SCELBI AUDIO CASSETTE TAPE UNIT INTERFACE

INTER-CONNECTION INSTRUCTIONS

TO CONNECT THE CASSETTE TAPE UNIT INTERFACE TO THE STANDARD SCELBI-8H CHASSIS, PLUG ONE END OF AN I/O CABLE INTO P1 AND THE OTHER END INTO THE OUTPUT PORT SOCKET TO BE USED (OUTPUT PORT 13 IF STANDARD SCELBI SOFTWARE IS TO BE USED). PLUG ONE END OF ANOTHER I/O CABLE INTO P2 AND THE OTHER END INTO THE INPUT PORT SOCKET TO BE USED.(INPUT PORT 3 IF STANDARD SCELBI SOFTWARE IS TO BE USED.) CONNECT +5 TO THE RED BINDING POST AND GROUND TO THE BLACK BINDING POST.

IMPORTANT

THE 'SYNC' LINE FROM THE CPU CARD MUST BE WIRED TO PIN 1Ø OF BOTH THE INPUT PORT SOCKET AND THE OUTPUT PORT SOCKET ON THE SCELBI-8H CHASSIS. TO DO THIS CONNECT A WIRE FROM XAØ2 B-A TO PIN 1Ø OF THE INPUT PORT SOCKET AND PIN 1Ø OF THE OUTPUT PORT SOCKET TO BE USED.

TO CONNECT THE AUDIO CASSETTE TAPE INTERFACE TO A CUSTOM CHASSIS, CONNECT P1-1 THRU -8 TO THE DATA LINES Ø THRU 7, RESPECTIVELY, OF THE OUTPUT PORT DATA BUSS AND P1-9 TO THE OUTPUT PORT STROBE OF THE OUTPUT PORT TO BE USED. (DØL THRU D7L AND ØS3 IF STANDARD SCELBI SOFTWARE IS TO BE USED.) CONNECT P2-7 TO AN INPUT PORT DATA LEAD AND P2-8 TO A DIFFERENT DATA LEAD ON THE SAME PORT. (INP36 AND INP37, RESPECTIVELY, IF SCELBI SOFTWARE IS TO BE USED.) CONNECT +5 VOLTS TO THE RED BINDING POST AN GROUND TO THE BLACK BINDING POST.

IMPORTANT

CONNECT THE 'SYNC' LINE FROM THE CPU CARD PIN B-A TO P1-1Ø AND P2-1Ø OF THE TAPE INTERFACE.

TO CONNECT THE TAPE INTERFACE TO THE AUDIO CASSETTE RECORDER, PLUG THE RECORDER INTERFACE CABLE INTO S1. THE LARGE SILVER PHONE PLUG IS CONNECTED TO THE EARPHONE JACK OF THE TAPE RECORDER. THE LARGE RED (OR BLACK) PHONE PLUG IS CONNECTED TO THE MICROPHONE OR AUXILIARY INPUT OF THE TAPE RECORDER AND THE SMALL JACK IS CONNECTED TO THE MOTOR CONTROL JACK.

PROGRAMMING CONSIDERATIONS

ANY PROGRAM WHICH USES THE TAPE INTERFACE SHOULD OUTPUT A ZERO WORD (OCTAL CODE 000) TO THE TAPE INTERFACE IN IT'S INITIALIZATION ROUTINE.

THE OUTPUT PORT GOING TO THE TAPE INTERFACE USES BITS 6 AND 7 AS CONTROL BITS AND BITS 0 THRU 3 AS DATA BITS. THE DEFINITIONS FOR THE BITS ARE GIVEN IN TABLE 1.

OUTPUT PORT DATA BIT	DEFINITION

BIT 7	IF =1, READ MODE
	IF =0, WRITE MODE

BIT 6	IF =1, MOTOR IS ON
	IF =0, MOTOR IS OFF

BIT 5	NOT USED
BIT 4	NOT USED
BIT 3	MOST SIGNIFICANT DATA BIT
BIT 2	DATA BIT
BIT 1	DATA BIT
BIT 0	LEAST SIGNIFICANT DATA BIT

TABLE 1

THE INPUT PORT GOING TO THE TAPE INTERFACE USES BIT 6 FOR TRANSFER OF TAPE INTERFACE STATUS AND BIT 7 FOR TRANSFER OF DATA FROM THE TAPE TO THE COMPUTER (READ MODE). THE DEFINITION FOR THESE BITS ARE GIVEN IN TABLE 2.

INPUT PORT DATA BIT	DEFINITION
BIT 7	EQUALS THE DATA VALUE READ FROM THE TAPE IN THE READ MODE.

BIT 6	IF =1, IN THE WRITE MODE, INDICATES THE INTERFACE IS READY TO ACCEPT THE NEXT DATA TRANSFER.
	IF =0, IN THE WRITE MODE, INDICATES THE INTERFACE IS BUSY WRITING OUT TO TAPE.

BITS 0 THRU 5	NOT USED

TABLE 2

FOR RECORDING DATA, THE OUTPUT FROM THE TAPE INTERFACE MAY GO TO EITHER THE MICROPHONE OR THE AUXILIARY INPUT OF THE TAPE UNIT. DETERMINE WHICH INPUT IS BEST FOR THE PARTICULAR RECORDER BEING USED.

FOR BOTH READING AND WRITING, THE VOLUME AND TONE CONTROLS OF THE TAPE UNIT SHOULD BE SET AS FOLLOWS. THE TONE CONTROL SHOULD BE SET TO MAXIMUM TREBLE. THE VOLUME CONTROL SETTING SHOULD BE SET IN THE FOLLOWING MANNER. SET THE TAPE RECORDER TO PLAYBACK RECORDED DATA WITH THE VOLUME TURNED ALL THE WAY DOWN. OBSERVE THE INDICATOR DS1 (FIGURE 1) THRU THE LARGE HOLE IN THE TOP OF THE INTERFACE AND TURN UP THE VOLUME UNTIL THE INDICATOR IS JUST ON, THEN TURN UP THE VOLUME CONTROL ANOTHER 1/8 TURN. WHEN DATA IS BEING READ IN THE INDICATOR WILL CHANGE IN INTENSITY BUT NEVER GO OUT.

