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

PROGRAM

Single Precision Binary to ASCII Decimal

TAPES

ASCII Source: 090-000030

ABSTRACT

This routine converts a single precision, two's complement binary number to a string of ASCII characters representing its equivalent decimal value.

093-000026-00

1. REQUIREMENTS

1.1 Memory

1K or larger alterable memory

1.2 Equipment

NOVA central processor

1.3 External Subroutines

A user supplied routine for accepting the ASCII output characters (see 2.3).

1.4 Other

None

2. OPERATING PROCEDURE

2.1 Calling Sequence

```
JSR  .BIND
return
```

2.2 Input Format

A single precision, two's complement integer is passed in AC1.

2.3 Output Format

The output is an ASCII character string of the form:

+DDDDD(null)

or -DDDDD(null)

The "D's" represent the ASCII decimal digits. The user must supply a routine that accepts these output characters. The address of this routine must be stored by the user in location 41 of page zero. The characters will be passed one at a time, right adjusted (bit 8 = Ø) in ACØ. Seven characters

will be passed; the sign, most significant digit, . . ., least significant digit, null (all Ø). The user routine need not save any registers (except AC3 if used). Return to the conversion routine should be made with a JMP Ø,3.

2.4 Error Returns

None

2.5 State of Active Registers upon Exit

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

2.6 Cautions to User

None

3. DISCUSSION

3.1 Algorithms

The sign of the result is determined and passed to the user output routine. Conversion is then made on the absolute value of the input.

The principle of the algorithm is to determine how many 1Ø,ØØØ's, 1,ØØØ's, . . ., 1's are contained in the number. This can be calculated by successively subtracting the appropriate power of ten from the original value until the result is negative. Each subtraction that gives a result greater than or equal to zero causes octal 6Ø (ASCII Ø) to be incremented. When the result becomes negative its previous value is restored, the ASCII digit is passed to the user, and the next lower power of ten is subtracted. After 1Ø**Ø is used, conversion is complete.

For example, assume an input of 43Ø. The following steps illustrate the process.

Step	Digit	Value	Number		Power of Ten
1	5	6Ø	43Ø	-	1ØØØØ
2	4	6Ø	43Ø	-	1ØØØ
3	3	6Ø	43Ø	-	1ØØ
4	3	61	33Ø	-	1ØØ
5	3	62	23Ø	-	1ØØ
6	3	63	13Ø	-	1ØØ
7	3	64	3Ø	-	1ØØ
8	2	6Ø	3Ø	-	1Ø
9	2	61	2Ø	-	1Ø
1Ø	2	62	1Ø	-	1Ø
11	2	63	Ø	-	1Ø
12	1	6Ø	Ø	-	1

The successive ASCII digits generated are thus:

ØØ43Ø.

3.2 Limitations and Accuracy

The routine is exact for all 16-bit, two's complement numbers.

3.3 Size and Timing

The routine is 51 (octal) words in length.

Execution time is (378.3 + N * 14.1) μ seconds, where N is the sum of the digits of the result. For example, the number ØØ43Ø requires

378.3 + (Ø + Ø + 4 + 3 + Ø) * 14.1 = 477 μ seconds.

3.4 References

None

3.5 Flow Diagrams

None

4. EXAMPLES AND APPLICATIONS

The ASCII source of binary to decimal convert is provided with the NOVA software. If a user routine requires .BIND, this tape should be edited into the user's source.

5. PROGRAM LISTING

A listing of .BIND follows. No origin is given in the source, enabling the tape to be edited anywhere within a user routine.

```
                ; BINARY TO DECIMAL ASCII CONVERT
                ; CONVERTS A SINGLE PRECISION, TWO'S COMPLEMENT
                ;       NUMBER TO AN ASCII CHARACTER STRING

                ; INPUT:         N IN AC1

                ; OUTPUT:        ASCII CHARACTER STRING, TERMINATED BY A
                ;                NULL WORD
                ;                CHARACTERS ARE RIGHT ADJUSTED IN AC0
                ;                PASSED TO THE ROUTINE WHOSE ADDRESS
                ;                MUST BE IN LOCATION 41 OF PAGE 0
                ;                STRING OF FORM:
                ;                        +DDDDD(NULL)
                ;                OR      -DDDDD(NULL)

                ; CALLING SEQUENCE:
                ;       JSR     .BIND
                ;       RETURN


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

00000 054034 .BIND:  STA 3,.ED03     ; SAVE RETURN
00001 050033         STA 2,.ED02     ; SAVE AC2
00002 040032         STA 0,.ED00     ; SAVE AC0
00003 034050         LDA 3,.ED30     ; ADDRESS OF POWER OF TEN TABLE
00004 054043         STA 3,.ED10     ; INITIALIZE POINTER
00005 020045         LDA 0,.ED20     ; ASSUME NEGATIVE
00006 125112         MOVL# 1,1,SZC
00007 124401         NEG 1,1,SKP
00010 020046         LDA 0,.ED21     ; NO, IT IS POSITIVE; GET PLUS
00011 044044 .ED97:  STA 1,.ED11     ; SAVE N
00012 006041         JSR @.ED40      ; PUT OUT SIGN OR DIGIT
00013 024044         LDA 1,.ED11     ; GET CURRENT VALUE OF N
00014 036043         LDA 3,@.ED10    ; GET CURRENT  POWER OF TEN
00015 010043         ISZ .ED10       ; BUMP POINTER
00016 161005         MOV 3,0,SNR
00017 000026         JMP .ED98       ; PUT OUT NULL
00020 020047         LDA 0,.ED22     ; GET ASCII "0"
00021 166422 .ED99:  SUBZ 3,1,SZC    ; DOES POWER OF TEN GO IN?
00022 101401         INC 0,0,SKP     ; YES, BUMP RESULT DIGIT
00023 167001         ADD 3,1,SKP     ; NO, RESTORE PREVIOUS VALUE
00024 000021         JMP .ED99       ; CONTINUE SUBTRACTING
00025 000011         JMP .ED97       ; PUT OUT DIGIT

00026 006041 .ED98:  JSR @.ED40      ; PUT OUT NULL WORD
00027 020032         LDA 0,.ED00     ; RESTORE AC0
00030 030033         LDA 2,.ED02     ; RESTORE AC2
00031 002034         JMP @.ED03      ; RETURN
```

```
00032 000000 .ED00:  0                   ; SAVE AC0
00033 000000 .ED02:  0                   ; SAVE AC2
00034 000000 .ED03:  0                   ; SAVE AC3

      000012         .RDX 10
00035 023420 .ED05:  10000               ; POWER OF TEN TABLE
00036 001750         1000                ; 10**3
00037 000144         100                 ; 10**2
00040 000012         10                  ; 10**1
00041 000001         1                   ; 10**0
00042 000000         0                   ; END OF TABLE INDICATION
      000010         .RDX 8

00043 000000 .ED10:  0                   ; ADDRESS OF CURRENT POWER OF
                                         ; TEN ENTRY
00044 000000 .ED11:  0                   ; RUNNING SUM WORD

00045 000055 .ED20:  "-                  ; ASCII "-"
00046 000053 .ED21:  "+                  ; ASCII "+"
00047 000060 .ED22:  60                  ; ASCII "0"

00050 000035 .ED30:  .ED05               ; ADDRESS OF POWER OF TEN TABLE

      000041 .ED40=41                    ; PAGE ZERO PUT CHARACTER
                                         ; ROUTINE ADDRESS
```