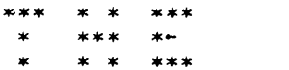
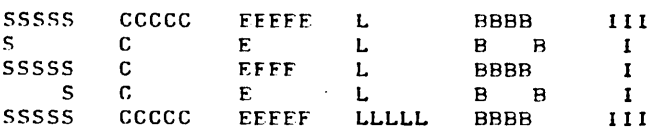




COMPUTER DIGEST

AND

USER'S BULLETIN

VOLUME I - ISSUE III

JULY 1975

EDITOR: N. WADSWORTH

ASSISTANT EDITOR: R. FINDLEY

SCELBI COMPUTER CONSULTING, INC.
1322 REAR - BOSTON POST ROAD
MILFORD, CT. 06460

BEFORE GETTING INTO THIS ISSUE, A FEW COMMENTS FROM THE EDITOR ARE NECESSARY. WE AT SCELBI ARE SOMEWHAT CONCERNED OVER THE SMALL PERCENTAGE OF READER'S WHO ARE CONTRIBUTING TO THE "SCELBI COMPUTER DIGEST." THIS PUBLICATION WAS LARGELY STARTED IN RESPONSE TO NUMEROUS INDIVIDUALS WHO REQUESTED A "USER FORUM" PUBLICATION. HOWEVER, DESPITE THE FACT THAT WE HAVE BEEN RECEIVING MANY FAVORABLE RESPONSES, AND MANY "PROMISES" OF CONTRIBUTIONS TO THE PUBLICATION, IT SEEMS THAT MOST OF THE SUBSCRIBERS ARE CONTENT TO READ RATHER THAN CONTRIBUTE. WE KNOW, FROM PERSONAL CONVERSATIONS AND LETTERS, THAT MANY OF OUR USER'S ARE DOING INTERESTING PROJECTS AND HAVE DEVELOPED GENERALLY USEFUL ROUTINES, BUT THEY APPARENTLY ARE SO INVOLVED THAT THEY FIND IT DIFFICULT TO FIND A FEW MOMENTS TO SHARE THEIR PROJECTS WITH OTHERS.

WHEN WE STARTED THIS PUBLICATION WE DID SO IN THE HOPES THAT WE WOULD BE PERFORMING A USEFUL SERVICE TO OUR USER'S BY PROVIDING AN INFORMATION EXCHANGE MEDIUM THAT MANY CUSTOMERS HAD BEEN CLAMORING FOR. HOWEVER, WE HAVE NOT FOUND THE USER INVOLVMENT TO BE ANY WHERE NEAR WHAT WE HAD ANTICIPATED. SHOULD THIS TREND CONTINUE, WE SHALL BE FORCED TO DISCONTINUE PUBLICATION OF THIS JOURNAL AT THE END OF THIS CALENDAR YEAR. THUS, WHETHER OR NOT PUBLICATION CONTINUES WILL BE UP TO YOU. THE FIELD OF COMPUTER'S FOR INDIVIDUALS IS STILL IN IT'S INFANCY. WE KNOW THAT ALMOST EVERY READER OUT THERE HAS SOMETHING TO SHARE WITH HIS/HER FELLOW COMPUTER ENTHUSIASTS. THE AIM OF THIS PUBLICATION HAS BEEN TO PROVIDE A MEDIUM FOR THAT EXCHANGE. PLEASE USE IT AS SUCH. THOSE LITTLE GENERAL OR SPECIAL PURPOSE ROUTINES THAT YOU HAVE STRUGGLED TO "CREATE" AND GET OPERATIONAL MAY BE OF USE TO OTHERS. YOUR SHARING THEM CAN STOP THE PROCESS OF "RE-CREATING THE WHEEL" FOR EACH AND EVERY OTHER NEW-COMER. WHEN THAT PROCESS IS ELIMINATED, PEOPLE WILL HAVE MORE TIME TO CREATE NEW AND MORE POWERFUL PROGRAMS. THERE IS PLENTY OF ROOM IN THE FIELD FOR ROUTINES, ARTICLES, AND CIRCUITS AT ALL LEVELS OF SOPHISTICATION. WHY NOT HELP THE FIELD ADVANCE AS YOU HELP YOURSELF?

ONE OF OUR MOST ACTIVE CONTRIBUTORS (AND ONE OF OUR EARLIEST CUSTOMERS!) IS DR. GEORGE L. HALLER (SUMMER ADDRESS: HOUND EARS CLUB, BLOWING ROCK, N.C. 28605). DR. HALLER CONTRIBUTED SEVERAL ITEMS TO THE LAST ISSUE AND HE HAS DONE IT AGAIN. FIRST, ON THE NEXT PAGE YOU WILL FIND A HANDY TABLE HE HAS PREPARED FOR CONVERTING OCTAL NUMBERS TO DOUBLE PRECISION NUMBERS. IT SHOULD BE VERY HELPFUL TO THOSE THAT ARE WORKING WITH MATHEMATICAL ROUTINES.

THEN, ON THE NEXT SEVERAL PAGES YOU WILL FIND HIS SOLUTION TO INTERFACING THE POPULAR "TELEVISION TYPEWRITER" (TVT) TO THE SCELBI-8H ALONG WITH SEVERAL ROUTINES TO DRIVE THE TVT. (THIS SHOULD PLEASE A NUMBER OF READERS WHO HAVE EXPRESSED AN INTEREST IN THIS "HOOK-UP.") THANK YOU ONCE AGAIN DR. HALLER. WE HOPE READERS WILL RECIPROCATE WITH SOME DATA/ROUTINES/CIRCUITS THAT YOU CAN USE!

A NEW CONTRIBUTOR THIS TIME IS MR. T. F. CALDWELL. (ADDRESS: PO BOX 116, BURGESS, VA 22432.) MR. CALDWELL JUST RECENTLY JOINED OUR USER'S GROUP AND HAS A SCELBI-8H TO WHICH HE WANTS TO ADD A TTY, TVT (NOTE THE TVT INTERFACE IN THIS ISSUE MR. CALDWELL!) AND MAG-TAPE UNIT. MR. CALDWELL CONSTRUCTED HIS COMPUTER FROM THE KIT VERSION AND HAS MADE UP A LITTLE "AUDIBLE" CIRCUIT TESTER TO HELP IN CHECKING OUT THE CHASSIS WIRING ETC.. THE CIRCUIT SURE LOOKS SIMPLE AND INEXPENSIVE AND PERHAPS SOME OF YOU WILL FIND IT AS USEFUL AS HE DOES. THE CIRCUIT IS SHOWN IMMEDIATELY FOLLOWING DR. HALLER'S TVT PROGRAM ON PAGE 5. THANK YOU ON BEHALF OF THE READERS MR. CALDWELL.

