Appendix D

Universal Serial Test Program – QuickBASIC Version

In Chapter 6 we made extensive use of the Universal Serial Test Program; either the QuickBASIC version (USTPQB) or the Visual Basic version (USTPVB). That's really all that's required because it's not necessary to understand how the test programs work internally but simply how to use them when you need them. However, for readers desiring an understanding of the inner workings of Universal Serial Test Programs – this appendix is the place to find that information for the QuickBASIC version. The Visual Basic version is covered in Appendix E.

Although grasping the internal details of the USTPQB program isn't required to use its testing capability, having such knowledge does:

- Significantly increase one's appreciation of how the overall system functions
- Provide a baseline for porting the testing techniques to languages other than Basic
- Help with system debugging if problems develop
- Enhance programming skills by studying how the program is written.

I'll go into considerable detail explaining the program's content. My underlying philosophy regarding all C/MRI documentation is that, "I would rather it be said that I provide too much detail rather than not enough."

Because of the universal nature of the program (performs output card test and wraparound card test for every type of node including USIC, SUSIC, 24 and 32-bit I/O cards as well as SMINI) the program is rather lengthy with considerable branching. Fortunately though, taking each group of statements individually, the program is quite easy to understand. I'll walk you through the listing to highlight the programming techniques.

PROGRAMMING HIGHLIGHTS (USTPQB)

A listing of the USTPQB program in provided at the end of this appendix. Also, it's included on the disk supplied with this manual as file USTPQB.BAS. Let's now examine some of the key programming statements.

The USTPQB program utilizes the CALL version of the Serial Protocol Subroutines; therefore I've included the 5 DECLARE and the corresponding DIM SHARED and COMMON SHARED statements defined back in Chapter 13. Instead of using a single PND$( ) array to handle card-port title printouts, there are 2 string arrays; PNU$( ) for USIC/SUSIC nodes and PNM$( ) for SMINI nodes.

The first thing the program needs to know is: "Are you testing an SMINI (or a USIC/SUSIC) node?" This is accomplished at line label NODEIN by using QuickBASIC's standard INPUT statement which prints out a question and then waits for a keyboarded reply. Two IF-THEN statements are used to check the user's input reply, which I've stored in the variable CMD$. Because an uppercase reply is an entirely different character than a lowercase reply, both need to be checked before branching. Thus, if the user's reply is a "Y" or a "y" the program branches to line label INSTM (which I picked to represent – initialize setup test for an SMINI node). If the user's reply is a "N" or "n" then the program branches to INSTU for handling the USIC/SUSIC case.

A key point is that it is not enough to simply check and branch on the "Y" or "y" case and if not then let the program fall through to handle the USIC/SUSIC case. Why? Because a user can enter erroneous inputs such as "X" or "USIC" or whatever. **To prevent accepting ALL "invalid" inputs, a separate IF-THEN statement is used for each valid input.** Fundamentally, each IF-THEN statement checks for a valid input reply and branches accordingly. Then, once through all the IF-THEN statements and no valid response is found, it's known that the inputted reply is invalid. For this case the program prints the message "INVALID ENTRY – TRY AGAIN" and branches back to the label (in this case NODIN) to reenter inputs. This type of user keyboarded input validity checking is vital to providing a robust program. We'll use the technique over-and-over again as each user input is requested.

At the line labels INSTU and INSTM, multiple PRINT lines are executed defining to the user the setup requirements for performing each test. Directly after the printouts, the program sets the Node Definition Parameter (NDP$). For the SMINI case it's easy; NDP$ = "M". However, to do this for the USIC/SUSIC case it's required to know if the SUSIC is using 24- or 32-bit I/O cards. Thus, the INPUT statement is used followed by 2 IF-THEN statements to check for the 2 valid inputs (24 and 32). If the user input is 24, NDP$ is set to "N". If the input is 32, then NDP$ is set to "X". If the input is neither 24 nor 32, then an "INVALID ENTRY – TRY AGAIN" message is printed followed by a branch back to BITIN to try again.