TO WRITE A WORD OR WORDS ONTO TAPE, USING THE AUTOMATIC MOTOR CONTROL SET THE TAPE UNIT TO THE RECORD MODE. THE PROGRAM SHOULD THEN PERFORM THE FOLLOWING STEPS:

1. OUTPUT A MOTOR START COMMAND TO SET UP THE TAPE INTERFACE AND TO START THE TAPE MOVING. THIS COMMAND IN BINARY IS 11ØØØØØØ.
2. OUTPUT A WRITE COMMAND WITH THE FIRST HALF OF THE DATA TO BE RECORDED IN BITS Ø THRU 3 OF THE COMMAND.
   Ø1ØØWXYZ WHERE WXY AND Z CORRESPOND TO THE FIRST HALF
   OF THE DATA TO BE RECORDED.
3. DELAY APPROXIMATE 2 INSTRUCTIONS (FOR EXAMPLE: LAA, LAA)
4. READ STATUS OF THE TAPE UNIT BY LOOKING AT BIT 6 OF THE INPUT PORT.
5. IF STATUS IS =Ø, JUMP TO STEP 4.
6. DELAY ABOUT 2 MILLISECONDS FOR STOP BIT.
7. OUTPUT THE 2ND HALF OF THE DATA TO BE WRITTEN BY OUTPUTTING A WRITE COMMAND WITH THE SECOND HALF OF THE DATA IN BITS Ø THRU 3.
8. READ THE TAPE STATUS BY LOOKING AT BIT 6 OF THE INPUT PORT.
9. IF STATUS =Ø, JUMP TO STEP 8.
1Ø. DELAY ABOUT 2 MILLISECONDS FOR A STOP BIT.
11. IF MORE IS TO BE WRITTEN RETURN TO STEP 2.
12. OUTPUT A ZERO WORD (ØØ ØØØ ØØØ) TO TURN OFF, THE TAPE UNIT.

FOR MANUAL MOTOR START AND STOP, DELETE STEP 1 AND MANUALLY START THE MOTOR BEFORE STARTING THE PROGRAM.

TO READ A WORD OR BLOCK OF WORDS FROM THE TAPE USING THE AUTOMATIC MOTOR CONTROL, FIRST MANUALLY POSITION THE TAPE TO THE CONSTANT TONE WHICH PRECEEDS THE DATA ON THE TAPE. THE PROGRAM SHOULD THEN PERFORM THE FOLLOWING STEPS:

1. OUTPUT A READ COMMAND TO THE TAPE INTERFACE TO SET UP A READ AND TO START THE MOTOR RUNNING. THIS COMMAND IN BINARY IS 11 ØØØ ØØØ.
2. WAIT APPROXIMATELY 1 SECOND TO ALLOW THE TAPE TO GET UP TO SPEED.
3. LOOK FOR THE START BIT BY LOOPING ON A TEST FOR BIT 7 OF THE INPUT PORT TO GO HIGH.
4. UPON RECEIPT OF THE START BIT, DELAY APPROXIMATELY 76Ø MICROSECONDS (1/2 BIT).
5. CHECK THE START BIT AGAIN TO MAKE SURE IT IS REAL, IF IT IS NOT, JUMP TO STEP 3.

6. DELAY TO GET 1538 MICROSECONDS OF PROCESSING PLUS DELAY TIME BETWEEN BITS.
7. READ IN THE DATA BIT AND SAVE IT.
8. IF THIS IS NOT THE FOURTH DATA BIT RETURN TO STEP 6.
9. SAVE THIS HALF OF THE WORD.PERFORM STEPS 3 THRU 8 AGAIN TO READ IN THE SECOND HALF OF THE WORD.
1Ø. PUT THE TWO HALFS TOGETHER AND STORE WHERE DESIRED.
11. IF MORE DATA IS TO BE READ IN, JUMP TO STEP 3.
12. WHEN FINISHED, OUTPUT A ZERO WORD TO TURN OFF THE TAPE.

FOR MANUAL MOTOR CONTROL, START THE TAPE MOTOR BEFORE STARTING THE PROGRAM.

JUST AS A REMINDER, SCELBI HAS SEVERAL PROGRAMS AVAILABLE, AT VERY REASONABLE PRICES, WHICH PERFORM THE FUNCTIONS DESCRIBED ABOVE. ORDER THE 8H TAPE WRITE PROGRAM #61XX-ØØØ2 AND THE 8H TAPE READ PROGRAM #61XX-ØØ3.