A TABLE OF DOUBLE PRECISION WORDS FOR AN 8 BIT MACHINE

If we wish to use numbers higher than 255 decimal in computer calculations using 8 bit words we find that we must concatenate two or more words which is called double or higher precision. Suppose we have the decimal number 2783, we would find that the binary equivalent is;

1 0 1 0 1 1 0 1 1 1 1 1 which in octal is 5337.

Now if we split this into two words of 8 bits each it would be

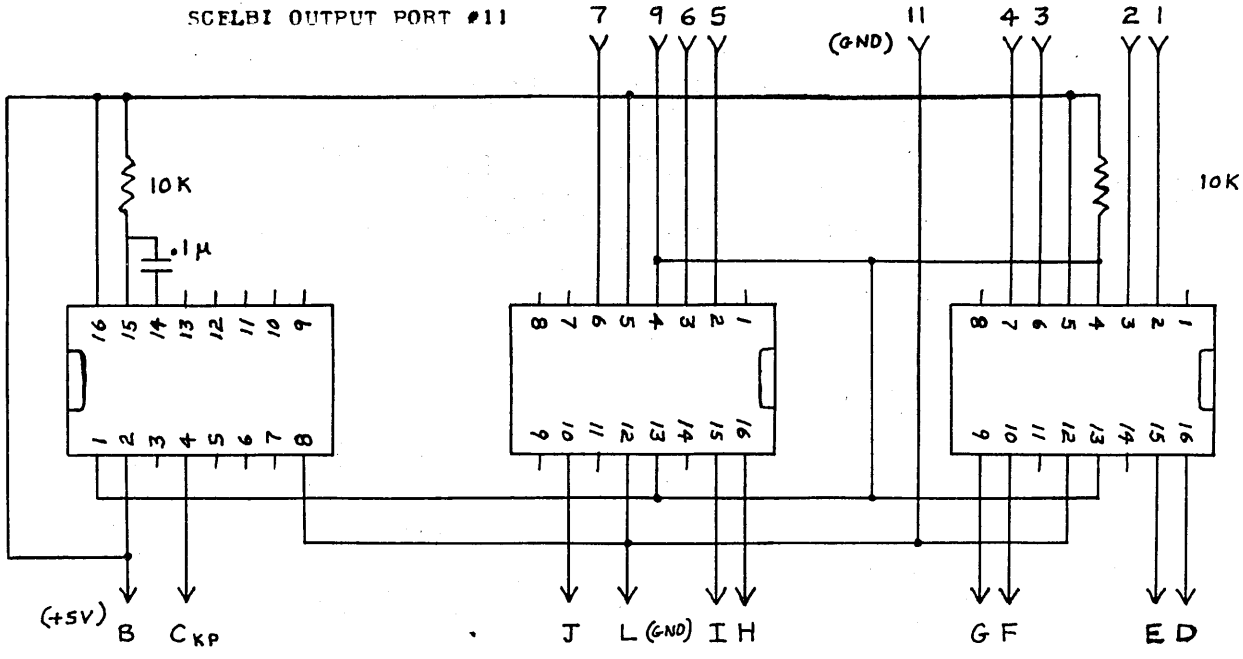
0 0 0 0 1 0 1 0 and 1 1 0 1 1 1 1 1

Note that the bit arrangement is the same but the octal is now 012 and 337. The first number is called the high order and the second number is called the low order of the double precision number.

Tables of decimal to octal numbers are found in many computer texts and following is the table of octal to double precision numbers.

00XX = 000 0XX	30XX = 006 0XX	60XX = 014 0XX
01XX = 000 1XX	31XX = 006 1XX	61XX = 014 1XX
02XX = 000 2XX	32XX = 006 2XX	62XX = 014 2XX
03XX = 000 3XX	33XX = 006 3XX	63XX = 014 3XX
04XX = 001 0XX	34XX = 007 0XX	64XX = 015 0XX
05XX = 001 1XX	35XX = 007 1XX	65XX = 015 1XX
06XX = 001 2XX	36XX = 007 2XX	66XX = 015 2XX
07XX = 001 3XX	37XX = 007 3XX	67XX = 015 3XX
10XX = 002 0XX	40XX = 010 0XX	70XX = 016 0XX
11XX = 002 1XX	41XX = 010 1XX	71XX = 016 1XX
12XX = 002 2XX	42XX = 010 2XX	72XX = 016 2XX
13XX = 002 3XX	43XX = 010 3XX	73XX = 016 3XX
14XX = 003 0XX	44XX = 011 0XX	74XX = 017 0XX
15XX = 003 1XX	45XX = 011 1XX	75XX = 017 1XX
16XX = 003. 2XX	46XX = 011 2XX	76XX = 017 2XX
17XX = 003 3XX	47XX = 011 3XX	77XX = 017 3XX
20XX = 004 0XX	50XX = 012 0XX	
21XX = 004 1XX	51XX = 012 1XX	* Example shown above.
22XX = 004 2XX	52XX = 012 2XX	
23XX = 004 3XX	53XX = 012 3XX *	
24XX = 005 0XX	54XX = 013 0XX	
25XX = 005 1XX	55XX = 013 1XX	
26XX = 005 2XX	56XX = 013 2XX	
27XX = 005 3XX	57XX = 013 3XX	

