HIGH-SPEED ZERO-SUM DETECTION

Arnold Weinberger
P.O.Box 390
International Business Machines Corporation
Poughkeepsie, New York 12602

SUMMARY

A common requirement accompanying high-speed parallel addition is the early detection that the sum is equal to zero. Normally, this condition is detected from the sum, generally at least two logic gate levels after the sum.

This paper derives expressions for detecting a zero-sum condition concurrently with or even earlier than the determination of the sum digits proper. As a result, a branch operation based on detecting a zero-sum can be executed earlier.

For example, a zero-sum condition, labeled ZEROS, during addition of two n-bit binary numbers A and B, can be expressed by:

$$ZEROS = \left[H_0+G_1+\ldots.+G_{n-1}\right]\bullet\left[H_0+\overline{P}_1\right]\bullet\ldots\bullet\left[H_{n-2}+\overline{P}_{n-1}\right]\bullet\left[H_{n-1}-\overline{C}_{in}\right]$$

where H_i, G_i, and P_i are the individual bit position functions:

$$\left.\begin{array}{l} H_i = A_i \forall B_i \\ G_i = A_i \bullet B_i \\ P_i = A_i + B_i \end{array}\right\} \quad \text{for } i=0,\ldots.,n-1; \text{ high-to-low-order bit positions}$$

and C_{in} = input carry to the adder

Other expressions for ZEROS are derived that make use of the output carry (C_0) or a carry from an intermediate bit position (C_k). Zero-sum detection is also extended to higher radices, notably decimal.

Similar expressions are also derived for detecting a binary sum of all 1's and generalized to a sum with diminished-radix digits.

INTRODUCTION

High-speed addition is usually accompanied by high-speed detection of certain sum conditions that critically determine the timing of the execution of the next operation. One of these conditions is that the sum digits or a string of consecutive sum digits are zero.

A zero-sum condition can be detected directly from the adder inputs without first generating the sum digits. In general, a string of consecutive zero-sum digits can be shown to be a simple function of the corresponding addend and augend input digits and the input carry to the string. If the string includes all sum digits, or the low-order truncated portion of the sum, the input carry is generally available concurrently with the input digits, so that the zero-sum condition can be detected in the earliest possible manner. The expressions derived here are more general and economical than one described earlier.[1]

Alternative expressions for zero-sum detection make use of the fast carry network[2,3] that accompanies high-speed addition. The output carry and some intermediate carries from a fast carry network are generally also available prior to the sum, so that zero-sum detection can still be achieved earlier than from the sum digits, yet with greater economy. The use of the output carry or some intermediate carry for zero-sum detection is restricted to binary.

The expressions for zero-sum detection will be derived first for a 32-bit adder, then extended to n-digit adders of higher integer radices, with special emphasis on binary-coded decimal.

The methods of deriving expressions for zero-sum detection are also applied to the detection of a string of 1's in a binary sum. This is similarly extended to detecting a string of diminished-radix sum digits for higher radices, with special emphasis to detecting a string of nines for decimal.

ZERO-SUM DETECTION OF A 32-BIT SUM

Let $A=(A_0,A_1,\ldots,A_{31})$ = addend of a 32-bit adder

$B=(B_0,B_1,\ldots,B_{31})$ = augend

where the subscripts refer to the bit positions, 0-31, high-to-low order, respectively.

The following functions of individual bit positions will be used in the description:

$H_k = A_k \forall B_k = \overline{A}_k \bullet B_k + A_k \bullet \overline{B}_k$ = half-sum of position k

$G_k = A_k \bullet B_k$ = carry generate of bit position k

$P_k = A_k + B_k$ = carry propagate of bit position k

It follows that:

$$H_k = P_k \bullet \overline{G}_k \qquad \overline{H}_k = \overline{P}_k + G_k$$

$$G_k = \overline{H}_k \bullet P_k \qquad \overline{G}_k = H_k + \overline{P}_k$$

$$P_k = H_k + G_k \qquad \overline{P}_k = \overline{H}_k \bullet \overline{G}_k$$

In addition, an input carry, C_{in}, is included among the inputs.

The conditions for generating a sum of all zeros are enumerated in Eq. (1). In words, a zero-sum labeled ZEROS is present if one of the three conditions is satisfied:

1. All inputs are zeros ($\overline{P}$ and $\overline{C}_{in}$).

2. In one and only one bit position both addend and augend are 1(G), producing a carry. At the same time, the trailing bit positions produce $\overline{P}$, C_{in} = 0, and the leading bit positions produce H to propagate the generated carry through them and leaving zeros in its wake.

3. C_{in} = 1, which generates a carry, and all bit positions produce H to propagate the carry through them and leave zeros in its wake.

$$\begin{aligned}\text{ZEROS} = {} & \overline{P}_0 \bullet \overline{P}_1 \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & + G_0 \bullet \overline{P}_1 \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & \vdots \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet G_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet H_{31} \bullet C_{in}\end{aligned} \qquad (1)$$

The first two terms of Eq. (1) are combined to yield Eq. (2). since $(\overline{P}_0 + G_0) = \overline{H}_0$.

$$\begin{aligned}\text{ZEROS} = {} & \overline{H}_0 \bullet \overline{P}_1 \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet G_1 \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & \vdots \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet G_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet H_{31} \bullet C_{in}\end{aligned} \qquad (2)$$

The first two terms of Eq. (2) are combined to yield Eq. (3),noting that $\overline{H}_0 \bullet H_0 = G_1 \bullet \overline{P}_1 = 0$

$$\begin{aligned}\text{ZEROS} = {} & (\overline{H}_0 + G_1) \bullet (H_0 + \overline{P}_1) \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet H_1 \bullet G_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & \vdots \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet G_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet H_{31} \bullet C_{in}\end{aligned} \qquad (3)$$

