@#KW.STA
                                                                                ;* * * END IF
167242                          SE150:
                                                                                ;* * END IF
167242                          SE200:
                                                                                ;* * IF RETURN ADDRESS = 0
167242  005701                                  TST     R1
167244  001001                                  BNE     SE300
                                                                                ;* * THEN
                                                                                ;* * * RETURN FROM CALL
167246  000207                                  RETURN
                                                                                ;* * ELSE
167250                          SE300:
                                                                                ;* * * JUMP TO RETURN ADDRESS
167250  000111                                  JMP     (R1)
                                                                                ;* END SET LEDS

```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 58
TRAP4 - TRAP 4 EMULATOR

                                        .SBTTL  TRAP4 - TRAP 4 EMULATOR

                                        .ENABLE LC,GBL
                                ;
                                ; Module name: TRAP4 - TRAP 4 EMULATOR MODULE
                                ;
                                ; System: KXT11-CA Native Firmware
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ; Functional Description:
                                ;
                                ;       This module emulates the Trap to 4. This emulation
                                ;       is performed by placing PSW and the vector address onto the stack
                                ;       and performing an RTI. If the stack pointer is not pointing
                                ;       to writeable memory (but not NXM) this routine will force a
                                ;       stack NXM situation where the Restart Interrupt handler (RESTRT)
                                ;       will re-establish the stack and retry the trap.
                                ;
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 59
TRAP4 - TRAP 4 EMULATOR

                                                                ;* PROCEDURE TRAP4
167252                          TRAP4::
                                                                ;* * SET UP STACK FOR TRAP TO 4
167252  013746  000006                  MOV     @#6,-(SP)
167256  013746  000004                  MOV     @#4,-(SP)
                                                                ;* * IF STACK IS POINTING TO RAM
167262  023766  000006  000002          CMP     @#6,2(SP)
167270  001004                          BNE     TR410
167272  023716  000004                  CMP     @#4,(SP)
167276  001001                          BNE     TR410
                                                                ;* * THEN
                                                                ;* * * EMULATE TRAP (NON STRUCTURED EXIT)
167300  000002                          RTI
                                                                ;* * ELSE
167302                          TR410:
                                                                ;* * * FAKE A STACK NXM SITUATION (NON STRUCTURED EXIT)
167302  052737  004000  175002          BIS     #KX$SXT,@#KW.STA
167310  005737  177776                  TST     @#177776    ;(FORCE A NXM RESTART INT.)
                                                                ;* * END IF
                                                                ;* END TRAP4
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 60
TRAP10 - TRAP 10 EMULATOR

                                   .SBTTL  TRAP10 - TRAP 10 EMULATOR

                                   .ENABLE LC,GBL
                         ;
                         ; Module name: TRAP10 - TRAP 10 EMULATOR MODULE
                         ;
                         ; System: KXT11-CA Native Firmware
                         ;
                         ;
                         ;
                         ;
                         ;
                         ;
                         ; Functional Description:
                         ;
                         ;       This module emulates the Trap to 10. The trap emulation
                         ;       is performed by placing the vector address and PSW on the stack
                         ;       and performing an RTI. If the stack pointer is not pointing
                         ;       to writable memory (but not NXM) this routines will force a
                         ;       stack NXM situation where the Restart Interrupt handler (RESTRT)
                         ;       will re-establish the stack and retry the trap.

```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 61
TRAP10 - TRAP 10 EMULATOR

                                                               ;* PROCEDURE TRAP4
167314                        TRAP10::
                                                               ;* * SET UP STACK FOR TRAP TO 4
167314  013746  000012                     MOV     @#12,-(SP)
167320  013746  000010                     MOV     @#10,-(SP)
                                                               ;* * IF STACK IS POINTING TO RAM
167324  023766  000012  000002             CMP     @#12,2(SP)
167332  001004                             BNE     TR1010
167334  023716  000010                     CMP     @#10,(SP)
167340  001001                             BNE     TR1010
                                                               ;* * THEN
                                                               ;* * * EMULATE TRAP (NON STRUCTURED EXIT)
167342  000002                             RTI
                                                               ;* * ELSE
167344                        TR1010:
                                                               ;* * * FAKE A STACK NXM SITUATION (NON STRUCTURED EXIT)
167344  052737  004000  175002             BIS     #KX$SXT,@#KW.STA
167352  005737  177776                     TST     @#177776    ;(FORCE A NXM RESTART INT.)
                                                               ;* * END IF
                                                               ;* END TRAP10
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 62
TRAP24 - TRAP 24 EMULATOR

                                      .SBTTL  TRAP24 - TRAP 24 EMULATOR

                                      .ENABLE LC,GBL
                              ;
                              ; Module name: TRAP24 - TRAP 24 EMULATOR MODULE
                              ;
                              ; System: KXT11-CA Native Firmware
                              ;
                              ;
                              ;
                              ;
                              ;
                              ;
                              ; Functional Description:
                              ;
                              ;       This module emulates the Trap to 24. This emulation
                              ;       is performed by placing data at locations 26 into the PSW
                              ;       and 24 int the PC.
                              ;
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 63
TRAP24 - TRAP 24 EMULATOR

                                                              ;* PROCEDURE TRAP24
 167356                       TPAP24::
                                                              ;* * MOVE CONTENTS OF 26 TO PSW
 167356  106437  000026               MTPS    @#26
                                                              ;* * MOVE CONTENTS OF 24 TO PC
 167362  013707  000024               MOV     @#24,PC
                                                              ;* END TRAP24
```

```
                                        .SBTTL  TRAPX - TRAP COMMAND EMULATOR

                                        .ENABLE LC,GBL
                                ;
                                ; Module name: TRAPX - TRAP COMMAND EMULATOR MODULE
                                ;
                                ; System: KXT11-CA Native Firmware
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ; Functional Description:
                                ;
                                ;       This module emulates the Trap specified by the DPR. This emulation
                                ;       is performed by placing data in DPR register 2 into the PSW
                                ;       and DPR register 3 into the PC.
                                ;
                                                                ;* PROCEDURE TRAPX
167366                          TRAPX::
                                                                ;* * CLEAR COMMAND REGISTER
167366  005037  175000                  CLR     @#KW.CMD
                                                                ;* * SET PSW TO DPR REGISTER 3
167372  106437  175006                  MTPS    @#KW.SC3
                                                                ;* * JUMP TO ADDRESS SPEC. IN DPR REGISTER 2
167376  013707  175004                  MOV     @#KW.SC2,PC
                                                                ;* END TRAPX

```

```
        .SBTTL  WAITST - WAIT FOR COMMAND STATE
        .ENABLE LC,GBL

;
; Module name: WAITST
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module is used to place the IOP in the wait state. In this state
;       the IOP waits for a command to be received from the Q bus.
;       The LED if not displaying an error will be set to the wait state
;       display. The state field of status register of the DPR will be
;       set to the wait state if the Root/Self Test Switch is not set to 10.
;       If the switch is set to 10 then the state field is set to the Q Bus
;       controlled test state. The routine continues to loop setting the LEDs
;       and the state field of the status register so that if the serial ODT
;       mode is entered via Break, while in this routine and a Proceed
;       command is executed the LED display and state field will be correct.
;
;Input Parameters:
;
;       NONE
;
;Output Parameters
;
;       NONE
;
;Routines Used
;
;       NONE
```

```
                                                            ;* PROCEDURE WAITST
167402                         WAITST::
                                                            ;* * DO UNTIL INTERRUPTED
167402                         WA100:
                                                            ;* * * SET LED DISPLAY TO WAIT STATE
167402  012700  000003                 MOV     #WAITD,R0
167406  004767  177566                 CALL    SETLED
                                                            ;* * * GET STATUS REGISTER
167412  013700  175002                 MOV     @#KW.STA,R0
167416  042700  000007                 BIC     #KM$STF,R0
                                                            ;* * * IF THE BOOT/SELF TEST SWITCH = 10
167422  113701  177522                 MOVB    @#KL.CSB,R1
167426  142701  177417                 BICB    #^C<KM$SWS>,R1
167432  122701  000240                 CMPB    #<10.*16.>,R1
167436  001003                         BNE     WA200
                                                            ;* * * THEN
                                                            ;* * * * SET STATE FIELD OF STATUS REG. TO
                                                            ;         QBUS CONTROLLED TEST STATE
167440  052700  000002                 BIS     #KP$QTS,R0
167444  000402                         BR      WA300
167446                         WA200:
                                                            ;* * * ELSE
                                                            ;* * * * SET STATE FIELD OF STATUS REG. TO WAIT STATE
167446  052700  000004                 BIS     #KP$WST,R0
                                                            ;* * * END IF
167452                         WA300:
                                                            ;* * * * WRITE STATUS REGISTER DATA
167452  010037  175002                 MOV     R0,@#KW.STA
                                                            ;* * END UNTIL
167456  000751                         BR      WA100
                                                            ;* END WAITST
```

```
                .SBTTL  QCOMIN - Q BUS COMMAND INTERP.
                .ENABLE GBL,LC

        ;
        ; Module name: QCOMIN - Q BUS COMMAND INTERPRITOR
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module decodes the Q bus commands received through the command
        ;       register of the dual port RAM. Once a valid command is recognised
        ;       the appropriate routine is called. If the command received is a test
        ;       command the test command decode routine (DECTST) is jumped to.
        ;
        ;Input Parameters:
        ;
        ;       NONF
        ;
        ;Output Parameters
        ;
        ;       NONF
        ;
        ;Data Structures Used
        ;
        ;       1.  IOP COMMAND RFGISTER
        ;
        ;       2.  IOP STATUS RECISTER
        ;
        ;Routines Used
        ;
        ;       1.  REINIT (JMP)
        ;
        ;       2.  DECTST (JMP)
        ;
        ;       3.  WAITST (JMP)
```

```
                                                                ;* PROCEDURE Q BUS COMMAND INTERPRETER
167460                          QCOMIN::
                                                                ;* * CLEAR COMMAND ERROR AND DMA LOAD ERROR
                                                                ;               BITS OF STATUS REGISTER
167460  042737  140000  175002          BIC     #<KX$CME ! KX$DME>,@#KW.STA
                                                                ;* * IF IN SERIAL ODT MODE
167466  032737  000100  175002          BIT     #KX$SOF,@#KW.STA
167474  001406                          BEQ     QC50
                                                                ;* * THEN
                                                                ;* * * SET COMMAND ERROR BIT OF STATUS REGISTER
167476  052737  100000  175002          BIS     #KX$CME,@#KW.STA
                                                                ;* * * CLEAR COMMAND REGISTER
167504  005037  175000                  CLR     @#KW.CMD
                                                                ;* * * RETURN FORM INTERRUPT
167510  000002                          RTI             ;(NON STRUCTURED EXIT)
                                                                ;* * END IF
167512                          QC50:
                                                                ;* * IF QODT FLAG OF STATUS REGISTER  = 1
167512  032737  000200  175002          BIT     #KX$QOF,@#KW.STA
167520  001401                          BEQ     QC70
                                                                ;* * THEN
                                                                ;* * * UNBLOCK QODT WAIT FOR COMMAND
167522  000002                          RTI             ;(NON STRUCTURED EXIT)
                                                                ;* * END IF
167524                          QC70:
                                                                ;* * IF TEST/COMMAND BIT = 0
167524  032737  100000  175000          BIT     #BIT15,@#KW.CMD
167532  001147                          BNE     QC500
                                                                ;* * THEN
                                                                ;* * * CASE COMMAND
                                                                ;* * * WHEN Q BUS ODT
167534  022737  000010  175000          CMP     #KX$ODC,@#KW.CMD
167542  001006                          BNE     QC100
                                                                ;* * * AND STATUS REGISTER SERIAL ODT FLAG = 0
167544  032737  000100  175002          BIT     #KX$SOF,@#KW.STA
167552  001002                          BNE     QC100
                                                                ;* * * * ENTER ODT Q BUS ODT MODE
                                                                ;      (NON STRUCTURED JUMP TO Q BUS ODT MONITOR)
167554  000167  001652                  JMP     QODTM
                                                                ;* * * WHEN RE-INITIALIZE
167560  022737  000004  175000  QC100:  CMP     #KX$INC,@#KW.CMD
167566  001024                          BNE     QC200
                                                                ;* * * AND DPR REGISTER 3 = 0 THROUGH 6 AND ID
                                                                ;               SWITCH < 7
167570  022737  000006  175006          CMP     #6,@#KW.SC3
167576  103407                          BLO     QC120
167600  013700  177522                  MOV     @#KW.CSB,R0
167604  042700  177417                  BIC     #^C<KM$SWS>,R0
167610  022700  000160                  CMP     #<7*16.>,R0
167614  101004                          BHI     QC150
                                                                ;* * * OR DPR REGISTER 3 = 8
167616                          QC120:
167616  022737  000010  175006          CMP     #8.,@#KW.SC3
167624  001005                          BNE     QC200
167626                          QC150:
                                                                ;* * * * DISABLE DUAL PORT PAM
167626  042737  000100  177520          BIC     #KX$DEN,@#KW.CSA
```

```
                                                        ;* * * * RE-INITIALIZE
                                                        ;      (NON STRUCTURED JUMP TO POWERUP MODULE RE-INITIALIZE
167634  000167  003156                  JMP     REINIT
                                                        ;* * * WHEN DMA LOAD
167640  022737  000002  175000  QC200:  CMP     #KX$DMC,@#KW.CMD
167646  001004                          BNE     QC300
                                                        ;* * * * REMOVE RETURN ADDR. AND PS FROM STACK
167650  062706  000004                  ADD     #4,SP
                                                        ;* * * * PERFORM DMA TRANSFER
                                                        ;      (NON STRUCTURED JUMP TO DMA LOADER MODULE)
167654  000167  001152                  JMP     DMALD
                                                        ;* * * WHEN TRAP COMMAND
167660  022737  000001  175000  QC300:  CMP     #KX$TRC,@#KW.CMD
167666  001023                          BNE     QC400
                                                        ;* * * * IF PC VALUE IN DPR REGISTER 2 IS EVEN
167670  032737  000001  175004          BIT     #BIT0,@#KW.SC2
167676  001016                          BNE     QC380
                                                        ;* * * * THEN
                                                        ;* * * * * SET LEDS TO NON NATIVE CODE STATE
167700  012700  000000                  MOV     #NNCED,R0
167704  012701  167714                  MOV     #QC350,R1
167710  000167  177266                  JMP     SETLQJ
167714                          QC350:
                                                        ;* * * * * SET STATUS REGISTER NON NATIVE CODE STATE
167714  042737  000007  175002          BIC     #KM$STF,@#KW.STA
167722  052737  000007  175002          BIS     #KP$NNC,@#KW.STA
                                                        ;* * * * * EMULATE TRAP AT ADDRESS
                                                        ;          PER DUAL PORT REGISTER 2
                                                        ;          SET PSW PER DUAL PORT REGISTER 3
                                                        ;          (NON STRUCTURED EXIT)
167730  000167  177432                  JMP     TRAPX
                                                        ;* * * * END IF
167734  000442                  QC380:  BR      QC450
                                                        ;* * * WHEN SHOW STATUS COMMAND
167736                          QC400:
167736  022767  000020  005034          CMP     #KX$SHC,KW.CMD
167744  001027                          BNE     QC447
                                                        ;* * * * PLACE BOOT SWITCH AND MAP SWITCH
                                                        ;              VALUES TO DPR REGISTER 3
167746  113737  177522  175006          MOVB    @#KL.CSB,@#KL.SC3
                                                        ;* * * * CLEAR ALL NON RELATED BITS
167754  042737  177401  175006          BIC     #^C<KM$MAP ! KM$SWS>,@#KW.SC3
                                                        ;* * * * IF USER SOCKETS MAPPED LOW
167762  032737  000010  177522          BIT     #KX$MP2,@#KL.CSB
167770  001402                          BEQ     QC420
                                                        ;* * * * THEN
                                                        ;* * * * * TEST FOR RAM AT 0
167772  005000                          CLR     R0
167774  000402                          BR      QC440
                                                        ;* * * * ELSE
167776                          QC420:
                                                        ;* * * * * TEST FOR RAM AT 100000
167776  012700  100000                  MOV     #100000,R0
                                                        ;* * * * END IF
170002                          QC440:
                                                        ;* * * * IF RAM PRESENT
170002  011001                          MOV     (R0),R1
```

```
170004  005110                                 COM     (R0)
170006  020110                                 CMP     R1,(R0)
170010  001404                                 BEQ     QC445
                                                                       ;* * * * THEN
                                                                       ;* * * * * RESTORE LOCATION
170012  010110                                 MOV     R1,(R0)
                                                                       ;* * * * * SET RAM PRESENT BIT OF DPR REG. 3
170014  052737  000001  175006                 BIS     #BIT0,@#KW.SC3
                                                                       ;* * * * END IF
170022                                 QC445:
170022  000412                                 BR      QC470
                                                                       ;* * * WHEN NOP COMMAND
170024                                 QC447:
170024  022737  000040  175000                 CMP     #KX$NOP,@#KW.CMD
170032  001003                                 BNE     QC450
                                                                       ;* * * * CLEAR COMMAND REGISTER
170034  005037  175000                         CLR     @#KW.CMD
                                                                       ;* * * * RETURN FROM INTERRUPT
170040  000002                                 RTI     ;(NON STRUCTURED EXIT)
                                                                       ;* * * ELSE (BAD COMMAND)
170042                                 QC450:
                                                                       ;* * * * SET COMMAND ERROR BIT OF STATUS REGISTER
170042  052737  100000  175002                 BIS     #KX$CME,@#KW.STA
                                                                       ;* * * END CASE
170050                                 QC470:
170050  000402                                 BR      QC600
                                                                       ;* * ELSE (TEST/COMMAND BIT = 1)
170052                                 QC500:
                                                                       ;* * * DECODE TEST COMMAND (NON-STRUCTURED JUMP TO DECTST)
170052  000167  000014                         JMP     DECTST
                                                                       ;* * END IF
170056                                 QC600:
                                                                       ;* * REMOVE RETURN ADDRESS AND PSW FROM STACK
170056  062706  000004                         ADD     #4,SP
                                                                       ;* * CLEAR COMMAND REGISTER
170062  005037  175000                         CLR     @#KW.CMD
                                                                       ;* * ENTER WAIT STATE (NON STRUCTURED EXIT)
170066  000167  177310                         JMP     WAITST
                                                                       ;* END Q BUS COMMAND INTERPRETER
```

```
        .SBTTL  DECTST - DECODE AND DISPATCH QBUS SELF TEST REQUESTS
        .ENABL  GBL


;
; MODULE NAME: DECTST - DECODE AND DISPATCH QBUS SELF TEST REQUESTS
;
;
;
;
;
;
; FUNCTIONAL DESCRIPTION:
;
;       THIS MODULE DECODES THE QBUS SELF TEST REQUESTS AND DISPATCHES TO THE
;       APPROPRIATE SELF TEST MODULE. ON RETURN FROM THE SELF TEST MODULE,
;       DECTST UPDATES THE LED DISPLAY, UPDATES THE ACCUMULATED ERROR WORD,
;       AND PERFORMS THE ACKNOWLEDGEMENT PORTION OF THE PROTOCOL TO NOTIFY THE
;       ARBITER OF COMPLETION.
;
; INPUTS:
;       TPR 0 CONTAINS THE SELF TEST REQUEST CODE.
;       TPR 3 CONTAINS AN OPTIONAL CALLING PARAMETER, IF ANY.
;
; CALLING SEQUENCE:
;       A SPECIAL CASE JMP TO DECTST FROM QCOMIN.
;
; OUTPUTS:
;       TPR 0 IS CLEAR.
;       TPR 1 IS THE SAME AS WHEN CALLED UNLESS DECTST SETS THE COMMAND ERROR
;               BIT, BIT 15. IF THE BOOT SWITCH SELECTION IS 10 (QBUS
;               CONTROLLED TEST), THE STATE FIELD IS SET TO "WAITING FOR
;               QBUS TEST COMMAND". ELSE, THE STATE FIELD IS SET TO "WAITING
;               FOR QBUS COMMAND".
;       TPR 2 HAS ITS APPROPRIATE TEST BIT CLEARED IF NO ERROR OCCURRED
;               AND SET IF AN ERROR OCCURRED. TPR 2 IS THE ACCUMULATED SELF
;               TEST RESULTS.
;       IF TPR 2'S APPROPRIATE ERROR BIT IS SET, TPR 3 CONTAINS THE TEST CODE
;               IN BITS <15:12> AND THE DISCRETE ERROR CODE IN BITS <11:00>.
;
;       IF AN ERROR OCCURRED, THE LEDS ARE UPDATED WITH A CALL TO TNNERR.
;
; ROUTINES USED:
;       SPR5SU, TNNERR, & THE VARIOUS SELF TEST ROUTINES
;
; OTHER NOTES AND COMMENTS:
;
;       1. THE ARBITER IS RESPONSIBLE FOR CLEARING THE ACCUMULATED VALUE IN
;       TPR 2.
;
;       3. ALL REGISTERS ARE USEABLE WITHOUT RESTORATION.
```

```
170072                          DECTST::
170072  012700  170102                  MOV     #3$,R0            ;SET UP A RETURN ADDRESS FROM SPR5SU.
170076  000137  173706                  JMP     @#SPR5SU          ;GO SET UP R5 AND A DEFAULT STACK POINTER.
170102                          3$:
170102  012700  170354                  MOV     #STSTAB,R0        ;POINT TO THE SELF TEST DISPATCH TABLE.
170106  005710                  5$:     TST     (R0)              ;ARE WE AT THE END OF THE TABLE?
170110  001005                          BNE     10$               ;NO.
170112  013700  175002                  MOV     @#KW.STA,R0       ;YES, GET TPR 1.
170116  052700  100000                  BIS     #KX$CME,R0        ;SET THE COMMAND ERROR BIT IN THE SAVED TPR 1.
170122  000465                          BR      45$               ;GO HANDLE THE REST OF UPDATING TPR 1 & TPR 0.
170124  021037  175000          10$:    CMP     (R0),@#KW.CMD     ;NO, DOES THE TABLE ENTRY MATCH TPR 0?
170130  001403                          BEQ     15$               ;YES.
170132  062700  000010                  ADD     #STSENL,R0        ;NO, ADD THE LENGTH OF A TABLE ENTRY TO THE
                                                                  ;POINTER.
170136  000763                          BR      5$                ;GO TRY THE NEXT ENTRY IN THE TABLE.
170140                          15$:                              ;WE HAVE FOUND A VALID COMMAND, SO SAVE
170140  013746  175002                  MOV     @#KW.STA,-(SP)    ;TPR 1 (THE STATUS REGISTER),
170144  013746  175004                  MOV     @#KW.SC2,-(SP)    ;TPR 2,
170150  012046                          MOV     (R0)+,-(SP)       ;THE COMMAND (WITH BIT 15 CLEARED),
170152  042716  100000                  BIC     #BIT15,(SP)
170156  012046                          MOV     (R0)+,-(SP)       ;THE ERROR CONDITION ROUTINE'S ADDRESS ON
                                                                  ;THE STACK,
170160  013746  175006                  MOV     @#KW.SC3,-(SP)    ;TPR 3, AND
170164  013746  177524                  MOV     @#KW.CSC,-(SP)    ;THE LEDS.
170170  005001                          CLR     R1                ;CLEAR R1 TO DEFAULT TO NO PARAMETER IN TPR 3.
170172  105720                          TSTB    (R0)+             ;NOW, IS THERE A PARAMETER IN TPR 3?
170174  001402                          BEQ     20$               ;NO, R1 HAS ALREADY BEEN CLEARED.
170176  013701  175006                  MOV     @#KW.SC3,R1       ;YES, PUT THE PARAMETER IN R1.
170202  105720                  20$:    TSTB    (R0)+             ;IS THE TEST ROUTINE IN OVERLAY 0?
170204  001404                          BEQ     25$               ;YES.
170206  052737  000040  177520          BIS     #KX$DPE,@#KW.CSA ;NO, IT'S IN OVERLAY 1, SO ENABLE THE
                                                                  ;OVERLAY 1 MAP.
170214  000403                          BR      30$
170216  042737  000040  177520  25$:    BIC     #KX$DPE,@#KW.CSA ;IT'S IN OVERLAY 0, SO DISABLE THE
                                                                  ;OVERLAY 1 MAP.
170224  052737  010000  175002  30$:    BIS     #KX$NXF,@#KW.STA ;ENABLE C-BIT SETTING ON REFERENCES TO
                                                                  ;NON-EXISTENT MEMORY (RATHER THAN A TRAP THRU
                                                                  ;4). KX$NXF WILL BE RESTORED WHEN TPR 1 IS
                                                                  ;RESTORED.
170232  004730                          JSR     PC,@(R0)+         ;CALL THE TEST ROUTINE VIA THE "VECTOR".
170234  012637  177524                  MOV     (SP)+,@#KW.CSC    ;RESTORE THE LEDS.
170240  005700                          TST     R0                ;WAS AN ERROR DETECTED?
170242  001406                          BEQ     35$               ;NO.
170244  010037  175006                  MOV     R0,@#KW.SC3       ;YES, SET TPR 3 FOR THIS ERROR.
170250  012601                          MOV     (SP)+,R1          ;PUT THE VALUE OF TPR 3, WHEN CALLED, IN R1
                                                                  ;(IN CASE THE LED HANDLING ROUTINE NEEDS IT).
170252  004736                          JSR     PC,@(SP)+         ;YES, GO TO THE LED UPDATING ROUTINE.
170254  052616                          BIS     (SP)+,(SP)        ;SET THE TEST'S ERROR BIT IN THE SAVED TPR 2.
170256  000404                          BR      40$               ;GO FINISH UP TPR 2, 1, AND 0 UPDATING.
170260  012637  175006          35$:    MOV     (SP)+,@#KW.SC3    ;RESTORE TPR 3.
170264  005726                          TST     (SP)+             ;FLUSH THE ERROR HANDLER ADDRESS OFF THE
                                                                  ;STACK.
170266  042616                          BIC     (SP)+,(SP)        ;CLEAR THE TEST'S ERROR BIT IN THE SAVED
                                                                  ;TPR 2.
170270  012637  175004          40$:    MOV     (SP)+,@#KW.SC2    ;STORE THE RESULT IN TPR 2.
170274  012600                          MOV     (SP)+,R0          ;GET THE SAVED TPR 1.
170276  042700  000007          45$:    BIC     #KM$STF,R0        ;CLEAR THE SAVED STATE FIELD.
```

```
170302  013701  177522                    MOV     @#KW.CSB,R1        ;GET THE BOOT SWITCH SETTING FROM KXTCSPB BITS
                                                                     ;4 TO 7.
170306  042701  177417                    BIC     #^C<KM$SWS>,R1     ;ISOLATE THE SWITCH SETTING BITS.
170312  022701  000240                    CMP     #10.*20,R1         ;IS THE SWITCH SETTING FOR "QBUS CONTROLLED
                                                                     ;TESTS"?
170316  001003                            BNE     47$                ;NO.
170320  052700  000002                    BIS     #KP$QTS,R0         ;YES, SET THE STATE TO "QBUS CONTROLLED TEST
                                                                     ;STATE".
170324  000402                            BR      50$
170326  052700  000004            47$:    BIS     #KP$WST,R0         ;SET THE STATE TO "WAITING FOR QBUS COMMAND".
170332  010037  175002            50$:    MOV     R0,@#KW.STA        ;RESTORE THE STATUS REGISTER.
170336  005037  175000                    CLR     @#KW.CMD           ;CLEAR THE COMMAND REGISTER, TPR 0.
170342  052737  000100  177530            BIS     #KX$DEN,@#KW.CSD ;RE-ENABLE THE TWO PORT RAM (SOMETIMES IT'S
                                                                     ;NEEDED, SOMETIMES IT'S NOT, BUT DO IT
                                                                     ;ALWAYS).
170350  000137  167402                    JMP     @#WAITST           ;GO TO THE WAIT STATE AND WAIT FOR THE NEXT
                                                                     ;QBUS COMMAND.