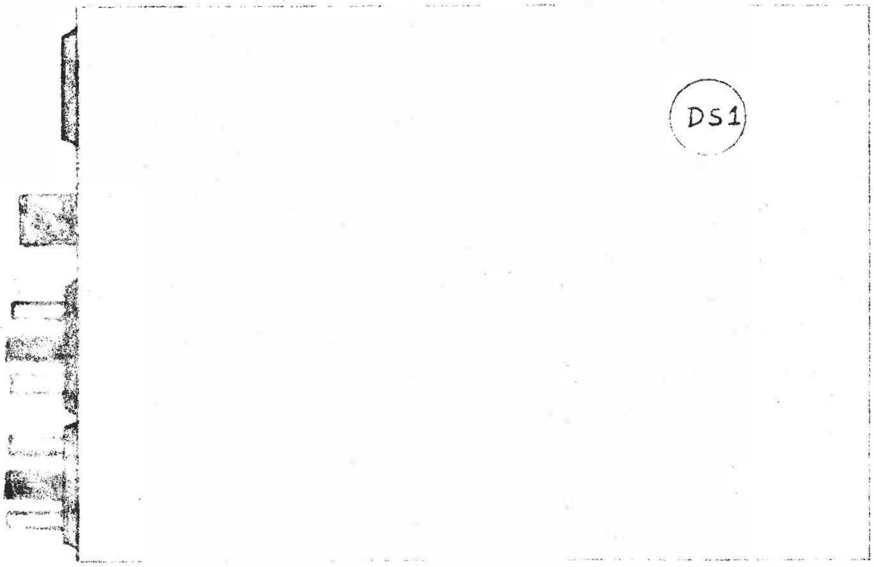

FIGURE 1

SCELBI SPECIAL FEATURE

AN IMPROVED TAPE READ/WRITE PROGRAM FOR THE SCELBI TAPE INTERFACE

IT WAS RECENTLY DECIDED TO DEVELOP AN IMPROVED GENERAL PURPOSE TAPE READ AND WRITE PROGRAM FOR THE SCELBI TAPE INTERFACE. THE OBJECTIVE WAS TO DEVISE A PROGRAM THAT WOULD HAVE GENERAL PURPOSE APPLICATION INSTEAD OF USING VARIOUS VERSIONS FOR DIFFERENT TYPES OF PROGRAMS AS HAS BEEN THE CASE IN THE PAST. ONE GOAL OF THE DEVELOPMENT EFFORT WAS TO HAVE THE ENTIRE PROGRAM FIT ON A SINGLE 1702 TYPE PROM SO THAT IT COULD SERVE AS A GENERAL PURPOSE UTILITY ROUTINE FOR A WIDE VARIETY OF PROGRAMS. PLACING THE PROGRAM ON A PROM, AS THE READER KNOWS, GIVES A SYSTEM INSTANT CAPABILITY AS SOON AS THE COMPUTER IS POWERED UP. THE PROGRAM DESCRIBED HERE HAS BEEN DESIGNATED AS THE NEW STANDARD TAPE READ/WRITE PROGRAM FOR THE SCELBI TAPE INTERFACE. IT IS BEING PRESENTED FOR THE BENEFIT OF ALL PREVIOUS SCELBI TAPE INTERFACE OWNERS.

THERE ARE NUMEROUS NEW FEATURES IN THE TAPE READ/WRITE PROGRAM AS WILL BE EXPLAINED SHORTLY. FIRST, HOWEVER, A REVIEW OF THE GENERAL OPERATION OF PREVIOUS SCELBI TAPE PROGRAMS IS IN ORDER.

AS SCELBI TAPE INTERFACE OWNERS KNOW, THE SCELBI TAPE INTERFACE, WHEN IN THE WRITE MODE, ACCEPTS FOUR "DATA" BITS FROM THE COMPUTER AT A TIME. THE INTERFACE THEN ADDS A "START" BIT AND SENDS THE START BIT AND FOUR DATA BITS AS ONE SERIAL GROUP. TO SEND A COMPLETE EIGHT BIT WORD THUS REQUIRES TWO WRITE OPERATIONS OF THE INTERFACE. FOR EXAMPLE, FIRST THE FOUR MOST SIGNIFICANT BITS OF A WORD ARE SENT TO THE INTERFACE, THEN THE FOUR LEAST SIGNIFICANT BITS.

IN THE RECEIVE MODE, THE TAPE INTERFACE RECEIVES DATA ASYNCHRONOUSLY. AS THE TAPE IS READ BACK, THE INTERFACE SEARCHES FOR A "START" BIT AND WHEN ONE IS DETECTED, A SOFTWARE ROUTINE IS UTILIZED TO SAMPLE THE NEXT FOUR UNITS OF TIME FOR THE FOUR DATA BITS IN A GROUP. THIS SEQUENCE IS REPEATED TWICE FOR EACH FULL EIGHT BIT WORD THAT IS RECEIVED TO BE PLACED IN THE COMPUTER'S MEMORY.

THE NEW TAPE READ/WRITE PROGRAM DOES NOT, OF COURSE, ALTER THE BASIC OPERATION OF THE TAPE INTERFACE AS IT HAS BEEN DESCRIBED IN THE PREVIOUS TWO PARAGRAPHS. HOWEVER, THE GENERAL "FORMAT" OF HOW DATA IS SENT TO AND RECEIVED FROM THE TAPE UNIT HAS BEEN ALTERED.

IN PREVIOUS TAPE READ/WRITE VERSIONS, DATA TO BE TRANSMITTED TO THE TAPE UNIT WAS FORMATTED IN SEVERAL WAYS DEPENDING ON THE TYPE OF PROGRAM BEING USED. FOR INSTANCE, IN THE BASIC TAPE READ/WRITE VERSION, WHEN A PERSON WANTED TO WRITE DATA TO THE TAPE UNIT, THE OPERATOR SET UP CPU REGISTERS "H&L" TO THE STARTING ADDRESS OF THE DATA BLOCK IN MEMORY, AND CPU REGISTER "E" TO A "WORD COUNT" WHICH INDICATED THE NUMBER OF CONSECUTIVE WORDS THAT WERE TO BE WRITTEN. THE BASIC PROGRAM LIMITED EACH WRITE OPERATION TO ONE PARTICULAR PAGE IN MEMORY. THE PROGRAM WOULD THEN SEND THE DATA TO THE TAPE UNIT BY SPLITTING EACH WORD IN HALF TO BE COMPATIBLE WITH THE TAPE INTERFACE. AS EACH MEMORY WORD WAS PROCESSED, THE TAPE WRITE PROGRAM COMPILED A "CHECK SUM" BY ADDING UP THE VALUE OF ALL THE "WORDS" SENT AND AT THE END OF THE PROGRAM IT WOULD SEND THE TWO'S COMPLEMENT OF THAT CALCULATED VALUE AS THE LAST "WORD" OF THE STRING OF DATA. IN THE READ MODE, THE OPERATOR WOULD AGAIN SET

CPU REGISTERS "H&L" TO THE STARTING ADDRESS WHERE DATA WAS TO BE LOADED, AND PLACE THE WORD COUNT IN REGISTER "E." THE READ PROGRAM WOULD THEN READ BACK THE DATA, COMBINING THE GROUPS OF FOUR BITS INTO EIGHT BIT GROUPS FOR STORAGE IN MEMORY WORDS. IT ALSO CALCULATED A "CHECK SUM" AS DATA WAS READ AND AT THE END OF A STRING OF DATA CHECKED TO SEE IF THE DATA HAD BEEN READ CORRECTLY BY ADDING THE TWO'S COMPLEMENT VALUE RECEIVED AS THE LAST "WORD" TO THE VALUE IT CALCULATED AS DATA WAS READ AND CHECKING TO SEE THAT THE SUM WAS ZERO.