The process of combining the first two terms is repeated until only a single term remains, as shown in Eq. (4).

$$\begin{aligned}\text{ZEROS} = {} & (\overline{H}_0 + G_1 + \ldots + G_{31} + C_{in}) \bullet (H_0 + \overline{P}_1) \bullet (H_1 + \overline{P}_2) \bullet \\ & \ldots \bullet (H_{30} + \overline{P}_{31}) \bullet (H_{31} + \overline{C}_{in})\end{aligned} \qquad (4)$$

The iteration can be proved by induction by showing that

$$\begin{aligned}& (\overline{H}_0 + G_1 + \ldots + G_k + G_{k+1}) \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet (H_{k-1} + \overline{P}_k) \bullet \\ & (H_k + \overline{P}_{k+1}) = (\overline{H}_0 + G_1 + \ldots + G_k) \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet \\ & (H_{k-1} + \overline{P}_k) \bullet \overline{P}_{k+1} + H_0 \bullet H_1 \bullet \ldots \bullet H_k \bullet G_{k+1}\end{aligned} \qquad (5)$$

Eq. (5) is proved by expanding the left side to:

$$\begin{aligned}& (\overline{H}_0 + G_1 + \ldots + G_k) \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet (H_{k-1} + \overline{P}_k) \bullet H_k \\ & + (\quad\downarrow\quad) \bullet (\quad\downarrow\quad) \bullet \ldots \bullet (\quad\downarrow\quad) \bullet \overline{P}_{k+1} \\ & + G_{k+1} \bullet (\quad\downarrow\quad) \bullet \ldots \bullet (\quad\downarrow\quad) \bullet H_k \\ & + G_{k+1} \bullet (\quad\downarrow\quad) \bullet \ldots \bullet (\quad\downarrow\quad) \bullet \overline{P}_{k+1}\end{aligned}$$

The first term of the expansion is equal to 0, as shown by multiplying through, as follows:

$$\begin{aligned}& (H_0 + G_1 + \ldots + G_k) \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet (H_{k-1} + \overline{P}_k) \bullet H_k \\ & = (\quad\downarrow\quad) \bullet (\quad\downarrow\quad) \bullet \ldots \bullet H_{k-1} \bullet H_k \\ & \vdots \\ & = (\quad\downarrow\quad) \bullet H_0 \bullet \ldots \bullet H_{k-1} \bullet H_k \qquad \text{since } \overline{P}_i \bullet H_i = 0\end{aligned}$$

$= 0$ since $G_i \bullet H_i = \overline{H}_0 \bullet H_0 = 0$

The second term of the expansion corresponds to the first term of the right side of Eq.(5). The third term of the expansion reduces to the second term of the right side of Eq. (5), as follows:

$$\begin{aligned}& G_{k+1} \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet (H_{k-1} + \overline{P}_k) \bullet H_k \\ & = G_{k+1} \bullet (\quad\downarrow\quad) \bullet \ldots \quad H_{k-1} \bullet H_k \\ & \vdots \\ & = G_{k+1} \bullet H_0 \bullet \ldots \bullet H_{k-1} \bullet H_k \qquad \text{since } \overline{P}_i \bullet H_i = 0\end{aligned}$$

The fourth term of the expansion is also equal to 0 since

$$G_{k+1} \bullet \overline{P}_{k+1} = 0$$

Eq. (5) also applies to the last iteration where

$$\overline{P}_{k+1} \equiv \overline{C}_{in} \text{ and } G_{k+1} \equiv C_{in}$$

Q.E.D

An alternate implementation makes use of the output carry of the adder, C_{out}, to replace the expression, $(G_1 + \ldots + G_{31} + C_{in})$. The output carry can be generated early, i.e., prior to the sum, so that ZEROS is available concurrently with or earlier than the sum. The alternate equation is:

$$\begin{aligned}\text{ZEROS} = {} & (C_{out} + \overline{P}_0) \bullet (H_0 + \overline{P}_1) \bullet \ldots \bullet (H_{30} + \overline{P}_{31}) \bullet \\ & (H_{31} + \overline{C}_{in})\end{aligned} \qquad (6)$$

It is derived as follows: C_{out} can be expressed as in Eq. (7).

$$\begin{aligned}C_{out} = {} & G_0 \\ & + H_0 \bullet G_1 \\ & \vdots \\ & + H_0 \bullet H_1 \bullet \ldots \bullet G_{31} \\ & + H_0 \; H_1 \bullet \ldots \bullet H_{31} \bullet C_{in}\end{aligned} \qquad (7)$$

Eq. (7) is now substituted for each G_k in Eq. (1) to yield Eq. (8). For each substitution of a G_k with a C_{out}, only the corresponding

$$\begin{aligned}\text{ZEROS} = {} & \overline{P}_0 \bullet \overline{P}_1 \bullet \overline{P}_2 \bullet \ldots \; \overline{P}_{31} \bullet \overline{C}_{in} \\ & + C_{out} \bullet \overline{P}_1 \; \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & + H_0 \bullet C_{out} \bullet \overline{P}_2 \bullet \ldots \bullet \overline{P}_{31} \bullet \overline{C}_{in} \\ & \vdots \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet C_{out} \bullet \overline{C}_{in} \\ & + H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet H_{31} \bullet C_{out}\end{aligned} \qquad (8)$$

term in Eq.(7) (i.e., the term containing

G_k) is relevant; the other terms drop out because H_k, $\overline{P}_k$, and G_k are mutually exclusive.

