

- 1. IDENTIFICATION
- 1.1 Digital-8-14-U-Sym
- 1.2 Binary to Binary Coded Decimal Conversion
- 1.3 March 10, 1965

2. ABSTRACT

This subroutine provides the basic means of converting binary data to binary-coded-decimal (BCD) data for typeout, magnetic tape recording, etc.

3. REQUIREMENTS

3.1 Storage

This subroutine uses 33 (decimal) storage locations.

3.3 Equipment

Standard PDP-8.

4. USAGE

4.2 Calling Sequence

The subroutine is called by the JMS instruction. When called, the binary number to be converted must be in the accumulator (AC).

The subroutine will return to the instruction immediately following the calling JMS with the BCD result in the AC.

6. DESCRIPTION

6.1 Discussion

Reference to the Flow Chart (Section 11.1) will illustrate this discussion.

On input the binary number is stored, a pointer is initialized, the link is cleared and a counter to control the number of passes through the computation loop proper is properly set.

The loop is now entered. It will be repeated eight times.

The binary equivalents of 800, 400, 200, 100, 80, 40, 20, and 10 are subtracted successively from the original binary number.

After each subtraction, a test on the link is made. If the result of the test shows the link to be 0, the next lower equivalent is subtracted from the same quantity after the contents of the links (0) are shifted into the developing BCD number (Location NUMBER).

If the subtraction leaves a negative link, the contents of the accumulator replace the binary representation currently being processed after the contents of the link (1) have been shifted into the growing BCD number.

After eight passes through the basic loop, the developed BCD representation is shifted left four bits and the "residual" least significant digit is added before exit.

6.2 Example

As an example consider the conversion of the binary equivalent of 512 decimal:

001 000 000 000

Successive calculations to obtain the most significant BCD digit are as follows:

<u>Link</u>	<u>Addition</u>	
	001 000 000 000	
	110 011 100 000	
0	<u>111 011 100 000</u>	-800
	001 000 000 000	
	111 001 110 000	
1	<u>000 001 110 000</u>	-400
	000 001 110 000	
	111 110 001 000	
0	<u>111 111 111 000</u>	-200
	000 001 110 000	
	111 110 011 100	
1	<u>000 000 001 100</u>	-100

Notice that the remainder is the binary representation of 12 decimal. Writing the link bits in the order they are developed gives 0101 the BCD character denoting 5.

6.3 Scaling

The original binary number must be no larger than 999 (decimal) which is equivalent to 1747 (octal). The binary point is assumed to be at the extreme right end of the word (to the right of bit position 11) and the decimal point is also so positioned.

In other words, this subroutine converts binary integers to BCD integers.

Note that the subroutine is designed for positive input only!

7. METHOD

7.2 Algorithm

The algorithm is straightforward and is fully described in sections 6.1, 10.4, and 11.1.


```

/ TIME=216.0-235.2 MICRO-SECONDS PDP-8
/ IF INPUT >999 (10) RESULT IS UNSPECIFIED
0200 0000 BCD, 0
0201 3226 DCA INPUT /STORE BINARY
0202 1225 TAD CONTRL /SET UP TABLE
0203 3210 DCA POINTR /POINTERS
0204 7100 CLL
0205 1230 TAD COUNT /SET BIT 7=1; 8RAL'S
0206 3227 DCA NUMBER /WILL PUT IT IN LINK
0207 1226 TAD INPUT
0210 1231 POINTR, TAD TABLE /OR TABLE+1, TABLE+2, ETC.
0211 7430 SZL /IF C(L)=1, INPUT>-TABLE
0212 3226 DCA INPUT /IF SO: INPUT=INPUT+TABLE
0213 7200 CLA
0214 1227 TAD NUMBER
0215 7004 RAL /PUT THIS BIT IN ANSWER
0216 2210 ISZ POINTR /UPDATE TABLE POINTER
0217 7420 SNL /IF LINK=1, ALL DONE
0220 5206 JMP POINTR.-2
0221 7106 CLL RTL /CONVERTED 2 BCD
0222 7006 RTL /CHARACTERS
0223 1226 TAD INPUT /SHIFT LEFT AND ADD
0224 5600 JMP I BCD /THE THIRD
0225 1231 CONTRL, TAD TABLE
0226 0000 INPUT, 0
0227 0000 NUMBER, 0
0230 0020 COUNT, 0020
0231 6340 TABLE, -1440 /-800(10)
0232 7160 -0620 /-400
0233 7470 -0310 /-200
0234 7634 -0144 /-100
0235 7660 -0120 /-80
0236 7730 -0050 /-40
0237 7754 -0024 /-20
0240 7766 -0012 /-10

/EXAMPLE: INPUT 0726 (8)
/ OUTPUT 0100/0111/0000 = 470 (10)

```

11. DIAGRAM

11.1 Flow Chart