                                  ;STSTAB, THE SELF TEST DISPATCH TABLE, DEFINES THE LEGAL SELF TEST CODES, THE
                                  ;EXISTENCE OF A CALLING PARAMETER IN TPR 3, THE "ROM PAGE" CONTAINING THE TEST
                                  ;CODE, AND THE ADDRESS OF THE CODE.
                                  ;
                                  ;       .WORD   TPR 0 VALUE OR 0            ;TPR 0 VALUE IF LEGAL DISPATCH CODE &
                                  ;                                           ;0 IF END OF LIST (THUS IMPLYING
                                  ;                                           ;COMMAND ERROR)
                                  ;
                                  ;       .WORD   TNNERR                      ;ADDRESS OF ROUTINE THAT HANDLES LED
                                  ;                                           ;UPDATING IF AN ERROR IS INDICATED.
                                  ;
                                  ;       .BYTE   0 OR NON-0                  ;NON-0 IMPLIES THAT A CALLING
                                  ;                                           ;PARAMETER IS IN TPR 3. 0
                                  ;                                           ;IMPLIES NO PARAMETER IN TPR 3.
                                  ;
                                  ;       .BYTE   0 OR NON-0                  ;NON-0 IMPLIES THAT THE TEST ROUTINE
                                  ;                                           ;IS IN THE HIGH OVERLAY, OVERLAY 1, OF
                                  ;                                           ;ROM, THUS IMPLYING THAT KXTCSRA
                                  ;                                           ;BIT 5 MUST BE 1 TO MAP THE CODE
                                  ;                                           ;TO THE 160000 BASE ADDRESS. 0
                                  ;                                           ;IMPLIES THAT THE TEST ROUTINE IS
                                  ;                                           ;IN THE LOW OVERLAY, OVERLAY 0,
                                  ;                                           ;OF ROM, THUS KXTCSRA BIT 05 MUST
                                  ;                                           ;BE 0 TO MAP THE CODE TO THE
                                  ;                                           ;160000 BASE ADDRESS.
                                  ;
                                  ;       .WORD   ADDRESS                     ;THE ADDRESS OF THE ROUTINE'S "VECTOR"
                                  ;                                           ;IN THE FIRST 32 (DECIMAL) BYTES OF
                                  ;                                           ;THE ROM OVERLAY.
                                  ;
170354                            STSTAB:
                                  ;CSR TEST
170354  100001                            .WORD   KX$T01                      ;TPR 0 VALUE
170356  170506                            .WORD   T1ERR                       ;LED UPDATING ROUTINE ON ERROR
170360     000                            .BYTE   0                           ;NO INPUT PARAMETER
170361     000                            .BYTE   0                           ;IN ROM OVERLAY 0
170362  160004                            .WORD   T1                          ;VECTOR ADDRESS

        000010                    STSENL  =       .-STSTAB                    ;DEFINE THE TABLE ENTRY LENGTH
```

```

                          ;RAM TEST
170364  100002                    .WORD   KX$T02          ;TPR 0 VALUE
170366  170546                    .WORD   T2ERR           ;LED UPDATING ROUTINE ON ERROR
170370     000                    .BYTE   0               ;NO INPUT PARAMETER
170371     000                    .BYTE   0               ;IN ROM OVERLAY 0
170372  160010                    .WORD   T2              ;VECTOR ADDRESS
                          ;ROM TEST
170374  100004                    .WORD   KX$T03          ;TPR 0 VALUE
170376  170576                    .WORD   T3ERR           ;LED UPDATING ROUTINE ON ERROR
170400     001                    .BYTE   1               ;TPR 3 CONTAINS PARAMETER
170401     000                    .BYTE   0               ;IN ROM OVERLAY 0
170402  160014                    .WORD   T3              ;VECTOR ADDRESS
                          ;CPU TEST
170404  100010                    .WORD   KX$T04          ;TPR 0 VALUE
170406  170642                    .WORD   T4ERR           ;LED UPDATING ROUTINE ON ERROR
170410     000                    .BYTE   0               ;NO INPUT PARAMETER
170411     000                    .BYTE   0               ;IN ROM OVERLAY 0
170412  160020                    .WORD   T4              ;VECTOR ADDRESS
                          ;BEVENT TEST
170414  100020                    .WORD   KX$T05          ;TPR 0 VALUE
170416  170650                    .WORD   T5ERR           ;LED UPDATING ROUTINE ON ERROR
170420     000                    .BYTE   0               ;NO INPUT PARAMETER
170421     000                    .BYTE   0               ;IN ROM OVERLAY 0
170422  160024                    .WORD   T5              ;VECTOR ADDRESS
                          ;SLU 1 TEST
170424  100040                    .WORD   KX$T06          ;TPR 0 VALUE
170426  170656                    .WORD   T6ERR           ;LED UPDATING ROUTINE ON ERROR
170430     000                    .BYTE   0               ;NO INPUT PARAMETER
170431     000                    .BYTE   0               ;IN ROM OVERLAY 0
170432  160030                    .WORD   T6              ;VECTOR ADDRESS
                          ;SLU 2 TEST
170434  100100                    .WORD   KX$T07          ;TPR 0 VALUE
170436  170664                    .WORD   T7ERR           ;LED UPDATING ROUTINE ON ERROR
170440     001                    .BYTE   1               ;TPR 3 CONTAINS PARAMETER, CHANNEL A /
                                                          ;CHANNEL B TEST SELECTION
170441     000                    .BYTE   0               ;IN ROM OVERLAY 0
170442  160034                    .WORD   T7              ;VECTOR ADDRESS
                          ;PIO TEST
170444  100200                    .WORD   KX$T10          ;TPR 0 VALUE
170446  170720                    .WORD   T10ERR          ;LED UPDATING ROUTINE ON ERROR
170450     000                    .BYTE   0               ;NO INPUT PARAMETER
170451     001                    .BYTE   1               ;IN ROM OVERLAY 1
170452  160000                    .WORD   T10             ;VECTOR ADDRESS
                          ;DMA TEST
170454  100400                    .WORD   KX$T11          ;TPR 0 VALUE
170456  170726                    .WORD   T11ERR          ;LED UPDATING ROUTINE ON ERROR
170460     001                    .BYTE   1               ;TPR 3 CONTAINS PARAMETER, BITS
                                                          ;<21:06> OF A USEABLE QBUS DMA BUFFER
170461     001                    .BYTE   1               ;IN ROM OVERLAY 1
170462  160004                    .WORD   T11
                          ;QIR TEST
170464  101000                    .WORD   KX$T12          ;TPR 0 VALUE
170466  170734                    .WORD   T12ERR          ;LED UPDATING ROUTINE ON ERROR
170470     001                    .BYTE   1               ;TPR 3 CONTAINS PARAMETER, THE QBUS
                                                          ;INTERRUPT VECTOR ADDRESS
170471     001                    .BYTE   1               ;IN ROM OVERLAY 1
```

```
 170472   160010                  .WORD   T12             ;VECTOR ADDRESS
                          ;TWO PORT RAM TEST
 170474   102000                  .WORD   KX$T13          ;TPR 0 VALUE
 170476   170742                  .WORD   T13ERR          ;LED UPDATING ROUTINE ON ERROR
 170500      000                  .BYTE   0               ;NO INPUT PARAMETER
 170501      001                  .BYTE   1               ;IN ROM OVERLAY 1
 170502   160014                  .WORD   T13             ;VECTOR ADDRESS
                          ;END OF TABLE                   ;IF NO MATCH, THEN COMMAND ERROR.
 170504   000000                  .WORD   0

```

```
                .SBTTL  TNNERR - UPDATE LEDS AFTER SELFTEST NN
                .ENABL  GBL


        ;
        ; MODULE NAME: TNNERR - UPDATE LEDS AFTER SELFTEST NN
        ;
        ;
        ;
        ;
        ;
        ; FUNCTIONAL DESCRIPTION:
        ;
        ;       T1ERR THROUGH T13ERR HANDLE SETTING THE LEDS DEPENDING ON THE ERROR
        ;       CODE RETURNED IN R0 BY A SELF-TEST ROUTINE. EACH TEST HAS A SET OF
        ;       BITS GIVING "SIGNIFICANT" ERROR CONDITIONS. "INSIGNIFICANT" ERROR
        ;       CONDITIONS ARE SUCH ERROR CODES AS "ROM MAPPED LOW SO INTERRUPT
        ;       TESTS WERE NOT EVEN TRIED". AFTER DECIDING ON AN LED VALUE, THE
        ;       ROUTINE CALLS SETLED TO SET THE VALUE. SETLED PRIORITIZES THE LED
        ;       SETTINGS SUCH THAT A NON-FATAL CODE WILL NOT REPLACE A FATAL CODE,
        ;       FOR EXAMPLE.
        ;
        ; INPUTS:
        ;       R0 CONTAINS THE TEST'S DISCRETE ERROR CODE.
        ;       R1 CONTAINS THE VALUE IN R1 WHEN THE SELFTEST ROUTINE WAS CALLED. THIS
        ;               VALUE IS OPTIONALLY USED BY TNNERR.
        ;
        ; CALLING SEQUENCE:
        ;       JSR     PC,TNNERR
        ;
        ; OUTPUTS:
        ;       THE LEDS ARE UPDATED VIA A CALL TO SETLED.
        ;
        ;       R0 CONTAINS 0 IF NO SIGNIFICANT ERROR OCCURRED. R0 CONTAINS THE LED
        ;       VALUE USED IN THE CALL TO SETLED IF A SIGNIFICANT ERROR OCCURRED.
        ;
        ;       NOTE: A SIGNIFICANT ERROR IS AN ERROR THAT RESULTS IN A REQUEST TO
        ;       UPDATE THE LEDS. WHEN A SELFTEST CAN NOT BE RUN (TYPICALLY BECAUSE ROM
        ;       IS MAPPED LOW), THE RESULTING ERROR IS NOT SIGNIFICANT AND, THEREFORE,
        ;       THERE IS NO REQUEST TO UPDATE THE LEDS.
        ;
        ; ROUTINES USED:
        ;       SETLED
        ;
        ; REGISTERS CHANGED:
        ;       R0,R2
        ;
```

```
                                    ;THE CSR TEST, TEST 1, IS A SPECIAL CASE BECAUSE WE MUST DIFFERENTIATE BETWEEN
                                    ;THE FATAL NON-STANDALONE TPR ERROR AND THE OTHER NON-FATAL ERRORS.
170506  032700  000004              T1ERR:: BIT     #4,R0           ;IS THERE A TPR ERROR?
170512  001407                              BEQ     T1ERR1          ;NO.
170514  032737  000340  177524              BIT     #340,@#KW.CSC   ;YES, BUT ARE WE IN STANDALONE MODE (I. E.
                                                                    ;WITH ID EQUAL TO 0 OR 1)? NOTE: THE 340 MASK
                                                                    ;IS A SUBSET OF THE KW$IDS DEFINITON. IF THE
                                                                    ;LOCATION OF KW$IDS CHANGES, THE 340 MUST BE
                                                                    ;CHANGED.
170522  001406                              BEQ     T1ERR2          ;YES, SO IT'S A NON-FATAL ERROR.
170524  012700  000017                      MOV     #FERRD,R0       ;NO, WE ARE IN NON-STANDALONE MODE SO IT'S A
                                                                    ;FATAL ERROR.
170530  000511                              BR      TNNCM1
170532  032700  000373              T1ERR1: BIT     #373,R0         ;ARE THERE OTHER CSR ERRORS?
170536  001511                              BEQ     TNNINX          ;NO.
170540  012700  000010              T1ERR2: MOV     #NFERRD,R0      ;YES, SET UP THE NON-FATAL LED DISPLAY.
170544  000503                              BR      TNNCM1
                                    ;
                                    ;THE RAM TEST, TEST 2, IS A SPECIAL CASE BECAUSE WE MUST DIFFERENTIATE BETWEEN
                                    ;FATAL NATIVE RAM ERRORS AND NON-FATAL USER RAM ERRORS.
170546  032700  000037              T2ERR:: BIT     #37,R0          ;ARE THERE NATIVE RAM ERRORS?
170552  001403                              BEQ     T2ERR1          ;NO.
170554  012700  000017                      MOV     #FERRD,R0       ;YES, SET UP FATAL LED DISPLAY.
170560  000475                              BR      TNNCM1
170562  032700  001740              T2ERR1: BIT     #1740,R0        ;ARE THERE USER RAM ERRORS?
170566  001475                              BEQ     TNNINX          ;NO.
170570  012700  000011                      MOV     #UROMOD,R0      ;YES, SET UP NON-FATAL LED DISPLAY.
170574  000467                              BR      TNNCM1
                                    ;
                                    ;THE ROM TEST, TEST 3, ERROR HANDLING IS A SPECIAL CASE BECAUSE THERE ARE
                                    ;3 POSSIBLE LED SETTINGS THAT MAY RESULT.
170576  032700  000003              T3ERR:: BIT     #3,R0           ;ARE THERE NATIVE ROM ERRORS?
170602  001403                              BEQ     T3ERR1          ;NO.
170604  012700  000017                      MOV     #FERRD,R0       ;YES.
170610  000461                              BR      TNNCM1
170612  032700  000004              T3ERR1: BIT     #4,R0           ;IS THERE A LOW BYTE USER ROM ERROR?
170616  001403                              BEQ     T3ERR2          ;NO.
170620  012700  000011                      MOV     #UROMOD,R0      ;YES.
170624  000453                              BR      TNNCM1
170626  032700  000010              T3ERR2: BIT     #10,R0          ;IS THERE A HIGH BYTE USER ROM ERROR?
170632  001453                              BEQ     TNNINX          ;NO.
170634  012700  000012                      MOV     #UROM1D,R0      ;YES.
170640  000445                              BR      TNNCM1
170642  012702  170766              T4ERR:: MOV     #T4ETB,R2       ;AS ABOVE FOR THE CPU TEST.
170646  000437                              BR      TNNCOM
170650  012702  170772              T5ERR:: MOV     #T5ETB,R2       ;AS ABOVE FOR THE BEVNT TEST.
170654  000434                              BR      TNNCOM
170656  012702  170776              T6ERR:: MOV     #T6ETB,R2       ;AS ABOVE FOR THE DLART TEST.
170662  000431                              BR      TNNCOM
                                    ;
                                    ;THE 2ND SERIAL LINE UNIT TEST, TEST 7, IS A CODED SPECIAL CASE TO HANDLE
                                    ;CHANNEL A / CHANNEL B LED SETTINGS.
170664  005701                      T7ERR:: TST     R1              ;DID WE DO THE "BOTH CHANNELS GO/NO GO" TEST?
170666  001403                              BEQ     T7ERR1          ;YES, USE CHANNEL A'S LED SETTING.
170670  032701  000001                      BIT     #BIT0,R1        ;NO, DID WE SELFTEST ONLY CHANNEL A?
170674  001403                              BEQ     T7ERR2          ;NO.
170676  012702  171002              T7ERR1: MOV     #T7AETB,R2      ;YES.
```

```
170702  000421                          BR      TNNCOM
170704  032701  000002          T7ERR2: BIT     #BIT1,R1        ;DID WE SELFTEST ONLY CHANNEL B?
170710  001424                          BEQ     TNNINX          ;NO.
170712  012702  171006                  MOV     #T7BETB,R2      ;YES.
170716  000413                          BR      TNNCOM
170720                          T10ERR::
170720  012702  171012                  MOV     #T10ETB,R2      ;AS ABOVE FOR THE PIO CHIP TEST.
170724  000410                          BR      TNNCOM
170726                          T11ERR::
170726  012702  171016                  MOV     #T11ETB,R2      ;AS ABOVE FOR THE DMA CHIP TEST.
170732  000405                          BR      TNNCOM
170734                          T12ERR::
170734  012702  171022                  MOV     #T12ETB,R2      ;AS ABOVE FOR THE QIR TEST.
170740  000402                          BR      TNNCOM
170742                          T13ERR::
170742  012702  171026                  MOV     #T13ETB,R2      ;AS ABOVE FOR THE TPR TEST.
                                ;
                                ;TNNCOM IS THE COMMON ENTRY TO USE WHEN R2 POINTS TO A TABLE ENTRY OF
                                ;SIGNIFICANT ERROR BITS FOLLOWED BY LED VALUE.
170746  032200                  TNNCOM: BIT     (R2)+,R0        ;ARE ANY OF THE "SIGNIFICANT" ERROR BITS
                                                                ;SET IN THE RETURNED ERROR CODE?
170750  001404                          BEQ     TNNINX          ;NO, DON'T EVEN TRY TO UPDATE THE LEDS.
170752  011200                          MOV     (R2),R0
                                ;
                                ;TNNCM1 IS THE COMMON ENTRY TO USE WHEN R0 CONTAINS AN LED SETTING.
170754  004737  167200          TNNCM1: JSR     PC,@#SETLED     ;UPDATE THE LEDS USING SETLED SO THAT A MORE
                                                                ;IMPORTANT SETTING DOES NOT GET OVERWRITTEN.
170760  000401                          BR      TNNXIT
                                ;
                                ;TNNINX IS THE EXIT WHEN THERE ARE NO SIGNIFICANT ERRORS.
170762  005000                  TNNINX: CLR     R0              ;SHOW THAT NO SIGNIFICANT ERROR OCCURRED.
170764  000207                  TNNXIT: RTS     PC

                                ;ERROR TABLES GIVING THE "SIGNIFICANT" ERROR BIT SETTINGS AND THE ASSOCIATED
                                ;LED SETTING FOR THE ERROR. THE SPARSENESS OF THE BITS IS BECAUSE CERTAIN
                                ;TESTS RETURN NON-ZERO ERROR CODES THAT REALLY MEAN "COULDN'T RUN THE TEST".
                                ;THERE ARE SEVERAL TESTS WHERE INTERRUPTS CANNOT BE TESTED BECAUSE ROM IS IN
                                ;LOW MEMORY.
                                ;
                                ;T1, T2, & T3 MUST BE SPECIAL CASED IN CODE.
170766  007777  000017          T4ETB:  .WORD   7777,FERRD
170772  000016  000010          T5ETB:  .WORD   16,NFERRD
170776  000176  000013          T6ETB:  .WORD   176,S1LBD
171002  007476  000014          T7AETB: .WORD   7476,S2ALBD     ;NOTE: BITS 6 & 7 ARE NOT INCLUDED BECAUSE A
                                                                ;CONFIGURATION JUMPER ON THE KXT11-CA SWITCHES
                                                                ;THE DMA CHIP CONNECTION FROM SLU2 CHAN A TO
                                                                ;THE PIO CHIP. WHEN THE DMA CHIP IS CONNECTED
                                                                ;TO THE PIO CHIP, BITS 6 & 7 WILL ALWAYS BE 1,
                                                                ;EVEN THOUGH NO TRUE ERROR OCCURRED.
                                                                ;THEREFORE, WE DO NOT SET THE LEDS.
171006  007776  000015          T7BETB: .WORD   7776,S2BLBD     ;NOTE: DMA (AND THEREFORE BITS 6 & 7) DOES
                                                                ;NOT APPLY TO CHANNEL B. IF THE BITS COME ON
                                                                ;WE HAVE A SERIOUS ERROR, SO WE OUGHT TO SET
                                                                ;THE LEDS.
171012  000376  000016          T10ETB: .WORD   376,PARLBD
171016  000074  000010          T11ETB: .WORD   74,NFERRD
171022  000172  000010          T12ETB: .WORD   172,NFERRD
171026  007767  000017          T13ETB: .WORD   7767,FERRD

```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 73
DMALD - DMA LOADER

                                      .SBTTL  DMALD - DMA LOADER
                                      .ENABLE LC,GBL

                              ;
                              ; Module name: DMALD - DMA LOAD
                              ;
                              ; System: KXT11-CA Native Firmware
                              ;
                              ;
                              ;
                              ;
                              ; Functional Description:
                              ;
                              ;       This is the module that contols the loading of the IOP's application
                              ;       from the Q bus by the DMA controller. It is used by the Q bus command
                              ;       interpreter module on reception of a DMA load command. This module
                              ;       expects the data which is to be placed into the chain control registers
                              ;       of the DMA controller to be in the dual port registers 2 and 3.
                              ;       On completion of the load, this module places the IOP in the
                              ;       Wait state.

                              ;Input Parameters:

                              ;       NONE

                              ;Output Parameters

                              ;       NONE

                              ;DATA STRUCTURES USED

                              ;       1) DMA MASTER MODE REGISTER
                              ;       2) DMA COMMAND REGISTER
                              ;       3) DMA CHANNEL 0 REGISTER
                              ;       4) DUAL PORT RAM REGISTER 2 AND 3
                              ;       5) IOP STATUS REGISTER
                              ;       6) IOP COMMAND REGISTER

                              ;Routines Used

                              ;       1) WAITST (JUMP TO)

                              ;Global Symbols Defined

```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 74
DMALD - DMA LOADER

        174442                CHNOFO  =       174442          ;DMA CHAIN OFFSET REGISTER ADDRESS OF CHANNEL 0
        174446                CHNSTO  =       174446          ;DMA CHAIN SEGMENT/TAG REGISTER ADDRESS OF CHANNEL 0
        174470                MMREGA  =       174470          ;DMA MASTER MODE REGISTER ADDRESS
        174454                CMREGA  =       174454          ;DMA COMMAND REGISTER ADDRESS
        174456                SOREGA  =       174456          ;DMA CHANNEL 0 STATUS REGISTER ADDRESS
        010000                CABIT   =       10000           ;CHAIN ABORT BIT OF STATUS
        000002                EOPBIT  =       2               ;END OF PROCESS BIT OF STATUS
        004000                NACBIT  =       4000            ;NAC BIT OF STATUS
```

```
                                                                ;* PROCEDURE DMA LOAD
171032                              DMALD::
                                                                ;* * SET LEDS TO DMA DISPLAY
171032  012700  000002                      MOV     #DMAED,R0
171036  012701  171046                      MOV     #DMA50,R1
171042  000167  176134                      JMP     SETLDJ
171046                              DMA50:
                                                                ;* * SET DMA MASTER MODE REGISTER TO NO INTERRUPT
171046  112737  000151  174470              MOVB    #151,@#MMREGA
                                                                ;* * COPY CHAIN DATA TO DMA CHAIN REGISTERS
171054  013737  175004  174442              MOV     @#KW.SC2,@#CHNOF0
171062  013737  175006  174446              MOV     @#KW.SC3,@#CHNST0
                                                                ;* * SET HARDWARE MASK
171070  112737  000202  174454              MOVB    #202,@#CMREGA
                                                                ;* * START CHAIN LOAD CHANNEL 0
171076  112737  000240  174454              MOVB    #240,@#CMREGA
                                                                ;* * DO UNTIL DMA CONTROLLER STOPPED
171104                              DMA100:
                                                                ;* * * TEST FOR END OF TRANSFER
171104  032737  014002  174456              BIT     #<CABIT ! EOPBIT ! NACBIT>,@#SOREGA
171112  001774                              BEQ     DMA100
                                                                ;* * END UNTIL
171114                              DMA200:
                                                                ;* * COPY DMA STATUS REGISTER CHANNEL 0
                                                                ;                TO DUAL PORT REGISTER 2
171114  013737  174456  175004              MOV     @#SOREGA,@#KW.SC2
                                                                ;* * IF NORMAL COMPLETION
171122  032737  010002  174456              BIT     #<CABIT ! EOPBIT>,@#SOREGA
171130  001004                              BNE     DMA300
                                                                ;* * THEN
                                                                ;* * * CLEAR DMA ERROR BIT OF DPR
171132  042737  040000  175002              BIC     #<KX$DME>,@#KW.STA
171140  000403                              BR      DMA400
                                                                ;* * ELSE
171142                              DMA300:
                                                                ;* * * SET DMA ERROR AND COMMAND ERROR BITS OF DPR
171142  052737  140000  175002              BIS     #<KX$CME ! KX$DME>,@#KW.STA
                                                                ;* * END IF
171150                              DMA400:
                                                                ;* * END IF
171150                              DMA500:
                                                                ;* * SETUP STACK POINTER AND R5
171150  012700  171160                      MOV     #DMA600,R0
171154  000167  002526                      JMP     SPR5SU
171160                              DMA600:
                                                                ;* * CLEAR COMMAND REGISTER
171160  005037  175000                      CLR     @#KW.CMD
                                                                ;* * ENTER WAIT STATE (NON STRUCTURED EXIT)
171164  000167  176212                      JMP     WAITST
                                                                ;* END DMA LOADER
```

```
                .SBTTL  BSUPV - BOOT SUPERVISOR
                .ENABLE LC,GBL


        ;
        ; Module name: BSUPV - BOOT SUPERVISOR
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module supervises the boot process. It makes the determination
        ;       of which device the application will be loaded from and transfers
        ;       control to the appropriate routines.