At label COMIN, the user is asked to input the PC serial communications port number which must be 1, 2, 3 or 4 as validated by the corresponding 4 IF-THEN statements. The same general procedure is used to have the user input the desired baud rate.

At line label ADJPAR (picked for adjustable parameters), the default values for MAXTRIES, DL, SLOWDIS and INFILTER are set. Later on we'll see that these default values can be overridden by the user. For example by varying the value of SLOWDIS, the user can control the speed at which the lighted LED moves across the Test Card. By varying the value of INFILTER, the user can compensate for the hardware delay resulting from adding the RC low pass filters to Input Cards.

At label CHKIN (picked for check to see if the user wants to input changes to the default parameters), we first print out the default values and then using the INPUT statement ask if changes are desired. If the answer is "N" or "n", the program branches directly to label TESTIN, thus skipping around reading in parameter changes. If the answer is "Y" or "y", the program branches to have the user input the desired parameter changes. If the user response is not "N", "n", "Y" or "y" an "INVALID ENTRY – TRY AGAIN" message is printed with a loop back to label CHKIN to try again.

At label DFLTIN, an INPUT statement is used to read in each of the desired values for MAXTRIES, DL, SLOWDIS and INFILTER. Once all 4 values are inputted, they are written back out to the screen for user verification. An INPUT statement is then used to ask the user if the entered values are correct followed by the 2 IF-THEN statements to respond to the reply. If the reply is a "Y" or "y" the program branches to TESTIN. If the reply is an "N" or "n" the program branches back to DFLTIN for data reentry. If the reply is not "Y", "y", "N" or "n" the program branches back to label RPTIN to repeat the entry of the yes or no request.

At label TESTIN, the program requests user input to define the desired test to be performed. If the user input is "O" or "o", a branch is made to begin an Output Card Test at line label OTEST. If the user input is "W" or "w", a branch is made to begin a Wraparound Card Test at line label WTEST. If the user input is "E" or "e", a branch is made to end testing at line label OPSYS. If the user input isn't any of these valid entries, an "INVALID ENTRY – TRY AGAIN" message is printed with a branch back to TESTIN to try again.

From the above, I hope you begin to see how easy it is to generate C/MRI application programs. Fundamentally, learn a few basic statements and techniques, use them over and over again changing variable names and numbers with the result: you soon have a working program. Copy, paste and change a little is the primary technique used to create unique C/MRI application programs to exactly meet your requirements.

At this point, the program really splits into two separate regions:

- At line label OTEST, the program goes into performing an Output Card Test
- At line label WTEST, the program goes into performing a Wraparound Test.

I'll walk through the Output Card Test first (starting at line label OTEST) and then we'll look at the Wraparound Test. The first step is to check NDP$ to see if the node is an SMINI. If it is, the program branches to line label OTESTM to initialize the SMINI node set up for Output Card testing. If it's not an SMINI, the program falls through the IF-THEN statement to line label OTESTU for initializing the USIC/SUSIC node. Both sets of node initialization statements follow the exact pattern we've been using throughout this manual.

Once the proper node is initialized, the program reaches line label OTESTR (picked for Output Card Test repeating) where the actual test code is located. The statements following the OTESTR label pretty much follow what we've written for the previous Output Demonstration Programs in Chapters 7 and 12.

One exception is the use of the IF-THEN-ELSE statements to select if the printout of card-port information is for an SMINI or for a USIC/SUSIC. Also, the delay loop used to allow time to read the Test Card LEDs is changed to use the SLOWDIS factor rather than a "fixed constant" as in previous examples.

