Eliminating cascading carries

In this note we describe an adder for numbers in the following decimal representation

(0) $\quad n = \langle \Sigma i : 0 \leq i : d_i \cdot 10^i \rangle$

(1) $\quad -5 \leq d_i \leq 6 \quad$ for all i .

All integers can be represented this way, but note that the representation is not necessarily unique: for instance, 26 can be represented in two decimals by (2, 6) as well as by (3, −4) .

In our "adder", which can calculate $\pm a \pm b$, the sum of two digits ranges from −12 through +12 ; its addition table represents these values as $c \cdot 10 + s$ —from "carry" and "sum digit"— , with c, s satisfying

(2) $\quad -1 \leq c \leq +1 \quad$ and $\quad -4 \leq s \leq +5$,

i.e., from −12 through −5 , $c = -1$,
from −4 through +5 , $c = 0$, and
from +6 through +12 , $c = +1$.

Because (2) excludes for s the extreme digit values −5 and +6 and $|c| \leq 1$, each sum digit can absorb a carry from the right without generating a new one. In long parallel adders the problem of carry propagation has thus been eliminated; alternatively we can add from left to right with a "look-ahead" of only 1 position.

Remark If we so desire, we may replace (1) by the weaker, symmetric $-6 \leq d_i \leq +6$. With (2) weakened accordingly, some freedom in the addition table is introduced. (End of Remark)

Please note that (i) the representation of zero is unique, and (ii) the sign of a non-zero value is determined by the sign of its most-significant non-zero digit.

The above, which was designed decades ago, was inspired by the implementation of the end-around carry in the serial adder of the ARMAC. Since the advent of systolic arrays it must look like something familiar.

Austin, 23 September 1999

prof. dr Edsger W. Dijkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712-1188
USA