        ;Input Parameters:
        ;
        ;     1.  BOOTFO of FLGWRD
        ;
        ;
        ;Output Parameters
        ;
        ;       NONE
        ;
        ;Data Structures Used
        ;
        ;     1.  LED DISPLAY
        ;
        ;     2.  IOP STATUS REGISTER
        ;
        ;     3.  TCSRB, bits of BOOTSW
        ;
        ;Routines Used
        ;
        ;     1.  TU58 Boot (TU58BT)
        ;
        ;     2.  Loop back tests (LBTEST)
        ;
        ;     3.  Serial ODT monitor (SODTM)
        ;
        ;     4.  Set LEDs (SETLED)
        ;
        ;     5.  Trap 24 emulator (TRAP24)
        ;
        ;Global Symbols Defined
        ;
        ;       NONE
```

```
                                                                  ;* PROCEDURE BOOTSTRAP SUPERVISOR
171170                              BSUPV::
                                                                  ;* * SET SP TO DEFAULT USER STACK
171170  010500                              MOV     R5,R0
171172  062700  177700                      ADD     #ADSTKO,R0
171176  010006                              MOV     R0,SP
                                                                  ;* * CASE BOOT SELECTION
                                                                  ;* * WHEN 0
                                                                  ;* * OR 1
                                                                  ;* * OR 2
171200  022765  000002  177774              CMP     #2,BOOTFO(R5)
171206  103414                              BLO     BS100
                                                                  ;* * * SET LEDS TO NON NATIVE CODE STATE
171210  012700  000000                      MOV     #NNCED,R0
171214  004767  175760                      CALL    SETLED
                                                                  ;* * * SET STATUS REGISTER TO NON NATIVE
                                                                  ;      CODE STATE
171220  042737  000007  175002              BIC     #KM$STF,@#KW.STA
171226  052737  000007  175002              BIS     #KP$NNC,@#KW.STA
                                                                  ;* * * EMULATE TRAP TO 24 (NON STRUCTURED EXIT)
171234  000167  176116                      JMP     TRAP24
                                                                  ;* * WHEN 3
171240  022765  000003  177774      BS100:  CMP     #3,BOOTFO(R5)
171246  001014                              BNE     BS200
                                                                  ;* * * SET LED TO PRIMARY BOOTSTRAP STATE
171250  012700  000001                      MOV     #PRIBD,R0
171254  004767  175720                      CALL    SETLED
                                                                  ;* * * SET STATUS REGISTER PRIMARY BOOTSTRAP STATE
171260  042737  000007  175002              BIC     #KM$STF,@#KW.STA
171266  052737  000005  175002              BIS     #KP$PBS,@#KW.STA
                                                                  ;* * * PERFORM TU58 PRIMARY BOOTSTRAP (JUMP TO TU58 MODULE)
171274  000167  000434                      JMP     TU58BT
                                                                  ;* * WHEN 4
171300  022765  000004  177774      BS200:  CMP     #4,BOOTFO(R5)
171306  001007                              BNE     BS300
                                                                  ;* * * DISABLE BREAKS
171310  052737  020000  175002              BIS     #KX$BDF,@#KW.STA
                                                                  ;* * * PUSH 0 ADDRESS AND PSW ON STACK
171316  005046                              CLR     -(SP)
171320  005046                              CLR     -(SP)
                                                                  ;* * * PERFORM SERIAL ODT
171322  000167  173300                      JMP     SODTM   ;(NON-STRUCTURED EXIT TO SERIAL ODT)
                                                                  ;* * WHEN 5
171326  022765  000005  177774      BS300:  CMP     #5,BOOTFO(R5)
171334  001404                              BEQ     BS400
                                                                  ;* * OR 6
171336  022765  000006  177774              CMP     #6,BOOTFO(R5)
171344  001002                              BNE     BS500
171346                              BS400:
                                                                  ;* * * WAIT FOR COMMAND
171346  000167  176030                      JMP     WAITST  ;(NON STRUCTURED EXIT)
                                                                  ;* * WHEN 8
171352                              BS500:
171352  022765  000010  177774              CMP     #8.,BOOTFO(R5)
171360  001404                              BEQ     BS550
                                                                  ;* * OR 9
171362  022765  000011  177774              CMP     #9.,BOOTFO(R5)
```

```
 171370  001005                            BNE     BS570
 171372                           BS550:
                                                                    ;* * * ENABLE OVERLAY 0
 171372  042737  000040  177520            BIC     #KX$DPE,@#KW.CSA
                                                                    ;* * * PERFORM LOOP BACK TESTS
 171400  000167  172306                    JMP     LBTEST  ;(NON-STRUCTURED EXIT)
                                                                    ;* * WHEN 10
 171404                           BS570:
 171404  022765  000012  177774            CMP     #10.,BOOTFO(R5)
 171412  001002                            BNE     BS600
                                                                    ;* * * WAIT FOR COMMAND
 171414  000167  175762                    JMP     WAITST  ;(NON-STRUCTURED EXIT)
                                                                    ;* * ELSE
 171420                           BS600:
                                                                    ;* * * SET LEDS TO FATAL ERROR
 171420  012700  000017                    MOV     #FERRD,R0
 171424  004767  175550                    CALL    SETLED
                                                                    ;* * * INVALID SELECTION STOP IOP
 171430  000777                   FAS::    BR      .
                                                                    ;* * END CASE
                                                                    ;* END BOOTSTRAP SUPERVISOR
```

```
                .SBTTL  QODTM - Q BUS ODT MONITOR

                .ENABLE GBL,LC
;
; Module name: QODTM - Q BUS ODT MONITOR
;
; System: KXT11-CA Native Firmware
;
;
;
;
; Functional Description:
;
;       This module is the main controlling module for the Q bus ODT mode.
;       It supervises the ODT functions and on completion returns control
;       back to the application or places the IOP in the wait state.
;
;Input Parameters:

;       None

;Output Parameters

;       None

;Routines Used

;       1) Save Registers (SAVERG)
;       2) Set LEDs (SETLED)
;       3) Wait State (WAITST)
;       4) Go/Proceed (PROC)
;       5) Go/Proceed (GO)
;       6) Deposit (DEPOS)
;       7) Examine (EXAMIN)

;Registers Used Outside  of Output Parameters:


```

```
                                                                        ;* PROCEDURE Q BUS ODT MONITOR
171432                          QODTM::
                                                                        ;* * SET ODT FORMAT FLAG OF STATUS REGISTER
                                                                        ;* * SET NXM FLAG OF STATUS REGISTER
171432  052767  010200  003342          BIS     #<KX$NXF ! KX$QOF>,KW.STA
                                                                        ;* * SAVE REGISTERS (NON STRUCTURED JUMP TO
                                                                        ;               AND BACK FROM SAVE REGISTER MODULE)
171440  000167  173006                  JMP     SAVERG
171444                          QSRET::
                                                                        ;* * SET STATUS REGISTER TO Q ODT MODE STATE AND
171444  042737  000007  175002          BIC     #KM$STF,@#KW.STA
171452  052737  000003  175002          BIS     #KP$QOD,@#KW.STA
                                                                        ;* * SET LEDS TO Q BUS ODT STATE
171460  012700  000005                  MOV     #QODTMD,R0
171464  004767  175510                  CALL    SETLED
                                                                        ;* * CLEAR ALL FLAGS
171470  005003                          CLR     R3
                                                                        ;* * CLEAR COMMAND REGISTER
171472  005037  175000                  CLR     @#KW.CMD
                                                                        ;* * DO UNTIL RECEIVE EXIT COMMAND
171476                          QS100:
                                                                        ;* * OR PROCEED COMMAND
                                                                        ;* * OR GO COMMAND
                                                                        ;* * * DO UNTIL COMMAND REGISTER NOT EMPTY
171476                          QS200:
                                                                        ;* * * END UNTIL
171476  000001                          WAIT
                                                                        ;* * * CLEAR COMMAND ERROR BIT
171500  042737  100000  175002          BIC     #KX$CME,@#KW.STA
                                                                        ;* * * CASE COMMAND RECEIVED
171506  013700  175000                  MOV     @#KW.CMD,R0
                                                                        ;* * * WHEN EXIT
171512  022700  100000                  CMP     #KX$EXO,R0
171516  001007                          BNE     QS300
                                                                        ;* * * * CLEAR QBUS ODT FLAG OF STATUS REG.
171520  042737  000200  175002          BIC     #KX$QOF,@#KW.STA
                                                                        ;* * * * CLEAR COMMAND REGISTER
171526  005037  175000                  CLR     @#KW.CMD
                                                                        ;* * * * ENTER WAIT STATE
171532  000167  175644                  JMP     WAITST  ;(NON STRUCTURED EXIT)
                                                                        ;* * * WHEN PROCEED
171536                          QS300:
171536  022700  000020                  CMP     #KX$PRO,R0
171542  001006                          BNE     QS400
                                                                        ;* * * * EXECUTE PROCEED
171544  004767  175240                  CALL    PROC
                                                                        ;* * * * IF ERROR (RETURN FROM PROC)
                                                                        ;* * * * THEN
                                                                        ;* * * * * SET COMMAND ERROR BIT OF STATUS REG.
171550  052737  100000  175002          BIS     #KX$CME,@#KW.STA
                                                                        ;* * * * END IF
171556  000454                          BR      QS1000
                                                                        ;* * * WHEN GO
171560                          QS400:
171560  022700  000010                  CMP     #KX$GOO,R0
171564  001012                          BNE     QS500
                                                                        ;* * * * SET GO ADDRESS = DPR 3
```

```
171566  013700  175006                       MOV     @#KW.SC3,R0
                                                                        ;* * * * PERFORM GO
171572  004767  175174                       CALL    GO
                                                                        ;* * * * IF RETURNED
                                                                        ;* * * * THEN
                                                                        ;* * * * * SET ERROR FLAG
171576  052703  100000                       BIS     #ERRFGB,R3
                                                                        ;* * * * * RE-ENABLE DUAL PORT RAM (DISABLED
                                                                        ;            BY THE RESET PERFORMED BY GO ROUTINE)
171602  052737  000100  177530               BIS     #KX$DEN,@#KW.CSD
171610  000437                               BR      QS1000
                                                                        ;* * * WHEN DEPOSIT
171612                               QS500:
171612  022700  000004                       CMP     #KX$DEO,R0
171616  001005                               BNE     QS700
                                                                        ;* * * * SET DATA = DPR 2
171620  013704  175004                       MOV     @#KW.SC2,R4
                                                                        ;* * * * PERFORM DEPOSIT
171624  004767  174644                       CALL    DEPOS
171630  000427                               BR      QS1000
                                                                        ;* * * WHEN OPEN REGISTER
171632                               QS700:
171632  022700  000002                       CMP     #KX$ORP,R0
171636  001007                               BNE     QS800
                                                                        ;* * * * SET TARGET = DPR 3
171640  013702  175006                       MOV     @#KW.SC3,R2
                                                                        ;* * * * SET REG/ADDR FLAG = REGISTER
171644  052703  000200                       BIS     #REGFGB,R3
                                                                        ;* * * * PERFORM OPEN ON REGISTER
171650  004767  174066                       CALL    EXAMIN
171654  000415                               BR      QS1000
                                                                        ;* * * WHEN OPEN MEMORY
171656                               QS800:
171656  022700  000001                       CMP     #KX$OMO,R0
171662  001007                               BNE     QS900
                                                                        ;* * * * SET TARGET = DPR 3
171664  013702  175006                       MOV     @#KW.SC3,R2
                                                                        ;* * * * SET REG/ADDR FLAG = ADDRESS
171670  042703  000200                       BIC     #REGFGB,R3
                                                                        ;* * * * PERFORM OPEN MEMORY
171674  004767  174042                       CALL    EXAMIN
171700  000403                               BR      QS1000
                                                                        ;* * * ELSE
171702                               QS900:
                                                                        ;* * * * SET COMMAND ERROR BIT
171702  052737  100000  175002               BIS     #KX$CME,@#KW.STA
                                                                        ;* * * END CASE
171710                               QS1000:
                                                                        ;* * * IF ERROR FLAG SET
171710  005703                               TST     R3
171712  100005                               BPL     QS1100
                                                                        ;* * * THEN
                                                                        ;* * * * SET COMMAND ERROR FLAG OF STATUS REGISTER
171714  052737  100000  175002               BIS     #KX$CME,@#KW.STA
                                                                        ;* * * * CLEAR ERROR FLAG
171722  042703  100000                       BIC     #ERRFGB,R3
                                                                        ;* * * END IF
```

```
171726                           QS1100:
                                                                   ;* * * CLEAR COMMAND REGISTER
171726  005037  175000                   CLR     @#KW.CMD
                                                                   ;* * END UNTIL
171732  000661                           BR      QS100
                                                                   ;* END Q BUS ODT MONITOR
```

```
                                         .SBTTL  TU58BT - TU58 BOOT STRAP

                                         .ENABLE GBL,LC
                                 ;
                                 ; Module name: LJD
                                 ;
                                 ; System: KXT11-CA Native Firmware
                                 ;
                                 ;
                                 ;
                                 ;
                                 ;
                                 ;
                                 ; Functional Description:
                                 ;
                                 ; This is the primary loader for TU58s. It will attempt to load and transfer
                                 ; control to the code in the boot block of a TU58 drive connected to the
                                 ; console port (SLU1).  If the first word of the block is in the range of
                                 ; 240 to 277 it is considered a valid boot block. This routine will
                                 ; continue to try to boot until it suceeds.
```

```

                                                                       ;* PROCEDURE TU58 BOOT
171734                            TU58BT::
                                                                       ;* * SET DRIVE FLAG TO 0
171734  012700  000001                    MOV     #1,R0
                                                                       ;* * DO UNTIL SUCESSFUL BOOT
171740                            TU100:
                                                                       ;* * * TOGGLE DRIVE FLAG
171740  005100                            COM     R0
171742  042700  177776                    BIC     #^C1,R0
                                                                       ;* * * SET POINTER = 0
171746  005004                            CLR     R4
                                                                       ;* * * SEND BREAK
171750  004767  000260                    CALL    SNDBRK
                                                                       ;* * * SEND INIT
171754  012702  000002                    MOV     #2,R2
171760  012701  172320                    MOV     #INITST,R1
171764  004767  000310                    CALL    SEND
                                                                       ;* * * DO UNTIL TIMED OUT OR CONTINUE RECEIVED
171770  012703  000002                    MOV     #2,R3
171774                            TU120:
171774  005002                            CLR     R2
171776                            TU125:
171776  105737  177560                    TSTB    @#RCSR1A
172002  100403                            BMI     TU130
172004  077204                            SOB     R2,TU125
172006  077306                            SOB     R3,TU120
                                                                       ;* * * END UNTIL
172010  000753                            BR      TU100    ;NON STRUCTURED BRANCH
172012                            TU130:
172012  122737  000020  177562            CMPB    #20,@#RBUF1A
172020  001365                            BNE     TU120


                                                                       ;* * * SEND BREAK
172022  004767  000206                    CALL    SNDBRK
                                                                       ;* * * SEND BOOT COMMAND
172026  012702  000002                    MOV     #2,R2
172032  012701  172322                    MOV     #BOOTST,R1
172036  004767  000236                    CALL    SEND
                                                                       ;* * * DO UNTIL TRANSMIT BUFFER EMPTY
172042                            TU150:
172042  105737  177564                    TSTB    @#XCSR1A
172046  100375                            BPL     TU150
                                                                       ;* * * END UNTIL
                                                                       ;* * * SEND DRIVE NUMBER
172050  110067  005512                    MOVB    R0,XBUF1A
                                                                       ;* * * DO UNTIL 512 BYTES RECIEVED
172054  012702  001000                    MOV     #512.,R2
172060                            TU200:
                                                                       ;* * * OR TIMED OUT
172060  012703  000100                    MOV     #100,R3
172064  005001                    TU300:  CLR     R1
172066  105737  177560            TU400:  TSTB    @#RCSR1A
172072  100403                            BMI     TU500
172074  077104                            SOB     R1,TU400
172076  077306                            SOB     R3,TU300
                                                                       ;* * * * IF TIMED OUT
```

```
                                                                 ;* * * * THEN
                                                                 ;* * * * * RETRY
172100  000717                               BR      TU100   ;(NON STRUCTURED BRANCH)
                                                                 ;* * * * ELSE
172102                              TU500:
                                                                 ;* * * * * INPUT DATA
172102  113724  177562                       MOVB    @#RBUF1A,(R4)+
                                                                 ;* * * * * DECREMENT BYTE COUNTER
172106  005302                               DEC     R2
                                                                 ;* * * * END IF
172110                              TU600:
                                                                 ;* * * * IF 3 BYTES HAVE BEEN RECEIVED
172110  022702  000775                       CMP     #509.,R2
172114  001013                               BNE     TU600
                                                                 ;* * * * AND BYTE 1 = 2
172116  005001                               CLR     R1
172120  122721  000002                       CMPB    #2,(R1)+
172124  001007                               BNE     TU800
                                                                 ;* * * * AND BYTE 2 = 12
172126  122721  000012                       CMPB    #12,(R1)+
172132  001004                               BNE     TU800
                                                                 ;* * * * AND BYTE 3 = 100
172134  122711  000100                       CMPB    #100,(R1)
172140  001001                               BNE     TU800
                                                                 ;* * * * THEN
                                                                 ;* * * * * RETRY
172142  000676                               BR      TU100
                                                                 ;* * * * END IF
172144                              TU600:
                                                                 ;* * * END UNTIL
172144                              TU900:
172144  005702                               TST     R2
172146  001344                               BNE     TU200
                                                                 ;* * END UNTIL
172150                              TU1000:
172150  005002                               CLR     R2
172152  021227  000240                       CMP     (R2),#240
172156  103670                               BLO     TU100
172160  021227  000277                       CMP     (R2),#277
172164  101265                               BHI     TU100
                                                                 ;* * SAVE UNIT NO.
172166  010004                               MOV     R0,R4
                                                                 ;* * SET LED DISPLAY TO NON NATIVE CODE EXECUTING
172170  012700  000000                       MOV     #NNCED,R0
172174  004767  175000                       CALL    SETLED
                                                                 ;* * SET STATUS REGISTER TO NON NATIVE CODE STATE
172200  042737  000007  175002               BIC     #KM$STF,@#KW.STA
172206  052737  000007  175002               BIS     #KP$NNC,@#KW.STA
                                                                 ;* * SET STACK POINTER TO DEFAULT USER STACK
172214  012706  177700                       MOV     #ADSTK0,SP
172220  060506                               ADD     R5,SP
                                                                 ;* * PASS CSR ADDRESS TO SECONDARY LOADER
172222  012701  177560                       MOV     #RCSR1A,R1
                                                                 ;* * PASS UNIT NUMBER TO SECONDARY LOADER
172226  010400                               MOV     R4,R0
                                                                 ;* * EXECUTE AT 0
172230  000137  000000                       JMP     @#0
```

```
                                                          ;* END TU58 BOOT
```

```
                                                          ;* PROCEDURE BREAK
172234                          SNDBRK:
                                                          ;* * SEND BREAK FOR 8 CHAR. TIMES
172234  012702  000010                  MOV     #8.,R2
172240  105737  177564          SN100:  TSTB    @#XCSR1A
172244  100375                          BPL     SN100
172246  052737  000001  177564          BIS     #1,@#XCSR1A
172254  005037  177566                  CLR     @#XBUF1A
172260  077211                          SOB     R2,SN100
                                                          ;* * END UNTIL
                                                          ;* * DO UNTIL TRANSMIT BUFFER EMPTY
172262  105737  177564          SN200:  TSTB    @#XCSR1A
172266  100375                          BPL     SN200
                                                          ;* * END UNTIL
                                                          ;* * CLEAR BREAK
172270  042737  000001  177564          BIC     #1,@#XCSR1A
                                                          ;* * RETURN FROM CALL
172276  000207                          RETURN
                                                          ;* END BREAK







                                                          ;* PROCEDURE SEND
172300                          SEND:
                                                          ;* * DO UNTIL DATA COUNT = 0
                                                          ;* * * DO UNTIL TRANSMIT BUFFER EMPTY
172300                          SD100:
172300  105737  177564                  TSTB    @#XCSR1A
172304  100375                          BPL     SD100
                                                          ;* * * END UNTIL
                                                          ;* * * OUTPUT NEXT CHAR. PER POINTER
172306  112137  177566                  MOVB    (R1)+,@#XBUF1A
                                                          ;* * * DECREMENT DATA COUNTER
172312  005302                          DEC     R2
                                                          ;* * END UNTIL
172314  001371                          BNE     SD100
                                                          ;* * RETURN FROM CALL
172316  000207                          RETURN
                                                          ;* END SEND
```

```
 172320      004                 INITST: .BYTE   4
 172321      004                         .BYTE   4

 172322      004                 BOOTST: .BYTE   4
 172323      010                         .BYTE   10
                                         .EVEN


```

```
                                         .SBTTL  PUTEST - POWER UP SELF TEST
                                         .ENABLE GBL,LC

                                 ;
                                 ; Module name: POWER UP SELF TEST - PUTEST
                                 ;
                                 ; System: KXT11-CA Native Firmware
                                 ;
                                 ;
                                 ;
                                 ;
                                 ; Functional Description:
                                 ;
                                 ; This module controls the execution of the Power Up Auto Self Tests.
                                 ; It detemines which of the test are to be executed per the Boot/Self Test
                                 ; selection. This routine also tests the Status Register of the DPR regardless
                                 ; of the Boot/Self Test Selection and halts the KXT if the test fails.
                                 ;
                                 ;       Note:
                                 ;
                                 ;       There are two important addresses with regard to the Power Up Self
                                 ;       Tests. FARSTR (defined in this module) is the address the KXT will be
                                 ;       looping at if the Status Register is found to be faulty. FTLST (defined
                                 ;       in the EXTEST module) is the address that the KXT will be looping at if
                                 ;       any other fatal self test error is found.
                                 ;
                                 ;
                                 ; Input Parameters:
                                 ;
                                 ;       BOOTFO(R5) - Boot selection location
                                 ;
                                 ; Output Parameters:
                                 ;
                                 ;       None
                                 ;
                                 ; Routines Used:
                                 ;
                                 ;       1) Set LEDs (SETLED)
                                 ;       2) Execute Test (EXTEST)



```

