CB-2

CB-2 Easy Logarithms for COP400

Literature Number: SNOA633

Easy Logarithms for COP400

National Semiconductor
COP Brief 2
September 1986

Logarithms have long been a convenient tool for the simplification of multiplication, division, and root extraction. Many assembly language programmers avoid the use of logarithms because of supposed complexity in their application to binary computers. Logarithms conjure up visions of time consuming iterations during the solution of a long series. The problem is far simpler than imagined and its solution yields, for the applications programmer, the classical benefits of logarithms:

1) Multiplication can be performed by a single addition.

2) Division can be performed by a single subtraction.

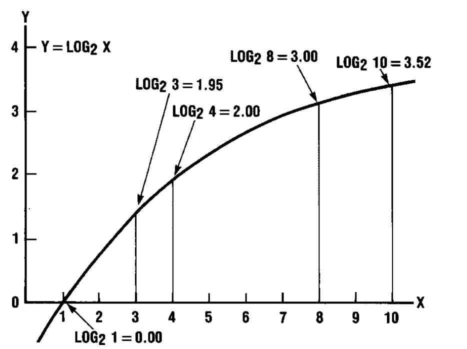
3) Raising a number to a power involves a single multiply.

4) Extracting a root involves a single divide.

When applied to binary computer operation logarithms yield two further important advantages. First, a broad range of values can be handled without resorting to floating point techniques (other than implied by the characteristic). Second, it is possible to establish the significance of an answer during the body of a calculation, again, without resorting to floating point techniques.

Implementation of $base_{10}$ logarithms in a binary system is cumbersome and unnecessary since logarithmic functions can be implemented in a number system of any base. The techniques presented here deal only with logarithms to the $base_2$.

A logarithm consists of two parts: an integer characteristic and a fractional mantissa.



TL/DD/6942–1

	CHARACTERISTIC	MANTISSA
LOG_2 3 =	1	0.95
LOG_2 4 =	2	0.00
LOG_2 8 =	3	0.00
LOG_2 10 =	3	0.52

FIGURE 1. The Logarithmic Function and Some Example Values

In *Figure 1* some points on the logarithmic curve are identified and evaluated to the $base_2$. Notice that the characteristic in each case represents the highest even power of 2 contained in the value of X. This is readily seen when binary notation is used.

X_{10}	X_2: 2^4	2^3	2^2	2^1	2^0	Log_2 X Characteristic	Log_2 X Where X = Even Power of 2
3	0	0	0	↑	1	1	
4	0	0	↑	0	0	2	010.0000
8	0	↑	0	0	0	3	011.0000
10	0	↑	0	1	0	3	

FIGURE 2. Identification of the Characteristic

In *Figure 2* each point evaluated in *Figure 1* has been repeated using binary notation. An arrow subscript indicates the highest even power of 2 appearing in each value of X. Notice that in X = 3 the highest even power of 2 is 2^1. Thus the characteristic of the log_2 3 is 1. Where X = 10 the characteristic of the log_2 10 is 3.

To find the log_2 X is very easy where X is an even power of 2. We simply shift the value of X left until a carry bit emerges from the high order position of the register. This procedure is illustrated in *Figure 3*. This characteristic is found by counting the number of shifts required and subtracting the result from the number of bits in the register. In practice it is easier to being with the number of bits and count down once prior to each shift.

Counter for Characteristic	Value of X in Binary		
1 0 0 0	0 0 0 0	1 0 0 0	Initial
0 1 1 1	0 0 0 1	0 0 0 0	First Shift
0 1 1 0	0 0 1 0	0 0 0 0	Second Shift
0 1 0 1	0 1 0 0	0 0 0 0	Third Shift
0 1 0 0	1 0 0 0	0 0 0 0	Fourth Shift
0 0 1 1	0 0 0 0	0 0 0 0	Fifth Shift
Characteristic	**Mantissa**		**Final**
0 1 1 . 0 0 0 0	0 0 0 0		Log_2 X = 3.00

FIGURE 3. Conversion to $Base_2$ Logarithm by Base Shift

Examination of the final value obtained in *Figure 3* reveals no bits in the mantissa. The value 3 in the characteristic, however, indicates that a bit did exist in the 2^3 position of the original number and would have to be restored in order to reconstruct the original value (antilog).