George L. Haller



TVT KEYBOARD EDGE CONNECTOR

"A" NEAREST RF TWIN LEAD

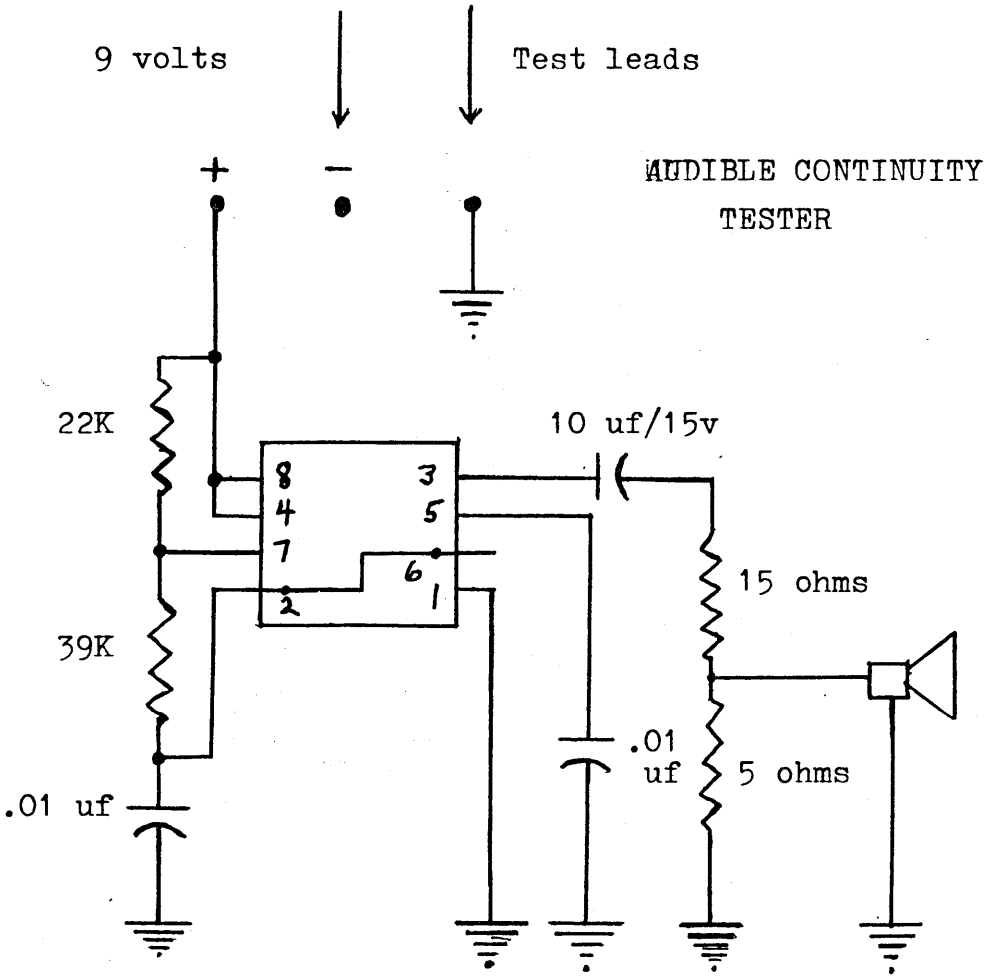
"L" NEAREST J1

10 UFD CAPACITOR (SHOWN AS C17 ON CURSOR DIAGRAM - WHICH IS C16 ON CURSOR BOARD) CHANGED TO 0.68 UFD.

+5 VOLTS TAKEN FROM TVT

DR. HALLER'S TVT INTERFACE

```
                        /
                        /DR. HALLER'S TVT DRIVER PROGRAM
                        /
                        /TVT WILL DISPLAY AS MEMORY IS
                        /BEING LOADED.
                        /
                        /MEMORY CAN BE DUMPED TO TVT BY
                        /STARTING PROGRAM AT LOCATION 042
                        /
                        /USE THE "RO" KEY FOR A HALT
                        /
                        /PROGRAM RESIDES ON PAGE 03
                        /CHARACTER STORAGE ON PAGE 05
                        /AND UP
                        /
                        ORG 003 000
                        /
003 000  056 005        INIT, LHI 05          /SET PNTR TO PAGE 05
003 002  066 000        LLI 000
003 004  125            NXCHR, OUT 12         /SET UP KEYBOARD
003 005  111            KEYIN, INP 4          /INPUT FM KEYBOARD
003 006  240            NDA                   /SET STATUS FLAGS
003 007  120 005 003    JFS KEYIN             /IF NO CHAR GO BACK
003 012  370            LMA
003 013  060            INL
003 014  110 020 003    JFZ AHEAD             /IF "L" NOT 0 JUMP AHEAD
003 017  050            INH                   /IF "L" = 0, ADV "H"
003 020  106 026 003    AHEAD, CAL OUTPT      /GO TO OUTPUT ROUTINE
003 023  104 004 003    JMP NXCHR             /GET NEXT CHAR FM KEYBD
                        /
003 026  123            OUTPT, OUT 11         /OUTPUT TO TVT
003 027  026 004        LCI 004               /LOOP COUNTER
003 031  031            DELAY, DCD            /TIME DELAY
003 032  110 031 003    JFZ DELAY
003 035  021            DCC
003 036  110 031 003    JFZ DELAY
003 041  007            RET
                        /
003 042  056 005        DSPLY, LHI 005        /MEMORY DISPLAY ROUTINE
003 044  066 000        LLI 000               /SET PNTR
003 046  307            NXBYT, LAM
003 047  074 377        CPI 377               /IS CHAR A "377" ?
003 051  150 067 003    JTZ DONE              /HALT IF CHAR IS "377"
003 054  106 026 003    CAL OUTPT             /IF NOT "377"
003 057  060            INL                   /OUTPUT TO TVT, THEN
003 060  110 064 003    JFZ AGAIN             /ADVANCE PNTR(S)
003 063  050            INH
003 064  104 046 003    AGAIN, JMP NXBYT      /OUTPUT NEXT CHARACTER
                        /
003 067  377            DONE, 377             /HALT - END OF ROUTINE
                        /
                        END
```



IC is 555 Timer

Values shown give about 700 hz at 30% duty cycle

The low-value resistors in the speaker circuit could be replaced with a 20-ohm pot if volume control is desired.

There is nine volts between the test leads, therefore the Tester should be used for continuity testing only.

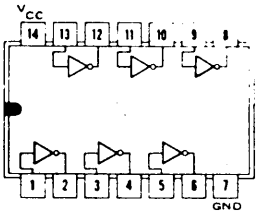
MR. CALDWELL'S CIRCUIT CONTINUITY TESTER

MORE ROUTINES FOR THE SCELBI INTEGRATED CIRCUIT TESTER PERIPHERAL

IN THE APRIL, 1975 ISSUE OF "THE SCELBI COMPUTER DIGEST AND USER'S BULLETIN" AN INTEGRATED CIRCUIT TESTER WAS DESCRIBED. SEVERAL SAMPLE ROUTINES FOR DRIVING THE TESTER TO TEST COMMON TTL DEVICES SUCH AS THE 7400 AND 7402 DEVICES WERE PROVIDED IN THAT ARTICLE. IN THIS ISSUE, WE ARE PRESENTING SEVERAL MORE ROUTINES FOR OTHER TYPES OF TTL DEVICES.

THE FIRST ROUTINE TO BE PRESENTED HERE IS FOR THE TYPE 7404 TTL INVERTER PACKAGE. A DIAGRAM OF THE DEVICE AND IT'S PIN ASSIGNMENTS ARE SHOWN BELOW. THE DEVICE IS SIMPLY A PACKAGE OF SIX TTL INVERTERS. THE FOLLOWING ROUTINE WILL TEST THE DEVICE AS WELL AS IT'S "OPEN COLLECTOR" EQUIVALENTS SUCH AS THE 7416.

FOR THIS DEVICE, POWER SHOULD BE CONNECTED TO TEST POINTS 7 (COMMON) AND TP16 (+5 VOLTS). REMEMBER TO LEAVE SWITCHES 7 AND 16 OPEN FOR THE POWER CONNECTION POINTS. SWITCHES 1, 3, 5, 11, 13 AND 15 ON THE TESTER SHOULD BE "ON" (CLOSED) TO PROVIDE INPUTS TO THE DEVICE UNDER TEST. ALL OTHER SWITCHES SHOULD BE "OFF" (OPEN).