```
172324                          PUTEST::
                                                                ;* PROCEDURE POWER UP SELF TEST
                                                                ;* * CLEAR ALL ERROR BITS IN DPR 2
172324  005037  175004                  CLR     @#KW.SC2
                                                                ;* * TEST DPR STATUS REGISTER BY WRITING
                                                                ;     ALTERATING 1'S AND 0'S INTO IT AND
                                                                ;     COMPARING WHAT IS READ BACK
172330  012702  175002                  MOV     #KW.STA,R2
172334  011200                          MOV     (R2),R0
172336  012701  052525                  MOV     #52525,R1
172342  010112                          MOV     R1,(R2)
172344  020112                          CMP     R1,(R2)
172346  001004                          BNE     PU20
172350  005112                          COM     (R2)
172352  005101                          COM     R1
172354  020112                          CMP     R1,(R2)
172356  001405                          BEQ     PU30
                                                                ;* * IF TEST FAILED
172360                          PU20:
                                                                ;* * THEN
                                                                ;* * * SET LEDS AND STATUS REGISTER TO FATAL ERROR
172360  012700  000017                  MOV     #FERRD,R0
172364  004767  174610                  CALL    SETLED
                                                                ;* * * STOP
172370  000777                  FARSTR:: BR
                                                                ;* * END IF
172372                          PU30:
                                                                ;* * RESTORE STATUS REGISTER
172372  010012                          MOV     R0,(R2)
                                                                ;* * IF BOOT/SELF TEST SELECTION = 1,2,3
172374  016500  177774                  MOV     BOOTFO(R5),R0
172400  005700                          TST     R0
172402  001414                          BEQ     PU50
172404  022700  000004                  CMP     #4,R0
172410  101013                          BHI     PU100
                                                                ;* * OR 5
172412  022700  000005                  CMP     #5,R0
172416  001410                          BEQ     PU100
                                                                ;* * OR 8
172420  022700  000010                  CMP     #8.,R0
172424  001405                          BEQ     PU100
                                                                ;* * OR 10
172426  022700  000012                  CMP     #10.,R0
172432  001402                          BEQ     PU100
172434  000167  000126          PU50:   JMP     PU1010
172440                          PU100:
                                                                ;* * THEN
                                                                ;* * * SET STATUS REGISTER TO AUTO TEST MODE STATE
172440  042712  177770                  BIC     #^C<KM$STF>,(R2)
172444  052712  000001                  BIS     #KP$STS,(R2)
                                                                ;* * * SET LEDS TO POWER UP TEST DISPLAY
172450  012700  000007                  MOV     #PUTSDS,R0
172454  004767  174520                  CALL    SETLED
                                                                ;* * * ENABLE DPR
172460  052767  000100  005042          BIS     #KX$DEN,KW.CSR
                                                                ;* * * TEST CPU
172466  012702  000004                  MOV     #4,R2
```

```
172472  004767  000072                        CALL    EXTEST
                                                                      ;* * * TEST RAM
172476  012702  000002                        MOV     #2,R2
172502  004767  000062                        CALL    EXTEST
                                                                      ;* * * IF BOOT/SELF TEST SELECTION = 2
172506  022765  000002  177774                CMP     #2,BOOTFO(R5)
172514  001404                                BEQ     PU400
                                                                      ;* * * OR 9
172516  022765  000011  177774                CMP     #9.,BOOTFO(R5)
172524  001003                                BNE     PU500
172526                            PU400:
                                                                      ;* * * THEN
                                                                      ;* * * * INCLUDE USER ROM IN TEST
172526  012701  000001                        MOV     #1,R1
172532  000401                                BR      PU600
                                                                      ;* * * ELSE
172534                            PU500:
                                                                      ;* * * * DO NOT INCLUDE USER ROM IN TEST
172534  005001                                CLR     R1
                                                                      ;* * * END IF
172536                            PU600:
                                                                      ;* * * TEST ROM
172536  012702  000003                        MOV     #3,R2
172542  004767  000022                        CALL    EXTEST
                                                                      ;* * * TEST CSR
172546  012702  000001                        MOV     #1,R2
172552  004767  000012                        CALL    EXTEST
                                                                      ;* * * TEST LOCAL DMA
172556  012702  000011                        MOV     #11,R2
172562  004767  000002                        CALL    EXTEST
                                                                      ;* * END IF
172566                            PU1010:
                                                                      ;* * RETURN FROM CALL
172566  000207                                RETURN
                                                                      ;* END POWER UP SELF TEST
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 86
EXTEST - EXICUTE TEST

                                              .SBTTL  EXTEST - EXICUTE TEST

                                              .ENABLE GBL,LC
                                  ;
                                  ; Module name: EXTEST - EXECUTE TEST
                                  ;
                                  ; System: KXT11-CA Native Firmware
                                  ;
                                  ;
                                  ;
                                  ;
                                  ; Functional Description:
                                  ;
                                  ; This module is used by PUTEST to call a particular self test and display any
                                  ; error results in the LEDs and the DPR. It will also halt the KXT if a fatal
                                  ; error is found.
                                  ;
                                  ; Input Parameters:
                                  ;
                                  ;           R1 = TEST PARAMETERS
                                  ;           R2 = TEST NUMBER
```

```
                                                                ;* PROCEDURE TEST
172570                              EXTEST::
                                                                ;* * INITIALIZE IO
172570  004767  000616                      CALL    DINIT
                                                                ;* * SAVE TEST NUMBER
172574  010246                              MOV     R2,-(SP)
                                                                ;* * SAVE LED STATE
172576  013746  177524                      MOV     @#KW.CSC,-(SP)
                                                                ;* * SAVE TEST PARAMETER
172602  010146                              MOV     R1,-(SP)
                                                                ;* * SET NXM FLAG OF STATUS REGISTER
172604  052737  010000  175002              BIS     #KX$NXF,@#KW.STA
                                                                ;* * IF TEST NUMBER < 7
172612  022702  000007                      CMP     #7,R2
172616  103404                              BLO     TE100
                                                                ;* * THEN
                                                                ;* * * CLEAR ROM MAP BIT
172620  042737  000040  177520              BIC     #KX$DPE,@#KL.CSA
172626  000405                              BR      TE200
                                                                ;* * ELSE
172630                              TE100:
                                                                ;* * * SET ROM MAP BIT
172630  052737  000040  177520              BIS     #KX$DPE,@#KL.CSA
                                                                ;* * * SUBTRACT 8 FROM TEST NUMBER
172636  162702  000010                      SUB     #10,R2
                                                                ;* * END IF
172642                              TE200:
                                                                ;* * PERFORM TEST
172642  006302                              ASL     R2      ;MULT. TEST NO. BY 4
172644  006302                              ASL     R2
172646  004762  160000                      CALL    T0(R2)
                                                                ;* * CLEAR NXM FLAG OF STATUS REGISTER
172652  042737  010000  175002              BIC     #KX$NXF,@#KW.STA
                                                                ;* * RETRIEVE TEST PARAMETER
172660  012601                              MOV     (SP)+,R1
                                                                ;* * RESTORE LED STATE
172662  012637  177524                      MOV     (SP)+,@#KW.CSC
                                                                ;* * RETRIEVE TEST NUMBER (BUT LEAVE ON STACK)
172666  011602                              MOV     (SP),R2
                                                                ;* * SAVE TEST RESULTS
172670  010003                              MOV     R0,R3
                                                                ;* * SET LEDS PER RESULTS
172672  006302                              ASL     R2
172674  004772  172746                      CALL    @<ERRTB-2>(R2)
                                                                ;* * RETRIEVE TEST NUMBER
172700  012602                              MOV     (SP)+,R2
                                                                ;* * IF SIGNIFICANT ERROR
172702  005700                              TST     R0
172704  001410                              BEQ     TE300
                                                                ;* * THEN
                                                                ;* * * SET ERROR BIT OF TEST IN DPR 2
172706  005000                              CLR     R0
172710  000261                              SEC
172712  006100                      TE250:  ROL     R0
172714  077202                              SOB     R2,TE250
172716  050037  175004                      BIS     R0,@#KW.SC2
                                                                ;* * * PLACE DISCRETE ERROR DATA IN DPR 3
```

```
 172722  010337  175006                  MOV     R3,@#KW.SC3
                                                                  ;* * END IF
 172726                          TE300:
                                                                  ;* * IF FATAL ERROR
 172726  032737  000040  175002          BIT     #KX$FEF,@#KW.STA
 172734  001401                          BEQ     TE400
                                                                  ;* * THEN
                                                                  ;* * * STOP
 172736  000777                  FTLST:: BR      .
                                                                  ;* * END IF
 172740                          TE400:
                                                                  ;* * ENABLE OVERLAY 0
 172740  042737  000040  177520          BIC     #KX$DPE,@#KW.CSA
                                                                  ;* * RETURN FROM CALL
 172746  000207                          RETURN
                                                                  ;* END TEST


                                 ;TNNERR ENTRY ADDRESS/TEST TABLE

 172750  170506                  ERRTB:  .WORD   T1ERR
 172752  170546                          .WORD   T2ERR
 172754  170576                          .WORD   T3ERR
 172756  170642                          .WORD   T4ERR
 172760  170650                          .WORD   T5ERR
 172762  170656                          .WORD   T6ERR
 172764  170664                          .WORD   T7ERR
 172766  170720                          .WORD   T10ERR
 172770  170726                          .WORD   T11ERR
 172772  170734                          .WORD   T12ERR
 172774  170742                          .WORD   T13ERR
```

```

                                         .SBTTL  ENTRY POINT
         173000                          .=173000
                                         .ENABLE GBL,LC
                                 ; Functional Description:
                                 ;

                                 ; These are the entry points to the native firmware

                                 ; Power Up entry point

 173000  000167  000006          PERV::  JMP     PWRUP

                                 ; Restart entry point

 173004  000167  170770          RSTV::  JMP     RESTRT

```

```

                                        .SBTTL PWRUP - POWER UP HANDLER
                                        .ENABLE GBL,LC
        173012                          .=173012
                                ;
                                ; Module name: PWRUP - POWER UP
                                ;
                                ; System: KXT11-CA Native Firmware
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ; Functional Description:
                                ;
                                ;This routine services the Power Up Interrupt at location 173000.
                                ;
                                ;Input Parameters:
                                ;
                                ;       NONE
                                ;
                                ;Output Parameters
                                ;
                                ;       1. R5 = Top of RAM Pointer
                                ;
                                ;
                                ;Data Structures
                                ;
                                ;       1.  BATTERY BACKUP POWER UP FLAG (STATUS REGISTER)
                                ;
                                ;       2.  POWER UP FLAG (STATUS REGISTER)
                                ;
                                ;       3.  BATTERY BACKUP FLAG 1
                                ;
                                ;       4.  BATTERY BACKUP FLAG 2
                                ;
                                ;       5.  Map Selection Bits (KXTCSR B, bits 1-3)
                                ;
                                ;       6.  Boot Flag (BOOTF0)
                                ;
                                ;Routines Used
                                ;
                                ;       1.  INIT
                                ;
                                ;       2.  PUTEST
                                ;
                                ;       3.  BSUPV (jump to)
                                ;
                                ;       4.  SETLED
                                ;
                                ;       5.  TRAP24 (jump to)

```

```
                                                                ;* PROCEDURE POWER UP
173012                          PWRUP::
                                                                ;* * IF POWER UP VECTOR ENTRY
                                                                ;* * THEN
                                                                ;* * * SET R4 = 0
173012  005004                          CLR     R4
173014  000404                          BR      PW050
                                                                ;* * ELSE
173016                          REINIT::
                                                                ;* * * SET R4 = 1
173016  012704  000001                  MOV     #1,R4
                                                                ;* * * SAVE DPR REGISTER 3
173022  013703  175006                  MOV     @#KW.SC3,R3
                                                                ;* * END IF
173026                          PW050:
                                                                ;* * INITIALIZE THE 8255
173026  012737  000212  177526          MOV     #212,@#C8255
                                                                ;* * CLEAR ALL KXT CSR'S
173034  005037  177530                  CLR     @#KW.CSD
                                                                ;* * CLEAR THE DUAL PORT RAM
173040  012700  000020                  MOV     #20,R0
173044  012701  175000                  MOV     #KW.CMD,R1
173050  005021                  PW100:  CLR     (R1)+
173052  077002                          SOB     R0,PW100
                                                                ;* * READ SLU1 TO CLEAR ANY PENDING BREAKS
173054  005737  177562                  TST     @#RBUF1A
                                                                ;* * SEQUENCE LEDS (ON-.5 SEC.-OFF)
173060  152737  000017  177524          BISB    #<KX$LD0 ! KX$LD1 ! KX$LD2 ! KX$LD3>,@#KW.CSC
173066  005000                          CLR     R0
173070  077001                          SOB     R0,.
173072  077001                          SOB     R0,.
173074  142737  000017  177524          BICB    #<KX$LD0 ! KX$LD1 ! KX$LD2 ! KX$LD3>,@#KW.CSC
                                                                ;* * INITIALIZE STACK POINTER AND SET R5 TO TOP OF RAM
173102  012700  173112                  MOV     #PW200,R0
173106  000167  000574                  JMP     SPR5SU
173112                          PW200:
                                                                ;* * IF POWER UP ENTRY
173112  005704                          TST     R4
173114  001403                          BEQ     PW300
                                                                ;* * OR REINIT ENTRY AND DPR 3 = 8
173116  022703  000010                  CMP     #8.,R3
173122  001013                          BNE     PW500
173124                          PW300:
                                                                ;* * THEN
                                                                ;* * * SET BOOT FLAG = BOOT SWITCH SETTING
173124  113700  177522                  MOVB    @#KL.CSB,R0
173130  042700  177417                  BIC     #^C<KM$SWS>,R0
173134  006200                          ASR     R0
173136  006200                          ASR     R0
173140  006200                          ASR     R0
173142  006200                          ASR     R0
173144  010065  177774                  MOV     R0,BOOTFO(R5)
173150  000402                          BR      PW600
                                                                ;* * ELSE
173152                          PW500:
                                                                ;* * * SET BOOT FLAG = BOOT SELECT PASSED WITH COMMAND
173152  010365  177774                  MOV     R3,BOOTFO(R5)
```

```
                                                            ;* * END IF
173156                          PW600:
                                                            ;* * IF POWER UP ENTRY
173156  005704                          TST     R4
173160  001034                          BNE     PW700
                                                            ;* * AND BATTERY BACKUP FLAG 1 = 125252
173162  022765  125252  000000          CMP     #125252,BBF10(R5)
173170  001030                          BNE     PW700
                                                            ;* * AND BATTERY BACKUP FLAG 2 = 52525
173172  022765  052525  177776          CMP     #52525,BBF20(R5)
173200  001024                          BNE     PW700
                                                            ;* * THEN
                                                            ;* * * SET BATTERY BACKUP POWER UP FLAG
                                                            ;       OF STATUS REGISTER = 1
173202  052737  002000  175002          BIS     #KX$BAT,@#KW.STA
173210  042737  001000  175002          BIC     #KX$PWF,@#KW.STA
                                                            ;* * * INITIALIZE I/O
173216  004767  000060                  CALL    INIT
                                                            ;* * * SET STATUS REGISTER TO NON NATIVE
                                                            ;       CODE STATE
173222  042737  000007  175002          BIC     #KM$STF,@#KW.STA
173230  052737  000007  175002          BIS     #KP$NNC,@#KW.STA
                                                            ;* * * SET LEDS TO NON NATIVE CODE STATE DISP.
173236  012700  000000                  MOV     #NNCED,R0
173242  004767  173732                  CALL    SETLED
                                                            ;* * * EMULATE TRAP TO 24 (NON STRUCTURED EXIT)
173246  000167  174104                  JMP     TRAP24
                                                            ;* * ELSE
173252                          PW700:
                                                            ;* * * RUN SELF TEST
173252  004767  177046                  CALL    PUTEST
                                                            ;* * * INITIALIZE I/O
173256  004767  000020                  CALL    INIT
                                                            ;* * * SET POWER UP FLAG OF STATUS REGISTER = 1
173262  052737  001000  175002          BIS     #KX$PWF,@#KW.STA
173270  042737  002000  175002          BIC     #KX$BAT,@#KW.STA
                                                            ;* * * BOOT IOP (JUMP TO BOOTSTRAP SUPERVISOR)
173276  000167  175666                  JMP     BSUPV
                                                            ;* * END IF
                                                            ;* END POWER UP
```

```
        .SBTTL  INIT - INITIALIZATION
        .ENABLE LC,GBL


;
; Module name: INIT - INITIALIZATION
;
; System: KXT11-CA Native Firmware
;
;
;
;
;
;
; Functional Description:
;
;       This module resets and initializes the DMA controller, 7201 and
;       PIO. It also places the configuration data in the dual port RAM
;       and enables the dual port RAM.
;
;Input Parameters:
;
;       NONE
;
;Output Parameters
;
;     1.  Reset DMA,7201,PIO
;
;     2.  PIO Master Interrupt Register
;
;     3.  " Port 1 Intr.  Vec.  Reg.
;
;     4.  "  "   2  "      "     "
;
;     5.  " Timmer  "      "     "
;
;
```

```
                                                                      ;* PROCEDURE INITIALIZATION
173302                            INIT::
                                                                      ;* * INITIALIZE IO
173302  004767  000104                      CALL    DINIT
                                                                      ;* * CLEAR ALL OF STATUS REGISTER EXCEPT STATE FIELD
                                                                      ;          AND POWER UP FLAGS
173306  042737  174770  175002              BIC     #^C<KM$STF ! KX$PWF ! KX$BAT>,@#KW.STA
                                                                      ;* * IF NO AUTO SELF TEST ERRORS FOUND
173314  005737  175004                      TST     @#KW.SC2
173320  001030                              BNE     IN400
                                                                      ;* * THEN
                                                                      ;* * * PLACE IOP MAP & BOOT CONFIGURATION IN DPR REGISTER 3
173322  013700  177522                      MOV     @#KW.CSB,R0
173326  042700  177401                      BIC     #^C<KM$MAP ! KM$SWS>,R0
173332  010037  175006                      MOV     R0,@#KW.SC3
                                                                      ;* * * IF USER SOCKETS MAPPED TO 0
173336  032737  000010  177522              BIT     #KX$MP2,@#KL.CSB
173344  001402                              BEQ     IN100
                                                                      ;* * * THEN
                                                                      ;* * * * TEST FOR RAM AT 0
173346  005000                              CLR     R0
173350  000402                              BR      IN200
                                                                      ;* * * ELSE
173352                            IN100:
                                                                      ;* * * * TEST FOR RAM AT 100000
173352  012700  100000                      MOV     #100000,R0
                                                                      ;* * * END IF
173356                            IN200:
                                                                      ;* * * IF RAM PRESENT
173356  011001                              MOV     (R0),R1
173360  005101                              COM     R1
173362  005110                              COM     (R0)
173364  020110                              CMP     R1,(R0)
173366  001005                              BNE     IN300
                                                                      ;* * * THEN
                                                                      ;* * * * RESTORE CONTENTS
173370  005101                              COM     R1
173372  010110                              MOV     R1,(R0)
                                                                      ;* * * * SET RAM PRESENT BIT OF DPR REG. 3
173374  052737  000001  175006              BIS     #BIT0,@#KW.SC3
                                                                      ;* * * END IF
173402                            IN300:
                                                                      ;* * END IF
173402                            IN400:
                                                                      ;* * ENABLE THE DUAL PORT RAM
173402  052737  000100  177530              BIS     #KX$DEN,@#KW.CSD
                                                                      ;* * RETURN
173410  000207                              RETURN
                                                                      ;* END INITIALIZATION
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 93
DINIT - DEVICE INITIALIZATION

                                            .SBTTL  DINIT - DEVICE INITIALIZATION
                                            .ENABLE LC,GBL



                                    ;
                                    ; Module name: DINIT - I/O DEVICE INITIALIZATION
                                    ;
                                    ; System: KXT11-CA Native Firmware
                                    ;
                                    ;
                                    ;
                                    ;
                                    ; Functional Description:
                                    ;
                                    ; Initialize all programmable I/O devices. Disables interrupts and sets
                                    ; programable interrupt vector addresses.
                                    ;
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 94
DINIT - DEVICE INITIALIZATION

                                                          ;* PROCEDURE DINIT
 173412                           DINIT::
                                                          ;* * RESET DMA
 173412  005037  174454                   CLR     @#DMACDA
                                                          ;* * RESET 7201
 173416  113700  175700                   MOVB    @#SL2SAA,R0
 173422  112737  000030  175704           MOVB    #30,@#SL2CAA
 173430  112737  000030  175714           MOVB    #30,@#SL2CBA
 173436  112737  000002  175704           MOVB    #2,@#SL2CAA
 173444  112737  000024  175704           MOVB    #24,@#SL2CAA
 173452  112737  000001  175714           MOVB    #1,@#SL2CBA     ;(SET STATUS EFFECTS VECTOR BIT)
 173460  112737  000004  175714           MOVB    #4,@#SL2CBA
                                                          ;* * RESET PIO
 173466  112737  000001  177000           MOVB    #01,@#PIOICA
 173474  105037  177000                   CLRB    @#PIOICA
                                                          ;* * SETUP PIO PORT A INTERRUPT VECTOR REGISTER
 173500  112737  000200  177004           MOVB    #200,@#PIOAVA
                                                          ;* * SETUP PIO PORT B INTERRUPT VECTOR REGISTER
 173506  112737  000204  177006           MOVB    #204,@#PIOBVA
                                                          ;* * SETUP PIO TIMER INTERRUPT VECTOR REGISTER
 173514  112737  000210  177010           MOVB    #210,@#PIOTVA
                                                          ;* * SETUP DMA CHANNEL 0 INTERRUPT VECTOR REGISTER
 173522  112737  000110  174532           MOVB    #110,@#DMA0VA
                                                          ;* * SETUP DMA CHANNEL 1 INTERRUPT VECTOR REGISTER
 173530  112737  000114  174530           MOVB    #114,@#DMA1VA
                                                          ;* END DINIT
 173536  000207                           RETURN
```

```
                        .SBTTL  ACCUM - ACCUMULATE NUMBER
                        .ENABLE LC,GBL

                ;
                ; Module name: ACCUMULATE - ACCUM
                ;
                ; System: KXT11-CA Native Firmware
                ;
                ;
                ;
                ;
                ;
                ; Functional Description:
                ;
                ;       This routine accumulates a 16 bit number by shifting the previous
                ; accumulation left 3 and "or"ing the new input value (which is a single
                ; octal digit value) into the accumulation.


                ;Input Parameters:

                ;       1) R0 = OCTAL DIGIT INPUT
                ;       2) R4 = PREVIOUS ACCUMULATED NUMBER

                ;Output Parameters:

                ;       1) R4 = ACCUMULATED NUMBER

                ;Routines Used:

                ;       None
```

```
                                                ;* PROCEDURE ACCUMULATE
173540                  ACCUM::
                                                ;* * SHIFT INPUT LEFT 3
173540  006304                  ASL     R4
173542  006304                  ASL     R4
173544  006304                  ASL     R4
                                                ;* * SET ACCUMULATED NUMBER = ACCUMULATED NUMBER [OR] DATA
173546  050004                  BIS     R0,R4
                                                ;* END ACCUMULATE
173550  000207                  RETURN
```

```
                                        .SBTTL  GETIN - GET INPUT
                                        .ENABLE LC,GBL

                                ;
                                ; Module name: GET INPUT - GETIN
                                ;
                                ; System: KXT11-CA Native Firmware
                                ;
                                ;
                                ;
                                ;
                                ;
                                ; Functional Description:
                                ;
                                ;       This routine inputs a character and sets the non numeric flag if the
                                ; input is not a number. If the input is a number it is converted to binary,
                                ; clears the non-numeric flag and sets the data received flag.
                                ;
                                ;       Output Parameters:
                                ;
                                ;       1) Data Received flag - DARFGB of R3
                                ;       2) Non Numeric Flag - Carry Flag
                                ;       3) Input Data - R0
```

```

 173552                         GETIN::
                                                                ;* PROCEDURE GET INPUT
                                                                ;* * DO UNTIL INPUT
 173552  004767  000036         GE100:  CALL    CHKCHR
                                                                ;* * END UNTIL
 173556  103375                         BCC     GE100
                                                                ;* * IF INPUT = 0 THRU 7
 173560  122700  000060                 CMPB    #'0,R0
 173564  101011                         BHI     GE200
 173566  122700  000067                 CMPB    #'7,R0
 173572  103406                         BLO     GE200
                                                                ;* * THEN
                                                                ;* * * CONVERT TO BINARY
 173574  162700  000060                 SUB     #60,R0
                                                                ;* * * SET DATA RECEIVED FLAG
 173600  052703  000002                 BIS     #DARFGB,R3
                                                                ;* * * CLEAR NON NUMERIC FLAG
 173604  000241                         CLC
 173606  000401                         BR      GE300
                                                                ;* * ELSE
 173610                         GE200:
                                                                ;* * * SET NON NUMERIC FLAG
 173610  000261                         SEC
                                                                ;* * END IF
 173612                         GE300:
                                                                ;* * RETUTN FROM CALL
 173612  000207                         RETURN
                                                                ;* END GET INPUT

```

```
                .SBTTL  CHKCHR - CHECK CHARACTER
                .ENABLE GBL,LC

        ;
        ; Module name: CHKCHR - CHECK CHARACTER
        ;
        ; System: Micro-PAX RTS
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module tests the console serial line receive buffer for a
        ;       character, signals if one was received ,echos the character
        ;       and returns that character to the calling routine.
        ;       This module is to be used for input polling. It
        ;       will handle XON XOFF protocol. This routine will not ecco
        ;       the XON or XOFF characters.