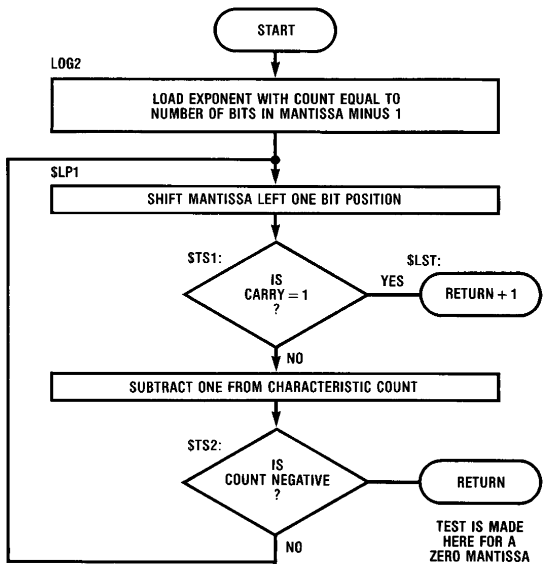
RRD-B30M105/Printed in U. S. A.

The log of any even power of 2 can be found in this way:

Decimal	Binary	Log_2
128	1 0 0 0 0 0 0 0	0 1 1 1 . 0 0 0 0 0 0 0 0
64	0 1 0 0 0 0 0 0	0 1 1 0 . 0 0 0 0 0 0 0 0
32	0 0 1 0 0 0 0 0	0 1 0 1 . 0 0 0 0 0 0 0 0
4	0 0 0 0 0 1 0 0	0 0 1 0 . 0 0 0 0 0 0 0 0
2	0 0 0 0 0 0 1 0	0 0 0 1 . 0 0 0 0 0 0 0 0
1	0 0 0 0 0 0 0 1	0 0 0 0 . 0 0 0 0 0 0 0 0

FIGURE 4. $Base_2$ Logarithms of Even Powers of 2

A simple flow chart, and program, can be devised for generating the values found in the table and, as will be apparent, a straight line approximation for values that are not even powers of 2. The method, as already illustrated in *Figure 3,* involves only shifting a binary number left until the most significant bit moves into the carry position. The characteristic is formed by counting. Since a carry on each successive shift will yield a decreasing power of 2, we must start the characteristic count with the number of bits in the binary value (x) and count down one each shift.



TL/DD/6942–2

FIGURE 5. Log Flowchart

```
COP CROSS ASSEMBLER       PAGE:  1
LOGS

                         ; TITLE LOGS                       ; BINARY LOGARITHMS
              01A4                     . CHIP 420
                                       ; ------→ CONVERT TO LOGARITHM ←------;
                         ;
                         ;                                  RAM ASSIGNMENT
                         ;
                         ; DIGIT:
                         ;
                         ; REG 0
                         ;
                         ; REG 1
                         ;
                         ; REG 2
                         ;
                         ; REG 3
                         ;
                         ;
```