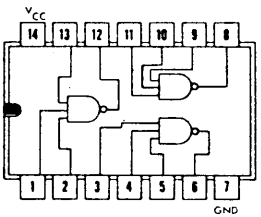

```
                        /
                        /7404 I.C. TEST
                        /
                        ORG 000 270
000 270  016 000        B7404, LBI 0            /SET TEST CNTR
000 272  006 250        D7404, LAI 250          /SEND 1'S TO INVERTER 1,2,3
000 274  121            OUT 10
000 275  101            INP 0                   /GET RESULTS (0)
000 276  044 124        NDI 124                 /TEST
000 300  110 350 000    JFZ BAD
000 303  006 052        LAI 52                  /SEND 1 TO INVERTER 4,5,6
000 305  123            OUT 11
000 306  103            INP 1                   /GET RESULTS  (0)
000 307  044 124        NDI 124                 /TEST
000 311  110 350 000    JFZ BAD
000 314  006 000        LAI 0                   /SEND 0 TO INVERTERS 1,2,3
000 316  121            OUT 10
000 317  101            INP 0                   /GET RESULTS  (1)
000 320  044 124        NDI 124                 /TEST
000 322  054 124        XRI 124
000 324  110 350 000    JFZ BAD
000 327  123            OUT 11                  /SEND 0 TO INVERTERS 4,5,6
000 330  103            INP 1
000 331  044 124        NDI 124                 /TEST
000 333  054 124        XRI 124
000 335  110 350 000    JFZ BAD
000 340  010            INB                     /SEE IF TEST IS FINISHED
000 341  110 272 000    JFZ D7404
```

```
000 344   000           ALDONE, 0              /DUT PASSED - LIGHTS OUT
000 345   104 270 000   JMP B7404
000 350   377           BAD, 377               /DUT FAILED - LIGHTS ON
000 351   104 270 000   JMP B7404
                        END
```

A VERY SIMILAR PROGRAM CAN BE USED TO TEST 7407 "BUFFERS" AND THEIR "OPEN COLLECTOR" EQUIVALENTS SUCH AS THE 7417. FOR THE FOLLOWING PROGRAM LEAVE THE SWITCHES AND POWER CONNECTIONS EXACTLY AS FOR THE PREVIOUS PROGRAM FOR 7404 DEVICES AND SUBSTITUTE THE FOLLOWING TESTER PROGRAM.

```
                        /
                        /7417 I.C. TEST
                        /
                        ORG 001 130
001 130   016 000       B7417, LBI 0           /SET TEST CNTR
001 132   006 250       D7417, LAI 250         /SEND 1'S TO BUFFERS 1,2,3
001 134   121           OUT 10
001 135   101           INP 0                  /GET RESULTS  (1)
001 136   044 124       NDI 124
001 140   054 124       XRI 124                /TEST
001 142   110 210 001   JFZ BAD
001 145   006 052       LAI 52                 /SEND 1 TO BUFFER 4,5,6
001 147   123           OUT 11
001 150   103           INP 1                  /GET RESULTS  (1)
001 151   044 124       NDI 124
001 153   054 124       XRI 124                /TEST
001 155   110 210 001   JFZ BAD
001 160   006 000       LAI 0                  /SEND 0 TO BUFFER 1, 2 & 3
001 162   121           OUT 10
001 163   101           INP 0                  /GET RESULTS  (0)
001 164   044 124       NDI 124                /TEST
001 166   110 210 001   JFZ BAD
001 171   123           OUT 11                 /SEND 0 TO BUFFER 1, 2 & 3
001 172   103           INP 1                  /GET RESULTS  (0)
001 173   044 124       NDI 124                /TEST
001 175   110 210 001   JFZ BAD
001 200   010           INB                    /SEE IF TEST IS FINISHED
001 201   110 132 001   JFZ D7417
001 204   000           ALDONE, 0              /DUT PASSED - LIGHTS OUT
001 205   104 130 001   JMP B7417
001 210   377           BAD, 377               /DUT FAILED - LIGHTS ON
001 211   104 130 001   JMP B7417
                        END
```

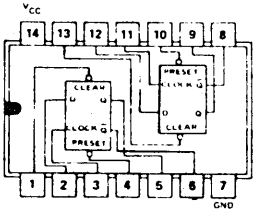
THE NEXT PROGRAM IS FOR TESTING THE TYPE 7410 TRIPLE THREE-INPUT "NAND" GATE. A DIAGRAM OF THE DEVICE WITH IT'S PIN ASSIGNMENTS IS ILLUSTRATED HERE:



```
                        /
                        /7410 I.C. TEST
                        /
                        ORG 000 360
000 360  016 000        B7410, LBI 000      /SET TEST CNTR
000 362  006 370        D7410, LAI 370      /SEND 111 TO GATES 1,2,3
000 364  121            OUT 10
000 365  006 072        LAI 072
000 367  123            OUT 11
000 370  101            INP 0               /GET RESULTS FM #2 (0)
000 371  044 004        NDI 004             /TEST
000 373  110 116 001    JFZ BAD
000 376  103            INP 1               /GET RESULTS FM 1 & 3 (0)
000 377  044 104        NDI 104             /TEST
001 001  110 116 001    JFZ BAD
001 004  006 360        LAI 360             /SEND 110 TO GATES 1,2,3
001 006  121            OUT 10
001 007  006 060        LAI 060
001 011  123            OUT 11
001 012  101            INP 0               /GET RESULTS FM #2 (1)
001 013  044 004        NDI 004
001 015  054 004        XRI 004             /TEST
001 017  110 116 001    JFZ BAD
001 022  103            INP 1               /GET RESULTS FM 1 & 3 (1)
001 023  044 104        NDI 104
001 025  054 104        XRI 104             /TEST
001 027  110 116 001    JFZ BAD
001 032  006 250        LAI 250             /SEND 101 TO GATES 1,2,3
001 034  121            OUT 10
001 035  006 052        LAI 052
001 037  123            OUT 11
001 040  101            INP 0               /GET RESULTS FM #2 (1)
001 041  044 004        NDI 004
001 043  054 004        XRI 004             /TEST
001 045  110 116 001    JFZ BAD
001 050  103            INP 1               /GET RESULTS FM 1 & 3 (1)
001 051  044 104        NDI 104
001 053  054 104        XRI 104             /TEST
001 055  110 116 001    JFZ BAD
001 060  006 130        LAI 130             /SEND 011 TO GATES 1,2,3
001 062  121            OUT 10
001 063  006 032        LAI 032
001 065  123            OUT 11
001 066  101            INP 0               /GET RESULTS FM # 2 (1)
001 067  044 004        NDI 004
001 071  054 004        XRI 004             /TEST
001 073  110 116 001    JFZ BAD
001 076  103            INP 1               /GET RESULTS FM 1 & 3 (1)
001 077  044 104        NDI 104
001 101  054 104        XRI 104             /TEST
001 103  110 116 001    JFZ BAD
001 106  010            INB                 /SEE IF FINISHED
001 107  110 362 000    JFZ D7410
001 112  000            ALDONE, 0           /DUT PASSED - LIGHTS OUT
001 113  104 360 000    JMP B7410
001 116  377            BAD, 377            /DUT FAILED - LIGHTS ON
001 117  104 360 000    JMP B7410
                        /
                        END
```

WHEN USING THE 7410 TESTER PROGRAM SHOWN ON THE PREVIOUS PAGE POWER IS CONNECTED TO TP 7 (COMMON) AND TP16 (+5 VOLTS). SWITCHES 1, 2, 3, 4, 12, 13, 14 AND 15 ARE "ON" (CLOSED) TO PROVIDE INPUTS TO THE DEVICE UNDER TEST. ALL OTHER SWITCHES ARE "OPEN" (OFF).