The Wraparound Test begins at line label WTEST. The first step is a user request to determine if every test line is to be printed or simply to print data lines when there's an error. A user reply of "N" or "n" sets variable PRTIO (used as a print I/O flag), equal to 0 to suppress every line printout. A user reply of "Y" or "y" sets PRTIO equal to 1 to enable every line printout. It's nice to run with every I/O line printed for general testing and to only print error lines when letting a test run for long periods of time (like overnight) for counting the number of random communication errors.

At line label CHKNDP, the program checks NDP$ to see if the node is an SMINI. If it is, the program branches to line label WTESTM to initialize the SMINI node set up for wraparound testing. If it's not an SMINI the program falls through the IF-THEN statement to line label WTESTU for initializing the USIC/SUSIC node. Both sets of node initialization statements follow the exact pattern we've been using throughout this manual.

Once the proper node is initialized, the program reaches line label WTESTR (picked for Wraparound Test repeating) where the actual test code is located. The statements following the WTESTR label pretty much follow what we've written for the previous Wraparound Display Programs in Chapters 7 and 12. One exception is the addition of the INFILTER delay loop to account for adding input line filtering to input ports. Another exception is the built-in branching based upon the value of the PRTIO flag. When PRTIO equals 1 every I/O line is printed and when PRTIO equals 0 only I/O lines with errors are printed.

That's really all there is to understanding the content of the USTPQB program.

PROGRAM SOURCE LISTING (USTPQB.BAS)

