1. IDENTIFICATION

1.1 Digital-8-10-U-Sym

1.2 Binary-Coded-Decimal to Binary Conversion Subroutine

1.3 March 1, 1965
2. ABSTRACT

A basic subroutine for converting binary-coded-decimal numbers to their equivalent binary value. Conversion is accomplished by "radix deflation."

3. REQUIREMENTS

3.1 Storage
This subroutine requires 23 (decimal) memory locations.

3.3 Equipment
Standard PDP-8.

4. USAGE

4.1 Loading
Load the subroutine with the Binary or RIM Loader.

4.2 Calling Sequence
Call with the number to be converted in the AC. Return will be to the location following the calling JMS with the result in the AC.

6. DESCRIPTION

6.1 Discussion

The method used is that of "radix deflation." Upon entry, the BCD number may be considered to be in the following form.

6.1.1.E
\[ D_2 16^2 + D_1 16 + D_0 \]

What is desired is the number in the form.

6.1.2.E
\[ D_2 10^2 + D_1 10 + D_0 \]

The PDP-8 can shift (rotate) and add. A right shift is equivalent to a division by a power of two. An appropriate series of shifts, additions, and subtractions is used to convert the number from the form of 6.1.1.E to that of 6.1.2.E.

6.2 Example

Consider the BCD number
representing the decimal number 519.

First, the whole number is stored and then brought back into the AC. Next, the four most significant bits are masked out. At this point, the accumulator contains 16x16xA or

\[
\begin{array}{cccc}
0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 \\
\end{array}
\]

A shift to the right of one bit yields

\[
\begin{array}{cccc}
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 \\
\end{array}
\]

This number is stored and then brought back to the AC, shifted right two bits, and the stored value added as follows

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 1 & 0 & 1 \\
\end{array}
\]

Now the original number is added to this result

\[
\begin{array}{cccc}
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 \\
\end{array}
\]

and the most significant eight bits masked out as

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 \\
\end{array}
\]

This is stored, brought back and shifted right once, and the stored value added.

\[
\begin{array}{cccc}
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 \\
\end{array}
\]

Next the result of this addition is shifted right two places dividing the number by four as follows

\[
\begin{array}{cccc}
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 \\
\end{array}
\]

negated and the original number added

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{array}
\]

This result represents in binary 512 plus 4 plus 2 plus 1 or 519, the original number.

6.3 Scaling

This subroutine assumes an integral BCD number and yields an integral binary equivalent.
7. **METHOD**

7.2 **Algorithm**

The algorithm used is illustrated step by step in Section 10.4.

9. **EXECUTION TIME**

9.2 **Maximum**

The maximum (and invariant) execution time of this subroutine is 49.6 microseconds.

10. **PROGRAM**

10.4 **Program Listing**

A listing of the subroutine with BCDBIN located at 0200 is given below. To simplify mnemonics D_{2}', D_{1}', and D_{0}' have been replaced respectively with A, B, and C.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0200</td>
<td>0000</td>
<td>BCDBIN,</td>
<td>0</td>
<td>/ABC IN BCD CODE IN AC</td>
<td></td>
</tr>
<tr>
<td>0201</td>
<td>3223</td>
<td>DCA TEMPPP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0202</td>
<td>1223</td>
<td>TAD TEMPPP</td>
<td>/16 (16A + B) + C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0203</td>
<td>0225</td>
<td>AND MASKKA</td>
<td>/16 (16A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0204</td>
<td>7110</td>
<td>CLL RAR</td>
<td>/8 (16A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0205</td>
<td>3224</td>
<td>DCA TEMPPQ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0206</td>
<td>1224</td>
<td>TAD TEMPPQ</td>
<td>/8 (16A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0207</td>
<td>7012</td>
<td>RTR</td>
<td>/2 (16A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0210</td>
<td>1224</td>
<td>TAD TEMPPQ</td>
<td>/10 (16A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0211</td>
<td>1223</td>
<td>TAD TEMPPP</td>
<td>/16 (26A + B) + C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0212</td>
<td>0226</td>
<td>AND MASKKB</td>
<td>/16 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0213</td>
<td>3224</td>
<td>DCA TEMPPQ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0214</td>
<td>1224</td>
<td>TAD TEMPPQ</td>
<td>/16 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0215</td>
<td>7110</td>
<td>CLL RAR</td>
<td>/8 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0216</td>
<td>1224</td>
<td>TAD TEMPPQ</td>
<td>/24 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0217</td>
<td>7012</td>
<td>RTR</td>
<td>/6 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0220</td>
<td>7041</td>
<td>CIA</td>
<td>/-6 (26A + B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0221</td>
<td>1223</td>
<td>TAD TEMPPP</td>
<td>/16 (16A + B) + C - 6 (26A + B)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0222</td>
<td>5600</td>
<td>JMP 1 BCDBIN</td>
<td>/BINARY VALUE IN AC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0223</td>
<td>0000</td>
<td>TEMPPP;</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0224</td>
<td>0000</td>
<td>TEMPPQ;</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0225</td>
<td>7400</td>
<td>MASKKA,</td>
<td>7400</td>
<td>/MASK FOR MOST SIG. FOUR BITS</td>
<td></td>
</tr>
<tr>
<td>0226</td>
<td>7760</td>
<td>MASKKB,</td>
<td>7760</td>
<td>/MASK FOR MOST SIG. EIGHT BITS</td>
<td></td>
</tr>
</tbody>
</table>
12. REFERENCES

12.3 DECUS Programs

See DECUSCOPE January 1965, article entitled "Accelerated Radix Deflation on the PDP-7 and PDP-8."

14. ACKNOWLEDGEMENTS

Mr. Donald V. Weaver, Consultant, of New York City, who first described the algorithm used by this subroutine in reference 12.3 has granted his kind permission to include this subroutine in the PDP-8 library so that a detailed description may be available.