THE FINAL TESTER PROGRAM TO BE ILLUSTRATED IN THIS ISSUE IS A MORE COMPLICATED ONE THAN THE ONES ILLUSTRATED FOR GATES, INVERTERS, AND BUFFERS. THIS PROGRAM IS FOR TESTING A "FLIP-FLOP" DEVICE. SUCH A DEVICE MUST BE "CLOCKED" ON ONE PIN WHILE A SIGNAL "INPUT" IS APPLIED TO ANOTHER, AND THEN A TEST MADE TO ASCERTAIN THAT THE "OUTPUT" FROM THE DEVICE IS PROPER AFTER THE "CLOCKING" HAS OCCURED. THE PROGRAM TO BE ILLUSTRATED IS FOR A TYPE 7474 DUAL FLIP-FLOP. A DIAGRAM ILLUSTRATING THE PIN ASSIGNMENTS FOR THE DEVICE IS SHOWN NEXT.



IN ADDITION TO THE "CLOCKING" MODE OF A STANDARD "FLIP-FLOP," THIS DEVICE ALSO HAS "PRESET" AND "CLEAR" SIGNALS THAT MAY BE USED TO CONDITION THE OUTPUT SIGNAL. THE FOLLOWING PROGRAM ALSO TESTS THAT THE "PRESET" AND "CLEAR" LINES OPERATE PROPERLY.

TO OPERATE THE FOLLOWING PROGRAM, POWER TO THE "DUT" IS PROVIDED AT TP 7 (COMMON) AND TP 16 (+5 VOLTS). SWITCHES 1, 2, 3, 4, 12, 13, 14 AND 15 ARE TURNED "ON." ALL OTHER SWITCHES SHOULD BE "OFF." THE READER MAY NOTICE THAT THE ROUTINE UTILIZES A NUMBER OF SUBROUTINES TO REDUCE THE TOTAL AMOUNT OF MEMORY SPACE USED BY THE PROGRAM.

```
                        /
                        /7474 I.C. TEST PROGRAM
                        /
                        ORG 004 000
004 000   016 000       B7474, LBI 0      /SET TEST CNTR
004 002   006 220       D7474, LAI 220    /SET D = 0
004 004   121           OUT 10
004 005   006 260       LAI 260           /CLOCK IT IN
004 007   106 136 004   CAL ONENOT
004 012   006 320       LAI 320           /SET D = 1
004 014   121           OUT 10
004 015   006 360       LAI 360           /CLOCK IT IN
004 017   106 116 004   CAL ONETRU
004 022   006 160       LAI 160
004 024   121           OUT 10
004 025   006 360       LAI 360           /NEGATIVE PULSE ON CLEAR LINE
004 027   106 136 004   CAL ONENOT
004 032   006 340       LAI 340
004 034   121           OUT 10
004 035   006 360       LAI 360           /NEGATIVE PULSE ON SET LINE
004 037   106 116 004   CAL ONETRU
004 042   006 022       LAI 022           /SET D = 0
004 044   123           OUT 11
```

```
004 045  006 032       LAI 032          /CLOCK IT IN
004 047  106 156 004   CAL TWONOT
004 052  006 026       LAI 026          /SET D = 1
004 054  123            OUT 11
004 055  006 036       LAI 036          /CLOCK IT IN
004 057  106 176 004   CAL TWOTRU
004 062  006 034       LAI 034
004 064  123            OUT 11
004 065  006 036       LAI 036          /NEGATIVE PULSE ON CLEAR LINE
004 067  106 156 004   CAL TWONOT
004 072  006 016       LAI 016
004 074  123            OUT 11
004 075  006 036       LAI 036          /NEGATIVE PULSE ON SET LINE
004 077  106 176 004   CAL TWOTRU
004 102  010            INB
004 103  110 002 004   JFZ D7474        /SEE IF FINISHED TEST
004 106  000            ALDONE, 0        /DUT PASSED - LIGHTS OUT
004 107  104 000 004   JMP B7474
004 112  377            BAD, 377         /DUT FAILED - LIGHTS ON
004 113  104 000 004   JMP B7474
004 116  121            ONETRU, OUT 10
004 117  101            INP 0            /GET RESULTS
004 120  044 010       NDI 010
004 122  054 010       XRI 010          /Q = 1
004 124  110 112 004   JFZ BAD
004 127  101            INP 0
004 130  044 004       NDI 004          /QN = 0
004 132  110 112 004   JFZ BAD
004 135  007            RET
004 136  121            ONENOT, OUT 10
004 137  101            INP 0
004 140  044 010       NDI 010          /NOW Q = 0
004 142  110 112 004   JFZ BAD
004 145  101            INP 0
004 146  044 004       NDI 004
004 150  054 004       XRI 004          /AND QN = 1
004 152  110 112 004   JFZ BAD
004 155  007            RET
004 156  123            TWONOT, OUT 11
004 157  103            INP 1
004 160  044 040       NDI 040          /Q = 0
004 162  110 112 004   JFZ BAD
004 165  103            INP 1
004 166  044 100       NDI 100
004 170  054 100       XRI 100          /QN = 1
004 172  110 112 004   JFZ BAD
004 175  007            RET
004 176  123            TWOTRU, OUT 11
004 177  103            INP 1
004 200  044 040       NDI 040
004 202  054 040       XRI 040          /Q = 1
004 204  110 112 004   JFZ BAD
004 207  103            INP 1
004 210  044 100       NDI 100          /QN = 0
004 212  110 112 004   JFZ BAD
004 215  007            RET
                        /
                        END
```