IN THE "BLOCK" FORMAT TAPE READ WRITE PROGRAM, THE USER FIRST SET UP THE STARTING PAGE IN REGISTER "H" AND THE NUMBER OF PAGES TO BE PROCESSED IN CPU REGISTER "E." THIS PROGRAM ALLOWED MULTIPLE PAGES OF DATA TO BE PROCESSED AT ONE TIME AND ALSO USED A CHECK SUM TECHNIQUE BUT IT HAD A DRAW BACK OF REQUIRING FULL PAGES TO BE WRITTEN AT A TIME.

WHEN USING THE TAPE INTERFACE TO RECORD THE OBJECT CODE FOR PROGRAMS PRODUCED BY THE ORIGINAL SCELBI ASSEMBLER PROGRAMS, A SIMILAR APPROACH THAT UTILIZED A "WORD COUNT" AND CHECK SUM WAS UTILIZED, BUT NOW THE TAPE WAS AUTOMATICALLY FORMATTED SO THAT THE FIRST TWO WORDS IN A "FILE" ON A TAPE WERE TAKEN TO BE AN "ADDRESS" AND THE THIRD WORD WAS A "WORD COUNT." THE TECHNIQUE ALLOWED A LARGE PROGRAM, SCATTERED AT MANY LOCATIONS IN MEMORY TO BE AUTOMATICALLY CREATED AND READ BACK IN ONE OPERATION, BUT THE COMBINED READ AND WRITE PROGRAMS WERE RATHER LARGE.

OTHER TYPES OF PROGRAMS USED OTHER KINDS OF TAPE FORMATTING TO ACCOMPLISH SPECIFIC OBJECTIVES. ESSENTIALLY, EACH TYPE OF PROGRAM UTILIZING THE TAPE INTERFACE HAD A DIFFERENT TYPE OF FORMAT RESULTING IN A PROLIFERATION OF TAPE READ/WRITE PROGRAMS.

THE NEW SCELBI STANDARD TAPE READ/WRITE PROGRAM ESTABLISHES A FORMAT THAT CAN BE USED BY A WIDE VARIETY OF PROGRAMS WITH WIDELY RANGING FUNCTIONS. THE KEY TO THE PROGRAM'S SUCCESS HAS BEEN THE DEVELOPMENT OF A FORMAT FOR PLACING DATA ON THE TAPE WHICH IS EXPLAINED BELOW.

A BYTE OF DATA IN THE COMPUTER, THAT IS TO BE STORED ON THE TAPE UNIT CAN BE CONSIDERED AS CONSISTING OF EIGHT BINARY BITS ARRANGED AND SYMBOLIZED FROM MOST SIGNIFICANT TO LEAST SIGNIFICANT BITS AS SHOWN:

```
7   6   5   4   3   2   1   0
```

IN THE NEW TAPE FORMAT, EACH GROUP OF EIGHT BITS FROM THE COMPUTER IS SPLIT INTO TWO GROUPS OF FOUR BITS (A MOST SIGNIFICANT HALF AND A LEAST SIGNIFICANT HALF). THEN A NEW WORD OF EIGHT BITS IS FORMED BY ADDING FOUR BITS OF INFORMATION TO THE RIGHT OF EACH "HALF" OF THE ORIGINAL EIGHT BIT COMPUTER WORD. THESE FOUR BITS OF INFORMATION ARE USED TO DIRECT THE OPERATION OF THE TAPE UNIT AS WILL BE EXPLAINED SHORTLY. FOR NOW, ONE CAN VIEW THE ORIGINAL FORMAT OF AN EIGHT BIT WORD BEING SPLIT IN HALF AND COUPLED TO FOUR "INFORMATION" BITS SO THAT THE ORIGINAL DATA WOULD APPEAR AS:

```
P   H   L   T   7   6   5   4

P   -   -   -   3   2   1   0
```

THUS, WHAT WAS ORIGINALLY AN EIGHT BIT DATA WORD IS TRANSFORMED INTO TWO EIGHT BIT WORDS. EACH NEW EIGHT BIT WORD CONTAINS FOUR BITS OF THE ORIGINAL DATA AND FOUR NEW "STATUS" BITS ARRANGED AS SHOWN.

THE "STATUS" BITS SHOWN IN THE PREVIOUS ILLUSTRATION CARRY THE FOLLOWING INFORMATION:

THE "P" POSITION IS USED AS A PARITY BIT. THIS BIT IS SET BY THE WRITE PROGRAM SO THAT A GROUP OF EIGHT BITS IS ALWAYS "EVEN PARITY." THAT IS, THERE WILL BE AN EVEN NUMBER (0, 2, 4, 6 OR 8) BITS IN THE LOGIC "ONE" STATE.

THE "H" BIT IS SET TO A ONE IF THE "DATA BITS" ARE TO CONTAIN A HIGH ADDRESS (PAGE).

THE "L" BIT IS SET TO A ONE IF THE "DATA BITS" ARE TO CONTAIN A LOW ADDRESS.

THE "T" BIT IS SET TO A ONE TO SIGNIFY "TRAILER" CODE. TRAILER CODE SIGNIFIES THE END OF A TAPE "FILE" AND THE "DATA BITS" IN THE GROUP WILL BE IGNORED.

IF NEITHER THE "H," "L," OR "T" BIT IS SET TO A ONE THE "DATA BITS" ARE CONSIDERED TO BE INFORMATION TO BE LOADED INTO MEMORY.