        ;INPUT PARAMETERS

        ;       NONE

        ;OUTPUT PARAMETERS

        ;       1) CARRY BIT = 1 FOR CHARACTER RECEIVED, = 0 FOR NO CHARACTER
        ;       2) R0 = RECEIVE BUFFER DATA, IF DATA RECEIVED

        ;DATA STRUCTURES USED

        ;       1) SLU1 CSR
        ;       2) SLU1 RECEIVE BUFFER

        ;ROUTINES USED

        ;       NONE
```

```
        000021                  XON     =       ^Q-100
        000023                  XOFF    =       ^S-100
```

```
                                                                        ;* PROCEDURE CHECK CHARACTER
173614                            CHKCHR::
173614  000241                            CLC
                                                                        ;* * IF RECEIVE DONE
173616  105737  177560                    TSTB    @#RCSR1A
173622  100030                            BPL     CH400
                                                                        ;* * THEN
                                                                        ;* * * READ RECEIVE BUFFER
173624  113700  177562                    MOVB    @#RBUF1A,R0
                                                                        ;* * * CLEAR PARITY BIT
173630  042700  000200                    BIC     #BIT7,R0
                                                                        ;* * * CASE CHARACTER
                                                                        ;* * * WHEN XOFF
173634  122700  000023                    CMPB    #XOFF,R0
173640  001004                            BNE     CH100
                                                                        ;* * * * SET XOFF FLAG = 1
173642  052703  000040                    BIS     #XOFFGB,R3
173646  000241                            CLC
173650  000415                            BR      CH300
                                                                        ;* * * WHEN XON
173652                            CH100:
173652  122700  000021                    CMPB    #XON,R0
173656  001004                            BNE     CH200
                                                                        ;* * * * SET XOFF FLAG = 0
173660  042703  000040                    BIC     #XOFFGB,R3
173664  000241                            CLC
173666  000406                            BR      CH300
                                                                        ;* * * ELSE
173670                            CH200:
                                                                        ;* * * * IF CHARACTER NOT LF
173670  122700  000012                    CMPB    #LF,R0
173674  001402                            BEQ     CH250
                                                                        ;* * * * THEN
                                                                        ;* * * * * ECHO CHARACTER
173676  010037  177566                    MOV     R0,@#XBUF1A
                                                                        ;* * * * END IF
173702                            CH250:
                                                                        ;* * * * SIGNAL CHARACTER RECEIVED
173702  000261                            SEC
                                                                        ;* * * END CASE
173704                            CH300:
                                                                        ;* * END IF
173704                            CH400:
173704  000207                            RETURN
```

```
                .SBTTL  SPR5SU - SP/R5 SETUP
                .ENABLE LC,GBL

        ;
        ; Module name: SP AND R5 SETUP - SPR5SU
        ;
        ; System: KXT11-CA Native Firmware
        ;
        ;
        ;
        ;
        ;
        ;
        ; Functional Description:
        ;
        ;       This module sets up the SP to the native firmware stack area and
        ; setting R5 to the address of the last word of native RAM.
        ;
        ;
        ; Input Parameters:
        ;
        ;       1) Return address in R0
        ;
        ; Output Parameters:
        ;
        ;       1) Top of RAM address in R5
        ;       2) Stack Pointer
```

```

                                                                  ;* PROCEDURE SP AND R5 SETUP
173706                          SPR5SU::
                                                                  ;* * CASE MAP SELECTION
                                                                  ;* * WHEN MAP BIT 2 = 0
173706  132737  000010  177522          BITB    #KX$MP2,@#KW.CSB
173714  001003                          BNE     SP200
                                                                  ;* * * SET STACK POINTER TO STACK 1
173716  012706  077762                  MOV     #SP1,SP
173722  000415                          BR      SP400
                                                                  ;* * WHEN MAP BITS 0 AND 1 = 11
173724  132737  000002  177522  SP200:  BITB    #KX$MP0,@#KW.CSB
173732  001407                          BEQ     SP300
173734  132737  000004  177522          BITB    #KX$MP1,@#KW.CSB
173742  001403                          BEQ     SP300
                                                                  ;* * * SET STACK POINTER TO STACK 3
173744  012706  157762                  MOV     #SP3,SP
173750  000402                          BR      SP400
                                                                  ;* * ELSE
173752                          SP300:
                                                                  ;* * * SET STACK POINTER TO STACK 2
173752  012706  137762                  MOV     #SP2,SP
                                                                  ;* * END CASE
173756                          SP400:
                                                                  ;* * SET R5 = TOP OF RAM (SP + STACK OFFSET)
173756  010605                          MOV     SP,R5
173760  062705  000014                  ADD     #SPOFF,R5
                                                                  ;* * RETURN TO CALLER
173764  000110                          JMP     (R0)
                                                                  ;* END SP AND R5 SETUP
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 104
VERS - VERSION NUMBER AND TEST

                                          .SBTTL  VERS - VERSION NUMBER AND TEST
          173774                          .=173774
                                          .ENABLE LC

                                  ;
                                  ; Module name: VERS - VERSION NUMBER
                                  ;
                                  ; System: KXT11-CA Native Firmware
                                  ;
                                  ;
                                  ;
                                  ;
                                  ;
                                  ;
                                  ; Functional Description:
                                  ;
                                  ;This module places the version number in the last word of the
                                  ;permanently mapped native ROM.

    173774     126     061     056  VERS::  .ASCII  'V1.0'
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 105
SELF TEST OVERLAY 1

                                          .SBTTL  SELF TEST OVERLAY 1
                                          .LIST   ME,MEB,SEQ,LOC,BIN      ; NORMAL LISTING MODE
                                          .NLIST  MC,CND,BEX              ;       DITTO
                                          .ENABL  AMA,GBL
          174000                          .=174000

                                  ;
                                  ; Module name: SELF TEST OVERLAY 1
                                  ;
                                  ; System: KXT11-CA Native Firmware
                                  ;
                                  ;
                                  ;
                                  ;
                                  ;
                                  ;
                                  ; Functional Description:
                                  ;
                                  ;       This module contains overlay 1 of the self test code. The self test
                                  ;       modules in this overlay are:
                                  ;
                                  ;               Parallel I/O port Z8036,
                                  ;               DMA controller Z8016,
                                  ;               Q-BUS interrupts, and
                                  ;               Two-port RAM.
                                  ;
                                  ;       Overlay 1 physically occupies the top 2 KB of the boot ROM. It
                                  ;       executes starting at 160000 when the DIAG PROM EN bit, bit 5, in
                                  ;       KXTCSRA is 1.
```

```
                ;       SCRATCH RAM ALLOCATION.
                ;
                ; R5 IS ODT'S "TOP-OF-MEM" POINTER, AND IS PRESERVED AS SUCH.
                ; ALL SCRATCH REFERENCES ARE INDEXED FROM R5 AS FOLLOWS:
                ;
                ;                                   LO     MID      HI
                ;                                 ------  ------  ------
000005          $TOP=   %5                      ; 077776  137776  157776
                ;
150002          BUFR1=  -27776                  ; 050000  110000  130000  TOP-6KW
160002          BUFR2=  -17776                  ; 060000  120000  140000  TOP-4KW
170002          TMP1=   -07776                  ; 070000  130000  150000  TOP-2KW
170004          TMP2=   TMP1+2                  ;                         \
170006          TMP3=   TMP1+4                  ;                          \
170010          TMP4=   TMP1+6                  ;                           \
170012          TMP5=   TMP1+10                 ;                            > SCRATCH REGISTERS...
170014          TMP6=   TMP1+12                 ;                           /  ...AND STACK SPACE.
170016          TMP7=   TMP1+14                 ;                          /
170020          TMP8=   TMP1+16                 ;                         /
177776          $EF1=   -2                      ; LAST 2 WORDS OF NATIVE RAM (BATTERY-BACK-FLAGS)...
000005          $EF=    $TOP                    ;...ARE UTILIZED AS OUR ERROR FLAGS.
                ;
                ; MISCELLANEOUS STUFF.
                ;
000007          $TN=    7                       ; INIT TEST NUMBER SEQUENCE.
014000          K3K=    3072.*2                 ; HISEG CALLS REQUIRE 3KW OFFSET.
                ;
                ; A FEW MACROS.
                ;





                        .NLIST MD

```

```
                              ; A FEW DEFINITIONS
                              ;
                              ; STANDARD STUFF.
                              ;
        000000                PR0=    000             ; THE PRIORITIES.
        000040                PR1=    040
        000100                PR2=    100
        000140                PR3=    140
        000200                PR4=    200
        000240                PR5=    240
        000300                PR6=    300
        000340                PR7=    340
                              ;
                              ; PRIMARY CONTROL REGISTERS.
                              ;
        177560                $SL1=   177560          ; DEC DC319 (DLART) ASYNC SERIAL PORT.
        175700                $SL2=   175700          ; NEC7201 SYNC/ASYNC SERIAL PORT.
        175720                $I8254= 175720          ; I8254 PROGRAMMABLE BAUD GEN FOR ABOVE.
        177000                $PIO=   177000          ; Z8036 PIO/PIT.
        174400                $DMA=   174400          ; AMZ8016 DMA CONTROLLER.
        177520                $CSRA=  177520          ; \
        177522                $CSRB=  177522          ;  > SECONDARY CSR'S...
        177524                $CSRC=  177524          ; /  ...EMBEDDED IN I8255 CHIP.
        177526                $CSRCON= 177526         ;
        177530                $CSR=   177530          ; PRIMARY CSR.
        177532                $QIR=   177532          ; Q-BUS INTERRUPT REGISTER.
        175000                $DPR=   175000          ; DUAL-PORT RAM (LOCAL).
        160000                $QDPR1= 160000          ; DUAL-PORT RAM (GLOBAL BASE ID 0-7...
        175400                $QDPR2= 175400          ;                         ...AND ID 8-F).
                              ;
        175000                $IPV=   $DPR            ; IPV INTERRUPTS (VIA DPR WORDS 0, 4, 8, AND 12).
        177524                $LEDS=  $CSRC           ; CSRC<3:0> DRIVE THE LEDS.
        177524                $UID=   $CSRC           ; CSRC<7:4> SHOW THE ID SWITCH SETTING.
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 108
SELF TEST OVERLAY 1

                                ; DEFAULT VECTORS.
                                ;
        000024                  PWRV=   024            ; PWR-UP (BRESET)        (NON-MASKABLE).
        000060                  SL1RV=  060            ; CONSOLE (SLU1) RCVR    (PRI 4).
        000064                  SL1XV=  064            ; CONSOLE (SLU1) XMTR    (PRI 4).
        000070                  NECV=   070            ; SYNC/ASYNC BOTH CHANS (PRI 4).
        000100                  BEVNT=  100            ; LINE CLOCK             (PRI 6).
        000104                  PEVNT=  104            ; PROGRAMMABLE CLOCK(S) (PRI 6).
                                ;       110            ;       OPEN
        000114                  PRTYV=  114            ; MEMORY PARITY          (PRI 7).
        000120                  DPRV4=  120            ; DUAL-PORT RAM WORD 4   (PRI 5).
        000124                  DPRV8=  124            ; DUAL-PORT RAM WORD 8   (PRI 5).
        000130                  BIACKV= 130            ; Q-BUS "IACK"           (PRI 5).
        000134                  DPRV12= 134            ; DUAL-PORT-RAM WORD 12 (PRI 5).
                                ;       140            ;       RESERVED (FALCON).
        000144                  QIRV=   144            ; Q-BUS REQUEST          (ARBITER EXECUTES RTI).
        000150                  QIRV1=  150            ;   DITTO                (ARBITER EXECUTES RESET,RTI)
        000154                  PIOAV=  154            ; PIO PORT A             (PRI 4).
        000160                  PIOBV=  160            ; PIO PORT B             (PRI 4).
        000164                  DMAV=   164            ; DMA (BOTH CHANNELS)    (PRI 4).
```

```
KXTNF - KXT11CA Native Firmware MACRO V05.00   00:51  Page 109
SELF TEST OVERLAY 1

                                ; HIGH SEGMENT ENTRY BLOCK
                                ;
                                ; CALL: BIS  #BIT5,CSRA ; ENABLE HIGH SEGMENT.
                                ;       CALL TN         ; CALL TEST 10-17.
                                ;       BEQ  OK         ; OR BNE ERROR.
        160000                  T10==.-K3K
174000  000137  160040                  JMP     TST10-K3K       ;; EXECUTE TEST 10.
        160004                  T11==.-K3K
174004  000137  161106                  JMP     TST11-K3K       ;; EXECUTE TEST 11.
        160010                  T12==.-K3K
174010  000137  162102                  JMP     TST12-K3K       ;; EXECUTE TEST 12.
        160014                  T13==.-K3K
174014  000137  162432                  JMP     TST13-K3K       ;; EXECUTE TEST 13.
        160020                  T14==.-K3K
174020  005000                          CLR     R0      ;SINCE THERE IS NO TEST BY THIS NAME, SHOW NO ERRORS.
174022  000207                          RETURN
        160024                  T15==.-K3K
174024  005000                          CLR     R0      ;SINCE THERE IS NO TEST BY THIS NAME, SHOW NO ERRORS.
174026  000207                          RETURN
        160030                  T16==.-K3K
174030  005000                          CLR     R0      ;SINCE THERE IS NO TEST BY THIS NAME, SHOW NO ERRORS.
174032  000207                          RETURN
        160034                  T17==.-K3K
174034  005000                          CLR     R0      ;SINCE THERE IS NO TEST BY THIS NAME, SHOW NO ERRORS.
174036  000207                          RETURN
```

```
                                 .SBTTL  - T10   PARALLEL I/O PORT Z8036
                                 ;;*********************************************************************
                                 ;;* TEST 10 -- PARALLEL I/O PORT Z8036
                                 ;;*********************************************************************
174040                           TST10:
174040                           TPIO:
174040  013746  000104                   MOV     PEVNT,-(SP)     ;; SAVE PEVNT.
174044  013746  000106                   MOV     PEVNT+2,-(SP)
174050  013746  000154                   MOV     PIOAV,-(SP)     ;; SAVE PIOAV.
174054  013746  000156                   MOV     PIOAV+2,-(SP)
174060  013746  000160                   MOV     PIOBV,-(SP)     ;; SAVE PIOBV.
174064  013746  000162                   MOV     PIOBV+2,-(SP)
174070  005015                           CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                 ;
                                 ; INPUT:  NONE.
                                 ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                 ;
                                 ; ZILOG 8036 CHIP HAS 48 REGISTERS.
                                 ; RESET THE CHIP AND VERIFY THAT ALL REGISTERS RETURN ZERO.
                                 ;
                                 ;       T10     PARALLEL I/O PORT Z8036
                                 ;       .         E0 = ROM IN VECTOR SPACE -- CAN'T RUN
                                 ;       .         E1 = RESET STATE INCORRECT
                                 ;       .         E2 = TIMER DIDN'T START
                                 ;       .         E3 = TIMER NEVER STOPS.
                                 ;       .         E4 = INTERRUPT NOT MASKED AT LEVEL 4
                                 ;       .         E5 = INTERRUPT NOT RECEIVED.
                                 ;       .         E6 = LOOP TIME-OUT, DATA XFER INCOMPLETE
                                 ;       .         E7 = RECEIVED DATA INCORRECT

174072  004737  163602                   CALL    XRAMLO-K3K      ; VECTOR SPACE USABLE ??
174076  001404                           BEQ     4$              ; PROCEED IF SO.
174100  052715  000001                   BIS     #BIT0,($EF)     ;; E0 = ROM IN VECTOR SPACE -- CAN'T RUN
174104  000137  160746                   JMP     PIOXIT-K3K

174110  012703  000060           4$:     MOV     #48.,R3         ; CHECK 48 REGISTERS...
174114  012704  177000                   MOV     #$PIO,R4        ;...STARTING HERE.
174120  112714  000001                   MOVB    #1,(R4)         ; SET "CHIP-RESET" IN MICR.
174124  122714  000001                   CMPB    #1,(R4)         ; READ IT , SHOULD SEE THE 1 BIT.
174130  000403                           BR      2$              ; CHECK IT AT 2$.

174132  112714  177777           1$:     MOVB    #-1,(R4)        ; THEN, WRITES SHOULD BE IGNORED...
174136  105714                           TSTB    (R4)            ;...AND READS RETURN ZERO.
174140  001402                   2$:     BEQ     3$              ; BR IF OK.
174142  052715  000002                   BIS     #BIT1,($EF)     ;; E1 = RESET STATE INCORRECT
174146  005724                   3$:     TST     (R4)+           ; NEXT ADDRESS...
174150  077310                           SOB     R3,1$           ;...AND LOOP 'TIL DONE.
                                 ;
                                 ; NEXT, SEE THAT EACH TIMER FUNCTIONS AS ADVERTISED.
                                 ;
        000000                   MIC=    000             ; MASTER INTERRUPT CONTROL REG.
        000002                   MCC=    002             ; MASTER CONFIGURATION CONTROL REG.
        000010                   CTV=    010             ; TIMER(S) INTERRUPT VECTOR REG.
        000070                   CTMS=   070             ; 1ST (OF 3) TIMER MODE SPEC REG.
        000024                   CTCS=   024             ; 1ST (OF 3) TIMER CMD/STATUS REG.
        000054                   CTTC=   054             ; 1ST (OF 6) TIME CONSTANT REG (3 PAIR).
        000040                   CTCC=   040             ; 1ST (OF 6) CURRENT COUNT REG (3 PAIR).
```

```
                                ;
174152  012704  177000          PIT:    MOV     #$PIO,R4        ; GET BASE ADDRESS.
174156  142714  000001                  BICB    #1,(R4)         ; CLEAR "CHIP-RESET"...
174162  012703  000060                  MOV     #48.,R3
174166  005024                  2$:     CLR     (R4)+           ;...AND CLEAR ALL REGISTERS.
174170  077302                          SOB     R3,2$
174172  012704  177000                  MOV     #$PIO,R4        ; GET BASE AGAIN.
174176  112714  000200                  MOVB    #200,(R4)       ; SET "MASTER INT ENAB".
174202  112764  000104  000010          MOVB    #PEVNT,CTV(R4)  ; SET TO VECTOR THRU "PEVNT".
174210  112764  000160  000002          MOVB    #160,MCC(R4)    ; ENABLE ALL 3 TIMERS...
                                                                ;...CTMS = 0 = PULSE-OUT, ONCE-ONLY...
                                                                ;...CTTC = 0 = TIME K 65536.
174216  062704  000024                  ADD     #CTCS,R4        ; NOW RAISE R4 TO 1ST TIMER CSR.

174222  106427  000200          PIT1:   MTPS    #PR4            ; RAISE CPU TO PIO LEVEL.
174226  012737  160320  000104          MOV     #3$-K3K,PEVNT   ; SET VECTOR.
174234  012737  000300  000106          MOV     #PR6,PEVNT+2
174242  005000                          CLR     R0
174244  012701  000004                  MOV     #4,R1           ; SET A 1 SECOND KEEP-ALIVE TIMER.
174250  112714  000040                  MOVB    #40,(R4)        ; CLEAR "IUS!IP"...
174254  112714  000306                  MOVB    #306,(R4)       ;...AND SET "IE!GCB!TCB"...
174260  000240                          NOP                     ;...TIMER SHOULD TAKE-OFF !!
174262  132714  000001          1$:     BITB    #1,(R4)         ; TIMER RUNNING (CIP=1) ??
174266  001004                          BNE     2$              ; BR IF SO.
174270  077004                          SOB     R0,1$
174272  052715  000004                  BIS     #BIT2,($EF)     ;; E2 = TIMER DIDN'T START
174276  000426                          BR      7$              ; ABORT.
174300  132714  000001          2$:     BITB    #1,(R4)         ; TIMER DONE (CIP=0) ??
174304  001411                          BEQ     4$              ; BR IF SO.
174306  077004                          SOB     R0,2$
174310  077105                          SOB     R1,2$
174312  052715  000010                  BIS     #BIT3,($EF)     ;; E3 = TIMER NEVER STOPS.
174316  000416                          BR      7$              ; ABORT.

174320  000240                  3$:     NOP                     ; INTERRUPT -- SHOULD HAVE BEEN MASKED.
174322  052715  000020                  BIS     #BIT4,($EF)     ;; E4 = INTERRUPT NOT MASKED AT LEVEL 4
174326  000411                          BR      6$

174330  062737  000032  000104  4$:5$:  ADD     #6$-3$,PEVNT    ; NOW CHANGE THE VECTOR...
174336  106427  000000                  MTPS    #PR0            ;...LOWER THE CPU...
174342  000240                          NOP                     ;...AND LET INTERRUPT COME IN.
174344  052715  000040                  BIS     #BIT5,($EF)     ;; E5 = INTERRUPT NOT RECEIVED.
174350  000401                          BR      7$

174352  022626                  6$:     CMP     (SP)+,(SP)+     ; INTERRUPT -- FIX THE STACK.
174354  012714  000040          7$:     MOV     #40,(R4)        ; CLEAR "IUS!IP!GCB"...
174360  012714  000160                  MOV     #160,(R4)       ;...AND "IE" (TIMER OFF).
174364  005724                          TST     (R4)+           ; BUMP TO NEXT TIMER CMD/STAT...
174366  020427  177032                  CMP     R4,#$PIO+CTCS+6
174372  103713                          BLO     PIT1            ;...AND LOOP 'TIL ALL 3 DONE.