DIGIT:	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
REG 0									CH	HM	LM				TEMP	
REG 1				CH	HM	LM									TEMP	
REG 2					TEMP									CH	HM	LM
REG 3											TEMP			CH	HM	LM

```
                                       . LOCAL
                         ;
                         ; CH, HM, LM REPRESENT ANY THREE SEQUENTIAL MEMORY DIGITS. THEY
                         ; MAY BE DEFINED IN ANY REGISTER. THE SYMBOLIC NOTATION CH, HM,
                         ; AND LM ARE USED FOR ADDRESSING TO ALLOW USER FLEXIBILITY.
                         ; UPON ENTRY TO THE ROUTINE HM AND LM CONTAIN THE HI AND LO
                         ; OF SOME VALUE X. THE MEMORY POINTER MUST CONTAIN THE ADDRESS
                         ; OF THE CHARACTERISTIC (CH). THE CONTENTS OF THIS LOCATION ARE
                         ; IGNORED AND ARE LOST DURING EXECUTION.
                         ;
                         ; UPON EXIT CH, HM, LM CONTAIN A STRAIGHT LINE APPROXIMATION OF
                         ; THE LOG BASE 2 OF X. CH = CHARACTERISTIC  HM = HI ORDER MANTISSA
                         ; LM = LO ORDER MANTISSA. AN 8 BIT MEMORY AREA (TEMP) IS USED IN
                         ; THE REGISTER OPPOSITE DURING THE CORRECTION OF A STRAIGHT
                         LINE APPROXIMATION OF A LOG OR AN ANTILOG.
                         ;             A TEST IS MADE FOR X = 0. IF THE VALUE OF X
                         ; IS NOT ZERO AN INSTRUCTION IS SKIPPED UPON RETURN
                         ; TO THE CALLING ROUTINE.
                         ;
                         ;                                                 — EXAMPLE —
                         ;
                         ; SUBROUTINE CALL                                          JSR   LOG2
                         ; RETURN HERE IF X = 0 --→                                 JP    ZERO
                         ; RETURN HERE IF X>0 --→                                    CONTINUE
                         ;
                         ;
                         ;
                         ;
                         ;
                         ;
       000     00        LOG2:         CLRA                                       ; SET CHARACTERISTIC.
       001     57                      AISC          07                           ; TO REG LENGTH  −1.
       002     06                      X                                          ; STORE IN MEMORY.

COP CROSS ASSEMBLER       PAGE:  2
LOGS

       003     A4        $LP1:         JSRP          SDB2                         ; SET ADDRESS POINTER
                                                                                  ; BACK 2 DIGITS.
       004     A9                      JSRP          SHLR                         ; RESET CARRY AND SHIFT
                                                                                  ; REG LEFT ONE BIT.
       005     20        $TS1:         SKC                                        ; IS CARRY = 1 YET?
       006     C8                      JP            $NO                          ; NO — KEEP GOING.
       007     49        $LST:         RETSK                                      ; YES — FINISHED!!
       008     05        $NO:          LD                                         ; NO — LOAD COUNT IN ACC.
       009     5F                      AISC          −1                           ; SUBTRACT ONE.
       00A     48        $TS2:         RET                                        ; MANTISSA IS A 0! RETURN
       00B     06                      X                                          ; STORE CHARACTERISTIC.
       00C     C3                      JP            $LP1                         ; DO IT AGAIN!

                         ; 2 ROUTINES ARE CALLED FROM THE SUBROUTINE PAGE BY THIS
                         ; PROGRAM: SDB2, SHLR.
```

TL/DD/6942–5

FIGURE 6

The program shown develops the $\log_2$ of any even power of 2 by shifting and testing as previously described. Examine what happens to a value of X that is not an even power of 2. In *Figure 7,* the number 25 is converted to a base 2 log.

$$25_{10} = 00011002_2$$

Shift left until carry = 1

Characteristic	Carry	Mantissa	Log_2
0100	1	10010000	0100.10010000

Figure 7. Straight Line Approximation of Base_2 Log

The resulting number when viewed as an integer characteristic and a fractional mantissa is 4.5625_{10}. The fraction 0.5625 is a straight line approximation of the logarithmic curve between the correct values for the base_2 logs of 2^4 and 2^5. The accuracy of this approximation is sufficient for many applications. The error can be corrected, as will be seen later in this discussion, but for now let's look at the problem of exponents or the conversion to an antilog.

To reconstruct the original value of X, find the antilog, requires only restoration of the most significant bit and then its alignment with the power of 2 position indicated by the characteristic. In the example, approximation ($\log_2 25 =$ 0100.1001) restoration of MSB can be accomplished by shifting the mantissa (only) one position to the right. In the process a one is shifted into the MSB position.

Approximation of Log_2 X	**Restoration of MSB**
Char. Mantissa	Char. Mantissa
0100.10010000	0100.11001000

The value of the characteristic is 4 so the mantissa must be shifted to the right until MSB is aligned with the 2^4 position.

$$2^7 \quad 2^6 \quad 2^5 \quad 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$$

The completion of this operation restores the value of X (X = 25) and is the procedure used to find an antilog. *Figure 8* is a flow chart for finding an antilog using this procedure. Ths implementation in source code is shown in *Figure 9.*