LAA	300	LMA	370	SBA	230	RET	0X7
LAB	301	LMB	371	SBB	231		
LAC	302	LMC	372	SBC	232		
LAD	303	LMD	373	SBD	233	RFC	003
LAE	304	LME	374	SBE	234	RFZ	013
LAH	305	LMH	375	SBH	235	RFS	023
LAL	306	LML	376	SBL	236	RFP	033
LAM	307			SBM	237	RTC	043
						RTZ	053
LBA	310	LAI	006 DDD	CPA	270	RTS	063
LBB	311	LBI	016 DDD	CPB	271	RTP	073
LBC	312	LCI	026 DDD	CPC	272		
LBD	313	LDI	036 DDD	CPD	273		
LBE	314	LEI	046 DDD	CPE	274	JMP	1X4 LLL HHH
LBH	315	LHI	056 DDD	CPH	275		
LBL	316	LLI	066 DDD	CPL	276	JFC	100 LLL HHH
LBM	317	LMI	076 DDD	CPM	277	JFZ	110 LLL HHH
						JFS	120 LLL HHH
LCA	320			NDA	240	JFP	130 LLL HHH
LCB	321	INB	010	NDB	241	JTC	140 LLL HHH
LCC	322	INC	020	NDC	242	JTZ	150 LLL HHH
LCD	323	IND	030	NDD	243	JTS	160 LLL HHH
LCE	324	INE	040	NDE	244	JTP	170 LLL HHH
LCH	325	INH	050	NDH	245		
LCL	326	INL	060	NDL	246		
LCM	327			NDM	247	CAL	1X6 LLL HHH
LDA	330			ORA	260	CFC	102 LLL HHH
LDB	331	DCB	011	ORB	261	CFZ	112 LLL HHH
LDC	332	DCC	021	ORC	262	CFS	122 LLL HHH
LDD	333	DCD	031	ORD	263	CFP	132 LLL HHH
LDE	334	DCE	041	ORE	264	CTC	142 LLL HHH
LDH	335	DCH	051	ORH	265	CTZ	152 LLL HHH
LDL	336	DCL	061	ORL	266	CTS	162 LLL HHH
LDM	337			ORM	267	CTP	172 LLL HHH
LEA	340	ADA	200	XRA	250		
LEB	341	ADB	201	XRB	251	RST 0	005
LEC	342	ADC	202	XRC	252	RST 1	015
LED	343	ADD	203	XRD	253	RST 2	025
LEE	344	ADE	204	XRE	254	RST 3	035
LEH	345	ADH	205	XRH	255	RST 4	045
LEL	346	ADL	206	XRL	256	RST 5	055
LEM	347	ADM	207	XRM	257	RST 6	065
						RST 7	075
LHA	350	ACA	210	ADI	004 DDD		
LHB	351	ACB	211	ACI	014 DDD		
LHC	352	ACC	212	SUI	024 DDD	INP 0	101
LHD	353	ACD	213	SBI	034 DDD	INP 1	103
LHE	354	ACE	214	NDI	044 DDD	INP 2	105
LHH	355	ACH	215	XRI	054 DDD	INP 3	107
LHL	356	ACL	216	ORI	064 DDD	INP 4	111
LHM	357	ACM	217	CPI	074 DDD	INP 5	113
LLA	360	SUA	220	RLC	002	OUT 10	121
LLB	361	SUB	221	RRC	012	OUT 11	123
LLC	362	SUC	222	RAL	022	OUT 12	125
LLD	363	SUD	223	RAR	032	OUT 13	127
LLE	364	SUE	224			OUT 14	131
LLH	365	SUH	225	HLT	000	OUT 15	133
LLL	366	SUL	226	"	001	OUT 16	135
LLM	367	SUM	227	"	377	OUT 17	137

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 H I G H ADDRESS (PAGE).

THE "L" BIT IS SET TO A ONE IF THE "DATA BITS" ARE TO CONTAIN A L O W 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
X 0 0 0 3 2 1 0     DATA BITS = PAGE ADDRESS

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

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

          .
          .
          .              D A T A
          .
          .