The remainder of this appendix is devoted to the source code listing of the USTPQB program. This source code is also provided on disk as file USTPQB.BAS.

```
 DEFINT A-Z
 DECLARE SUB INIT ()
 DECLARE SUB INPUTS ()
 DECLARE SUB OUTPUTS ()
 DECLARE SUB RXBYTE ()
 DECLARE SUB TXPACK ()
   REM*****************USTPQB.BAS************************
   REM***** UNIVERSAL SERIAL CARD TEST PROGRAM          **
   REM**COMBINED OUTPUT CARD AND WRAPAROUND TEST PROGRAM**
   REM**              QuickBASIC CALL VERSION            **
   REM**   FOR USE WITH USIC, SUSIC AND SMINI NODES      **
   REM**   WRITTEN BY BRUCE CHUBB - August 18, 2003      **
   REM*****************************************************

   REM**DIMENSION VARIABLES USED IN MAINLINE ONLY
      DIM PNU$(4)  'Array used to print out port letter for SUSIC/USIC nodes
      DIM PNM$(6)  'Array used to print out card-port for SMINI nodes

   REM**GLOBALIZE SERIAL PROTOCOL HANDLING VARIABLES
      DIM SHARED OB(60), IB(60), CT(15), TB(80)
      COMMON SHARED UA, COMPORT, BAUD100, NDP$, DL, NS, NI, NO, MAXTRIES
      COMMON SHARED INBYTE, ABORTIN, INTRIES, INITERR, PA, LM, MT

   REM**DEFINE I/O CARD PORTS FOR USIC AND SUSIC NODES
      PNU$(1) = "A": PNU$(2) = "B": PNU$(3) = "C": PNU$(4) = "D"

   REM**DEFINE OUTPUT CARD-PORT NOMENCLATURE FOR SMINI NODE
      PNM$(1) = "0 A": PNM$(2) = "0 B": PNM$(3) = "0 C"
      PNM$(4) = "1 A": PNM$(5) = "1 B": PNM$(6) = "1 C"

   REM**CLEAR SCREEN AND PRINT OUT PROGRAM TITLE
      CLS    'Clear screen
      PRINT "************************************************************"
      PRINT "**COMBINED TEST PROGRAM FOR OUTPUT CARD AND WRAPAROUND TEST**"
      PRINT "**     FOR USE WITH SERIAL USIC, SUSIC AND SMINI NODES     **"
      PRINT "************************************************************"
      PRINT " "   'Print blank line used throughout program

   REM**CHECK IF TESTING SMINI NODE TO PRINT HARDWARE ASSUMPTIONS**
NODEIN:
      INPUT "ARE YOU TESTING AN SMINI NODE?, ENTER Y OR N ? ", CMD$
      IF CMD$ = "Y" OR CMD$ = "y" THEN GOTO INSTM
      IF CMD$ = "N" OR CMD$ = "n" THEN GOTO INSTU
      PRINT "INVALID ENTRY - TRY AGAIN"
      PRINT " "
      GOTO NODEIN

INSTU:
   REM**PRINT OUT HARDWARE SETUP ASSUMPTIONS FOR USIC AND SUSIC**
     PRINT " "
     PRINT "******FOR OUTPUT CARD TESTING WITH USIC AND SUSIC*****"
     PRINT "1) Node's USIC address must be 0 (UA DIP switch all off)"
     PRINT "2) Output card in card address slot 0 (DIP switch all off)"
     PRINT "3) Compatible test card to be mounted on output card"
     PRINT " "
     PRINT "******FOR WRAPAROUND TESTING WITH USIC AND SUSIC*****"
     PRINT "1) Node's USIC address must be 0 (UA DIP switch all off)"
     PRINT "2) Output card in card address slot 0 (DIP switch all off)"
```

```
     PRINT "3) Output card must be standard current sinking configuration"
     PRINT "4) Input card in card address slot 1 (DIP switch right segment on)"
     PRINT "5) Wraparound cable to be connected between output and input cards"
     PRINT "6) Adding a second output card, set to the same card address..."
     PRINT "   ...slot as the first output card, with an appropriate test..."
     PRINT "   ...card attached enhances wraparound test debug if a..."
     PRINT "   ...problem is detected with the input card."
     PRINT " "

   REM**DEFINE NODE DEFINITION PARAMETER FOR USIC AND SUSIC APPLICATIONS**