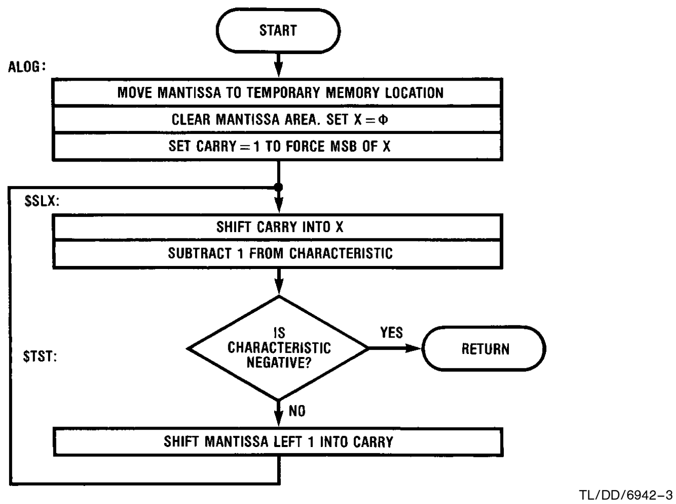


FIGURE 8. Flow Chart for Conversion to Antilog

```
COP CROSS ASSEMBLER PAGE 3
LOGS

                    . FORM         ; ------> CONVERT TO ANTILOG <------ ;


                    ; THE FOLLOWING SUBROUTINE CONVERTS THE STRAIGHT LINE
                    ; THE APPROXIMATION OF A BASE 2 LOGARITHM TO ITS CORRESPONDING
                    ; ANTILOG. UPON EXIT FROM THE ROUTINE THE CONTENTS OF CH
                    ; WILL BE EQUAL TO THE HEXADECIMAL VALUE OF 'ΦF'.

                                   . LOCAL


00D   A4            ALOG:          JSRP        SDB2            ; SET ACC TO 0.
00E   00            CLRA                                       ; CLEAR MANTISSA AREA.
00F   36                           X           03              ; AND MOVE MANTISSA TO
010   34                           XIS         03              ; TEMPORARY STORAGE.
011   00                           CLRA                        ; LEAVE POINTER AT LO
012   36                           X           03              ; ORDER OF MANTISSA.
013   37                           XDS         03
014   22                           SC                          ; RESTORE MSB OF X.
015   D8                           JP          $SLX
01    A9            $SLM:          JSRP        SHLR            ; SHIFT REMAINDER
                                                               ; LEFT INTO CARRY.
017   A3                           JSRP        SDR2            ; MOVE BACK 2 DIGITS.
018   AA            $SLX:          JSRP        SHLC            ; SHIFT X LEFT 1.
019   05                           LD                          ; LOAD CHARACTERISTIC.
01A   5F            $TST:          AISC        -1              ; CHARACTERISTIC -1.
01B   48            $LST:          RET                         ; IF NO CARRY — FINIS.
01C   36                           X           03              ; STORE REMAINDER AND MOVE
                                                               ; DOWN ONE REGISTER.
01D   A4                           JSRP        SDB2            ; MOVE BACK 2 DIGITS.
01E   D6                           JP          $SLM            ; DO IT AGAIN.


                    ; 4 ROUTINES ARE CALLED FROM THE SUBROUTINE PAGE BY THIS
                    ; PROGRAM: SDB2, SDR2, SHLR, SHLC.
```

TL/DD/6942–6

FIGURE 9

Using the linear approximation technique just described, some error will result when converting any value of X that is not an even power of 2.

Figure 10 contains a table of correct base 2 logarithms for values of X from 1 through 32 along with the error incurred for each when using linear approximation. Notice that no error results for values of X that are even powers of 2. Also notice that the error incurred for multiples of even powers of 2 of any given value of X is always the same.

Value of X	Error
5	0.12
2 × 5 = 10	0.12
4 × 5 = 20	0.12
3	0.15
2 × 3 = 6	0.15
4 × 3 = 12	0.15
8 × 3 = 24	0.15

X	Hexadecimal Log Base	Linear Approximation of Log Base 2	Error Hexadecimal	$E_M - 1 + \frac{EM - EM - 1}{2}$
1	0.00	0.00	0.00	
2	1.00	1.00	0.00	
3	1.95	1.80	0.15	
4	2.00	2.00	0.00	
5	2.52	2.40	0.12	
6	2.95	2.80	0.15	
7	2.CE	2.C0	0.0E	
8	3.00	3.00	0.00	
9	3.2B	3.20	0.0B	
10	3.52	3.40	0.12	
11	3.75	3.60	0.15	
12	3.95	3.80	0.15	
13	3.B3	3.A0	0.13	
14	3.CE	3.C0	0.0E	
15	3.E8	3.E0	0.08	
16	4.00	4.00	0.00	0.03
17	4.16	4.10	0.06	0.09
18	4.2B	4.20	0.0B	0.0D
19	4.3F	4.30	0.0F	0.11
20	4.52	4.40	0.12	0.15
21	4.67	4.50	0.17	0.16
22	4.75	4.60	0.15	0.16
23	4.87	4.70	0.17	0.16
24	4.95	4.80	0.15	0.15
25	4.A4	4.90	0.14	0.14
26	4.B3	4.IA0	0.13	0.12
27	4.C1	4.B0	0.11	0.10
28	4.CE;	4.C0	0.0E	0.0D
29	4.DB	4.D0	0.0B	0.0A
30	4.E8	4.E0	0.08	0.06
31	4.F4	4.F0	0.04	0.02
32	5.00	5.00	0.00	
33		5.1-		