Eq.(8) is now reduced iteratively by combining the first two terms.

$$\begin{aligned}
ZEROS &= (C_{out}+\overline{P}_0)\bullet\overline{P}_1\bullet\overline{P}_2\bullet\ldots\bullet\overline{P}_{31}\bullet\overline{C}_{in} \\
&\quad + \quad H_0\bullet C_{out}\bullet\overline{P}_2\bullet\ldots\bullet\overline{P}_{31}\bullet\overline{C}_{in} \\
&\quad +\ldots \\
&= (C_{out}+\overline{P}_0)\bullet(H_0+\overline{P}_1)\bullet\overline{P}_2\bullet\ldots\bullet\overline{P}_{31}\bullet\overline{C}_{in} \\
&\quad + \quad H_0\bullet H_1\bullet C_{out}\bullet\ldots\bullet\overline{P}_{31}\bullet C_{in} \\
&\quad +\ldots \\
&\quad \vdots \\
&= (C_{out}+\overline{P}_0)\bullet(H_0+\overline{P}_1)\bullet(H_1+\overline{P}_2)\bullet\ldots\bullet \\
&\qquad (H_{30}+\overline{P}_{31})\bullet(H_{31}+\overline{C}_{in}) \qquad (9)
\end{aligned}$$

The first two reductions are obvious. The remaining iterations can be proved by induction by showing that:

$$\begin{aligned}
&(C_{out}+\overline{P}_0)\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet(H_{k-1}+\overline{P}_k)\bullet(H_k+\overline{P}_{k+1}) = \\
&(C_{out}+\overline{P}_0)\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet(H_{k-1}+\overline{P}_k)\bullet\overline{P}_{k+1} \\
&\quad + \quad H_0\bullet H_1\bullet\ldots\bullet H_k\bullet C_{out} \qquad (10)
\end{aligned}$$

the left side of Eq.(10) is expanded to:

$$\begin{aligned}
&(C_{out}+\overline{P}_0)\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet(H_{k-1}+\overline{P}_k)\bullet\overline{P}_{k+1} \\
&+(\ \downarrow\ )\bullet(\ \downarrow\ )\bullet\ldots\bullet(\ \downarrow\ )\bullet H_k
\end{aligned}$$

The first term of the expansion corresponds to the first term of the right side of Eq.(10). The second term of the expansion is multiplied out to yield the second term of the right side of Eq.(10), since $\overline{P}_i\bullet H_i=0$. Again, for the last iteration, $\overline{P}_{32} = \overline{C}_{in}$. Q.E.D.

Since C_{out} includes G_0 as a term according to Eq.(7),

$$(C_{out}+\overline{P}_0) = (C_{out}+G_0+\overline{P}_0) = (C_{out}+\overline{H}_0)$$

so that

$$\begin{aligned}
ZEROS = &(C_{out}+\overline{H}_0)\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet(H_{30}+\overline{P}_{31})\bullet \\
&(H_{31}+\overline{C}_{in}) \qquad (11)
\end{aligned}$$

Generally, an adder design produces the single-bit functions, H, P, and G, earlier than C_{out}. Eq.(11) may therefore be restated as in Eq.(12) to permit a single level of delay between C_{out} and ZEROS.

$$\begin{aligned}
ZEROS = &\ C_{out}\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet(H_{30}+\overline{P}_{31})\bullet(H_{31}+\overline{C}_{in}) \\
&+ H_0\bullet(\ \downarrow\ )\bullet\ldots\bullet(\ \downarrow\ )\bullet(\ \downarrow\ ) \qquad (12)
\end{aligned}$$

ZEROS may also be expressed as a function of some intermediate carry, C_k, as follows:

$$\begin{aligned}
ZEROS = &(\overline{H}_0+G_1+\ldots+G_{k-1}+C_k)\bullet(H_0+\overline{P}_1)\bullet\ldots\bullet \\
&(H_{30}+\overline{P}_{31})\bullet(H_{31}+\overline{C}_{in}) \qquad (13)
\end{aligned}$$

The proof is similar to those for Eqs.(4) and (11).

EXTENSION TO RADICES >2

The zero-sum detect method can be generalized to radices ≥ 2. It is particularly useful for decimal.

Let r = integer radix ≥ 2
A = $(A_0, \ldots, A_{n-1})$=addend of an n-digit adder
B = $(B_0, \ldots, B_{n-1})$=augend
C_{in} = input carry to adder

where the subscripts, 0 through n-1, refer to digit positions high-to-low-order, respectively.

The following functions of a digit position, k, will be used:

$(k)_{r-1}$ = the normalized algebraic sum, A_k+B_k, of digit position k equals to r-1 (the weight of digit position k is normalized to $r^0=1$)

$(k)_r$ = the normalized algebraic sum, A_k+B_k, of digit position k equals to r.

$(k)_0$ = the normalized algebraic sum, A_k+B_k, of digit position k equals to 0.

For binary (r=2), the functions $(k)_{r-1}$, $(k)_r$, and $(k)_0$ correspond to H_k, G_k, $\overline{P}_k$, respectively.

The conditions for generating a sum of all zeros are enumerated in Eq. (14).

$$\begin{aligned}
ZEROS = &\ (0)_0\bullet(1)_0\bullet(2)_0\bullet\ldots\bullet(n-1)_0\bullet\overline{C}_{in} \\
&+(0)_r\bullet(1)_0\bullet(2)_0\bullet\ldots\bullet(n-1)_0\bullet\overline{C}_{in} \\
&+(0)_{r-1}\bullet(1)_r\bullet(2)_0\bullet\ldots\bullet(n-1)_0\bullet\overline{C}_{in} \\
&\quad \vdots \qquad (14) \\
&+(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet(n-1)_r\bullet\overline{C}_{in} \\
&+(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet(n-1)_r\bullet C_{in}
\end{aligned}$$

In words, the zero-sum labeled ZEROS is generated if one of the three conditions is satisfied:

1. The algebraic sum of each addend/augend pair is 0 and $C_{in}=0$.
2. The normalized algebraic sum of one and only one addend/augend pair is r that generates a carry in the respective digit position. In each trailing digit position, the addend/augend pair produces a normalized algebraic sum of 0 and $C_{in}=0$. In each leading digit position, the addend/augend pair produces a normalized algebraic sum of r-1 that permits the generated carry to pass through leaving 0's in its wake.
3. C_{in} is 1 which produces a carry. In each digit position the addend/augend pair produces a normalized algebraic sum of r-1 that permits the generated carry to pass through leaving 0's in its wake.