174374  112737  000001  177000          MOVB    #1,$PIO         ; THEN, RESET THE CHIP.
174402  000240                          NOP                     ; FALL THRU TO DATA TEST.
```

```
                                     ;
                                     ; DATA PORT(S) TEST.
                                     ;
                                     ; *** REQUIRES EXTERNAL LOOP-BACK ***
                                     ;
          000004                     PAV=     004              ; PORT A VECTOR...
          000020                     PACS=    020              ;...CMD/STATUS...
          000032                     PADB=    032              ;...DATA BUFFER...
          000100                     PAMS=    100              ;...MODE...
          000102                     PAHS=    102              ;...AND HANDSHAKE SPEC.
          000006                     PBV=     006              ; PORT B VECTOR...
          000022                     PBCS=    022              ;...CMD/STATUS...
          000034                     PBDB=    034              ;...DATA BUFFER...
          000120                     PBMS=    120              ;...MODE...
          000122                     PBHS=    122              ;...AND HANDSHAKE SPEC.
          000014                     PCDD=    014              ; PORT C DATA DIRECTION...
          000036                     PCDB=    036              ;...AND BUFFER (HANDSHAKE BITS).
          000140                     XCVR=    140              ; 1ST WORD ABOVE THE CHIP HOLDS...
          164377                     AI.BO=   <350*400>!377    ;...XCVR CONTROL BITS TTL/A(IN)!B(OUT).
          165400                     AO.BI=   <353*400>!0      ;        DITTO          TTL/A(OUT)!B(IN).
                                     ;
          170002                     BYTES=   TMP1             ; DEFINE A SCRATCH LOCATION.
                                     ;
 174404   012704  177000             PIOD:    MOV     #$PIO,R4         ; GET BASE ADDRESS.
 174410   142714  000001                      BICB    #1,(R4)          ; CLEAR "CHIP-RESET".
 174414   012701  000154                      MOV     #PIOAV,R1
 174420   010164  000004                      MOV     R1,PAV(R4)       ; SET PORT A VECTORS.
 174424   012721  161010                      MOV     #AI-K3K,(R1)+
 174430   012721  000200                      MOV     #PR4,(R1)+
 174434   010164  000006                      MOV     R1,PBV(R4)       ; SET PORT B VECTORS.
 174440   012721  161024                      MOV     #BI-K3K,(R1)+
 174444   012721  000200                      MOV     #PR4,(R1)+
 174450   112764  000100  000100     PIOD1:   MOVB    #100,PAMS(R4)    ; SET PORT A INPUT...
 174456   112764  000000  000102              MOVB    #000,PAHS(R4)    ;...INTERLOCKED HS...
 174464   112764  000300  000020              MOVB    #300,PACS(R4)    ;...INT ENABLED.
 174472   112764  000200  000120              MOVB    #200,PBMS(R4)    ; SET PORT B OUTPUT...
 174500   112764  000000  000122              MOVB    #000,PBHS(R4)    ;...INTERLOCKED HS...
 174506   112764  000300  000022              MOVB    #300,PBCS(R4)    ;...INT ENABLED.
 174514   112764  000200  000022              MOVB    #200,PBCS(R4)    ; SET OUT "IP" TO PRIME THE LOOP.
 174522   012764  164377  000140              MOV     #AI.BO,XCVR(R4)  ; SET XCVR CONTROL BITS.
 174530   000414                              BR      PIOD3
 174532   112764  000200  000100     PIOD2:   MOVB    #200,PAMS(R4)    ; CHANGE PORT A TO OUTPUT...
 174540   112764  000200  000020              MOVB    #200,PACS(R4)    ;...AND SET OUT "IP".
 174546   112764  000100  000120              MOVB    #100,PBMS(R4)    ; CHANGE PORT B TO INPUT.
 174554   012764  165400  000140              MOV     #AO.BI,XCVR(R4)  ; CHANGE XCVR CONTROLS.
 174562   106427  000200             PIOD3:   MTPS    #PR4             ; RAISE CPU.
 174566   112764  000377  000014              MOVB    #377,PCDD(R4)    ; PORT C DIRECTION ALWAYS "IN".
 174574   012702  163646                      MOV     #XFLT10-K3K,R2   ; XMT BUFFER POINTER => R2.
 174600   010503                              MOV     $TOP,R3
 174602   062703  150002                      ADD     #BUFR1,R3        ; RCV BUFFER POINTER => R3.
 174606   010346                              MOV     R3,-(SP)         ; SAVE A COPY OF R3.
 174610   012700  000020                      MOV     #16.,R0
 174614   005023                     1$:      CLR     (R3)+            ; CLEAR IT...
 174616   077002                              SOB     R0,1$
 174620   011603                              MOV     (SP),R3          ;...AND RESET POINTER.
 174622   005001                              CLR     R1
 174624   005065  170002                      CLR     BYTES($TOP)      ; CLEAR BYTE COUNTERS.
```

```
174630  112714  000200                    MOVB    #200,(R4)         ; SET "MIE"...
174634  112764  000224  000002            MOVB    #224,MCC(R4)      ;...AND ENABLE ALL THREE PORTS.
174642  116400  000032                    MOVB    PADB(R4),R0       ; DUMMY READ (FLUSH)...
174646  116400  000034                    MOVB    PBDB(R4),R0       ;...BOTH DB'S.
174652  106427  000000                    MTPS    #PR0              ; LOWER CPU...
                                                                    ;...DATA SHOULD BEGIN TO FLOW.
174656  026527  170002  011022  2$:       CMP     BYTES($TOP),#<18.*BITB>!18. ; 18 BYTES TRANSFERRED ??
174664  001403                            BEQ     3$                ; BR IF SO.
174666  077105                            SOB     R1,2$             ; HANG-AROUND FOR A WHILE.
174670  052715  000100                    BIS     #BIT6,($EF)       ;; E6 = LOOP TIME-OUT, DATA XFER INCOMPLETE

174674  012603                  3$:       MOV     (SP)+,R3          ; GET RCV'D...
174676  012702  163646                    MOV     #XFLT10-K3K,R2    ;...AND XMT'D POINTERS...
174702  012701  000022                    MOV     #18.,R1
174706  122223                  4$:       CMPB    (R2)+,(R3)+       ;...AND COMPARE XMTD/RCVD DATA.
174710  001402                            BEQ     5$
174712  052715  000200                    BIS     #BIT7,($EF)       ;; E7 = RECEIVED DATA INCORRECT
174716  077105                  5$:       SOB     R1,4$

174720  105764  000100          6$:       TSTB    PAMS(R4)          ; CHECK PORT A DIRECTION.
174724  100404                            BMI     7$                ; BR IF OUTPUT.
174726  105014                            CLRB    (R4)              ; ELSE, CLEAR MASTER...
174730  105064  000002                    CLRB    MCC(R4)           ;...CONTROL REGISTERS...
174734  000676                            BR      PIOD2             ;...AND GO 'ROUND ONCE MORE.

174736  112714  000001          7$:       MOVB    #1,(R4)           ; THEN, RESET THE CHIP...
174742  005064  000140                    CLR     XCVR(R4)          ;...AND TRANCEIVER CONTROLS.
174746                          PIOXIT:
174746  012637  000162                    MOV     (SP)+,PIOBV+2     ;; RESTORE PIOBV.
174752  012637  000160                    MOV     (SP)+,PIOBV
174756  012637  000156                    MOV     (SP)+,PIOAV+2     ;; RESTORE PIOAV.
174762  012637  000154                    MOV     (SP)+,PIOAV
174766  012637  000106                    MOV     (SP)+,PEVNT+2     ;; RESTORE PEVNT.
174772  012637  000104                    MOV     (SP)+,PEVNT
174776  011500                            MOV     ($EF),R0          ;; ERROR BITS => R0<11:00>...
175000  001402                            BEQ     30007$            ;;...AND SKIP IF NONE.
175002  052700  100000                    BIS     #<10*BIT12>,R0    ;; ELSE, ADD TEST NUM => R0<15:12>...
175006  000207                  30007$:   RETURN                    ;;...AND RETURN (NZ).
```

```
                                        ;
                                        ; PORT INTERRUPT HANDLER(S).
                                        ;
                                                .ENABL  LSB
175010  012700  000032                  AI:     MOV     #PADB,R0        ; PORT A -- GET DB INDEX.
175014  105764  000100                          TSTB    PAMS(R4)        ; CHECK DIRECTION.
175020  100407                                  BMI     1$              ;       OUTPUT
175022  000421                                  BR      3$              ;       INPUT

175024  012700  000034                  BI:     MOV     #PBDB,R0        ; PORT B -- DITTO.
175030  105764  000120                          TSTB    PBMS(R4)
175034  100401                                  BMI     1$              ;       OUTPUT
175036  000413                                  BR      3$              ;       INPUT

175040  060400                          1$:     ADD     R4,R0           ; OUTPUT -- POINT TO DATA BUFFER.
175042  105712                                  TSTB    (R2)            ; ANY BYTES LEFT ??
175044  001404                                  BEQ     2$              ; BR IF NOT.
175046  112210                                  MOVB    (R2)+,(R0)      ; ELSE, SEND NEXT BYTE...
175050  105265  170002                          INCB    BYTES($TOP)     ;...BUMP THE COUNT...
175054  000410                                  BR      4$              ;...AND RETURN.
175056  112760  000240  177766          2$:     MOVB    #240,-12(R0)    ; WHEN XMIT DONE, CLEAR "IP"...
175064  000404                                  BR      4$              ;...AND RETURN.

175066  060400                          3$:     ADD     R4,R0           ; INPUT -- POINT TO DATA BUFFER.
175070  111023                                  MOVB    (R0),(R3)+      ; STORE THE BYTE...
175072  105265  170003                          INCB    BYTES+1($TOP)   ;...AND BUMP THE COUNT.
175076  112760  000140  177766          4$:     MOVB    #140,-12(R0)    ; COMMON EXIT -- CLEAR "IUS"...
175104  000002                                  RTI                     ;...AND RETURN.
                                                .DSABL  LSB
```

```
                                   .SBTTL  - T11   DMA CONTROLLER AMZ8016
                                   ;;********************************************************************
                                   ;;* TEST 11 -- DMA CONTROLLER AMZ8016
                                   ;;********************************************************************
 175106                            TST11:
 175106                            TDMA:
 175106  013746  000164                    MOV     DMAV,-(SP)        ;; SAVE DMAV.
 175112  013746  000166                    MOV     DMAV+2,-(SP)
 175116  005015                            CLR     ($EF)             ;; CLEAR ERROR FLAG.
                                   ;
                                   ; INPUT:  R1 = Q-BUS ADDRESS BITS<21:6> OR ZERO.
                                   ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                   ;
                                   ;       T11     DMA CONTROLLER AMZ8016
                                   ;        .        E0 = ROM IN VECTOR SPACE -- INTERRUPTS NOT TESTED
                                   ;        .        E1 = Q-BUS ADDRESS UNDEFINED -- Q-BUS ACCESS NOT TESTED
                                   ;        .        E2 = CHAN INTERRUPT NOT RECEIVED
                                   ;        .        E3 = DMA CHANNEL HUNG (TC!EOP BOTH CLEAR)
                                   ;        .        E4 = DMA ABORTED (EOP = 1 = NXM)
                                   ;        .        E5 = DMA DATA ERROR

         170002                    QAHI=   TMP1              ; SAVERS FOR A TWO WORD Q-BUS ADDRESS...
         170004                    QALO=   TMP2
         170006                    CLISP=  TMP3              ;...THREE VARIABLE (CH1/CH2) COMMANDS...
         170010                    SETIE=  TMP4
         170012                    STACHN= TMP5
                                   ;
 175120  004737  163602                    CALL    XRAMLO-K3K        ; VECTOR SPACE USABLE ??
 175124  001402                            BEQ     1$                ; SKIP IF SO.
 175126  052715  000001                    BIS     #BIT0,($EF)       ;; E0 = ROM IN VECTOR SPACE -- INTERRUPTS NOT TESTED

 175132  010102                    1$:     MOV     R1,R2             ; Q-ADDR<21:6> DEFINED ??.
 175134  001002                            BNE     2$                ; SKIP IF SO.
 175136  052715  000002                    BIS     #BIT1,($EF)       ;; E1 = Q-BUS ADDRESS UNDEFINED -- Q-BUS ACCESS NOT TESTED

 175142  012700  000006            2$:     MOV     #6,R0
 175146  005001                            CLR     R1
 175150  006302                    3$:     ASL     R2                ; \
 175152  006101                             ROL    R1                ;  > ADDR<21:16> => R1<6:0>
 175154  077003                              SOB   R0,3$             ; /
 175156  000301                            SWAB    R1                ; REPOSITION TO DMA FORMAT <13:8>...
 175160  052701  100000                    BIS     #BIT15,R1         ;...SET "Q-BUS" BIT...
 175164  010165  170002                    MOV     R1,QAHI($TOP)     ;...AND SAVE Q-BUS HIGH (SEG/TAG)...
 175170  010265  170004                    MOV     R2,QALO($TOP)     ;...AND LOW (OFFSET) ADDRESS.
                                   ;
                                   ; NOW PUSH THE 6-LINK CHAIN SCRIPT ONTO THE STACK
                                   ; FILLING IN THE CORRECTED ADDRESSES AS WE GO.
                                   ;
 175174  012700  162102            TDMA1:  MOV     #CHAINX+2-K3K,R0        ; CHAIN (TOP) ADDRESS...
 175200  012701  000052                    MOV     #<CHAINX+2-CHAIN1>/2,R1 ;...AND SIZE.
 175204  014046                    3$:     MOV     -(R0),-(SP)       ; PUSH A WORD.
 175206  021627  150002                    CMP     (SP),#BUFR1       ; LOCAL BUFR1 ??
 175212  001002                            BNE     4$                ; NO.
 175214  060516                            ADD     $TOP,(SP)         ; YES, ADJUST IT.
 175216  000426                            BR      8$
 175220  021627  160002            4$:     CMP     (SP),#BUFR2       ; LOCAL BUFR2 ??
 175224  001002                            BNE     5$                ; NO.
```

```
175226  060516                          ADD     $TOP,(SP)       ; YES, ADJUST IT.
175230  000421                          BR      8$
175232  021627  120000          5$:     CMP     (SP),#120000    ; Q-BUS 20K ??
175236  001003                          BNE     6$              ; NO.
175240  016516  170004                  MOV     QALO($TOP),(SP) ; YES, SUBSTITUTE IT.
175244  000407                          BR      7$
175246  021627  130000          6$:     CMP     (SP),#130000    ; Q-BUS 22K ??
175252  001010                          BNE     8$              ; NO.
175254  016516  170004                  MOV     QALO($TOP),(SP) ; YES, SUBSTITUTE IT...
175260  062716  010000                  ADD     #4096.,(SP)     ;...AND BUMP UP 2KW.
175264  014046                  7$:     MOV     -(R0),-(SP)     ; ON EITHER Q-BUS ADDRESS, PUSH...
175266  016516  170002                  MOV     QAHI($TOP),(SP) ;...AND SUBSTITUTE THE HIGH PART...
175272  005301                          DEC     R1              ;...AND TWEAK THE COUNT ACCORDINGLY.
175274  077135                  8$:     SOB     R1,3$           ; LOOP 'TIL DONE...
                                                                ;...(SP) IS THE CHIP'S CHAIN ADDRESS.
                                ;
                                ; NOW, EXECUTE AND ERROR CHECK 5 DMA CHAINS AS FOLLOWS:
                                ;       LOCAL => LOCAL, 2KW.
                                ;       LOCAL I/O PAGE => LOCAL, 1KW.
                                ;       LOCAL => Q-BUS 20K, 2KW.
                                ;       Q-BUS 20K => Q-BUS 22K, 2KW.
                                ;       Q-BUS 22K => LOCAL, 2KW.
                                ;
        000070                  MMR=    70              ; MASTER MODE REGISTER.
        000056                  STAT1=  56              ; CHANNEL 1 STATUS.
        000054                  STAT2=  54              ; CHANNEL 2 STATUS...
        000054                  CMDR=   54              ;...AND COMMAND REGISTER (FOR BOTH).
        000046                  CHA1H=  46              ; CHAIN ADDRESS REGISTERS HIGH (SEG/TAG)...
        000044                  CHA2H=  44
        000042                  CHA1L=  42              ;...AND THE LOW (OFFSET) HALFS.
        000040                  CHA2L=  40
        000001                  TC=     1               ; TERMINAL COUNT STATUS BIT (BIT0).
        000002                  EOP=    2               ; END-OF-PROCESS (NXM) STATUS BIT (BIT2).
        020000                  IP=     20000           ; INTERRUPT POSTED (DONE) STATUS BIT (BIT13).
                                ;
175276  106427  000200          TDMA2:  MTPS    #PR4            ; RAISE TO CHIP LEVEL.
175302  012700  174400                  MOV     #$DMA,R0        ; BASE ADDRESS.
175306  012701  000056                  MOV     #46.,R1
175312  005060  000054                  CLR     CMDR(R0)        ; COMMAND "RESET"...
175316  005020                  1$:     CLR     (R0)+           ;...AND CLEAR ALL REGISTERS.
175320  077102                          SOB     R1,1$
175322  012701  150002                  MOV     #BUFR1,R1
175326  060501                          ADD     $TOP,R1         ; GET BUFR1 ADDRESS.
175330  012702  004000                  MOV     #2048.,R2
175334  010100                  2$:     MOV     R1,R0
175336  005061  010000                  CLR     4096.(R1)       ; CLEAR BUFR2...
175342  010021                          MOV     R0,(R1)+        ;...AND PUT ADDRESSES IN BUFR1.
175344  077205                          SOB     R2,2$

175346  012704  174400          TDMA3:  MOV     #$DMA,R4        ; GET BASE POINTER.
175352  012737  161500  000164          MOV     #21$-K3K,DMAV   ; INT SERVICE CHAN 1 (AND 2).
175360  012737  000200  000166          MOV     #PR4,DMAV+2
175366  012765  000054  170006          MOV     #40!14,CLISP($TOP) ; SET "CLEAR IUS!IP"...
175374  012765  000062  170010          MOV     #42!20,SETIE($TOP) ;..."SET IE"...
175402  012765  000240  170012          MOV     #240,STACHN($TOP) ;...AND "START CHAIN" COMMANDS FOR CH 1.
175410  012764  000115  000070          MOV     #115,MMR(R4)    ; SET MODE = VI, WAIT, CPINTLV, ENABLE.
175416  010664  000042                  MOV     SP,CHA1L(R4)    ; SET CHAIN ADDRESS(ES)...
```

```
175422  010664  000040                  MOV     SP,CHA2L(R4)     ;... = 1ST LINK.
175426  012703  161742          1$:     MOV     #WCTBL-K3K,R3    ; SET WC TABLE POINTER IN R3.
175432  005000                  2$:     CLR     R0               ; SET A KEEP-ALIVE TIMER.
175434  032715  000001                  BIT     #BIT0,($EF)      ; INTERRUPTS USEABLE ??
175440  001001                          BNE     25$              ; SKIP NEXT IF NOT.
175442  106400                          MTPS    R0               ; YES, LOWER CPU.
175444  016564  170010  000054  25$:    MOV     SETIE($TOP),CMDR(R4) ; SET "CIE"...
175452  016564  170012  000054          MOV     STACHN($TOP),CMDR(R4) ;...AND START CHAIN CHAN 1(2).
175460  077001                          SOB     R0,.             ; TC (OR EOP) INTERRUPT SHOULD...
                                                                 ;...HAPPEN BEFORE THIS TIMES OUT.
175462  052715  000004                  BIS     #BIT2,($EF)      ;; E2 = CHAN INTERRUPT NOT RECEIVED
175466  024646                          CMP     -(SP),-(SP)      ; FAKE IT...
175470  026527  170012  000240          CMP     STACHN($TOP),#240 ;...AND SERVICE CHAN 1...
175476  001003                          BNE     22$              ;...OR CHAN 2.
                                ;
                                ; PROCESS DMA CHANNEL INTERRUPTS.
                                ;
175500  016400  000056          21$:    MOV     STAT1(R4),R0     ; GET CHAN 1 STATUS.
175504  000402                          BR      27$              ;       OR
175506  016400  000054          22$:    MOV     STAT2(R4),R0     ; GET CHAN 2 STATUS.
175512  022626                  27$:    CMP     (SP)+,(SP)+      ; ADJUST STACK.
175514  032700  000001                  BIT     #TC,R0
175520  001010                          BNE     24$              ; BR IF TC IS SET.
175522  032700  000002                  BIT     #EOP,R0
175526  001003                          BNE     23$              ; BR IF EOP (NXM) IS SET.
175530  052715  000010                  BIS     #BIT3,($EF)      ;; E3 = DMA CHANNEL HUNG (TC!EOP BOTH CLEAR)
175534  000402                          BR      24$
175536                          23$:
175536  052715  000020                  BIS     #BIT4,($EF)      ;; E4 = DMA ABORTED (EOP = 1 = NXM)
175542  016564  170006  000054  24$:    MOV     CLISP($TOP),CMDR(R4) ; INT SERVICE COMPLETE, CLEAR "IUS!IP".

175550  012300                  3$:     MOV     (R3)+,R0         ; GET WORD COUNT FROM "WCTBL"...
175552  001423                          BEQ     6$               ;...AND SKIP IF DATA CHECK INHIBITED.
175554  012701  150002                  MOV     #BUFR1,R1
175560  060501                          ADD     $TOP,R1          ; ASSUME SRC WAS BUFR1...
175562  012702  160002                  MOV     #BUFR2,R2
175566  060502                          ADD     $TOP,R2          ;...AND DEST BUFR2.
175570  020027  004000                  CMP     R0,#2048.        ; WC = 2KW ??
175574  001404                          BEQ     4$               ; SKIP IF SO...
175576  012701  164000                  MOV     #164000,R1       ;...ELSE, SRC WAS ODTROM...
175602  162702  010000                  SUB     #4096.,R2        ;...AND DST BUFR1.
175606  022122                  4$:     CMP     (R1)+,(R2)+      ; COMPARE SRC/DST DATA.
175610  001403                          BEQ     5$               ; BR IF OK.
175612  052715  000040                  BIS     #BIT5,($EF)      ;; E5 = DMA DATA ERROR
175616  000401                          BR      6$               ; ESCAPE ON 1ST DATA ERROR.
175620  077006                  5$:     SOB     R0,4$

175622  005713                  6$:     TST     (R3)             ; NEXT LINK INVOLVE Q-BUS ??
175624  001003                          BNE     7$               ; BR IF NOT.
175626  032715  000002                  BIT     #BIT1,($EF)      ; YES, Q-BUS ACCESS PERMITTED ??
175632  001003                          BNE     8$               ; NO, QUIT WHILE YOU'RE AHEAD.
175634  021327  177777          7$:     CMP     (R3),#-1         ; CHECK FOR END OF TABLE...
175640  001274                          BNE     2$               ;...AND LOOP 'TIL ALL LINKS DONE.

175642  032765  000001  170012  8$:     BIT     #1,STACHN($TOP)  ; NOW, CHECK THE CHANNEL SELECT BIT...
175650  001012                          BNE     9$               ;...AND EXIT IF BOTH DONE.
175652  062737  000006  000164          ADD     #22$-21$,DMAV    ; ELSE, CHANGE VECTOR...
```

```
 175660  005265  170006          INC     CLISP($TOP)
 175664  005265  170010          INC     SETIE($TOP)      ;...AND COMMANDS FOR CHAN 2...
 175670  005265  170012          INC     STACHN($TOP)
 175674  000654                  BR      1$               ;...AND GO 'ROUND ONCE MORE.

 175676  005064  000054  9$:     CLR     CMDR(R4)         ; ALL DONE RESET CHIP.
 175702  062706  000124          ADD     #CHAINX+2-CHAIN1,SP ; POP CHAIN OFF THE STACK.
 175706  032715  000001          BIT     #BIT0,($EF)      ; DID WE RUN WITH INTERRUPTS ??
 175712  001402                  BEQ     10$              ; SKIP IF SO.
 175714  042715  000004          BIC     #BIT2,($EF)      ; NO, DISCARD INTERRUPT ERROR BIT.
 175720                  10$:DMAXIT:
 175720  012637  000166          MOV     (SP)+,DMAV+2     ;; RESTORE DMAV.
 175724  012637  000164          MOV     (SP)+,DMAV
 175730  011500                  MOV     ($EF),R0         ;; ERROR BITS => R0<11:00>...
 175732  001402                  BEQ     30008$           ;;...AND SKIP IF NONE.
 175734  052700  110000          BIS     #<11*BIT12>,R0   ;; ELSE, ADD TEST NUM => R0<15:12>...
 175740  000207          30008$: RETURN                   ;;...AND RETURN (NZ).
                         ;
                         ; WORD COUNT TABLE FOR POST-DMA DATA CHECKS.
                         ; FINAL CHECK (5.) VERIFIES THE RESULTS OF CHAINS 2, 3, 4, AND 5.
                         ;
 175742  004000          WCTBL:  2048.                    ; 1. 2KW, BUFR1 => BUFR2.
 175744  002000                  1024.                    ; 2. 1KW, ODTROM => BUFR1.
 175746  000000                  0                        ; 3. Q-BUS INVOLVED, NO DATA CHECK.
 175750  000000                  0                        ; 4. Q-BUS INVOLVED, NO DATA CHECK.
 175752  004000                  2048.                    ; 5. 2KW, BUFR1 => Q1 => Q2 => BUFR2.
 175754  177777                  -1                       ; TABLE TERMINATOR.
                         ;
                         ; AND THIS IS THE 6 LINK DMA CHAIN.
                         ;
                         ; THIS GETS PUSHED ONTO THE STACK WHERE LOCAL AND Q-BUS ADDRESSES
                         ; ARE ADJUSTED ACCORDING TO RAM CONFIG AND ENTRY OPTION (R2).
                         ;
 175756  001606          CHAIN1: 1606            ; LOAD ARA, ARB, OPK, VECT, AND CHAN MODE.
 175760  000000  150002          0, BUFR1        ; FROM LOCAL BUFR1...
 175764  000000  160002          0, BUFR2        ;...TO LOCAL BUFR2...
 175770  004000                  2048.           ;...2K WORDS.
 175772  000164                  DMAV            ; VECTOR.
 175774  000030  001340          30, 1340        ; SFTREQ, HDMSK, INT TC!EOP, INTLV, WORD-WORD.

 176000  001602          CHAIN2: 1602            ; RE-LOAD ARA, ARB, OPK, AND CHAN MODE ONLY.
 176002  000000  164000          0, 164000       ; FROM LOCAL ODTROM...
 176006  000000  150002          0, BUFR1        ;...TO LOCAL BUFR1...
 176012  002000                  1024.           ;...1K WORDS...
 176014  000030  001340          30, 1340        ;...SAME MODE AND TERMINATION.

                         ;Previously, CHAIN3 read the arbiter's boot ROM using the QBUS. However, the
                         ;Boot ROM of a KDF11-B does not respond on the QBUS. Therefore, CHAIN3 was
                         ;removed.

 176020  001602          CHAIN4: 1602            ;       DITTO.
 176022  000000  150002          0, BUFR1        ; FROM LOCAL BUFR1...
 176026  100000  120000          100000, 120000  ;...TO Q-BUS 20K...
 176032  004000                  2048.           ;...2K WORDS.
 176034  000030  001340          30, 1340        ;       DITTO.