FIGURE 10. Error Incurred by Linear Approximation of Base 2 Logs

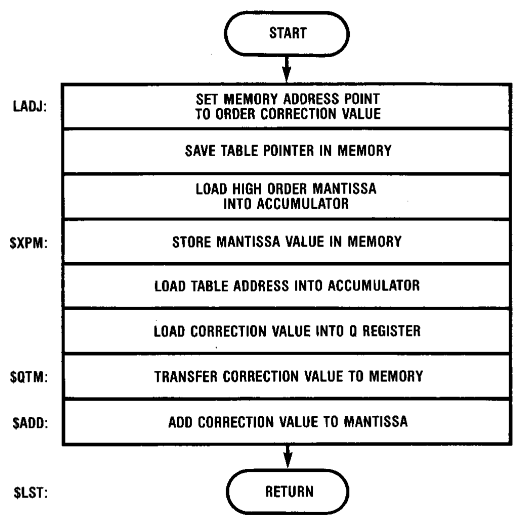
An error that repeats in this way is easily corrected using a look-up table. The greatest absolute error will occur for the least value of X not an even power of 2, X = 3, is about 8%. A 4 point correction table will eliminate this error but will move the greatest uncompensated error to X = 9 where it will be about 4%. This process continues until at 16 correction points the maximum error for the absolute value of the logarithm is less than 1 percent. This can be reduced to 0.3 percent by distributing the error. Interpolated error values are listed in *Figure 10* and are repeated in *Figure 11* as a binary table.

High Order 4 Mantissa Bits	Binary Correction Value	Hexadecimal Correction Value
0 0 0 0	0 0 0 0 0 0 0 0	0 0
0 0 0 1	0 0 0 0 1 0 0 1	0 9
0 0 1 0	0 0 0 0 1 1 0 1	0 3
0 0 1 1	0 0 0 1 0 0 0 1	1 1
0 1 0 0	0 0 0 1 0 1 0 1	1 5
0 1 0 1	0 0 0 1 0 1 1 0	1 6
0 1 1 0	0 0 0 1 0 1 1 0	1 6
0 1 1 1	0 0 0 1 0 1 1 0	1 6
1 0 0 0	0 0 0 1 0 1 0 1	1 5
1 0 0 1	0 0 0 1 0 1 0 0	1 4
1 0 1 0	0 0 0 1 0 0 1 0	1 2
1 0 1 1	0 0 0 1 0 0 0 0	1 0
1 1 0 0	0 0 0 0 1 1 0 1	0 D
1 1 0 1	0 0 0 0 1 0 1 0	0 A
1 1 1 0	0 0 0 0 0 1 1 0	0 6
1 1 1 1	0 0 0 0 0 0 1 0	0 2

FIGURE 11. Correction Table for L_2 X Linear Approximations

Notice in *Figure 10* that left justification of the mantissa causes its high order four bits to form a binary sequence that always corresponds to the proper correction value. This works to advantage when combined with the COP400 LQID instruction. LQID implements a table look-up function using the contents of a memory location as the address pointer. Thus we can perform the required table look-up without disturbing the mantissa.

Figure 12 is the flow chart for correction of a logarithm found by linear approximation. *Figure 13* is its implementation in COP400 assembly language. Notice that there are two entry points into the program. One is for correction of logs (LADJ:), the other is for correction of a value prior to its conversion to an antilog (AADJ:).