The first two terms of Eq.(14) are combined to yield Eq. (15).

$$\begin{aligned}\text{ZEROS} = {} & (0)_{0,r}\bullet(1)_0\bullet(2)_0\bullet\ldots\bullet(n-1)_0\bullet\bar{C}_{in}\\ & +(0)_{r-1}\bullet(1)_r\bullet(2)_0\bullet\ldots\bullet(n-1)_0\bullet C_{in}\\ & \vdots\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet(n-1)_r\bullet\bar{C}_{in}\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet(n-1)_r\bullet C_{in}\end{aligned} \qquad (15)$$

where $(0)_{0,r}$ means that the normalized algebraic sum, (A_0+B_0), is equal to 0 or r.

The first two terms of Eq.(15) are combined to yield Eq.(16), noting that $(0)_{0,r}\bullet(0)_{r-1} = (1)_r\bullet(1)_0 = 0$.

$$\begin{aligned}\text{ZEROS} = {} & [(0)_{0,r}+(1)_r]\bullet[(0)_{r-1}+(1)_0]\bullet(2)_0\bullet\ldots\bullet\\ & (n-1)_0\bullet\bar{C}_{in}\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_r\bullet\ldots\bullet\\ & (n-1)_0\bullet\bar{C}_{in}\\ & \vdots\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet\\ & (n-1)_r\bullet\bar{C}_{in}\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\ldots\\ & (n-1)_{r-1}\bullet C_{in}\end{aligned} \qquad (16)$$

The process of combining the first two terms is repeated until only a single term remains, as shown in Eq.(17).

$$\begin{aligned}\text{ZEROS} = {} & [(0)_{0,r}+(1)_r+\ldots+(n-1)_r+C_{in}]\bullet\\ & [(0)_{r-1}+(1)_0]\bullet\ldots\bullet[(n-2)_{r-1}+(n-1)_0]\bullet\\ & [(n-1)_{r-1}+\bar{C}_{in}]\end{aligned} \qquad (17)$$

The iteration can be proved by induction by showing that:

$$\begin{aligned}& [(0)_{0,r}+(1)_r+\ldots+(k)_r+(k+1)_r]\bullet[(0)_{r-1}+(1)_0]\\ & \bullet\ldots\bullet[(k-1)_{r-1}+(k)_0]\bullet[(k)_{r-1}+(k+1)_0] =\\ & [(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet\\ & [(k-1)_{r-1}+(k)_0]\bullet(k+1)_0+(0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet\\ & (k)_{r-1}\bullet(k+1)_r\end{aligned} \qquad (18)$$

The left side of Eq.(18) is expanded to:

$$\begin{aligned}& [(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet\\ & [(k-1)_{r-1}+(k)_0]\bullet(k)_{r-1}\\ & +[(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet\\ & [(k-1)_{r-1}+(k)_0]\bullet(k+1)_0\\ & +(k+1)_r\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet[(k-1)_{r-1}+(k)_0]\bullet\\ & (k)_{r-1}\\ & +(k+1)_r\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet[(k-1)_{r-1}+(k)_0]\bullet\\ & (k+1)_0\end{aligned}$$

The first term of the expansion is shown to be equal to 0, as follows:

$$\begin{aligned}& [(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet\\ & [(k-1)_{r-1}+(k)_0]\bullet(k)_{r-1}\\ & =[(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet(k-1)_{r-1}\bullet(k)_{r-1}\\ & \vdots\\ & =[(0)_{0,r}+(1)_r+\ldots+(k)_r]\bullet(0)_{r-1}\bullet\ldots\bullet(k-1)_{r-1}\bullet(k)_{r-1}\end{aligned} \quad \left.\begin{array}{l}\\ \\ \\ \\ \end{array}\right\} \text{since } (i)_0\bullet(i)_{r-1}=0$$

$= 0$ since $(0)_{0,r}\bullet(0)_{r-1} = (i)_r\bullet(i)_{r-1} = 0$

The second term of the expansion corresponds to the first term of the right side of Eq.(18). The third term of the expansion reduces to the second term of the right side of Eq.(18), as follows:

$$\begin{aligned}& (k+1)_r\bullet[(0)_{r-1}+(1)_0]\bullet\ldots\bullet[(k-1)_{r-1}+(k)_0]\bullet(k)_{r-1}\\ & = (k+1)_r\bullet[\ \downarrow\ ]\bullet\ldots\bullet(k-1)_{r-1}\bullet(k)_{r-1}\\ & \vdots\\ & = (k+1)_r\bullet(0)_{r-1}\bullet\ldots\bullet(k-1)_{r-1}\bullet(k)_{r-1}\end{aligned}$$

since $(i)_0\bullet(i)_{r-1}=0$

The fourth term of the expansion is also equal to 0, since $(k+1)_r\bullet(k+1)_0 = 0$.

Eq.(18) also applies to the last iteration where $(k+1)_r \equiv C_{in}$ and $(k+1)_0 \equiv \bar{C}_{in}$. Q.E.D.