BITIN:
      PRINT "Note: If using classic USIC node, I/O cards must be 24-bit"
      PRINT " "
      INPUT "INPUT NUMBER OF I/O LINES PER I/O CARD (24 OR 32)? ", NUMBITS
      IF NUMBITS = 24 THEN NDP$ = "N": GOTO COMIN
      IF NUMBITS = 32 THEN NDP$ = "X": GOTO COMIN
      PRINT "INVALID INPUT - TRY AGAIN"
      PRINT " "
      GOTO BITIN

INSTM:
   REM**PRINT OUT HARDWARE SETUP ASSUMPTIONS FOR SMINI**
     PRINT " "
     PRINT "******FOR OUTPUT CARD TESTING WITH SMINI*****"
     PRINT "1) Node's SMINI address must be zero (UA DIP switch all off)"
     PRINT "2) Compatible test card to be mounted on output card 0"
     PRINT "3) Additional test card can be mounted on output card 1 or..."
     PRINT "  ...the same test card moved to card 1 once card 0 is tested OK"
     PRINT " "
     PRINT "******FOR WRAPAROUND TESTING WITH SMINI*****"
     PRINT "1) Node's SMINI address must be 0 (UA DIP switch all off)"
     PRINT "2) Test card to be mounted on output card 0"
     PRINT "3) Output card 1 must be standard current sinking configuration"
     PRINT "4) Wraparound cable to be connected between output card 1 and..."
     PRINT "...input card 2 (make sure A0 on card 1 connects to A0 on card 2)"

   REM**DEFINE NODE DEFINITION PARAMETER FOR SMINI APPLICATIONS**
      NDP$ = "M"

   REM**READ KEYBOARD TO DEFINE DESIRED PC COM PORT**
COMIN:
      INPUT "TO BEGIN TESTING, INPUT PC COM PORT = 1, 2, 3, OR 4 ? ", COMPORT
      IF COMPORT = 1 THEN GOTO BAUDIN
      IF COMPORT = 2 THEN GOTO BAUDIN
      IF COMPORT = 3 THEN GOTO BAUDIN
      IF COMPORT = 4 THEN GOTO BAUDIN
      PRINT "INVALID ENTRY - TRY AGAIN"
      PRINT " "
      GOTO COMIN

   REM**READ KEYBOARD TO DEFINE DESIRED BAUD RATE**
BAUDIN:
      PRINT " "
      PRINT "Baud rate of 9600 or 19200 advised when testing I/O cards..."
      PRINT "...as this minimizes the chances that a communication error can..."
      PRINT "...occur that might be mistakenly identified as a board error"
      PRINT " "
      PRINT "The baud rate DIP switch, as well as the UA address DIP..."
      PRINT "...switch, are read only during the card's power up cycle."
      PRINT "Therefore, if you change either of these DIP switch..."
      PRINT "...settings, you must cycle power to the card to activate..."
      PRINT "...the new switch settings."
      PRINT " "
      PRINT "INPUT BAUD RATE/100 CORRESPONDING TO DIP SWITCH SETTINGS:"
```

```
      PRINT "   FOR   9600 enter 1"
      PRINT "   FOR  19200 enter 2"
      PRINT "   FOR  28800 enter 3"
      PRINT "   FOR  57600 enter 4"
      INPUT "   FOR 115200 enter 5 ? ", BAUD
      IF BAUD = 1 THEN BAUD100 = 96: GOTO ADJPAR
      IF BAUD = 2 THEN BAUD100 = 192: GOTO ADJPAR
      IF BAUD = 3 THEN BAUD100 = 288: GOTO ADJPAR
      IF BAUD = 4 THEN BAUD100 = 576: GOTO ADJPAR
      IF BAUD = 5 THEN BAUD100 = 1152: GOTO ADJPAR
      PRINT "INVALID ENTRY - TRY AGAIN"
      GOTO BAUDIN

ADJPAR:
   REM**DEFINE INITIALIZED DEFAULT PARAMETERS**
      PRINT " "
      MAXTRIES = 30000 'Define maximum number of reading input tries for PC
      DL = 0           'Define USIC delay between transmitted bytes to PC
      SLOWDIS = 400    'Define slow display factor to observe test card LEDs
      INFILTER = 0     'Define input filtering delay for wraparound testing

   REM**PRINT OUT DEFAULT VALUES OF PARAMETERS
      PRINT " "
      PRINT "DEFAULT PARAMETERS ARE:"