USING THE ABOVE FORMAT, A TYPICAL WRITE OPERATION WOULD RESULT IN INFORMATION BEING WRITTEN ON THE TAPE AS FOLLOWS:

```
X  H  0  0  7  6  5  4
                          DATA BITS = PAGE ADDRESS
X  0  0  0  3  2  1  0


X  0  L  0  7  6  5  4
                          DATA BITS = LOW ADDRESS
X  0  0  0  3  2  1  0


X  0  0  0  7  6  5  4
                          DATA BITS = DATA
X  0  0  0  3  2  1  0


          .
          .
          .                      D A T A
          .
          .


X  0  0  T  7  6  5  4
                          DATA BITS = DON'T CARE
X  0  0  0  3  2  1  0
```

THIS FORMAT OFFERS SEVERAL NICE FEATURES. FOR ONE, IT ALLOWS THE WRITING OF TAPES THAT WHEN READ BACK IN WILL AUTOMATICALLY READ DATA INTO THE CORRECT MEMORY ADDRESSES WITHOUT ANY "INITIALIZATION" PROCEDURES. FOR ANOTHER, ERROR CHECKING IS ACCOMPLISHED ON A BYTE-BY-BYTE BASIS. THUS, IF AN ERROR IS DETECTED, THE PROGRAM CAN BE STOPPED IM-

MEDIATELY INSTEAD OF HAVING TO WAIT FOR AN ENTIRE PROGRAM TO BE READ IN ONLY TO FIND BY A CHECK SUM TECHNIQUE THAT AN ERROR OCCURED. THIRD, BY DEVELOPING THE OVER ALL PROGRAM AS A SERIES OF SUBROUTINES, THE PROGRAM ALLOWS CONSIDERABLE FLEXIBILITY AS WILL BE ILLUSTRATED.

FOR INSTANCE, BY REFERRING TO THE PROGRAM LISTING WHICH IS PRESENTED AT THE END OF THIS DISCUSSION, (THE LISTING SHOWS THE PROGRAM AS IT WOULD APPEAR RESIDING ON PAGE 17), ONE CAN SEE A GROUP OF SUBROUTINES IN THE WRITE SECTION.

THE FIRST SUCH SUBROUTINE, LABELED "WLEAD," SIMPLY STARTS THE TAPE RECORDER'S MOTOR AND PROVIDES FOR ABOUT A THREE SECOND DELAY BEFORE THE ROUTINE IS EXITED. THIS SUBROUTINE WOULD BE CALLED WHEN ONE WANTED TO START A NEW TAPE "FILE." AS SCELBI TAPE INTERFACE OWNERS KNOW, WHEN THE TAPE UNIT IS NOT TRANSMITTING DATA, IT WILL WRITE ALL "ZEROS" SO USING THIS SUBROUTINE WOULD EFFECTIVELY CAUSE ABOUT THREE SECONDS OF "LEADER" (ALL ZEROS) CODE TO BE WRITTEN ON THE TAPE.

THE NEXT SUBROUTINE STARTING AT LOCATION 010 AND LABELED "WADDR" IS A SUBROUTINE THAT WILL WRITE THE CONTENTS OF THE "H" AND "L" REGISTERS IN THE CPU ONTO THE TAPE IN THE DESCRIBED FORMAT. THE "H" STATUS BIT WILL BE SET WHEN THE "PAGE" ADDRESS IS WRITTEN, AND THE "L" STATUS BIT WILL BE SET WHEN THE "LOW" ADDRESS IS WRITTEN. NOTE THAT ONE ALSO HAS THE OPTION OF ENTERING THE SUBROUTINE AT LOCATION 016 LABELED AS "WADRL" IN THE EVENT ONE ONLY WANTS TO WRITE A NEW "LOW" ADDRESS BYTE! THUS, TO SEND ADDRESSING INFORMATION OUT TO THE TAPE UNIT ONE MERELY HAS THE CALLING ROUTINE SET UP "H & L" TO THE DESIRED ADDRESS (OR JUST "L" IF THAT OPTION IS DESIRED) AND CALLS THE "WADDR" SUBROUTINE.

THE NEXT SUBROUTINE SHOWN AT LOCATION 024 AND LABELED "WDATA" WILL CAUSE THE CONTENTS OF CPU REGISTER "C" TO BE WRITTEN ON THE TAPE UNIT AS A "DATA" WORD. THUS, TO WRITE A STRING OF LOCATIONS IN MEMORY AS DATA ON THE TAPE UNIT, ONE JUST LOADS SUCCESSIVE WORDS INTO REGISTER "C" AND CALLS THE "WDATA" SUBROUTINE.

FOLLOWING THE "WDATA" SUBROUTINE AT LOCATION 036 IS A SUBROUTINE LABELED "WTRAL." CALLING THIS SUBROUTINE WILL CAUSE THE PROGRAM TO WRITE AN "END OF FILE" OR "TRAILER CODE" BYTE TO THE TAPE UNIT AND STOP THE TAPE UNIT'S MOTOR.

TO WRITE A CONTINUOUS BLOCK OF DATA FROM ONE ADDRESS IN MEMORY TO ANOTHER HIGHER ORDERED ADDRESS VALUE, ONE CAN USE THE SUBROUTINE LABELED "WRITE" SHOWN AT LOCATION 147. PRIOR TO CALLING THE "WRITE" SUBROUTINE ONE HAS THE CALLING PROGRAM SET "H & L" TO THE STARTING ADDRESS OF THE BLOCK OF DATA TO BE WRITTEN, AND "D & E" SET TO THE ENDING ADDRESS. THE "WRITE" ROUTINE WILL THEN CALL ON THE PREVIOUSLY DESCRIBED ROUTINES IN THE CORRECT ORDER TO WRITE A VARIABLE LENGTH "FILE."