The alternate implementation that makes use of the output carry, C_{out}, to replace the expression:

$$[(1)_r+\ldots+(n-1)_r+C_{in}]$$

is not applicable to r>2. The reason is apparent from Eq.(19). For r=2, $(k)_{\geq r}=(k)_r$, while for r>2, $(k)_{\geq r}\neq(k)_r$.

$$\begin{aligned}C_{out} = {} & (0)_{\geq r}\\ & +(0)_{r-1}\bullet(1)_{\geq r}\\ & \vdots\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet(n-1)_{\geq r}\\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet(n-1)_{r-1}\bullet C_{in}\end{aligned} \qquad (19)$$

Note that $(k)_{\geq r}$ means that the normalized algebraic sum, (A_k+B_k), is equal to or larger than r with a maximum possible value of 2(r-1).

ZERO-SUM DETECT OF AN n-DIGIT DECIMAL SUM

For decimal (r=10), Eq.(17) becomes:

$$\begin{aligned}\text{ZEROS} = {} & [(0)_{0,10}+(1)_{10}+\ldots+(n-1)_{10}+C_{in}]\bullet\\ & [(0)_9+(1)_0]\bullet\ldots\bullet[(n-2)_9+(n-1)_0]\bullet\\ & [(n-1)_9+\bar{C}_{in}]\end{aligned} \qquad (20)$$

Let (A_8,A_4,A_2,A_1) and (B_8,B_4,B_2,B_1) be the BCD (binary-coded decimal) representation of the decimal addend and augend, respectively. The subscripts, 8,4,2, and 1 refer to the weight of the respective bits of the decimal digit. Then,

$$(k)_0=(\overline{A}_8\bullet\overline{B}_8)\bullet(\overline{A}_4\bullet\overline{B}_4)\bullet(\overline{A}_2\bullet\overline{B}_2)\bullet(\overline{A}_1\bullet\overline{B}_1) \quad (21)$$

$$(k)_9 = [(A_8 \forall B_8)\bullet(\overline{A}_4\bullet\overline{B}_4)+(A_4\bullet B_4)]\bullet(\overline{A}_2\bullet\overline{B}_2) + (A_4\forall B_4)\bullet(A_2\bullet B_2)\}\bullet(A_1\forall B_1) \quad (22)$$

$$(k)_{10}= \{[(A_8\forall B_8)\bullet(\overline{A}_4\bullet\overline{B}_4)+(A_4\bullet B_4)]\bullet(\overline{A}_2\bullet\overline{B}_2) + (A_4\forall B_4)\bullet(A_2\bullet B_2)\bullet(A_1\bullet B_1) + [(A_8\forall B_8)\bullet(\overline{A}_4\bullet\overline{B}_4)+(A_4\bullet B_4)] \bullet(A_2\forall B_2)\bullet(\overline{A}_1\bullet\overline{B}_1) \quad (23)$$

Eqs.(21),(22), and (23) are the individual digit functions that are readily derived from a truth table. It is assumed that an input digit has a range of values 0-9 while values 10-15 are don't-care conditions.

DETECTION OF A SUM WITH DIMINISHED-RADIX DIGITS

The methods of deriving expressions for zero-sum detection are now applied to deriving comparable expressions for detecting a sum with diminished-radix digits (digits of radix-less-one). For binary it means detecting a string of ones, for decimal a string of nines, etc..

The corresponding equations will be numbered identically to the equations for zero-sum detection with the letter A appended.

Again, we begin with a 32-bit adder. The conditions for generating a sum are enumerated in Eq. (1A). In words, a sum of all ones labeled ONES is present if one of the three conditions is satisfied:

1. All inputs are ones (G and C_{in}) so that every bit position generates and accepts a carry to produce a sum equal to one.
2. In one and only one bit position are both addend and augend equal to zero ($\overline{P}$) converting a carry into this bit position into a sum bit equal to one and precluding a carry from this bit position. At the same time, the trailing bit position inputs as well as the input carry are all ones, so that they produce trailing sum bits of ones and a carry into the bit position of condition $\overline{P}$. Also, each of the leading bit positions produces a half-sum H that becomes a sum bit of one in the absence of a carry.
3. All bit positions produce H and $C_{in}=0$ so that no carries are produced and sums remain one.

$$\begin{aligned}\text{ONES} = {} & G_0\bullet G_1\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & +\overline{P}_0\bullet G_1\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & +H_0\bullet\overline{P}_1\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & \vdots\\ & +H_0\bullet H_1\bullet H_2\bullet\ldots\bullet\overline{P}_{31}\bullet C_{in}\\ & +H_0\bullet H_1\bullet H_2\bullet\ldots\bullet H_{31}\bullet\overline{C}_{in}\end{aligned} \quad (1A)$$

The first two terms of Eq.(1A) are combined to yield Eq.(2A), since $(G_0+\overline{P}_0)=\overline{H}_0$

$$\begin{aligned}\text{ONES} = {} & \overline{H}_0\bullet G_1\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & +H_0\bullet\overline{P}_1\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & \vdots\\ & +H_0\bullet H_1\bullet H_2\bullet\ldots\bullet\overline{P}_{31}\bullet C_{in}\\ & +H_0\bullet H_1\bullet H_2\bullet\ldots\bullet H_{31}\bullet\overline{C}_{in}\end{aligned} \quad (2A)$$

The first two terms of Eq.(2A) are combined to yield Eq.(3A), noting that $\overline{H}_0\bullet H_0=\overline{P}_1\bullet G_1=0$.

$$\begin{aligned}\text{ONES} = {} & (\overline{H}_0+\overline{P}_1)\bullet(H_0+G_1)\bullet G_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & + H_0\bullet H_1\bullet P_2\bullet\ldots\bullet G_{31}\bullet C_{in}\\ & \vdots\\ & + H_0\bullet H_1\bullet H_2\bullet\ldots\bullet\overline{P}_{31}\bullet C_{in}\\ & + H_0\bullet H_1\bullet H_2\bullet\ldots\bullet H_{31}\bullet\overline{C}_{in}\end{aligned} \quad (3A)$$