      PRINT "  MAXTRIES = "; MAXTRIES
      PRINT "  USIC DELAY (DL) = "; DL
      PRINT "  SLOW DISPLAY FACTOR (SLOWDIS) = "; SLOWDIS
      PRINT "  INPUT FILTER DELAY (INFILTER) = "; INFILTER

   REM**CHECK IF DESIRE TO CHANGE DEFAULT PARAMETERS
CHKIN:
      INPUT "WANT TO CHANGE DEFAULT PARAMETERS?, ENTER Y OR N ? ", CMD$
      IF CMD$ = "N" OR CMD$ = "n" THEN GOTO TESTIN
      IF CMD$ = "Y" OR CMD$ = "y" THEN GOTO DFLTIN
      PRINT "INVALID ENTRY - TRY AGAIN"
      GOTO CHKIN

   REM**READ KEYBOARD FOR UPDATING DEFAULT PARAMETERS
DFLTIN:
      PRINT " "
      INPUT "INPUT MAXTRIES ? ", MAXTRIES
      INPUT "INPUT USIC TRANSMISSION DELAY ? ", DL
      INPUT "INPUT SLOW DISPLAY FACTOR ? "; SLOWDIS
      INPUT "INPUT INPUT FILTER DELAY ? "; INFILTER

   REM**DISPLAY UPDATED DEFAULT PARAMETERS
      PRINT " "
      PRINT "DEFAULT PARAMETERS HAVE BEEN CHANGED TO:"
      PRINT "  MAXTRIES = "; MAXTRIES
      PRINT "  USIC DELAY = "; DL
      PRINT "  SLOW DISPLAY FACTOR = "; SLOWDIS
      PRINT "  INPUT FILTER DELAY = "; INFILTER

   REM**USER VERIFICATION OF CHANGES BEING OK
RPTIN:
      INPUT "ARE CHANGED VALUES CORRECT ?, ENTER Y OR N", CMD$
      IF CMD$ = "Y" OR CMD$ = "y" THEN GOTO TESTIN
      IF CMD$ = "N" OR CMD$ = "n" THEN GOTO DFLTIN
      PRINT "INVALID ENTRY - TRY AGAIN"
      GOTO RPTIN

   REM**READ KEYBOARD FOR DESIRED TEST TO BE PERFORMED
TESTIN:
      PRINT " "
```

```
      PRINT "INPUT DESIRED TEST TO BE PERFORMED:"
      PRINT "  For OUTPUT CARD TEST enter O (letter O)"
      PRINT "  For WRAPAROUND TEST enter W"
      INPUT "  For Ending Testing enter E ? ", DESTST$
      IF DESTST$ = "O" OR DESTST$ = "o" THEN GOTO OTEST
      IF DESTST$ = "W" OR DESTST$ = "w" THEN GOTO WTEST
      IF DESTST$ = "E" OR DESTST$ = "e" THEN GOTO OPSYS
      PRINT "INVALID ENTRY - TRY AGAIN"
      GOTO TESTIN

   REM************************************************
   REM**BEGIN USER ROUTINE TO PERFORM OUTPUT CARD TEST**
   REM************************************************
OTEST:
      CLS
      PRINT "PERFORMING OUTPUT CARD LED DRIVER TEST"
      PRINT " "

   REM**CHECK AND BRANCH TO SEPARATE TEST INITIALIZATION FOR SMINI
      IF NDP$ = "M" THEN GOTO OTESTM

   REM**INITIALIZE USIC AND SUSIC
OTESTU:
      PRINT "PERFORMING OUTPUT CARD TEST ON USIC OR SUSIC"
      PRINT " "
      UA = 0       'USIC address
      NS = 1       'Number of card sets of 4
      NI = 0       'Number of input ports
      IF NDP$ = "N" THEN NO = 3 'Define number of output ports 24-bit card
      IF NDP$ = "X" THEN NO = 4 'Define number of output ports 32-bit card
      CT(1) = 2    'Card type definition (one output card in slot 0)
      CALL INIT  'Invoke initialization subroutine
      PRINT " "
      PRINT "Node Initialization is complete - check blink rate of..."
      PRINT "...green status LED for 1 second on 1 second off blink rate"
      PRINT " "
      PRINT "Press any key to continue"
      SLEEP        'Sleep until key is pressed
      GOTO OTESTR  'Branch to start output card test repeating

   REM**INITIALIZE SMINI
OTESTM:
      PRINT "PERFORMING OUTPUT CARD TEST ON SMINI"
      PRINT " "
      UA = 0       'USIC address
      NS = 0       'Number of 2-lead yellow aspect oscillate signals
      NI = 3       'Number of input ports
      NO = 6       'Number of output ports
      CALL INIT    'Invoke initialization subroutine
      PRINT " "
