DATA GENERAL
CORPORATION
Southboro,
Massachusetts 01772
(617) 485-9100

PROGRAM

Double Precision BCD to Binary

TAPES

ASCII Source: 090-000033

ABSTRACT

This routine converts an eight digit (32-bit) number in BCD to its double precision binary equivalent.

093-000029-00

1. REQUIREMENTS

1.1 Memory

1K or larger alterable memory.

1.2 Equipment

NOVA Central Processor.

1.3 External Subroutines

None.

1.4 Other

None.

2. OPERATING PROCEDURE

2.1 Calling Sequence

```
JSR  .DBCB
return
```

2.2 Input Format

An eight digit BCD integer is passed in AC0, AC1. The most significant digit is in bits 0 - 3 of AC0, the least significant digit is in bits 12 - 15 of AC1. Its maximum value is 99999999.

2.3 Output Format

The binary equivalent of the input (always positive) is returned in AC0 (high order), AC1 (low order).

2.4 Error Returns

If a digit greater than binary 1001 is encountered in the input, Carry will be set and AC0 will contain the bad digit. On a normal return, Carry will be zero.

2.5 State of Active Registers upon Exit

AC2 is unchanged. ACØ, AC1, AC3, and Carry are destroyed.

2.6 Cautions to User

None.

3. DISCUSSION

3.1 Algorithms

Assume a BCD input of the form:

B(7) B(6)...B(Ø)

where the B's represent the 4-bit BCD digits. The routine uses Horner's method to compute the binary equivalent:

(...(B(7)*1Ø + B(6)) * 1Ø + ...B(1)) * 1Ø + B(Ø)

If an illegal BCD digit is encountered at any point, Carry is set and return made with erroneous results.

3.2 Limitations and Accuracy

The BCD input, B, must be in the range

$$Ø \leqslant B < 1ØØ,ØØØ,ØØØ.$$

For inputs in this range, the routine is exact.

3.3 Size and Timing

The routine is 76 (octal) words in length.

Execution time is 2.174 milliseconds.

3.4 References

Single precision BCD to binary is described by write-up 093-000023.

3.5 Flow Diagrams

None.

4. EXAMPLES AND APPLICATIONS

The ASCII source of .DBCB is provided with the NOVA software. If a user routine requires double BCD to binary, this tape should be edited into the user source.

5. PROGRAM LISTING

A listing of .DBCB follows. No origin is given in the source, enabling the tape to be edited anywhere within the user's source.

```
                ; CONVERT A DOUBLE PRECISION NUMBER
                ; IN BCD TO BINARY

                ; INPUT:         A DOUBLE PRECISION BCD INTEGER IN
                ;                AC0, AC1 (HIGH, LOW) OF MAXIMUM VALUE
                ;                99999999 DECIMAL

                ; OUTPUT:        BINARY EQUIVALENT OF BCD INPUT IN AC0,
                ;                AC1 (HIGH, LOW)

                ; CALLING SEQUENCE:
                ;       JSR     .DBCB
                ;       RETURN

                ; ERROR CONDITION:       IF A DIGIT >9 IS ENCOUNTERED IN
                ;                THE INPUT, CARRY WILL BE SET
                ;                AND AC0 WILL CONTAIN THE BAD DIGIT

                ; DESTROYED:     AC0, AC1, AC3, CARRY
                ; UNCHANGED:     AC2



00000 054063 .DBCB:   STA 3,.FA03      ; SAVE RETURN
00001 040067          STA 0,.FA12
00002 044070          STA 1,.FA12+1    ; SAVE INPUT
00003 176400          SUB 3,3
00004 054064          STA 3,.FA10
00005 054065          STA 3,.FA10+1    ; CLEAR RESULT WORDS
00006 034073          LDA 3,.FA20      ; LOOP 8 TIMES
00007 054066          STA 3,.FA11

00010 004020 .FA99:   JSR .FA50        ; GET A DIGIT
00011 000017          JMP .FA98        ; ERROR IN DIGIT, CARRY IS SET
00012 004040          JSR .FA51        ; MULTIPLY BY 10 AND ADD
00013 014066          DSZ .FA11        ; LOOP FOR 8 DIGITS
00014 000010          JMP .FA99
00015 020064          LDA 0,.FA10
00016 024065          LDA 1,.FA10+1
00017 002063 .FA98:   JMP @.FA03       ; RETURN
```

```
                ; RETURN NEXT BCD DIGIT
                ; RIGHT ADJUSTED IN AC0

00020 024067 .FA50:  LDA 1,.FA12      ; GET INPUT
00021 030070         LDA 2,.FA12+1
00022 020074         LDA 0,.FA21      ; COUNT FOR 4 LEFT SHIFTS
00023 040071         STA 0,.FA13
00024 102400         SUB 0,0
00025 151120         MOVZL 2,2
00026 125100         MOVL 1,1
00027 101100         MOVL 0,0
00030 014071         DSZ .FA13
00031 000025         JMP .-4

00032 044067         STA 1,.FA12      ; SAVE SHIFTED INPUT
00033 050070         STA 2,.FA12+1
00034 024075         LDA 1,.FA22      ; TEST CONSTANT (10 DECIMAL)
00035 106042         ADCO 0,1,SZC
00036 001400         JMP 0,3          ; ERROR: >9, CARRY SET
00037 001401         JMP 1,3          ; SUCCESS: <= 9


                ; MULTIPLY BY 10 AND ADD AC0

00040 024064 .FA51:  LDA 1,.FA10      ; GET CURRENT SUM
00041 030065         LDA 2,.FA10+1
00042 151120         MOVZL 2,2
00043 125100         MOVL 1,1
00044 151120         MOVZL 2,2
00045 125100         MOVL 1,1         ; *4
00046 054072         STA 3,.FA14      ; SAVE RETURN
00047 034065         LDA 3,.FA10+1    ; LOW ORDER
00050 173022         ADDZ 3,2,SZC
00051 125400         INC 1,1
00052 034064         LDA 3,.FA10
00053 167000         ADD 3,1          ; *(4+1)
00054 151120         MOVZL 2,2
00055 125100         MOVL 1,1         ; 2*(4+1)
00056 113022         ADDZ 0,2,SZC
00057 125420         INCZ 1,1
00060 044064         STA 1,.FA10      ; SAVE RESULTS
00061 050065         STA 2,.FA10+1
00062 002072         JMP @.FA14       ; RETURN



00063 000000 .FA03:  0                ; SAVE RETURN

      000002 .FA10:  .BLK 2           ; RESULT WORDS
00066 000000 .FA11:  0                ; MAIN LOOP COUNT WORD
      000002 .FA12:  .BLK 2           ; INPUT
00071 000000 .FA13:  0                ; DIGIT LOOP COUNT WORD
00072 000000 .FA14:  0                ; RETURN FROM MULTIPLY BY 10

00073 000010 .FA20:  10               ; MAIN COUNT OF 8
00074 000004 .FA21:  4                ; DIGIT SHIFT OF 4
00075 000012 .FA22:  12               ; ERROR TEST (DECIMAL 10)
```