 176040  001602          CHAIN5: 1602            ;       DITTO.
```

```
176042  100000  120000                    100000, 120000  ; FROM Q-BUS 20K...
176046  100000  130000                    100000, 130000  ;...TO Q-BUS 22K...
176052  004000                            2048.           ;...2K WORDS.
176054  000030  001340                    30, 1340        ;       DITTO.

176060  001602                    CHAIN6: 1602            ;       DITTO.
176062  100000  130000                    100000, 130000  ; FROM Q-BUS 22K...
176066  000000  160002                    0, BUFR2        ;...TO LOCAL BUFR2...
176072  004000                            2048.           ;...2K WORDS.
176074  000030  001340                    30, 1340        ;       DITTO.
176100  000000                    CHAINX: 0               ; TERMINATOR.
```

```
                                      .SBTTL  - T12   Q-BUS INTERRUPT
                                      ;;*****************************************************************
                                      ;;* TEST 12 -- Q-BUS INTERRUPT
                                      ;;*****************************************************************
   176102                             TST12:
   176102                             TQIR:
   176102  013746  000130                     MOV     BIACKV,-(SP)      ;; SAVE BIACKV.
   176106  013746  000132                     MOV     BIACKV+2,-(SP)
   176112  013746  000024                     MOV     PWRV,-(SP)        ;; SAVE PWRV.
   176116  013746  000026                     MOV     PWRV+2,-(SP)
   176122  005015                             CLR     ($EF)             ;; CLEAR ERROR FLAG.
                                      ;
                                      ; INPUT:  R1 = 1ST OF 2 CONSECUTIVE Q-BUS VECTORS.
                                      ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                      ;
                                      ; *** REQUIRES A RUNNING ARBITER CPU ***
                                      ;
                                      ; VERIFY THAT THE QIR CAN INTERRUPT THE ARBITER, AND THAT
                                      ; THE POST-INTERRUPT PROCESS WORKS AS ADVERTISED.
                                      ;
                                      ;       T12     Q-BUS INTERRUPT
                                      ;       .         E0 = Q-BUS VECTOR UNDEFINED -- CAN'T RUN
                                      ;       .         E1 = BREQ (CSR<14>) NEVER GOT SET
                                      ;       .         E2 = ROM IN VECTOR SPACE -- RESPONSE INTERRUPTS NOT TESTED
                                      ;       .         E3 = INT-ON-BIACK NOT MASKED AT PR5
                                      ;       .         E4 = INT-ON-BIACK NOT RECEIVED
                                      ;       .         E5 = BIACK DIDN'T CLEAR BREQ (CSR<14>)
                                      ;       .         E6 = BRESET TRAP THRU 24 DIDN'T HAPPEN

   176124  005701                             TST     R1               ; Q VECTOR DEFINED ??
   176126  001003                             BNE     1$               ; PROCEED IF SO.
   176130  052715  000001                     BIS     #BIT0,($EF)      ;; E0 = Q-BUS VECTOR UNDEFINED -- CAN'T RUN
   176134  000516                             BR      QIXIT

   176136  106427  000240             1$:     MTPS    #PR5             ; RAISE CPU.
   176142  042737  177677  177530             BIC     #^CBIT6,$CSR     ; ENSURE EVERYTHING OFF (EXCEPT DPR).
   176150  010137  177532                     MOV     R1,$QIR          ; WRITE A VECTOR IN QIR<9:2>.
   176154  032737  040000  177530             BIT     #BIT14,$CSR      ; REQUEST SHOULD BE SET.
   176162  001002                             BNE     2$               ; SKIP IF SO.
   176164  052715  000002                     BIS     #BIT1,($EF)      ;; E1 = BREQ (CSR<14>) NEVER GOT SET

   176170  004737  163602             2$:     CALL    XRAMLO-K3K       ; VECTOR SPACE USABLE ??
   176174  001403                             BEQ     3$               ; PROCEED IF SO.
   176176  052715  000004                     BIS     #BIT2,($EF)      ;; E2 = ROM IN VECTOR SPACE -- RESPONSE INTERRUPTS NOT TESTE
   176202  000473                             BR      QIXIT

   176204  012737  162232  000130     3$:     MOV     #4$-K3K,BIACKV   ; SET RESPONSE VECTOR.
   176212  012737  000240  000132             MOV     #PR5,BIACKV+2
   176220  052737  030000  177530             BIS     #BIT13!BIT12,$CSR ; SET ENABLES.
   176226  077001                             SOB     R0,.             ; RESPONSE SHOULD BE MASKED AT THIS LEVEL.
   176230  000403                             BR      5$
   176232                             4$:
   176232  052715  000010                     BIS     #BIT3,($EF)      ;; E3 = INT-ON-BIACK NOT MASKED AT PR5
   176236  000411                             BR      6$
   176240  062737  000030  000130     5$:     ADD     #6$-4$,BIACKV    ; CHANGE RESPONSE VECTOR.
   176246  106427  000000                     MTPS    #PR0             ; LOWER CPU...
   176252  077001                             SOB     R0,.             ;...INT SHOULD COME IN.
```

```
176254  052715  000020                  BIS     #BIT4,($EF)        ;; E4 = INT-ON-BIACK NOT RECEIVED
176260  000401                          BR      65$
176262  022626                  6$:     CMP     (SP)+,(SP)+        ; OK, FIX STACK.
176264  032737  040000  177530  65$:    BIT     #BIT14,$CSR        ; REQUEST SHOULD BE CLEAR.
176272  001405                          BEQ     7$
176274  052715  000040                  BIS     #BIT5,($EF)        ;; E5 = BIACK DIDN'T CLEAR BREQ (CSR<14>)
176300  077001                          SOB     R0,.
176302  000005                           RESET                     ;; LOCAL RESET.
176304  077001                            SOB   R0,.
                                ;
                                ; NOW ONCE MORE USING THE ALTERNATE VECTOR (IN ARBITER)
                                ; TO CAUSE A "BUS-INIT" AND LOCAL TRAP THRU VECTOR 24.
                                ;
176306  022121                  7$:     CMP     (R1)+,(R1)+        ; BUMP UP (+4) TO SECOND VECTOR.
176310  012737  162370  000024          MOV     #8$-K3K,PWRV       ; SET PWR-UP VECTOR.
176316  012737  000340  000026          MOV     #PR7,PWRV+2
176324  106427  000340                  MTPS    #PR7               ; SHOULD BE NON-MASKABLE.
176330  042737  020000  177530          BIC     #BIT13,$CSR        ; CLEAR IACK ENABLE.
176336  052737  002000  177530          BIS     #BIT10,$CSR        ; ARM BRESET TRAP.
176344  010137  177532                  MOV     R1,$QIR            ; POST Q-BUS REQUEST.
176350  005000                          CLR     R0                 ; ARBITER EXECUTES A BUS-RESET...
176352  077001                          SOB     R0,.               ;...AND A TRAP THRU 24 SHOULD OCCUR.
176354  052715  000100                  BIS     #BIT6,($EF)        ;; F6 = BRESET TRAP THRU 24 DIDN'T HAPPEN
176360  077001                          SOB     R0,.
176362  000005                           RESET                     ;; LOCAL RESET.
176364  077001                            SOB   R0,.
176366  000401                          BR      QIXIT
176370  022626                  8$:     CMP     (SP)+,(SP)+        ; PWR-UP TRAPPED TO HERE, FIX STACK.
176372  042737  177677  177530  QIXIT:  BIC     #^CBIT6,$CSR       ; CLEAR CSR.
176400  012637  000026                  MOV     (SP)+,PWRV+2       ;; RESTORE PWRV.
176404  012637  000024                  MOV     (SP)+,PWRV
176410  012637  000132                  MOV     (SP)+,BIACKV+2     ;; RESTORE BIACKV.
176414  012637  000130                  MOV     (SP)+,BIACKV
176420  011500                          MOV     ($EF),R0           ;; ERROR BITS => R0<11:00>...
176422  001402                          BEQ     30009$             ;;...AND SKIP IF NONE.
176424  052700  120000                  BIS     #<12*BIT12>,R0     ;; ELSE, ADD TEST NUM => R0<15:12>...
176430  000207                  30009$: RETURN                     ;;...AND RETURN (NZ).
```

```
                                .SBTTL  - T13   DUAL-PORT RAM AND IPV
                                ;;**********************************************************************
                                ;;* TEST 13 -- DUAL-PORT RAM AND IPV
                                ;;**********************************************************************
176432                          TST13:
176432                          TDPR:
176432  013746  000120                  MOV     DPRV4,-(SP)     ;; SAVE DPRV4.
176436  013746  000122                  MOV     DPRV4+2,-(SP)
176442  013746  000124                  MOV     DPRV8,-(SP)     ;; SAVE DPRV8.
176446  013746  000126                  MOV     DPRV8+2,-(SP)
176452  013746  000134                  MOV     DPRV12,-(SP)    ;; SAVE DPRV12.
176456  013746  000136                  MOV     DPRV12+2,-(SP)
176462  005015                          CLR     ($EF)           ;; CLEAR ERROR FLAG.
                                ;
                                ; INPUT:  NONE.
                                ; OUTPUT: R0 = ERROR FLAGS OR ZERO.
                                ;
                                ; FIRST, DISABLE THE DPR AND VERIFY WRITE/READ
                                ;       FROM THE LOCAL (II) SIDE.
                                ;
                                ;       T13     DUAL-PORT RAM AND IPV
                                ;       .         E0 = LOCAL DPR WRITE-READ ERROR
                                ;       .         E1 = BUS-ERROR ON Q-BUS/DPR READ
                                ;       .         E2 = NON-ZERO DATA RETURNED ON Q-BUS READ
                                ;       .         E3 = ROM IN VECTOR SPACE -- Q-WRITE AND INTERRUPT NOT TESTED
                                ;       .         E4 = BUS ERROR ON WRITE TO DPR WORD 0
                                ;       .         E5 = COMMAND INTERRUPT DIDN'T HAPPEN (OR DOESN'T WORK)
                                ;       .         E6 = Q-BUS WRITE AND/OR INTERRUPT WITH DPR DISABLED
                                ;       .         E7 = Q-BUS WRITE DIDN'T TIME-OUT
                                ;       .         E8 = Q-BUS WRITE TIMED-OUT
                                ;       .         E9 = DPR INTERRUPT 4 NOT RECEIVED
                                ;       .         E10 = DPR INTERRUPT 8 NOT RECEIVED
                                ;       .         E11 = DPR INTERRUPT 12 NOT RECEIVED

176464  005037  177530                  CLR     $CSR            ; ENSURE EVERYTHING OFF.
176470  012702  175000                  MOV     #$DPR,R2        ; DPR BASE ADDRESS (LOCAL).
176474  012701  125252                  MOV     #125252,R1      ; USE ALTERNATING 1'S AND 0'S.
176500  012700  000020                  MOV     #16.,R0
176504  010112                  1$:     MOV     R1,(R2)         ; WRITE...
176506  022201                          CMP     (R2)+,R1        ;...AND CHECK A WORD.
176510  001402                          BEQ     2$              ; SKIP IF RIGHT.
176512  052715  000001                  BIS     #BIT0,($EF)     ;; E0 = LOCAL DPR WRITE-READ ERROR
176516  077006                  2$:     SOB     R0,1$           ; LOOP 'TIL DONE.
                                ;
                                ; NOW THE DPR IS FULL OF ALTERNATING 1'S AND 0'S.
                                ; READ ALL WORDS FROM THE Q-BUS SIDE USING THE DMA ENGINE.
                                ; EXPECT ALL ADDRESSES TO RESPOND (NO EOP), BUT DATA = 0'S.
                                ;
        170004                  QADR=   TMP2                    ; SAVE Q-BUS...
        170006                  LADR=   TMP3                    ;...AND LOCAL ADDRESSES.
        000000                  RD=     0                       ; DMA DRIVER READ...
        000010                  WRT=    WQIO-RQIO               ;...WRITE...
        000062                  CNTNU=  CQIO-RQIO               ;...AND CONTINUE (AFTER EOP) ENTRIES.
                                ;
176520  013700  177524          TDPR2:  MOV     $UID,R0         ; GET CPU ID NUM (CSRC<7:4>).
176524  042700  177617                  BIC     #^C160,R0       ; STRIP <6:4> = NUM*16...
176530  006300                          ASL     R0              ;...TIMES 2 = NUM*32...
```

```
176532  062700  160000                  ADD     #$QDPR1,R0          ;...PLUS LO BASE = GLOBAL ADDRESS.
176536  032737  000200  177524          BIT     #BIT7,$UID          ; ID >= 8. ??
176544  001402                          BEQ     1$                  ; SKIP IF NOT.
176546  062700  015400                  ADD     #<$QDPR2-$QDPR1>,R0 ; YES, ADJUST TO HIGH RANGE.
176552  010065  170004          1$:     MOV     R0,QADR($TOP)       ; SAVE IT IN RAM.
176556  010500                          MOV     $TOP,R0
176560  062700  150002                  ADD     #BUFR1,R0           ; CALCULATE BUFFER ADDRESS...
176564  010065  170006                  MOV     R0,LADR($TOP)       ;...AND SAVE IT AS WELL.
176570  012700  163602                  MOV     #QIOC+2-K3K,R0      ; GET DPIVER ADDRESS...
176574  012701  000101                  MOV     #<QIOC+2-RQIO>/2,R1 ;...AND SIZE...
176600  014046                  2$:     MOV     -(R0),-(SP)         ;...AND PUSH IT ONTO THE STACK.
176602  077102                          SOB     R1,2$
176604  000403                          BR      35$                 ; SKIP NEXT.
176606  052765  002000  170004  3$:     BIS     #2000,QADR($TOP)    ; ALTERNATE BASE ADDRESS.
176614  016500  170004          35$:    MOV     QADR($TOP),R0       ; SET Q-BUS ADDRESS...
176620  016501  170006                  MOV     LADR($TOP),R1       ;...LOCAL ADDRESS...
176624  012702  000020                  MOV     #16.,R2             ;...AND WORD COUNT.
176630  004766  000000                  CALL    RD(SP)              ; READ FROM Q-BUS SIDE VIA DMA.
176634  102007                          BVC     4$                  ; BR IF OK (NOT "EOP").
176636  032765  002000  170004          BIT     #2000,QADR($TOP)    ; NOT THERE, BOTH BASES TRIED ??
176644  001760                          BEQ     3$                  ; NO, GO 'ROUND ONCE.
176646  052715  000002                  BIS     #BIT1,($EF)         ;; E1 = BUS-ERROR ON Q-BUS DPR READ
176652  000411                          BR      TIPV1
176654  016500  170006          4$:     MOV     LADR($TOP),R0       ; GET BUFFEP ADDRESS.
176660  012701  000020                  MOV     #16.,R1
176664  005720                  5$:     TST     (R0)+               ; ORSERVED DATA SHOULD BE ZERO.
176666  001402                          BEQ     6$                  ; BR IF SO.
176670  052715  000004                  BIS     #BIT2,($EF)         ;; E2 = NON-ZERO DATA RETURNED ON Q-BUS READ
176674  077105                  6$:     SOB     R1,5$
                                ;
                                ; SO FAR, SO GOOD -- NOW WRITE ALL FROM THE Q-BUS SIDE.
                                ; 1ST WRITE SHOULD FORCE A NON-MASKABLE RESTART TRAP (COMMAND).
                                ; ALL OTHERS SHOULD APPEAR NON-EXISTANT (EOP)..
                                ;
176676  004737  163602          TIPV1:  CALL    XRAMLO-K3K          ; VECTOR SPACE USABLE ??
176702  001403                          BEQ     1$                  ; PROCEED IF SO.
176704  052715  000010                  BIS     #BIT3,($EF)         ;; E3 = ROM IN VECTOR SPACE -- Q-WRITE AND INTERRUPT NOT TES
176710  000563                          BR      TIPXIT

176712  012700  175000          1$:     MOV     #$DPR,R0
176716  016501  170006                  MOV     LADR($TOP),R1
176722  012702  000020                  MOV     #16.,R2
176726  005020                  11$:    CLR     (R0)+               ; CLEAR DPR...
176730  005021                          CLR     (R1)+               ;...AND BUFFER.
176732  077203                          SOB     R2,11$

176734  012701  000120                  MOV     #DPRV4,R1
176740  012721  163350                  MOV     #IPV4-K3K,(R1)+ ; SET DPRV4...
176744  012721  000240                  MOV     #PR5,(R1)+
176750  012721  163356                  MOV     #IPV8-K3K,(R1)+ ;...DPRV8...
176754  012721  000240                  MOV     #PR5,(R1)+
176760  022121                          CMP     (R1)+,(R1)+
176762  012721  163364                  MOV     #IPV12-K3K,(R1)+ ;...AND DPRV12 VECTORS.
176766  012721  000240                  MOV     #PR5,(R1)+
176772  106427  000340                  MTPS    #PR7                ; RAISE CPU, COMMAND INTERRUPT...
                                                                    ;...SHOULD BE NON-MASKABLE.
176776  016500  170004                  MOV     QADR($TOP),R0       ; GET GLOBAL ADDRESS...
```

```
177002  016501  170006                   MOV     LADR($TOP),R1     ;...LOCAL ADDRESS...
177006  012702  000001                   MOV     #1,R2             ;...AND WORD COUNT.
177012  012711  000040                   MOV     #40,(R1)          ; SFT 1ST WORD = "NOP" COMMAND.
177016  004766  000010                   CALL    WRT(SP)           ; WRITE DPR WORD 0...
177022  102002                           BVC     2$                ;...SHOULD NOT HAVE ABORTED.
177024  052715  000020                   BIS     #BIT4,($EF)       ;; E4 = BUS ERROR ON WRITE TO DPR WORD 0
177030  005737  175000          2$:      TST     $DPR              ; COMMAND SHOULD HAVE BEEN CLEARED.
177034  001402                           BEQ     3$
177036  052715  000040                   BIS     #BIT5,($EF)       ;; E5 = COMMAND INTERRUPT DIDN'T HAPPEN (OR DOESN'T WORK)
177042                          3$:
177042  005004                  4$:      CLR     R4                ; CLEAR INT RECEIVED FLAG...
177044  106427  000000                   MTPS    #PR0              ;...AND LOWER CPU.
177050  012703  000017                   MOV     #15.,R3           ; SET LOOP FOR 15 MORE WRITES.
177054  012702  000001          5$:      MOV     #1,R2             ; SET WORD COUNT.
177060  004766  000010                   CALL    WRT(SP)           ; WRITE NEXT FROM Q-BUS SIDE...
177064  102002                           BVC     6$                ;...SHOULD GET "EOP"...
177066  005704                           TST     R4                ;...     AND      ...
177070  001402                           BEQ     7$                ;...NO INTERRUPT.
177072                          6$:
177072  052715  000100                   BIS     #BIT6,($EF)       ;; E6 = Q-BUS WRITE AND/OR INTERRUPT WITH DPR DISABLED
177076  077312                  7$:      SOB     R3,5$
                                ;
                                ; NOW, ENABLE THE DPR, AND WRITE WORDS 1-15 AGAIN.
                                ; EXPECT LEVEL 5 INTERRUPTS AT WORDS 4, 8, AND 12,
                                ; AND EOP AT WORDS 1, 5, AND 9.
                                ;
177100  052737  000170  177530  TIPV2:   BIS     #170,$CSR         ; SET ALL ENABLES.
177106  016500  170004                   MOV     QADR($TOP),R0     ; SET Q ADDRESS...
177112  062700  000002                   ADD     #2,R0             ;...FOR DPR1...
177116  016501  170006                   MOV     LADR($TOP),R1     ;...AND LOCAL ADDRESS.
177122  062701  000002                   ADD     #2,R1             ;...DITTO.
177126  012703  000002                   MOV     #BIT1,R3          ; BIT = CURRENT WORD UNDER TEST.
177132  005004                  1$:      CLR     R4                ; CLEAR "INT RECVD" FLAG.
177134  012702  000001                   MOV     #1,R2             ; SET WC...
177140  004766  000010                   CALL    WRT(SP)           ;...AND WRITE 1ST WORD.
177144  102406                           BVS     3$                ; BR IF IT TIMED-OUT.

177146  030327  001042          2$:      BIT     R3,#BIT1!BIT5!BIT9 ; DIDN'T TIME-OUT, SHOULD IT HAVE ??
177152  001410                           BEQ     4$                ; NO, PROCEED.
177154  052715  000200                   BIS     #BIT7,($EF)       ;; E7 = Q-BUS WRITE DIDN'T TIME-OUT
177160  000405                           BR      4$
177162  030327  001042          3$:      BIT     R3,#BIT1!BIT5!BIT9 ; DID TIME-OUT, SHOULD IT HAVE ??
177166  001002                           BNE     4$                ; YES, PROCEED.
177170  052715  000400                   BIS     #BIT8,($EF)       ;; E8 = Q-BUS WRITE TIMED-OUT

177174  030327  000020          4$:      BIT     R3,#BIT4          ; INTFRRUPT 4 EXPECTED ??
177200  001405                           BEQ     5$                ; NO
177202  020427  000004                   CMP     R4,#4.            ; YES, RECEIVED ??
177206  001402                           BEQ     5$                ; YES.
177210  052715  001000                   BIS     #BIT9,($EF)       ;; E9 = DPR INTERRUPT 4 NOT RECEIVED
177214  030327  000400          5$:      BIT     R3,#BIT8          ; INTERRUPT 8 EXPECTED ??
177220  001405                           BEQ     6$                ; NO.
177222  020427  000010                   CMP     R4,#8.            ; YFS, RECEIVED ??
177226  001402                           BEQ     6$                ; YES.
177230  052715  002000                   BIS     #BIT10,($EF)      ;; E10 = DPR INTERRUPT 8 NOT RECEIVED
177234  030327  010000          6$:      BIT     R3,#BIT12         ; INTERRUPT 12 EXPECTED ??
177240  001405                           BEQ     7$                ; NO
```

```
177242  020427  000014                  CMP     R4,#12.           ; YES, RECEIVED ??
177246  001402                          BEQ     7$                ; YES.
177250  052715  004000                  BIS     #BIT11,($EF)      ;; E11 = DPR INTERRUPT 12 NOT RECEIVED
177254  006303                  7$:     ASL     R3                ; SHIFT UP TO NEXT WORD (BIT)...
177256  001325                          BNE     1$                ;...AND LOOP 'TIL ALL DONE.

177260  062706  000202          TIPXIT: ADD     #<QIOC+2-RQIO>,SP ; DONE, POP DRIVER OFF THE STACK.
177264  042737  177677  177530          BIC     #^CBIT6,$CSR      ; CLEAR ALL EXCEPT "DPREN".
177272  012700  175000                  MOV     #$DPR,R0          ; GET LOCAL ADDRESS...
177276  012701  000020                  MOV     #16.,R1
177302  005020                  1$:     CLR     (R0)+             ;...AND FLUSH ANY CRAP.
177304  077102                          SOB     R1,1$
177306  012637  000136                  MOV     (SP)+,DPRV12+2    ;; RESTORE DPRV12.
177312  012637  000134                  MOV     (SP)+,DPRV12
177316  012637  000126                  MOV     (SP)+,DPRV8+2     ;; RESTORE DPRV8.
177322  012637  000124                  MOV     (SP)+,DPRV8
177326  012637  000122                  MOV     (SP)+,DPRV4+2     ;; RESTORE DPRV4.
177332  012637  000120                  MOV     (SP)+,DPRV4
177336  011500                          MOV     ($EF),R0          ;; ERROR BITS => R0<11:00>...
177340  001402                          BEQ     30010$            ;;...AND SKIP IF NONE.
177342  052700  130000                  BIS     #<13*BIT12>,R0    ;; ELSE, ADD TEST NUM => R0<15:12>...
177346  000207                  30010$: RETURN                    ;;...AND RETURN (NZ).