TL/DD/6942–4

FIGURE 12. Flow Chart for Correction of a Value Found by Straight Line Approximation

```
COP CROSS ASSEMBLER     PAGE:  4
LOGS

                  . FORM         ; ······→ ADJUST VALUE OF LOGARITHM ←······ ;

                                 . LOCAL


                  ; THE FOLLOWING TABLE IS USED DURING THE CORRECTION OF VALUES
                  ; FOUND BY STRAIGHT LINE APPROXIMATION. IT IS PLACED HERE IN
                  ; ORDER TO ALIGN ITS BEGINNING ELEMENT WITH A ZERO ADDRESS AS
                  REQUIRED BY THE LQID INSTRUCTION.

       01F   44                  NOP                                 ; REGISTER WITH ZERO ADDRESS.
       020   03   TPLS:          . WORD    03,09,0D,011
       021   09
       022   0D
       023   11
       024   15   . WORD         015,016,016,016
       025   16
       026   16
       027   16
       028   15                  . WORD    015,014,012,010
       029   14
       02A   12
       02B   10
       02C   0D                  . WORD    0D,0A,06,02
       02D   0A
       02E   06
       02F   02

                  ; THE FOLLOWING SUBROUTINE ADJUSTS THE VALUE OF A BASE 2
                  ; LOGARITHM FOUND BY STRAIGHT LINE APPROXIMATION. THE
                  ; CORRECTION TERMS ARE TAKEN FROM THE TABLE ABOVE. THE
                  ; SUBROUTINE HAS 2 ENTRY POINTS:
                  ;
                  ;              LADJ: — ADJUSTS A VALUE DURING CONVERSION TO A LOG
                  ;
                  ;              AADJ: — ADJUSTS A VALUE DURING CONVERSION TO ANTILOG
                  ;
                  ; THE CARRY FLAG IS SET UPON ENTRY TO DISTINGUISH BETWEEN LOG
                  ; (C = 1) AND ANTILOG (C = 0) CONVERSIONS. DURING A LOGARITHM
                  ; CONVERSION THE VALUE FOUND IN THE ABOVE TABLE IS ADDED TO
                  ; THE MANTISSA. DURING AN ANTILOG CONVERSION THE VALUE FOUND
                  ; IN THE ABOVE TABLE IS SUBTRACTED FROM THE MANTISSA.


       030   32   AADJ:          RC                                  ; C = 0 FOR ANTILOG
       031   F3                  JP        $LD                       ; CONVERSION.
       032   22   LADJ:          SC                                  ; C =  FOR LOG2 ADJ.
       033   05   $LD            LD                                  ; MOVE ADDRESS POINTER BACK
       034   07                  XDS                                 ; ONE LOCATION.
       035   05                  LD                                  ; LOAD CONTENTS OF HI MANTISSA
       036   37                  XDS       03                        ; AND STORE IT IN THE LO ORDER
       037   06                  X                                   ; OF THE TEMP MEMORY LOCATION.
       038   00                  CLRA                                ; SET TABLE POINTER
       039   52                  AISC      TBL                       ; (ACC) TO TABLE ADDRESS.

COP CROSS ASSEMBLER     PAGE:  5
LOGS

       03A   BF                  LQID                                ; LOAD CORRECTION VALUE TO Q.
       03B   332C $GTM:          CQMA                                ; TRANSFER Q REGISTER
       03D   04                  XIS                                 ; CONTENTS TO MEMORY.
       03F   07                  XDS
       03F   20                  SKC                                 ; ANTILOG?

       040   80                  JSRP      COMP                      ; YES — COMPLIMENT.
       041   98   $ADD:          JSRP      ADRO                      ; ADD CORRECTION VALUE
                                                                     ; TO MANTISSA.
       042   35                  LD        03                        ; SET POINTER TO
       043   48   $LST:          RET       ; CHARACTERISTIC AND
                                                                     ; RETURN.

                  ; 2 ROUTINES ARE CALLED FROM THE SUBROUTINE PAGE BY THIS
                  ; PROGRAM: COMP, ADRO

             0020                V1 = TPLS&OFF
             0002                TBL = V1/16
```