The process of combining the first two terms is repeated until only a single term remains, as shown in Eq.(4A).

$$\text{ONES} = (\overline{H}_0+\overline{P}_1+\ldots+\overline{P}_{31}+\overline{C}_{in})\bullet(H_0+G_1)\bullet\ldots\bullet (H_{30}+G_{31})\bullet(H_{31}+C_{in}) \quad (4A)$$

The iteration can be proved by induction by showing that

$$(\overline{H}_0+\overline{P}_1+\ldots+\overline{P}_k+\overline{P}_{k+1})\bullet(H_0+G_1)\bullet\ldots\bullet(H_{k-1}+G_k)\bullet(H_k+G_{k+1})=(\overline{H}_0+\overline{P}_1+\ldots+\overline{P}_k)\bullet(H_0+G_1)\bullet\ldots\bullet (H_{k-1}+G_k)\bullet G_{k+1} + H_0\bullet\ldots\bullet H_k\bullet\overline{P}_{k+1} \quad (5A)$$

Eq.(5A) is proved in a manner similar to Eq.(5). Eq.(5A) also applies to the last iteration where $\overline{P}_{k+1}\equiv\overline{C}_{in}$ and $G_{k+1}\equiv C_{in}$ Q.E.D.

An alternate implementation makes use of the output carry of the adder, C_{out}, to replace the expression,

$$(\overline{P}_1+\ldots+\overline{P}_{31}+\overline{C}_{in}).$$

The output carry can be generated early; i.e., prior to the sum, so that ONES is available concurrently with or earlier than the sum. The alternate equation is:

$$\text{ONES}=(\overline{C}_{out}+G_0)\bullet(H_0+G_1)\bullet\ldots\bullet (H_{30}+G_{31})\bullet(H_{31}+C_{in}) \quad (6A)$$

It is derived as follows: $\overline{C}_{out}$ can be expressed as in Eq.(7A).

$$\begin{aligned}\overline{C}_{out} = {} & \overline{P}_0 \\ & +H_0 \bullet \overline{P}_1 \\ & \cdot \\ & \cdot \\ & \cdot \\ & +H_0 \bullet H_1 \bullet \ldots \bullet \overline{P}_{31} \\ & +H_0 \bullet H_1 \bullet \ldots \bullet H31 \bullet \overline{C}_{in}\end{aligned} \qquad (7A)$$

Eq.(7A) is now substituted for each $\overline{P}$ in Eq.(1A) to yield Eq.(8A). For each substitution of a $\overline{P}_k$ with $\overline{C}_{out}$, only the

$$\begin{aligned}ONES = {} & G_0 \bullet G1 \bullet G_2 \bullet \ldots \bullet G31 \bullet C_{in} \\ & +\overline{C}_{out} \bullet G_1 \bullet G_2 \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & +H_0 \bullet \overline{C}_{out} \bullet G_2 \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & \cdot \\ & \cdot \\ & \cdot \\ & +H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet \overline{C}_{out} \bullet C_{in} \\ & +H_0 \bullet H_1 \bullet H_2 \bullet \ldots \bullet H_{31} \bullet \overline{C}_{out}\end{aligned} \qquad (8A)$$

corresponding term in Eq.(7A), (i.e., the term containing $\overline{P}_k$) is relevant; the other terms drop out because H_k, $\overline{P}_k$, and G_k are mutually exclusive.

Eq. (8A) is now reduced iteratively by combining the first two terms.

$$\begin{aligned}ONES = {} & (\overline{C}_{out}+G_0) \bullet G_1 \bullet G_2 \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & + \quad H_0 \bullet \overline{C}_{out} \bullet G_2 \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & +\ldots \\ & = (\overline{C}_{out}+G_0) \bullet (H_0+G_1) \bullet G_2 \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & + \quad H_0 \bullet H_1 \bullet \overline{C}_{out} \bullet \ldots \bullet G_{31} \bullet C_{in} \\ & +\ldots \\ & \cdot \\ & \cdot \\ & \cdot \\ & = (\overline{C}_{out}+G_0) \bullet (H_0+G_1) \bullet \ldots \bullet (H_{30}+G_{31}) \\ & \bullet (H_{31}+C_{in})\end{aligned} \qquad (9A)$$

The first two reductions are obvious. The remaining iterations can be proved by induction by showing that:

$$\begin{aligned}& (\overline{C}_{out}+G_0) \bullet (H_0+G_1) \bullet \ldots \bullet (H_{k-1}+G_k) \bullet (H_k+G_{k+1}) \\ & = (\overline{C}_{out}+G_0) \bullet (H_0+G_1) \bullet \ldots \bullet (H_{k-1}+G_k) \bullet G_{k+1} \\ & + \quad H_0 \bullet H_1 \bullet \ldots \bullet H_k \bullet \overline{C}_{out}\end{aligned} \qquad (10A)$$

The left side of Eq.(10A) is expanded to:

$$\begin{aligned}& (\overline{C}_{out}+G_0) \bullet (H_0+G_1) \bullet \ldots \bullet (H_{k-1}+G_k) \bullet G_{k+1} \\ & +(\downarrow) \bullet (\downarrow) \bullet \ldots \bullet (\downarrow) \bullet H_k\end{aligned}$$

The first term of the expansion corresponds to the first term of the right side of Eq. (10A). The second term of the expansion is multiplied out to yield the second term of the right side of Eq.(10A), since $G_k \bullet H_k = 0$. Again, for the last iteration, $G_{32} \equiv C_{in}$. Q.E.D.