OF COURSE, THERE ARE TIMES WHEN ONE DOES NOT WANT TO WRITE JUST ONE CONTINUOUS BLOCK OF DATA, BUT MAY INSTEAD DESIRE TO WRITE A SERIES OF VARIABLE LENGTH BLOCKS RESIDING IN DIFFERENT MEMORY LOCATIONS, WITHOUT STOPPING THE TAPE UNIT. A TYPICAL EXAMPLE OF WHEN SUCH CAPABILITY IS DESIRED IS WHEN ONE IS USING AN ASSEMBLER TO PRODUCE OBJECT CODE AT VARIOUS LOCATIONS IN MEMORY. IN SUCH A CASE, ONE CAN CALL ON THE VARIOUS DESCRIBED SUBROUTINES IN THE ORDER DESIRED. FOR INSTANCE, WHENEVER AN "ORG" STATEMENT WAS PROCESSED BY THE ASSEMBLER, ONE COULD CALL THE "WADDR" TO ALTER THE ADDRESSING INFORMATION. ONE COULD ALTERNATELY WRITE NEW ADDRESSES FOLLOWED BY BLOCKS OF DATA FOR AS LONG AS NECESSARY FOR THE ASSEMBLY PROCESS AND THEN TERMINATE THE FILE BY CALLING THE "WTRAL" SUBROUTINE. THERE ARE OTHER TYPES OF PROGRAMS WHERE SUCH FLEXIBILITY IS DESIRED.

OF COURSE, IF ONE HAS CRITICAL APPLICATIONS WHERE ONE DOES NOT FEEL SECURE BY JUST USING "PARITY" ERROR CHECKING, ONE CAN HAVE A CALLING ROUTINE GENERATE A "CHECK SUM" OR OTHER ADDITIONAL ERROR CHECKING PROCEDURE AND WRITE THAT INFORMATION AS "DATA" WHEN DESIRED. AN ADDITIONAL ROUTINE WOULD THEN PROCESS THAT INFORMATION AS DESIRED ON THE RECEIVE SIDE.

FINALLY, ONE CAN FIND AT LOCATION 374 A ROUTINE CALLED "BWRIT." THIS ROUTINE WAS INCLUDED FOR "PROM" VERSIONS SO THAT A USER COULD MANUALLY SET UP "H & L" AND "D & E" AND WRITE A "FILE" AS A "STAND ALONE" FUNCTION.

OPERATION OF THE RECEIVE SIDE IS SIMPLICITY ITSELF. ONE SIMPLY CALLS THE SUBROUTINE "READ" AT LOCATION 210. OPERATION FROM THERE IS AUTOMATIC. THE READ PROGRAM WILL PROCESS THE INFORMATION ON THE TAPE, SETTING UP "H" AND "L" AS DIRECTED BY THE "STATUS" CODES IT RECEIVES AND LOADING DATA INTO MEMORY LOCATIONS UNTIL A PARITY ERROR IS DETECTED OR A "TRAILER" CODE IS DETECTED. UPON EXIT FROM THE "READ" ROUTINE CPU REGISTER "C" WILL CONTAIN ALL ZEROS IF NO ERRORS WERE DETECTED, OR ALL ONES IF A PARITY ERROR OCCURED. ONE CAN THUS HAVE THE CALLING ROUTINE CHECK TO SEE IF THE FILE READ WAS "VALID." THE READ ROUTINE ALSO PROVIDES FOR STARTING AND STOPPING THE TAPE UNIT'S MOTOR. NATURALLY, BEFORE USING THE READ ROUTINE ONE WOULD MANUALLY SET UP THE TAPE UNIT SO THAT IT WAS ON THE "LEADER" PORTION AT THE BEGINNING OF A "FILE." THE USER MAY NOTE THAT THE READ PROGRAM INTRODUCES ABOUT A HALF SECOND DELAY FROM THE TIME IT DIRECTS THE MOTOR TO START SO THAT THE TAPE UNIT WILL BE UP TO SPEED BEFORE LOOKING FOR INFORMATION ON THE TAPE. THIS TECHNIQUE ALSO ENABLES THE TAPE UNIT TO SKIP OVER ANY "GARBAGE" THAT CAN EXIST BETWEEN THE END OF ONE FILE (WHEN THE TAPE UNIT IS STOPPED) AND THE BEGINNING OF THE NEXT FILE (WHEN THE TAPE UNIT IS FIRST STARTED) SO THAT THE PROGRAM CAN BE USED TO PROCESS A WHOLE SERIES OF CONSECUTIVELY WRITTEN "FILES." (A FILE IS DEFINED HERE AS THE STARTING AND STOPPING OF TAPE MOTION. A FILE ITSELF MAY HAVE MULTIPLE "BLOCKS" OF DATA AT VARIOUS ADDRESSES USING THE DESCRIBED TAPE FORMAT!)

FINALLY, JUST ENOUGH ROOM WAS PROVIDED IN THE PROM VERSION TO BE ABLE TO INCLUDE THE ROUTINE AT LOCATION 370 LABELED "BREAD." THIS ROUTINE MAY BE USED WHEN A UNIT IS INITIALLY POWERED UP TO ALLOW AN OPERATOR TO READ IN TAPES. WHEN THIS ROUTINE IS USED, THE OPERATOR SHOULD USE MANUAL METHODS TO CHECK THE CONTENTS OF CPU REGISTER "C" WHEN THE TAPE UNIT STOPS TO SEE THAT IT CONTAINS THE "VALID" ALL ZEROS INDICATOR.

IN ADDITION TO THE OVER ALL IMPROVEMENTS THIS NEW FORMAT YIELDS, IN DEVELOPING THE ROUTINES IT WAS FOUND THAT A SUBSTANTUAL IMPROVEMENT IN THE READ ROUTINE ALLOWS INCREASED VARIATIONS OF THE TAPE UNITS SPEED TO STILL BE RECEIVED PROPERLY AND THE PROGRAM MAY PROVIDE SUFFICIENT MARGIN FOR USER'S TO EXCHANGE TAPES MADE ON DIFFERENT MACHINES WITHOUT HAVING TO ALTER THE TIMING CONSTANTS OF THE PROGRAM.