TL/DD/6942–7

FIGURE 13

Subroutines Used by the Log and Antilog Programs

```
COP CROSS ASSEMBLER     PAGE:  6
LOGS

                              .FORM

          0080      .PAGE 02                     ; -----→ SUBROUTINES ←----- ;

                    ; THE FOLLOWING ROUTINES RESIDE ON THE SUBROUTINE PAGE. THEY
                    ; ARE CALLED BY THE LOGS PROGRAM BUT ARE GENERAL PURPOSE IN
                    ; NATURE AND FUNCTION AS UTILITY ROUTINES.



                              ; -----→ COMPLEMENT 8 BITS ←----- ;

                              .LOCAL

                    ; THIS ROUTINE FORMS IN MEMORY THE 2'S COMPLEMENT OF THE TWO
                    ; ADJACENT DIGITS IDENTIFIED BY THE ADDRESS POINTER. THE
                    ; CONTENTS OF THE ADDRESS POINTER ARE NOT ALTERED.

                    ; THERE ARE TWO ENTRY POINTS:
                    ;
                    ; COP: COMPLEMENT 8 BITS.
                    ;
                    ; CMPE: EXTEND THE COMPLEMENT TO AN ADDITIONAL 8 BITS
                    ;

 080   22           COMP:     SC
 081   00           CMPE:     CLRA                                  ; SET MINUEND = 0
 082   06                     X                                     ; AND STORE IN MEMORY.
 083   10                     CASC
 084   44                     NOP                                   ;
 085   04                     XIS                                   ;
 086   00                     CLRA                                  ; SET MINUEND = 0
 087   06                     X                                     ; AND STORE IN MEMORY.
 083   10                     CASC                                  ;
 089   44                     NOP
 08A   04                     XIS                                   ;
 08B   44                     NOP                                   ; AVOID SKIP IF DIGIT 15.
 08C   A4                     JP          SDB2                      ; RETURN THRU SDB2
                                                                    ; TO RESTORE POINTER.



                    ; -----→ ADD 8 BITS IN ADJACENT REGISTERS ←----- ;

                              .LOCAL



                    ; THIS ROUTINE ADDS TWO BINARY DIGITS (8 BITS) FROM ANY REGISTER
                    ; TO THE CORRESPONDING TWO BINARY DIGITS IN EITHER REGISTER
                    ; IMMEDIATELY ADJACENT. THERE ARE THREE ENTRY POINTS:
                    ;
                    ;         LADR: — RESET CARRY AND ADD 2 DIGIT PAIRS
```

TL/DD/6942-8

```
                  ;          LADD: — ADD 2 DIGIT PAIRS WITH UNMODIFIED CARRY
                  ;          ADD1: — ADD 2 SINGLE DIGITS WITH UNMODIFIED CARRY

08D   32          LADR:      RC                  ; RESET CARRY PRIOR TO ADD.
08E   15          LADD:      :D        01        ; LD ADDEND AND MOVE TO ADJ REG
08F   30                     ASC                 ; ADD AUGEND.
090   44                     NOP                 ; AVOID CARRY!
091   14                     XIS       01        ; STORE SUM AND MOVE TO ADDEND
092   15          ADD1:      LD        01        ; REPEAT PROCESS
093   30                     ASC                 ; FOR
094   44                     NOP                 ; HIGH ORDER
095   14                     XIS       01        ; DIGIT.
096   44                     NOP                 ; AVOID SKIP IF DIGIT 15.
097   48          $LST:      RET                 ; FINISHED — RETURN!!!!

                             ; ------→ ADD 8 BITS IN OPPOSITE REGISTERS ←------ ;

                             . LOCAL

                  ; THIS ROUTINE ADDS TWO BINARY DIGITS (8BITS) FROM ANY REGISTER
                  ; TO THE CORRESPONDING TWO BINARY DIGITS IN EITHER REGISTER
                  ; DIRECTLY OPPOSITE. THERE ARE THREE ENTRY POINTS:
                  ;
                  ;          ADR0: — RESET CARRY AND ADD 2 DIGIT PAIRS
                  ;          ADD0: — ADD 2 DIGIT PAIRS WITH UNMODIFIED CARRY
                  ;          AD01: — ADD 2 SINGLE DIGITS WITH UNMODIFIED CARRY

098   32          ADR0:      RC                  ; RESET CARRY PRIOR TO ADD.
099   35          ADD0:      LD        03        ; LD ADDEND AND MOVE TO OPP REG
09A   30                     ASC                 ; ADD AUGEND.
09B   44                     NOP                 ; AVOID CARRY!
09C   34                     XIS       03        ; STORE SUM AND MOVE TO ADDEND.
09D   15          AD01:      LD        01        ; REPEAT PROCESS
09E   30                     ASC                 ; FOR
09F   44                     NOP                 ; HIGH ORDER
0A0   34                     XIS       03        ; DIGIT.
0A1   44                     NOP                 ; AVOID SKIP IF DIGIT 15.
0A2   48          $LST:      RET                 ; FINISHED — RETURN!!!!

                             ; ------→ SET DIGIT ADDRESS BACK TWO ←------ ;
```