Since $\overline{C}_{out}$ includes $\overline{P}_0$ as a term according to Eq.(7A),

$$(\overline{C}_{out}+G_0) = (\overline{C}_{out}+\overline{P}_0+G_0) = (\overline{C}_{out}+\overline{H}_0)$$

so that

$$\begin{aligned}ONES = {} & (\overline{C}_{out}+\overline{H}_0) \bullet (H_0+G_1) \bullet \ldots \bullet \\ & (H_{30}+G_{31}) \bullet (H_{31}+C_{in})\end{aligned} \qquad (11A)$$

or

$$\begin{aligned}ONES = {} & \overline{C}_{out} \bullet (H_0+G_1) \bullet \ldots \bullet (H_{30}+G_{31}) \bullet (H_{31}+C_{in}) \\ & +\overline{H}_0 \bullet (\downarrow) \bullet \ldots \bullet (\downarrow) \bullet (\downarrow)\end{aligned} \qquad (12A)$$

ONES may also be expressed as a function of some intermediate carry, C_k, as follows:

$$\begin{aligned}ONES = {} & (\overline{H}_0+\overline{P}_1+\ldots+\overline{P}_{k-1}+\overline{C}_k) \bullet (H_0+G_1) \\ & \bullet \ldots \bullet (H_{30}+G_{31}) \bullet (H_{31}+C_{in})\end{aligned} \qquad (13A)$$

The proof is similar to those for Eqs.(4A), and (11A).

The method for detecting a sum of all ones is now generalized to integer radices ≥ 2. The corresponding condition is for each sum digit to be equal to radix-less-one when normalized to the lowest integer position ($radix^0=1$).

The following functions of a digit position, will be used:

$(k)_{2(r-1)}$ = the normalized algebraic sum, (A_k+B_k), is equal to $2(r-1)$

$(k)_{r-2}$ = the normalized algebraic sum, (A_k+B_K), is equal to $r-2$

$(k)_{r-1}$ = the normalized algebraic sum, (A_k+B_k), is equal to $r-1$

For binary ($r=2$), the functions $(k)_{2(r-1)}$, $(k)_{r-2}$, and $(k)_{r-1}$ correspond to $G_k, \overline{P}_k, H_k$, respectively.

The conditions for generating a sum of diminished-radix digits are enumerated in Eq.(14A). The function will be referred to as DRD.

$$\begin{aligned}DRD = {} & (0)_{2(r-1)} \bullet (1)_{2(r-1)} \bullet (2)_{2(r-1)} \\ & \bullet \ldots \bullet (n-1)_{2(r-1)} \bullet C_{in} \\ & +(0)_{r-2} \bullet (1)_{2(r-1)} \bullet (2)_{2(r-1)} \\ & \bullet \ldots \bullet (n-1)_{2(r-1)} \bullet C_{in} \\ & +(0)_{r-1} \bullet (1)_{r-2} \bullet (2)_{2(r-1)} \\ & \bullet \ldots \bullet (n-1)_{2(r-1)} \bullet C_{in} \\ & \cdot \\ & \cdot \\ & \cdot \\ & +(0)_{r-1} \bullet (1)_{r-1} \bullet (2)_{r-1} \bullet \ldots \bullet (n-1)_{r-2} \bullet C_{in} \\ & +(0)_{r-1} \bullet (1)_{r-1} \bullet (2)_{r-1} \bullet \ldots \bullet (n-1)_{r-1} \bullet \overline{C}_{in}\end{aligned} \qquad (14A)$$

In words DRD is generated if one of the three conditions is satisfied:

1. In each digit position, the normalized algebraic sum (A_k+B_k) is $2(r-1)$ and $C_{in}=1$.

Each digit position generates a carry with a remainder of r-2 which combined with the carry entering the digit produces a final sum of r-1.

2. The normalized algebraic sum in one and only one digit position (A_k+B_k) is r-2. In each trailing digit position (i>k) the normalized algebraic sum is 2(r-1) that produces a carry; $C_{in}=1$; and in each leading digit position (i<k) the normalized algebraic sum is r-1. Therefore, the respective carry that enters a trailing digit position as well as the digit position k produces a sum digit of r-1. Since no carry is generated in digit position k, the leading digit positions retain sums of r-1.
3. C_{in} is 0 and the normalized algebraic sum in each digit position (A_k+B_k) is r-1.

The first two terms of Eq.(14A) are combined to yield Eq. (15A).

$$\begin{aligned}DRD = {} & [(0)_{2(r-1)}+(0)_{r-2}]\bullet(1)_{2(r-1)}\bullet(2)_{2(r-1)} \\ & \bullet\ldots\bullet(n-1)_{2(r-1)}\bullet C_{in} \\ & + (0)_{r-1}\bullet(1)_{r-2}\bullet(2)_{2(r-1)} \\ & \bullet\ldots\bullet(n-1)_{2(r-1}\bullet C_{in} \\ & \vdots \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1} \\ & \bullet\ldots\bullet(n-1)_{r-2}\bullet C_{in} \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1} \\ & \bullet\ldots\bullet(n-1)_{r-1}\bullet\overline{C}_{in}\end{aligned} \qquad (15A)$$

The first two terms of Eq.(15A) are combined to yield Eq.(16A) noting that

$$[(0)_{2(r-1)}+(0)_{r-2}]\bullet(0)_{r-1}=(1)_{r-2}\bullet(1)_{2(r-1)}=0$$

$$\begin{aligned}DRD = {} & [(0)_{2(r-1)}+(0)_{r-2}+(1)_{r-2}] \\ & \bullet[(0)_{r-1}+(1)_{2(r-1)}]\bullet(2)_{2(r-1)}\bullet\ldots\bullet \\ & (n-1)_{2(r-1)}\bullet C_{in} \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-2}\bullet\ldots\bullet \\ & (n-1)_{2(r-1)}\bullet C_{in} \\ & \vdots \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet \\ & (n-1)_{r-2}\bullet C_{in} \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet(2)_{r-1}\bullet\ldots\bullet \\ & (n-1)_{r-1}\bullet\overline{C}_{in}\end{aligned} \qquad (16A)$$