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

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 WLDI            /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
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
017 353  032          RAR                  /SHIFT BITS TO MAKE RDY
017 354  340          LEA                  /FOR NEXT INCOMING BIT
017 355  016 024      LBI 024              /SET 1 BIT DELAY
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 CON-    *
*  TROL 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 RE-   *
*  MOVED OR INSERTED, TRANSIENT SIGNALS CAN OCCUR THAT WILL TRIGGER     *
*  THE RELAY.  THE SOLUTION TO THE PROBLEM IS A SIMPLE ONE.  SIMPLY     *
*  CONNECT A SEPARATE  P E R M A N E N T  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 AT-    *
*  TACHED 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.              *
*                                                                       *
*************************************************************************
```

WHAT IS A "BOOTSTRAP LOADER" ?

SOME NEWCOMERS TO THE FIELD OF COMPUTERS HAVE WRITTEN TO ASK ABOUT DEFINITIONS OF VARIOUS TERMS THEY HAVE HEARD ABOUT. ONE OF THE MOST COMMON QUESTIONS REFERS TO THE DEFINITION OF A "BOOTSTRAP LOADER PROGRAM." THE FOLLOWING IS A BRIEF DISCUSSION ON THE SUBJECT.

THE TERM "BOOTSTRAP LOADER" IS SOMEWHAT NEBULOUS. IT CAN REFER TO A VARIETY OF PROGRAMS, BUT THE COMMON DENOMINATOR AMONG THEM RELATES TO THE FOLLOWING CONCEPT. A "BOOTSTRAP LOADER" IS LITERALLY A PROGRAM THAT ENABLES A COMPUTER TO USE ITSELF TO LOAD IN A MORE POWERFUL PROGRAM. THE VERNACULAR ACTUALLY COMES FROM THE CONCEPT OF "LIFTING ONESELF UP BY ONE'S OWN BOOTSTRAPS!"

AS MOST READERS KNOW, WHEN A COMPUTER IS INITIALLY POWERED UP IT'S MEMORY CIRCUITS WILL BE IN A STATE OF DISARRAY. THE BINARY BITS THAT MAKE UP EACH WORD IN MEMORY WILL BE IN ESSENTIALLY RANDOM STATES OF EITHER A LOGIC "1" OR "0." IF A USER WERE TO PUT THE COMPUTER IN THE "RUN" MODE IMMEDIATELY FOLLOWING POWER TURN-ON, IT IS HIGHLY UNLIKELY THAT THE COMPUTER WOULD DO ANYTHING USEFUL. THE RANDOM STATES OF THE WORDS IN MEMORY WOULD BE INTERPRETED AS COMPLETELY RANDOM INSTRUCTIONS IF ANYTHING AT ALL.

THUS, THE FIRST THING THAT MUST BE ACCOMPLISHED WHEN A COMPUTER IS INITIALLY POWERED UP IS TO PLACE SOME KIND OF PROGRAM INTO IT'S MEMORY. FOR THE SMALL COMPUTER OWNER, THIS PROCESS IS USUALLY ACCOMPLISHED BY UTILIZING MANUAL METHODS TO "ADDRESS" INDIVIDUAL WORDS IN MEMORY AND PLACE SOME DESIRED "INSTRUCTIONS" INTO THOSE WORDS. IN OTHER WORDS, TO MANUALLY LOAD IN A "PROGRAM."

NOW, MOST PROGRAMS THAT NEED TO BE LOADED INTO MEMORY IN ORDER TO HAVE THE COMPUTER PERFORM SOMETHING USEFUL ARE LIKELY TO BE OF CONSIDERABLE LENGTH. THAT IS, THEY MAY CONSIST OF SEVERAL HUNDRED TO SEVERAL THOUSAND INSTRUCTIONS. IT WOULD BE A RATHER TEDIOUS PROCESS IF EACH TIME ONE TURNED ON A COMPUTER, THEY HAD TO MANUALLY "TOGGLE IN," SAY BY CONSOLE SWITCHES, SEVERAL THOUSAND INSTRUCTIONS INTO THE COMPUTER'S MEMORY. (SEVERAL HUNDRED INSTRUCTIONS IS TEDIOUS ENOUGH!)

NATURALLY, THERE IS A BETTER WAY. MOST COMPUTERS ARE CONNECTED TO I/O (INPUT/OUTPUT) DEVICES WHICH CAN BE USED TO AUTOMATE THE PROCESS OF "LOADING" PROGRAMS INTO MEMORY - AS WELL AS SERVING AS A COMMUNICATIONS MEDIUM BETWEEN THE OPERATOR AND A COMPUTER PROGRAM. HOWEVER, THOSE "I/O" DEVICES THEMSELVES REQUIRE SOME SORT OF "PROGRAM" IN ORDER TO EFFECTIVELY COMMUNICATE WITH THE COMPUTER. HOW COMPLEX A PROGRAM - AND THUS HOW LENGTHY, GENERALLY DEPENDS ON WHAT KINDS OF FUNCTIONS ARE GOING TO BE PERFORMED.

A "BOOTSTRAP PROGRAM," IN GENERAL TERMS IS A "MINIMUM LENGTH" PROGRAM THAT CAN BE PLACED IN MEMORY IN ORDER TO ALLOW AN INPUT DEVICE TO AUTOMATICALLY LOAD A MORE COMPLEX PROGRAM. OFTEN, IN ORDER TO SATISFY THE NEED FOR A MINIMUM LENGTH PROGRAM, THE BOOTSTRAP LOADER WILL NOT FULLY UTILIZE THE CAPABILITY OF THE INPUT DEVICE. FOR EXAMPLE, THE INPUT DEVICE MIGHT BE CAPABLE OF SENDING SAY EIGHT BITS OF INFORMATION IN ONE OPERATION TO THE COMPUTER. HOWEVER, IT MIGHT REQUIRE MORE INSTRUCTIONS IN A PROGRAM FOR THE COMPUTER TO ACCEPT THE EIGHT BITS IN ONE OPERATION THAN IF JUST ONE BIT WAS ACCEPTED PER OPERATION OF THE INPUT DEVICE. THE TRADE-OFF MIGHT BE FOR INSTANCE, THE OVER-ALL SPEED AT WHICH THE INPUT DEVICE COULD OPERATE. IT WOULD BE MUCH FASTER IF IT USED THE MODE WHERE IT SENT EIGHT BITS IN ONE OPERATION INSTEAD OF JUST ONE BIT. ANOTHER AREA THAT IS OFTEN SACRIFICED IN USING A "BOOTSTRAP

LOADER" PROGRAM IS OFTEN RELATED TO WHERE IN MEMORY DATA WILL BE LOADED. FOR INSTANCE, AN INPUT DEVICE, BY ONE SCHEME OR ANOTHER, MAY BE ABLE TO LOAD VIRTUALLY ANY "ADDRESS" IN MEMORY BY, FOR INSTANCE, SENDING A SPECIFIC "CODE" THAT WOULD CAUSE THE COMPUTER TO ALTER THE ADDRESS AT WHICH IT WAS DEPOSITING DATA FROM THE DEVICE. HOWEVER, ADDING THIS CAPABILITY OFTEN REQUIRES A MORE SOPHISTICATED "LOADER PROGRAM." A "BOOTSTRAP LOADER" MIGHT ELIMINATE THIS FEATURE TO SHORTEN THE PROGRAM BY ASSUMING THAT WHATEVER IS LOADED WILL BE DEPOSITED IN A SPECIFIC AREA IN MEMORY (THAT MIGHT BE DETERMINED BY THE OPERATOR'S INITIALIZING COMMANDS).