TL/DD/6942-9

```
COP CROSS ASSEMBLER      PAGE:  8
LOGS

                                      . LOCAL
                           ; THIS ROUTINE SUBTRACTS 2 FROM THE CONTENTS OF THE
                           ; DIGIT POINTER (B REGISTER). THE CONTENTS OF THE
                           ; ACCUMULATOR ARE LOST IN THE PROCESS. THE USE OF
                           ; SDB2 ALLOWS ADDRESSING WITHIN THE LOGS SUB
                           ; ROUTINE TO BE RELATIVE TO THE CONTENTS OF THE
                           ; ADDRESS POINTER (B REGISTER) UPON ENTRY.
                           ; SDB2 IS COMMONLY USED IN BYTE OPERATIONS TO RESTORE THE
                           ; DIGIT POINTER TO THE LOW ORDER POSITION.
                           ; THERE ARE TWO ENTRY POINTS:
                           ;
                           ; SDR2:        SET DIGIT ADDRESS BACK 2 AND MOVE TO OPPOSITE REGISTER.
                           ;
                           ; SDB2: SET DIGIT ADDRESS BACK 2 RETAINING PRESENT REGISTER.
  0A3    35                SDR2:      LD        03                ; MOVE TO OPPOSITE REGISTER.
  0A4    4E                SDB2:      CBA                         ; PLACE DIGIT COUNT IN ACC.
  0A5    5E                           AISC      -2                ; SUBTRACT 2.
  0A6    44                           NOP                         ; SHOULD ALWAYS SKIP.
  0A7    50                           CAB                         ; PUT DIGIT COUNT BACK.
  0A8    48                           RET                         ; FINISHED — RETURN!!
                                      ; ------→ SHIFT LEFT ←------ ;
                                      . LOCAL
                           ; THIS ROUTINE SHIFTS LEFT THE CONTENTS OF TWO MEMORY
                           ; LOCATIONS ONE BIT. THERE ARE THREE ENTRY POINTS:
                           ;                  SHLR: RESETS THE CARRY BEFORE SHIFTING
                           ;                        IN ORDER TO FILL THE LOW ORDER
                           ;                        BIT POSITION WITH A 0.
                           ;
                           ;                  SHLC: SHIFTS THE STATE OF THE CARRY INTO
                           ;                        THE LOW ORDER BIT POSITION.
                           ;
                           ;                  SHL1: SHIFTS LEFT THE CONTENTS OF ONLY
                           ;                        ONE MEMORY LOCATION. THE STATE
                           ;                        OF THE CARRY IS SHIFTED INTO THE
                           ;                        LOW ORDER POSITION OF MEMORY.
  0A9    32                SHLR:      RC                          ; CLEAR CARRY PRIOR TO SHIFT.
  0AA    05                SHLC:      LD                          ; LOAD FIRST MEM DIGIT.
  0AB    30                           ASC                         ; DOUBLE IT.
  0AC    44                           NOP                         ; AVOID SKIP.
  0AD    04                           XIS                         ; STORE SHIFTED DIGIT.
  0AE    05                SHL1:      LD                          ; LOAD NEXT MEM DIGIT.
  0AF    30                           ASC                         ; DOUBLE IT TOO.

COP CROSS ASSEMBLER      PAGE:  9
LOGS

  0B0    44                           NOP                         ; AVOID SKIP, IF ANY
  0B1    04                           XIS                         ; STORE SHIFTED DIGIT.
  0B2    48                $LST:      RET                         ; FINISHED — RETURN!
                                      . END
```

TL/DD/6942-10

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
1111 West Bardin Road
Arlington, TX 76017
Tel: 1(800) 272-9959
Fax: 1(800) 737-7018

National Semiconductor Europe
Fax: (+49) 0-180-530 85 86
Email: cnjwge@tevm2.nsc.com
Deutsch Tel: (+49) 0-180-530 85 85
English Tel: (+49) 0-180-532 78 32
Français Tel: (+49) 0-180-532 93 58
Italiano Tel: (+49) 0-180-534 16 80

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductor Japan Ltd.
Tel: 81-043-299-2309
Fax: 81-043-299-2408

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connectivity	www.ti.com/wirelessconnectivity		
		TI E2E Community Home Page	e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265