The process of combining the first two terms is repeated until only a single term remains, as shown in Eq.(17A).

$$\begin{aligned}DRD = {} & [(0)_{2(r-1)}+(0)_{r-2}+\ldots+(n-1)_{r-2}+\overline{C}_{in}] \\ & \bullet[(0)_{r-1}+(1)_{2(r-1)}]\bullet\ldots\bullet \\ & [(n-2)_{r-1}+(n-1)_{2(r-1)}]\bullet[(n-1)_{r-1}+C_{in}]\end{aligned} \qquad (17A)$$

The iteration can be proved by induction by showing that:

$$\begin{aligned}& [(0)_{2(r-1)}+(0)_{r-2}+\ldots+(k)_{r-2}+(k+1)_{r-2}]\bullet \\ & [(0)_{r-1}+(1)_{2(r-1)}]\bullet\ldots\bullet \\ & [(k-1)_{r-1}+(k)_{2(r-1)}]\bullet[(k)_{r-1}+(k+1)_{2(r-1)}] \\ & =[(0)_{2(r-1)}+(0)_{r-2}+\ldots+(k)_{r-2}] \\ & \bullet[(0)_{r-1}+(1)_{2(r-1)}]\bullet\ldots\bullet \\ & [(k-1)_{r-1}+(k)_{2(r-1)}]\bullet(k+1)_{2(r-1)} \\ & + (0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet(k)_{r-1}\bullet(k+1)_{r-2}\end{aligned} \qquad (18A)$$

The proof is similar to that for Eq.(18).

Eq.(18A) also applies to the last iteration where $(n)_{r-2}\equiv\overline{C}_{in}$ and $(n)_{2(r-1)}\equiv C_{in}$.

Q.E.D.

The alternate implementation that makes use of the output carry, C_{out}, to replace the expression,

$$[(0)_{r-2}+\ldots+(n-1)_{r-2}+\overline{C}_{in}]$$

is not applicable to r>2. The reason is apparent from Eq.(19A). For r=2, $(k)_{<(r-1)}=(k)_{r-2}$, while for r>2, $(k)_{<(r-1)}\neq(k)_{r-2}$

$$\begin{aligned}\overline{C}_{out} = {} & (0)_{<(r-1)} \\ & +(0)_{r-1}\ (1)_{<(r-1)} \\ & \vdots \\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet(n-1)_{<(r-1)} \\ & +(0)_{r-1}\bullet(1)_{r-1}\bullet\ldots\bullet(n-1)_{r-1}\bullet\overline{C}_{in}\end{aligned} \qquad (19A)$$

Note that $(k)_{<(r-1)}$ means that the normalized algebraic sum of a digit position (A_k+B_k) is less than r-1 with a minimum value of 0.

For decimal (r=10), eq.(17A) becomes:

$$\begin{aligned}NINES = {} & [(0)_{18}+(0)_{8}+\ldots+(n-1)_{8}+\overline{C}_{in}] \\ & \bullet[(0)_{9}+(1)_{18}]\bullet\ldots\bullet[(n-2)_{9}+(n-1)_{18}] \\ & \bullet[(n-1)_{9}+C_{in}]\end{aligned} \qquad (20A)$$

Let (A_8, A_4, A_2, A_1) and (B_8, B_4, B_2, B_1) be the BCD (binary-coded decimal) representation of the decimal addend and augend digits respectively. The subscripts (8,4,2,1) refer to the weight of the respective bits comprising a decimal digit.

Then,

$$\begin{aligned}(k)_9 = {} & \{[(A_8\forall B_8)\bullet(\overline{A}_4\bullet\overline{B}_4)+(A_4\bullet B_4)] \\ & \bullet(\overline{A}_2\bullet\overline{B}_2)+(A_4\forall B_4)\bullet(A_2\bullet B_2)\}\bullet(A_1\forall B_1)\end{aligned} \qquad (21A)$$

$$(k)_8 = \{[(A_8 \forall B_8) \bullet (\bar{A}_4 \bullet \bar{B}_4) + (A_4 \bullet B_4)] \bullet (\bar{A}_2 \bullet \bar{B}_2) + (A_4 \forall B_4) \bullet (A_2 \bullet B_2)\} \bullet (\bar{A}_1 \bullet \bar{B}_1) + (\bar{A}_8 \ \bar{B}_8) \bullet (A_4 \forall B_4) \bullet (A_2 \forall B_2) \bullet (A_1 \ B_1) \quad (22A)$$

$$(k)_{18} = (A_8 \bullet B_8) \bullet (A_1 \bullet B_1) \quad (23A)$$

Eqs.(21A),(22A), and (23A) are the individual digit functions that are readily derived from a truth table.

References

1. L.C. Queen, "Prediction Adding Circuit," IBM Technical Disclosure Bulletin, Vol. 13, No.9(Feb.1971),pp.2477-2478.
2. A Weinberger and J.L. Smith,"A One-Microsecond Adder Using One-Microsecond Circuitry,"IRE Trans. on Electronic Computers, Vol.EC-5,No.2(June 1956), pp. 65-73.
3. M.S. Schmookler and A. Weinberger, "High-Speed Decimal Addition," IEEE Trans. on Computers,Vol.C-20,No.8 (Aug. 1971,pp.862-868.