177350  012704  000004          IPV4:   MOV     #4.,R4            ; DPRV4 INTERRUPT RECEIVED.
177354  000405                          BR      IPVXIT
177356  012704  000010          IPV8:   MOV     #8.,R4            ; DITTO DPRV8.
177362  000402                          BR      IPVXIT
177364  012704  000014          IPV12:  MOV     #12.,R4           ; DITTO DPRV12.
177370  042737  000007  177530  IPVXIT: BIC     #7,$CSR           ; *** TEMP, ENSURE REQ'S ARE CLEAR. ***
177376  000002                          RTI
```

```
                                ;************************************************************************
                                ; SUBROUTINE TO ACCESS THE Q-BUS I/O PAGE USING THE DMA ENGINE.
                                ; THIS CODE IS PUSHED ONTO AND EXECUTED IN STACK SPACE.
                                ;
                                .DSABL  AMA     ; RELATIVE PIC.
                                ;
                                ; ON CALL, R0 = QIO ADDRESS, R1 = LOCAL ADDRESS, R2 = WORD COUNT.
                                ; ON RETURN, THEY CONTAIN UPDATED (TERMINAL) VALUES.
                                ; IF EOP TERMINATION, "V" BIT IS SET.
                                ;
        174400                  DM=     $DMA                ; DMA CHIP BASE.
                                ;
177400  042767  000020  000170  RQIO:   BIC     #BIT4,CM+2      ; READ -- CLEAR "FLIP" BIT.
177406  000403                          BR      WQIOA
177410  052767  000020  000160  WQIO:   BIS     #BIT4,CM+2      ; WRITE -- SET "FLIP" BIT.
177416  010067  000142          WQIOA:  MOV     R0,QA+2         ; SET Q-BUS I/O ADDRESS.
177422  010167  000142                  MOV     R1,LA+2         ; SET LOCAL ADDRESS.
177426  010267  000140                  MOV     R2,WC           ; SET WORD COUNT.
177432  012737  000115  174470          MOV     #115,@#DM+MMR   ; MODE = VI, WAIT, CPINTLV, ENABLE.
177440  012737  000074  174454          MOV     #40134,@#DM+CMDR ; CLEAR INTERRUPT ENABLES.
177446  005037  174446                  CLR     @#DM+CHAIH
177452  012700  000100                  MOV     #QIO-.-6,R0     ; GET (PIC) CHAIN START ADDRESS.
177456  060700                          ADD     PC,R0
177460  000403                          BR      CQIOA
177462  012700  000110          CQIO:   MOV     #QIOC-.-6,R0    ; GET (PIC) CHAIN CONTINUE (DUMMY).
177466  060700                          ADD     PC,R0
177470  010037  174442          CQIOA:  MOV     R0,@#DM+CHAIL   ; SET CHAIN ADDRESS.
177474  012737  000240  174454          MOV     #240,@#DM+CMDR  ; CHAIN-LOAD (OR RELOAD 0) REGISTERS.
177502  012737  000102  174454          MOV     #102,@#DM+CMDR  ; SET SOFT-REQ (START/CONTINUE DMA)...
177510  000240                          NOP
177512  032737  020000  174456  1$:     BIT     #IP,@#DM+STAT1  ;...AND WAIT FOR DONE/ABORT (IP=1).
177520  001774                          BEQ     1$
177522  012737  000054  174454          MOV     #40114,@#DM+CMDR ; OK, CLEAR IUS AND IP.
177530  013700  174412                  MOV     @#DM+12,R0      ; GET FINAL Q ADDR (ARAL)...
177534  013701  174402                  MOV     @#DM+2,R1       ;...LOCAL ADDR (ARBL)...
177540  013702  174462                  MOV     @#DM+62,R2      ;...AND WORD COUNT (OPK).
177544  032737  000002  174456          BIT     #EOP,@#DM+STAT1 ; "EOP" TERMINATION ??
177552  001401                          BEQ     2$              ; SKIP IF NOT...
177554  000262                          SEV                     ;...OTHERWISE, SET "V" IN PSW.
177556  000207                  2$:     RETURN

177560  001602                  QIO:    1602            ; LOAD CARA, CARB, COPK, AND CH MODE.
177562  177400  000000          QA:     177400, 0       ; Q-BUS I/O ADDRESS (CARA)
177566  000000  000000          LA:     0, 0            ; LOCAL ADDRESS (CARB)
177572  000000                  WC:     0               ; WORD COUNT (COPK)
177574  000010  001340          CM:     10, 1340        ; CHAN MODE (1360 IF "FLIP" = 1).
177600  000000                  QIOC:   0               ; DUMMY RELOAD TO CLEAR "NAC" AND CONTINUE.
                                        .ENABL  AMA
                                ;************************************************************************
```

```
                                  ;
                                  ; HI SEGMENT COPY OF ROM/RAM IN VECTOR SPACE ROUTINE.
                                  ;
                                  ; RETURN R2<00> = 1 IF LO BYTE NON-WRITABLE.
                                  ;        R2<01> = 1 IF HI BYTE NON-WRITABLE.
                                  ;
177602  005000                    XRAMLO: CLR     R0
177604  005002                            CLR     R2
177606  011046                            MOV     (R0),-(SP)      ; GET TARGET LOCATION...
177610  005116                            COM     (SP)            ;...AND COMPLIMENT IT.
177612  005110                            COM     (R0)            ; COMPLIMENT TARGET LOCATION.
177614  126066  000001  000001            CMPB    1(R0),1(SP)     ; HI BYTE WRITABLE ??
177622  001402                            BEQ     1$              ;       YES
177624  052702  000002                    BIS     #BIT1,R2        ;       NO
177630  121026                    1$:     CMPB    (R0),(SP)+      ; LO BYTE WRITABLE ??
177632  001402                            BEQ     2$              ;       YES
177634  052702  000001                    BIS     #BIT0,R2        ;       NO
177640  005110                    2$:     COM     (R0)            ; RESTORE TARGET LOCATION.
177642  005702                            TST     R2              ; SET CONDITION FOR CALLER.
177644  000207                            RETURN
                                  ;
                                  ; HIGH SEGMENT COPY OF THE FLOATING 1/0 DATA TABLE.
                                  ;
177646     015     012            XFLT10: .BYTE   15, 12          ; SYNC PAIR.
177650     001     002     004            .BYTE   001, 002, 004, 010, 020, 040, 100, 200  ; FLOATING 1.
177660     376     375     373            .BYTE   376, 375, 373, 367, 357, 337, 277, 177  ; FLOATING 0.
177670  000000                            .WORD   0               ; TERMINATOR.

                                          .DSABL  AMA
```

```
        000001                            .end
```

Symbol table

```
ABAUD   166656 G    BIT4  = 000020 G    CNTNU = 000062      DMA600  171160      IN300   173402
AB100   166712      BIT5  = 000040 G    CQIO    177462      DPRV12= 000134      IN400   173402
AB200   166722      BIT6  = 000100 G    CQIOA   177470      DPRV4 = 000120      IP    = 020000
AB300   166740      BIT7  = 000200 G    CR      000015 G    DPRV8 = 000124      IPVXIT  177370
AB400   166756      BIT8  = 000400 G    CRDS    166277 G    DS100   166346      IPV12   177364
AB500   166770      BIT9  = 001000 G    CRLF    166274 G    DS200   166346      IPV4    177350
AB600   166770      BOOTFO  177774 G    CSTMD   000006 G    ENDTAB  166655      IPV8    177356
ACC     161146      BOOTST  172322      CTCC  = 000040      EOP   = 000002      IRX     163566
ACCUM   173540 G    BRKENT  164674 G    CTCS  = 000024      EOPBIT= 000002      ISEXT   163600
ADDREG  165522 G    BSUPV   171170 G    CTMS  = 000070      ERRB  = 100000 G    ITX     163544
ADSTKO  177700 G    BS100   171240      CTRL  = 000000      ERRFGB  100000 G    IXIT    163630
AD100   165530      BS200   171300      CTTC  = 000054      ERROR   166264 G    KAT   = 170002
AD200   165530      BS300   171326      CTV   = 000010      ERRTB   172750      KC$DD   000022 G
AD220   165550      BS400   171346      C.CLR   162200      EXAMIN  165742 G    KC$DI   000006 G
AD240   165560      BS500   171352      C8255 = 177526 G    EXMDIS  165470      KC$ED   000020 G
AD250   165564      BS550   171372      DARFGB  000002 G    EXTEST  172570 G    KC$EI   000004 G
AD253   165574      BS570   171404      DATCOM  166404 G    EX10    165766      KC$GS   000010 G
AD255   165606      BS600   171420      DA100   166412      EX1000  166224      KC$MAX  000022 G
AD270   165614      BUFR1 = 150002      DA200   166412      EX1050  166242      KC$NOP  000000 G
AD280   165642      BUFR2 = 160002      DA370   166426      EX1200  166260      KC$RD   000014 G
AD290   165644      BYTES = 170002      DA400   166452      EX1300  166262      KC$RES  000002 G
AD300   165646      B1200 = 000022      DA500   166460      EX20    165776      KC$SS   000012 G
AD400   165652      B2400 = 000032      DA600   166464      EX300   166020      KC$WD   000016 G
AD500   165666      B300  = 000002      DA700   166472      EX400   166030      KC.COM  000036 G
AD600   165674      B4800 = 000042      DCDFGB  000001 G    EX500   166040      KC.EOM  004000 G
AD700   165720      B600  = 000012      DC.DAT  161614      EX600   166040      KC.ICC  000040 G
AD800   165724      B9600 = 000052      DC.LBO  161672      EX700   166056      KC.IDA  000100 G
AI      175010      CABIT = 010000      DC.RCV  161500      EX750   166072      KC.IDR  000200 G
AI.BO = 164377      CHA   =%000003      DC.XIT  161676      EX760   166140      KC.LEN  003400 G
AO.BI = 165400      CHAINX  176100      DC.XMT  161406      EX770   166146      KC.VEC  177400 G
APLPCO  177766 G    CHAIN1  175756      DECTST  170072 G    EX800   166156      KE$DNA  000014 G
APLPSO  177770 G    CHAIN2  176000      DEPOS   166474 G    EX900   166172      KE$ILC  000006 G
APLR00  177750 G    CHAIN4  176020      DE200   166526      EX950   166222      KE$ILL  000010 G
APLR10  177752 G    CHAIN5  176040      DE50    166512      EX970   166222      KE$ILV  000012 G
APLR20  177754 G    CHAIN6  176060      DINIT   173412 G    FAR     164116 G    KE$MAX  000014 G
APLR30  177756 G    CHA1H = 000046      DISPRG  165450      FARSTR  172370 G    KE$NDA  000002 G
APLR40  177760 G    CHA1L = 000042      DISPST  166346 G    FAS     171430 G    KE$NDR  000004 G
APLR50  177762 G    CHA2H = 000044      DISPWD  166532 G    FCHFGB  000010 G    KE$OK   000000 G
APLSPO  177764 G    CHA2L = 000040      DI100   166550      FERRD   000017 G    KH.CMD  175001 G
ARDFGB  000004 G    CHB   =%000004      DI200   166554      FLT0    163700      KH.CSD  177531 G
A.SET   162130      CHEND   165742      DM    = 174400      FLT1    163670      KH.IOC  177541 G
A4800   162126      CHKCHR  173614 G    DMACDA= 174454 G    FLT10   163666      KH.SC2  175005 G
BBF10   000000 G    CHNOFO= 174442      DMAED   000002 G    FRERRB= 020000 G    KH.SC3  175007 G
BBF20   177776 G    CHNSTO= 174446      DMALD   171032 G    FTLST   172736 G    KH.STA  175003 G
BEVNT = 000100      CHTAB   165734      DMAMMA= 174470 G    GETIN   173552 G    KL.CMD  175000 G
BI      175024      CHX   =%000004      DMAV  = 000164      GE100   173552      KL.CSA  177520 G
BIACKV= 000130      CH100   173652      DMAXIT  175720      GE200   173610      KL.CSB  177522 G
BIT0  = 000001 G    CH200   173670      DMA0OA= 174442 G    GE300   173612      KL.CSC  177524 G
BIT1  = 000002 G    CH250   173702      DMA0SA= 174446 G    GO      166772 G    KL.CSD  177530 G
BIT10 = 002000 G    CH300   173704      DMA0VA= 174532 G    GP300   167176      KL.IOC  177540 G
BIT11 = 004000 G    CH400   173704      DMA1VA= 174530 G    HZ800   162124      KL.SC2  175004 G
BIT12 = 010000 G    CKSUM   161042      DMA100  171104      INDATA  166640      KL.SC3  175006 G
BIT13 = 020000 G    CLISP = 170006      DMA200  171114      INHIB = 170010      KL.STA  175002 G
BIT14 = 040000 G    CM      177574      DMA300  171142      INIT    173302 G    KM$IDS  000360 G
BIT15 = 100000 G    CMDR  = 000054      DMA400  171150      INITST  172320      KM$MAP  000016 G
BIT2  = 000004 G    CMREGA= 174454      DMA50   171046      IN100   173352      KM$STF  000007 G
BIT3  = 000010 G    CNDFGB  000020 G    DMA500  171150      IN200   173356      KM$SWS  000360 G
```

```
KP$INI  000000 G      KX$DME  040000 G      KX$T04  100010 G      ODTSTK  177746 G      PWRV  = 000024
KP$NNC  000007 G      KX$DPE  000040 G      KX$T05  100020 G      OPENDS  166301 G      PW050   173026
KP$PBS  000005 G      KX$DRT  000200 G      KX$T06  100040 G      ORERRB= 040000 G      PW100   173050
KP$QOD  000003 G      KX$EXO  100000 G      KX$T07  100100 G      PACS  = 000020        PW200   173112
KP$QTS  000002 G      KX$FEF  000040 G      KX$T10  100200 G      PADB  = 000032        PW300   173124
KP$STS  000001 G      KX$F01  000001 G      KX$T11  100400 G      PAHS  = 000102        PW500   173152
KP$WST  000004 G      KX$F02  000002 G      KX$T12  101000 G      PAMS  = 000100        PW600   173156
KR$QRP  040000 G      KX$F03  000004 G      KX$T13  102000 G      PARLBD  000016 G      PW700   173252
KR$TNS  000001 G      KX$F04  000010 G      KX$2EN  000002 G      PAV   = 000004        QA      177562
KS.ALN  003400 G      KX$F05  000020 G      K3K   = 014000        PBCS  = 000022        QADR  = 170004
KS.DA   000200 G      KX$F06  000040 G      LA      177566        PBDB  = 000034        QAHI  = 170002
KS.DBG  020000 G      KX$F07  000100 G      LADR  = 170006        PBHS  = 000122        QALO  = 170004
KS.DR   000040 G      KX$F10  000200 G      LBTEST  163712 G      PBMS  = 000120        QCOMIN  167460 G
KS.EOM  000100 G      KX$F11  000400 G      LB100   163712        PBREB = 000002 G      QC100   167560
KS.ERC  000036 G      KX$F12  001000 G      LEDTSD  000017 G      PBRF  = 000070 G      QC120   167616
KS.ERR  100000 G      KX$F13  002000 G      LF      000012 G      PBV   = 000006        QC150   167626
KS.ON   040000 G      KX$GOO  000010 G      LS0     000017 G      PCDB  = 000036        QC200   167640
KW$PAB  040000 G      KX$INC  000004 G      LS1     000016 G      PCDD  = 000014        QC300   167660
KW$PB0  000001 G      KX$I20  000010 G      LS10    000005 G      PCRLF   165510        QC350   167714
KW$PB1  000002 G      KX$I24  000020 G      LS11    000004 G      PERV    173000 G      QC380   167734
KW$PB2  000004 G      KX$I34  000040 G      LS12    000003 G      PEVNT = 000104        QC400   167736
KW$PB3  000010 G      KX$LD0  000001 G      LS13    000002 G      PIOAV = 000154        QC420   167776
KW$PB4  000020 G      KX$LD1  000002 G      LS14    000001 G      PIOAVA= 177004 G      QC440   170002
KW$PB5  000040 G      KX$LD2  000004 G      LS15    000000 G      PIOBV = 000160        QC445   170022
KW$PB6  000100 G      KX$LD3  000010 G      LS2     000015 G      PIOBVA= 177006 G      QC447   170024
KW$PB7  000200 G      KX$MP0  000002 G      LS3     000014 G      PIOD    174404        QC450   170042
KW$PCO  100000 G      KX$MP1  000004 G      LS4     000013 G      PIOD1   174450        QC470   170050
KW$PHD  001000 G      KX$MP2  000010 G      LS5     000012 G      PIOD2   174532        QC50    167512
KW$PLD  000400 G      KX$NOP  000040 G      LS6     000011 G      PIOD3   174562        QC500   170052
KW$POD  002000 G      KX$NXF  010000 G      LS7     000010 G      PIOICA= 177000 G      QC600   170056
KW$P1D  004000 G      KX$NXM  100000 G      LS8     000007 G      PIOTVA= 177010 G      QC70    167524
KW$P2D  010000 G      KX$ODC  000010 G      LS9     000006 G      PIOXIT  174746        QIO     177560
KW$P3D  020000 G      KX$ODT  000001 G      MAINTB= 000004 G      PIT     174152        QIOC    177600
KW.CMD  175000 G      KX$OHL  000020 G      MCC   = 000002        PIT1    174222        QIRV  = 000144
KW.CSA  177520 G      KX$OMO  000001 G      MEM1    160304        PRIBD   000001 G      QIRV1 = 000150
KW.CSB  177522 G      KX$ORO  000002 G      MEM2    160312        PROC    167010 G      QIXIT   176372
KW.CSC  177524 G      KX$PRO  000020 G      MEM3    160320        PROMPT  166266 G      QODTM   171432 G
KW.CSD  177530 G      KX$PWF  001000 G      MEM3A   160324        PRTYV = 000114        QODTMD  000005 G
KW.DAT  000004 G      KX$QIE  020000 G      MIC   = 000000        PR0   = 000000        QSRET   171444 G
KW.DCO  000000 G      KX$QOF  000200 G      MMR   = 000070        PR1   = 000040        QS100   171476
KW.DC0  175010 G      KX$QRE  002000 G      MMRFGA= 174470        PR2   = 000100        QS1000  171710
KW.DC1  175020 G      KX$QRF  004000 G      MODE  = 170004        PR3   = 000140        QS1100  171726
KW.DST  000002 G      KX$RTE  000100 G      NACBIT= 004000        PR4   = 000200        QS200   171476
KW.IOC  177540 G      KX$R20  000001 G      NECV  = 000070        PR5   = 000240        QS300   171536
KW.SC2  175004 G      KX$R24  000002 G      NECXIT  163470        PR6   = 000300        QS400   171560
KW.SC3  175006 G      KX$R34  000004 G      NEC.AA  162262        PR7   = 000340        QS500   171612
KW.STA  175002 G      KX$SEF  000010 G      NEC.AS  162242        PUTEST  172324 G      QS700   171632
KX$BAT  002000 G      KX$SHC  000020 G      NEC.DA  162210        PUTSDS  000007 G      QS800   171656
KX$BDF  020000 G      KX$SMA  000004 G      NEC.DM  162624        PU100   172440        QS900   171702
KX$BHE  000400 G      KX$SOF  000100 G      NEC.EX  163640        PU1010  172566        RAMLO   160634
KX$BHF  001000 G      KX$SXT  004000 G      NEC.I   163522        PU20    172360        RAMLOA  160636
KX$BQI  010000 G      KX$TIS  000020 G      NEC.MD  163144        PU30    172372        RBOT1   000000 G
KX$CME  100000 G      KX$TRC  000001 G      NEC.SY  162250        PU400   172526        RBOT2   040000 G
KX$CTE  000200 G      KX$TTC  000010 G      NFERRD  000010 G      PU50    172434        RBOT3   100000 G
KX$DEN  000100 G      KX$T01  100001 G      NFFWDO  177772 G      PU500   172534        RBUF1A= 177562 G
KX$DEO  000004 G      KX$T02  100002 G      NNCED   000000 G      PU600   172536        RCSR1A= 177560 G
KX$DMC  000002 G      KX$T03  100004 G      ODTSTG  177702 G      PWRUP   173012 G      RCVACB= 004000 G
```

```
Symbol table

RCVBKB= 004000 G      SN200   172262        S2ALBD  000014 G      TST6    161344        T7     - 160034 G
RCVDNB= 000200 G      SODTM   164626 G      S2BLBD  000015 G      TST7    161730        T7AETB  171002
RCVIEB= 000100 G      SODTMD  000004 G      TC    = 000001        TU100   171740        T7BETB  171006
RD    = 000000        S0100   164746        TCLK    161164        TU1000  172150        T7ERR   170664 G
RDB   = 177776        S01000  165104        TCPU    161150        TU120   171774        T7ERR1  170676
REGDIS  166303 G      S01300  165142        TCXIT   161322        TU125   171776        T7ERR2  170704
REGFGB  000200 G      S01400  165170        TDC     161344        TU130   172012        UROMOD  000011 G
REINIT  173016 G      S01450  165174        TDMA    175106        TU150   172042        UROM1D  000012 G
RESTRT  164000 G      S01470  165174        TDMA1   175174        TU200   172060        VERS    173774 G
RE1000  164314        S01500  165240        TDMA2   175276        TU300   172064        WAITD   000003 G
RE1010  164352        S01550  165254        TDMA3   175346        TU400   172066        WAITST  167402 G
RE1020  164352        S01552  165304        TDPR    176432        TU500   172102        WA100   167402
RE1040  164352        S01555  165312        TDPR2   176520        TU58BT  171734 G      WA200   167446
RE1045  164372        S01557  165322        TE100   172630        TU600   172110        WA300   167452
RF1050  164412        S01560  165330        TE200   172642        TU800   172144        WC      177572
RE1060  164416        S01565  165356        TE250   172712        TU900   172144        WCTBL   175742
RE1070  164416        S01567  165364        TE300   172726        T0    = 160000 G      WQIO    177410
RE1072  164442        S01569  165376        TE400   172740        T1    = 160004 G      WQIOA   177416
RE1075  164450        S01570  165402        TIOR    160040        T1ERk   170506 G      WRT   = 000010
RE1080  164450        S01600  165404        TIPV1   176676        T1ERR1  170532        XBRKB = 000001 G
RE200   164074        S01700  165410        TIPV2   177100        T1ERR2  170540        XBUF1A= 177566 G
RE250   164104        S01800  165412        TIPXIT  177260        T10   = 160000 G      XCSR1A= 177564 G
RE400   164106        S01850  165412        TMP1  = 170002        T10ERR  170720 G      XCVR  = 000140
RE450   164120        S01900  165440        TMP2  = 170004        T10ETB  171012        XDB   = 000002
RE500   164142        S0200   164746        TMP3  = 170006        T11   = 160004 G      XFLT10  177646
RE600   164226        S0300   164776        TMP4  = 170010        T11ERP  170726 G      XIEB  = 000100 G
RE700   164260        S0400   165006        TMP5  = 170012        T11ETB  171016        XOFF  = 000023
RE800   164270        S0700   165054        TMP6  = 170014        T12   = 160010 G      XOFFGB  000040 G
RE900   164274        S0800   165060        TMP7  = 170016        T12ERR  170734 G      XON   = 000021
RNGFGB  000002 G      S0900   165062        TMP8  = 170020        T12ETB  171022        XRAMLO  177602
RQIO    177400        SPACE   166272 G      TNEC    161730        T13   = 160014 G      XRDYB = 000200 G
RSTV    173004 G      SPOFF   000014 G      TNNCM1  170754        T13ERR  170742 G      $CSR  = 177530
RTOP1   077776 G      SPR5SU  173706 G      TNNCOM  170746        T13ETB  171026        $CSRA = 177520
RTOP2   137776 G      SP1     077762 G      TNNINX  170762        T14   = 160020 G      $CSRB = 177522
RTOP3   157776 G      SP2     137762 G      TNNXIT  170764        T15   = 160024 G      $CSRC = 177524
SAVERG  164452 G      SP200   173724        TPIO    174040        T16   = 160030 G      $CSRCO= 177526
SD100   172300        SP3     157762 G      TQIR    176102        T17   = 160034 G      $DMA  = 174400
SEND    172300        SP300   173752        TRAM    160272        T2    = 160010 G      $DPR  = 175000
SETIE = 170010        SP400   173756        TRAPX   167366 G      T2ERR   170546 G      $EF   =%000005
SETLDJ  167202 G      SR100   164474        TRAP10  167314 G      T2ERR1  170562        $EF1  = 177776
SETLED  167200 G      SR200   164526        TPAP24  167356 G      T3    = 160014 G      $EN   = 000014
SE100   167222        SR300   164536        TRAP4   167252 G      T3ERR   170576 G      $IPV  = 175000
SE150   167242        SR400   164622        TPOM    160700        T3ERR1  170612        $I8254= 175720
SE200   167242        SSRET   164654 G      TR1010  167344        T3ERR2  170626        $LEDS = 177524
SE300   167250        STACHN= 170012        TR410   167302        T4    = 160020 G      $PIO  = 177000
SKTA    160546        STAT  = 177774        TST1    160040        T4ERR   170642 G      $QDPR1= 160000
SL1RV = 000060        STATS = 170006        TST10   174040        T4ETB   170766        $QDPR2= 175400
SL1XV = 000064        STAT1 = 000056        TST11   175106        T5    = 160024 G      $QIR  = 177532
SL2CAA= 175704 G      STAT2 = 000054        TST12   176102        T5ERR   170650 G      $SL1  = 177560
SL2CBA= 175714 G      STSENL= 000010        TST13   176432        T5ETB   170772        $SL2  = 175700
SL2SAA= 175700 G      STSTAB  170354        TST2    160272        T6    = 160030 G      $TN   = 000013
SL2SBA= 175710 G      S.SET   162150        TST3    160700        T6ERR   170656 G      $TOP  =%000005
SNDBRK  172234        SOREGA= 174456        TST4    161150        T6ETB   170776        $UID  = 177524
SN100   172240        S1LBD   000013 G      TST5    161164

. ABS.  177672     000    (RW,I,GBL,ABS,OVR)
        000000     001    (RW,I,LCL,REL,CON)
```