IN ANY EVENT, WE AT SCELBI HAVE FOUND THE OVER ALL PACKAGE TO BE MOST SATISFACTORY AND THINK OUR PREVIOUS TAPE INTERFACE OWNERS WILL FIND IT A CONSIDERABLE IMPROVEMENT.

```
                              /
017 000   026 220             WLEAD, LCI 220        /SETUP FOR 3 SEC DELAY
017 002   006 300             WLD1, LAI 300         /SET READ STATUS
017 004   127                 OUT 13                /START MOTOR
017 005   104 132 017         JMP DELXS             /TO DELAY ROUTINE
017 010                       /
017 010   016 100             WADDR, LBI 100        /SET ADDR STATUS CODE
017 012   325                 LCH                   /MOVE PG ADDR TO "C"
017 013   106 026 017         CAL WDAT1             /WRITE PAGE ADDRESS
017 016   016 040             LBI 040               /SET LOW ADDR STATUS
017 020   326                 LCL                   /MOVE LOW ADDR TO "C"
017 021   104 026 017         JMP WDAT1             /WRITE LOW ADDRESS
017 024                       /
017 024   016 000             WDATA, LBI 000        /SET DATA STATUS CODE
017 026   302                 WDAT1, LAC            /MOVE "C" INTO "A"
017 027   106 046 017         CAL PART1             /SEND STATUS & MSH
017 032   302                 LAC                   /RESTORE "C" TO "A"
017 033   104 071 017         JMP PART2             /SEND LSH
017 036                       /
017 036   016 020             WTRAL, LBI 020        /SET TRAILER STATUS
017 040   106 026 017         CAL WDAT1             /SEND TRAILER CODE
017 043   250                 XRA                   /CLEAR THE ACCUMULATOR
017 044   127                 OUT 13                /STOP THE MOTOR
017 045   007                 RET
017 046                       /
017 046   012                 PART1, RRC            /POSITION MSH TO LSB'S
017 047   012                 RRC
017 050   012                 RRC
017 051   012                 RRC
017 052   044 017             NDI 017               /MASK OFF RESIDUE
017 054   201                 ADB                   /ADD IN STATUS CODE
017 055   170 062 017         PCHEK, JTP SET1       /PARITY OK IF EVEN
017 060   004 200             ADI 200               /ELSE MAKE IT EVEN
017 062   310                 SET1, LBA             /SAVE IN REG "B"
017 063   106 102 017         CAL LSB               /SEND DATA HALF
017 066   104 076 017         JMP MSB               /SEND PARITY/STATUS
017 071                       /
017 071   044 017             PART2, NDI 017        /MASK OFF RESIDUE
017 073   104 055 017         JMP PCHEK             /FORM PARITY
017 076                       /
017 076   012                 MSB, RRC              /POSITION BITS
017 077   012                 RRC
017 100   012                 RRC
017 101   012                 RRC
017 102   044 017             LSB, NDI 017          /MASK OFF LEFT PART
017 104   004 100             LSBGO, ADI 100        /SET WRITE STATUS
017 106   127                 OUT 13                /WRITE TO TAPE
017 107   106 114 017         CAL WAIT              /LET TAPE WRITE
017 112   301                 LAB                   /RESTORE ORIG TO ACC
017 113   007                 RET
017 114                       /
017 114   107                 WAIT, INP 3           /CHECK TAPE STATUS
017 115   022                 RAL                   /MOVE BIT B6 TO B7
017 116   240                 NDA                   /SET FLAGS
017 117   120 114 017         JFS WAIT              /LOOP IF B7 IS ZERO
017 122   006 340             LAI 340               /SET DELAY CNTR VALUE
017 124   004 001             ACCLP, ADI 001        /FORM DELAY LOOP USING
017 126   110 124 017         JFZ ACCLP             /ONLY THE ACCUMULATOR
017 131   007                 RET
```

```
                        /
017 132   106 142 017   DELXS, CAL DELMOR    /LONG DELAY LOOP
017 135   021           DCC                  /FORMED BY NESTING ONE
017 136   110 132 017   JFZ DELXS            /COUNTER LOOP INSIDE
017 141   007           RET                  /ANOTHER (B INSIDE C)
017 142                 /
017 142   011           DELMOR, DCB          /SHORT DELAY LOOP
017 143   110 142 017   JFZ DELMOR           /DECREMENT REG B UNTIL
017 146   007           RET                  /IT REACHES ZERO VALUE
017 147                 /
017 147   106 000 017   WRITE, CAL WLEAD     /PROVIDE TAPE LEADER
017 152   106 010 017   CAL WADDR            /WRITE STARTING ADDRESS
017 155   327           WNEXT, LCM           /GET DATA FM MEMORY
017 156   106 024 017   CAL WDATA            /WRITE DATA
017 161   305           LAH                  /PUT CURRENT PG INTO ACC
017 162   273           CPD                  /COMPARE WITH "LAST" PAGE
017 163   110 173 017   JFZ WMORE            /KEEP GOING IF NOT EQUAL
017 166   306           LAL                  /PUT CURR LOCATION TO ACC
017 167   274           CPE                  /SEE IF AT LAST LOCA
017 170   150 201 017   JTZ WSTOP            /WRAP IT UP ON MATCH
017 173   106 204 017   WMORE, CAL ADVHL     /ELSE ADVANCE MEMORY PNTR
017 176   104 155 017   JMP WNEXT            /B4 CONTINUEING TO WRITE
017 201   106 036 017   WSTOP, CAL WTRAL     /PROVIDE TRAILER AT END
017 204   060           ADVHL, INL           /ADVANCE LOW MEM PNTR
017 205   013           RFZ                  /RETURN IF LOW PNTR NOT "0"
017 206   050           INH                  /ADV PG PNTR IF REQ'D
017 207   007           RET
                        /
017 210   026 060       READ, LCI 060        /SETUP FOR 0.5 SEC DELAY
017 212   106 002 017   CAL WLD1             /START MOTOR & DELAY
017 215   106 312 017   RNEXT, CAL RCHAR     /READ MSH OF A BYTE
017 220   130 305 017   JFP REROR            /ERROR IF ODD PARITY
017 223   330           LDA                  /STORE TEMP IN REG "D"
017 224   106 312 017   CAL RCHAR            /READ LSH OF A BYTE
017 227   130 305 017   JFP REROR            /ERROR IF ODD PARITY
017 232   044 017       NDI 017              /STRIP OFF PARITY HALF
017 234   340           LEA                  /HOLD IN "E" TEMPORARILY
017 235   313           LBD                  /SAVE "D" IN "B" TEMP
017 236   303           LAD                  /RESTORE "D" TO ACCUM
017 237   002           RLC
017 240   002           RLC                  /MOVE LSB'S OVER TO MSH
017 241   002           RLC
017 242   002           RLC
017 243   044 360       NDI 360              /GET RID OF RESIDUE
017 245   204           ADE                  /FORM COMPLETE BYTE
017 246   330           LDA                  /SAVE IN REG "D"
017 247   301           LAB                  /RESTORE STATUS TO ACC
017 250   044 160       NDI 160              /MASK OFF DATA & PARITY
017 252   150 276 017   JTZ RDATA            /DATA WORD IF NO STATUS
017 255   002           RLC                  /HAVE STATUS
017 256   002           RLC                  /MOVE TO TEST BY CARRY
017 257   100 266 017   JFC NOTPG            /IF CARRY NOT "1," - JUMP
017 262   353           LHD                  /SET PAGE ADDR IN REG "H"
017 263   104 215 017   JMP RNEXT            /FETCH NEXT BYTE FM TAPE
017 266   002           NOTPG, RLC           /IS BYTE FOR LOW ADDR ?
017 267   100 307 017   JFC RDONE            /HAVE TRAILER IF NOT
017 272   363           LLD                  /SET LOW ADDR IN REG "L"
017 273   104 215 017   JMP RNEXT            /FETCH NEXT BYTE FM TAPE
017 276   373           RDATA, LMD           /PUT DATA INTO MEMORY
017 277   106 204 017   CAL ADVHL            /ADVANCE MEMORY POINTER
017 302   104 215 017   JMP RNEXT            /FETCH NEXT BYTE FM TAPE
```

```
017 305  026 377       RERROR, LCI 377      /SET ERROR INDICATOR IN "E"
017 307  250           RDONE, XRA           /FOUND TRAILER MARKER
017 310  127           OUT 13               /CLR ACC & STOP MOTOR
017 311  007           RET
017 312                /
017 312  046 000       RCHAR, LEI 000       /CLEAR WORKING REGISTER
017 314  106 326 017   CAL BITS             /GET 4 LEAST SIG BITS
017 317  106 326 017   CAL BITS             /GET 4 MOST SIG BITS
017 322  304           LAE                  /RESTORE TO ACCUMULATOR
017 323  022           RAL                  /& GET LAST BIT FM CARRY
017 324  240           NDA                  /SET FLAGS AFTER ROTATE OP
017 325  007           RET                  /EXIT WITH INFO IN ACC
017 326                /
017 326  026 004       BITS, LCI 004        /SET A FOUR BIT COUNTER
017 330  107           START, INP 3         /LOOK FOR A START BIT
017 331  240           NDA                  /SET FLAGS AFTER INPUT
017 332  120 330 017   JFS START            /LOOP ON A LOW BIT B7
017 335  107           INP 3                /WHEN B7 GOES HIGH
017 336  240           NDA                  /PERFORM A DOUBLE CHECK
017 337  120 330 017   JFS START            /TO VERIFY A START BIT
017 342  016 037       LBI 037              /SET 1.5 BIT DELAY  1984 µs
017 344  106 142 017   CAL DELMOR           /CALL DELAY ROUTINE
017 347  107           BIT, INP 3           /SAMPLE INCOMING BIT
017 350  044 200       NDI 200              /MASK OFF UNUSED BITS
017 352  204           ADE                  /ADD TO PREVIOUS BITS     WRITE CIRCUIT
017 353  032           RAR                  /SHIFT BITS TO MAKE RDY   FLIPS N4886
017 354  340           LEA                  /FOR NEXT INCOMING BIT    SO MUST READ
017 355  016 024       LBI 024              /SET 1 BIT DELAY  1280 µs  BACKWARDS
017 357  106 142 017   CAL DELMOR           /CALL DELAY ROUTINE
017 362  021           DCC                  /DECREMENT BITS COUNTER
017 363  053           RTZ                  /EXIT WHEN HAVE 4 BITS
017 364  104 347 017   JMP BIT              /ELSE CONTINUE
017 367                /
017 367  000           000                  /SPARE
017 370                /
017 370                ORG 017 370
017 370  106 210 017   BREAD, CAL READ      /BOOT READ PROGRAM
017 373  000           000                  /HLT
017 374                /
017 374  106 147 017   BWRIT, CAL WRITE     /BOOT WRITE PROGRAM
017 377  000           000                  /HLT
020 000                /
```