THUS, A "BOOTSTRAP LOADER" PROGRAM IS OFTEN USED TO ENABLE THE COMPUTER TO "LIFT ITSELF UP" FROM A VERY SMALL PROGRAM TO A MUCH LARGER PROGRAM. A TYPICAL SEQUENCE OF EVENTS FOR GETTING A COMPUTER OPERATIONAL AFTER POWER TURN-ON MIGHT BE AS FOLLOWS:

1.) POWER IS INITIALLY TURNED ON. MEMORY IS IN RANDOM STATE AND COMPUTER IS ESSENTIALLY "FUNCTION-LESS."

2.) OPERATOR USES MANUAL CONSOLE SWITCHES TO LOAD IN A "MINIMAL LENGTH" BOOTSTRAP LOADER PROGRAM. THIS BOOTSTRAP LOADER PROGRAM WILL PROVIDE SOME SORT OF "LOW-LEVEL" INEFFICIENT INPUT CAPABILITY FROM AN INPUT DEVICE TO SPEED UP THE PROGRAM LOADING PROCESS OVER THAT ALLOWED BY MANUAL MEANS.

3.) THE "BOOTSTRAP LOADER PROGRAM" IS THEN USED TO ALLOW THE INPUT DEVICE TO LOAD IN, FOR INSTANCE, A MUCH MORE EFFICIENT "GENERAL PURPOSE" LOADER PROGRAM. THIS GENERAL LOADER PROGRAM IS TYPICALLY MUCH FASTER AND HAS MORE CAPABILITY.

4.) THE "GENERAL PURPOSE" LOADER PROGRAM IS THEN USED TO LOAD IN A DESIRED OPERATING PROGRAM SUCH AS, FOR INSTANCE, AN EDITOR, ASSEMBLER, CALCULATOR OR SPECIAL PURPOSE PROGRAM.

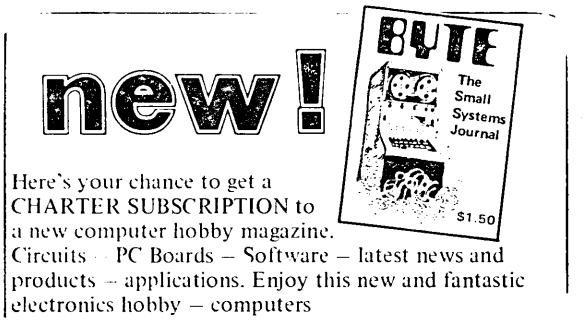
SO, ACTUALLY, TO SUMMARIZE, THE PROCESS IS REALLY ONE OF THE COMPUTER "HELPING ITSELF." A MINIMAL LENGTH PROGRAM IS USED TO LOAD IN A LONGER MORE SOPHISTICATED "LOADER" AND THAT IN TURN IS USED TO LOAD IN A LARGER GENERAL PURPOSE PROGRAM. THE COMPUTER USES ITS OWN "POWER" TO BUILD UP IT'S TOTAL CAPABILITY!

A "BOOTSTRAP LOADER" PROGRAM CAN TAKE ON MANY FORMS AND BE OF VARIABLE LENGTH DEPENDING ON THE TYPE OF INPUT DEVICE IT IS INTENDED TO FUNCTION WITH. NATURALLY, THE SHORTER THE PROGRAM THE BETTER FROM THE OPERATOR'S STAND-POINT OF HAVING TO MANUALLY LOAD IN THE BOOTSTRAP PROGRAM. FOR 8008 BASED MACHINES, IT IS GENERALLY DESIRABLE TO LIMIT THE SIZE OF A BOOTSTRAP LOADER TO UNDER, SAY, 100 OCTAL (64 DECIMAL) INSTRUCTIONS, AS THIS WILL KEEP THE MANUAL LOADING PROCESS UNDER ABOUT TEN MINUTES.

OF COURSE, FOR A PERSON WHO USES A COMPUTER ON A DAILY BASIS, EVEN TAKING FIVE OR 10 MINUTES EACH DAY TO MANUALLY INSERT A "BOOTSTRAP LOADER PROGRAM" CAN BECOME A DISTASTEFUL TASK. SO, MANY PEOPLE ELECT TO PUT A "BOOTSTRAP LOADER" ON A "READ ONLY MEMORY" (ROM) DEVICE. A "ROM" DEVICE HAS THE ATTRIBUTE OF RETAINING IT'S CONTENTS WHEN POWER IS REMOVED. SO, WITH A BOOTSTRAP PROGRAM ON A "ROM," ONE ELIMINATES THE PROCESS OF MANUALLY LOADING IN THE BOOTSTRAP PROGRAM WHEN POWER IS SUPPLIED. THE CONCEPT OF STILL UTILIZING A "BOOTSTRAP" PROGRAM THOUGH, IS STILL APPLICABLE, BECAUSE "ROM" ELEMENTS ARE CONSIDERABLY MORE EXPENSIVE THAN TYPICAL "READ AND WRITE" (RAM) MEMORY AND THE SHORTER THE PROGRAM ON "ROM" THE LESS EXPENSIVE THE "ROM" PORTION OF THE COMPUTER!

IT'S ABOUT TIME! SCELBI COMPUTER CONSULTING, INC., NOW IN IT'S 3'RD YEAR OF DELIVERING COMPUTERS TO PRIVATE INDIVIDUALS (AS WELL AS BUSINESS AND EDUCATIONAL INSTITUTIONS) HAS LONG BEEN AN ADVOCATE OF SOMEBODY PRODUCING A FULL FLEDGED MAGAZINE THAT WOULD CATER TO THE INDIVIDUAL COMPUTER OWNER. WE HAVE TRULY WONDERED WHY SOME OF THE ALREADY ESTABLISHED ELECTRONIC MAGAZINES DID NOT AT THE VERY LEAST DEVOTE A REGULAR SECTION IN THEIR PUBLICATIONS TO THIS FAST GROWING FIELD. WE'VE HAD PROOF FROM THE THOUSANDS OF LETTERS WE RECEIVE ASKING FOR ALL KINDS OF ADVICE AND GUIDANCE THAT THE PUBLIC WANTS TO KNOW MORE ABOUT COMPUTERS - THE KIND OF ADVICE THAT CANNOT BE PRACTICALLY DISPERSED BY A CONSULTING FIRM BUT RATHER NEEDS TO BE HANDLED BY A REGULAR PERIODICAL THAT CAN ATTRACT THE TALENTS OF A WIDE RANGE OF WRITERS AND COVER A DIVERSITY OF SPECIAL INTEREST WITHIN THE FIELD.

WE HAVE BEEN MOST PLEASED TO LEARN RECENTLY THAT SUCH A PUBLICATION IS NOW BEING STARTED. AND IT'S BEING STARTED BY AN ORGANIZATION WELL EQUIPPED TO HANDLE THE JOB. ANY COMPUTER OWNERS WHO ARE ALSO AMATEUR RADIO OPERATORS HAVE UNDOUBTABLY HEARD OF "73 MAGAZINE" AND IT'S PUBLISHER WAYNE GREEN. WELL, MR. GREEN HAS TAKEN THE BIG PLUNGE AND HAS STARTED ANOTHER MAGAZINE THAT IS DEVOTED TO SMALL COMPUTER OWNERS. THE NAME OF THE NEW MAGAZINE IS: B Y T E. THE FIRST ISSUE IS COMING OUT IN AUGUST. IN FACT, BY THE TIME YOU RECEIVE THIS THE FIRST ISSUE SHOULD BE COMING OFF THE PRESS. THEY ARE OFFERING "CHARTER" SUBSCRIPTION RATES AT $10.00 PER YEAR AT THE PRESENT TIME AND KNOWING THE QUALITY OF THE WAYNE GREEN ORGANIZATION, WE HAVE NO QUALMS ABOUT MAKING THE RECOMMENDATION THAT OUR READER'S CONSIDER OBTAINING A SUBSCRIPTION TO THIS NEW PUBLICATION. IF YOU ACT RIGHT AWAY, YOU SHOULD BE ABLE TO PICK UP ON THE "CHARTER" RATES. THE ADDRESS IS SHOWN BELOW.



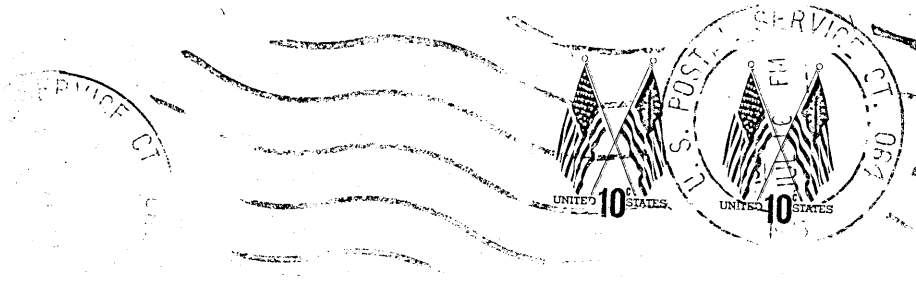
1 YEAR CHARTER SUBSCRIPTION RATE $10.00

BYTE
PETERBOROUGH
N. H. 03458

****

IN THIS ISSUE

SCELBI COMPUTER CONSULTING, INC.
1322 REAR - BOSTON POST ROAD
MILFORD, CT. 06460



FIRST CLASS MAIL!