      PRINT "Node Initialization is complete - check blink rate of..."
      PRINT "...green status LED for 1 second on 1 second off blink rate"
      PRINT " "
      PRINT "Press any key to continue"
      SLEEP        'Sleep until key is pressed

   REM**BEGIN COMMON CODE OF ALL OUTPUT CARD TESTING**
   REM**INITIALIZE ALL LEDS TO OFF
OTESTR:
      FOR I = 1 TO NO
         OB(I) = 0
      NEXT I

   REM**INCREMENT PORT TO BE TESTED IN A LOOP
```

```
      FOR PN = 1 TO NO

   REM**INCREMENT DISPLAYED BIT NUMBER IN A LOOP

      FOR N = 0 TO 7

   REM**OUTPUT TEST STATUS TO CRT FOR SMINI AND NON-SMINI
      IF NDP$ = "M" THEN
         PRINT "PORT IS   "; PNM$(PN); "   BIT NUMBER IS  "; N
      ELSE
         PRINT "PORT IS   "; PNU$(PN); "   BIT NUMBER IS  "; N
      END IF

   REM**SETUP TEST LED PATTERN FOR DISPLAY ON TEST CARD
      OB(PN) = 2 ^ N   'Integer 2 raised to the power N shifts...
                       '...the turn-on displayed LED one position...
                       '...to the left each loop through program.

   REM**OUTPUT LED DISPLAY DATA TO CARD
      CALL OUTPUTS    'Invoke transmit data subroutine

   REM**ADD DELAY TO BE ABLE TO READ LEDs
      IF SLOWDIS > 0 THEN               'Add slow display loop if non-zero
        FOR I = 1 TO SLOWDIS            'Delay by looping to give...
           FOR IJ = 1 TO 1000: NEXT IJ  '...time to visually check...
        NEXT I                          '...operation of LEDs on test card
      END IF

   REM**COMPLETE LOOPS
      NEXT N       'Update display to next LED bit position
      OB(PN) = 0   'Last LED within port tested so turn off port LED
      NEXT PN      'Update display to next port number

   REM**REPEAT OUTPUT CARD TEST
      GOTO OTESTR  'All ports completed, so branch back to repeat test

   REM************************************************
   REM**BEGIN USER ROUTINE TO PERFORM WRAPAROUND TEST**
   REM************************************************
WTEST:
      CLS
      PRINT "PERFORMING WRAPAROUND TEST"
      PRINT " "

   REM**CHECK IF DESIRE TO PRINT EVERY I/O LINE OR ONLY ERROR LINES
CHKPIO:
      PRINT " "
      PRINT "NOTE: Entering N to following will print only error lines"
      INPUT "DESIRE TO PRINT EVERY I/O LINE ?, ENTER Y OR N ? ", CMD$
      IF CMD$ = "N" OR CMD$ = "n" THEN PRTIO = 0: GOTO CHKNDP
      IF CMD$ = "Y" OR CMD$ = "y" THEN PRTIO = 1: GOTO CHKNDP
      PRINT "INVALID ENTRY - TRY AGAIN"
      GOTO CHKPIO

   REM**CHECK AND BRANCH TO SEPARATE TEST INITIALIZATION FOR SMINI
CHKNDP:
      IF NDP$ = "M" THEN GOTO WTESTM

   REM**INITIALIZE USIC AND SUSIC
WTESTU:
      PRINT "PERFORMING WRAPAROUND TEST ON USIC OR SUSIC"
      PRINT " "
      UA = 0      'USIC address
```

```
      NS = 1      'Number of card sets of 4
      IF NDP$ = "N" THEN NI = 3: NO = 3  'Define no. I/O ports 24-bit cards
      IF NDP$ = "X" THEN NI = 4: NO = 4  'Define no. I/O ports 32-bit cards
      CT(1) = 6   'Card type definition (output slot 1, input slot 2)
      CALL INIT   'Invoke initialization subroutine
      PRINT "Node Initialization is complete - check blink rate of..."
      PRINT "...green status LED for 1 second on 1 second off blink rate"
      PRINT " "
      PRINT "Press any key to continue"
      SLEEP       'Sleep until key is pressed
      GOTO WTESTR 'Branch to start wraparound test repeating

   REM**INITIALIZE SMINI
WTESTM:
      PRINT "PERFORMING WRAPAROUND TEST ON SMINI"
      PRINT " "
      UA = 0      'USIC address
      NS = 0      'Number of 2-lead yellow aspect oscillating signals
      NI = 3      'Number of input ports
      NO = 6      'Number of output ports
      CALL INIT   'Invoke initialization subroutine
      PRINT " "
      PRINT "Node Initilization is complete - check blink rate of..."
      PRINT "...green status LED for 1 second on 1 second off blink rate"
      PRINT " "
      PRINT "Press any key to continue"
      SLEEP       'Sleep until key is pressed

   REM**BEGIN COMMON CODE OF ALL WRAPAROUND TESTING
   REM**INITIALIZE ALL OUTPUTS TO ZERO
WTESTR:
      FOR I = 1 TO NO
         OB(I) = 0
      NEXT I

   REM**SETUP TO INCREMENT THROUGH ONLY 3 PORTS IF SMINI...
          '...AND THROUGH ALL NO PORTS IF USIC OR SUSIC
          'Note: During wraparound testing of SMINI...
          '...test program increments test pattern...
          '...through first 3 ports while setting the...
          '...second 3 ports equal to the same pattern.
          'During wraparound testing of SUSIC and USIC...
          '...test program increments test pattern...
          '...through all NO ports.
      IF NDP$ = "M" THEN N3NO = 3 ELSE N3NO = NO

   REM**INCREMENT PORT NUMBER TO BE TESTED IN A LOOP
      FOR PN = 1 TO N3NO

   REM**INCREMENT TEST PATTERN IN A LOOP
      FOR D = 0 TO 255
      OB(PN) = D
      IF NDP$ = "M" THEN OB(PN + 3) = D  'For SMINI case also set...
                                    '...second card's port to same bit pattern
   REM**OUTPUT BIT PATTERN TO OUTPUT CARD
      CALL OUTPUTS   'Invoke transmit subroutine

      FOR IJ = 1 TO 1000: NEXT IJ   'Basic delay to slow program...
                                    '...to read LEDs on output card

   REM**ADD DELAY TO ACCOUNT FOR INPUT LINE FILTERING
      IF INFILTER > 0 THEN      'Add delay loop if INFILTER non-zero
         FOR I = 1 TO INFILTER  'Delay to compensate for input card filtering
            FOR IJ = 1 TO 100   'Stretch length of effective delay
```

```
            NEXT IJ
         NEXT I
      END IF

   REM**READ BIT PATTERN FROM INPUT CARD
      CALL INPUTS  'Invoke receive subroutine

   REM**OUTPUT TEST STATUS TO CRT
      IF PRTIO = 1 THEN
       PRINT "PORT IS "; PNU$(PN); " DEC OUT = "; OB(PN); " DEC IN = "; IB(PN)
      END IF

   REM**COMPARE INPUT TO OUTPUT AND BRANCH TO CONTINUE LOOP IF NO ERROR
      IF IB(PN) = OB(PN) THEN GOTO CLOOP

   REM**ERROR FOUND SO PRINT TO CRT
      IF PRTIO = 1 THEN
       PRINT "ERROR FOUND IN ABOVE I/O, ENTER RETURN TO CONTINUE TEST"
       INPUT CR   'Input carriage return (enter key) to continue
      ELSE
       PRINT "PORT IS "; PNU$(PN); " DEC OUT = "; OB(PN); " DEC IN = "; IB(PN)
       PRINT "ERROR FOUND IN ABOVE I/O, CONTINUING TEST"
      END IF

   REM**COMPLETE LOOPS
CLOOP:
      NEXT D       'Increment to next bit pattern for port
      OB(PN) = 0   'Zero out port before starting new port
      IF NDP$ = "M" THEN OB(PN + 3) = 0  'For SMINI case also set...
                                         '...second card's port to 0
      NEXT PN    'Increment to next port

   REM**REPEAT WRAPAROUND TEST
      GOTO WTESTR  'At end of port loops so branch back to repeat test

   REM**TERMINATE TESTING
OPSYS:
      PRINT " "
      PRINT "TESTING IS BEING TERMINATED BY USER REQUEST"
      SYSTEM     'Return to system program
```