SCELBI TAPE INTERFACE UNIT OWNERS WHO USE THE TAPE MOTOR CONTROL FEATURE WILL FIND THAT REMOVING THE MOTOR CONTROL PLUG FROM THE TAPE UNIT JACK CAN CAUSE THE MOTOR CONTROL RELAY TO CHANGE STATES. THE REASON THIS CAN OCCUR IS BECAUSE WHEN THE PLUG IS REMOVED OR INSERTED, TRANSIENT SIGNALS CAN OCCUR THAT WILL TRIGGER THE RELAY. THE SOLUTION TO THE PROBLEM IS A SIMPLE ONE. SIMPLY CONNECT A SEPARATE PERMANENT GROUND WIRE BETWEEN THE TAPE UNIT AND THE TAPE INTERFACE. THIS WIRE CAN BE ATTACHED AT THE SHIELD (GROUND) OF EITHER THE CABLE FOR THE TAPE INTERFACE READ PLUG OR WRITE PLUG. THE OTHER END OF THE WIRE SHOULD BE ATTACHED TO A SCREW THAT IS CONNECTED TO THE TAPE UNIT'S SIGNAL GROUND, OR TO THE COMMON SIDE OF THE MOTOR CONTROL, EARPHONE, OR RECORDING JACK. A GOOD MEANS OF SECURING THE WIRE TO THE JACK IS TO INSERT A LOCKING OR FLAT WASHER UNDER THE NUT THAT HOLDS THE JACK IN PLACE AND SOLDER THE GROUND WIRE TO THE WASHER.