QUICKADDER:
A NEW SOLUTION TO AN OLD PROBLEM

TR - 92

March 1979

Jayadev Misra
University of Texas at Austin
Austin, Texas 78712

1. Introduction

We propose a new design for fixed point adders. The speed of the proposed circuit compares favorably with the ones in practice [1]. More importantly, the major part of the circuit is built out of one elementary functional block, called the router, which routes one of its input data to its output lines. $n\lceil \log_2 n \rceil$ routers are needed for adding n bit numbers. The interconnection among routers is quite simple; a router can have at most 2 input and 2 output lines and hence the total number of interconnection among routers does not exceed $2n\lceil \log_2 n \rceil$. Besides the routers, we use n 3-bit adders, each of which performs modulo 2 addition of the three inputs and produces one bit of output. These adders are used only in the last time unit of computation to complete the additon. One 3-bit adder is used for every bit position. We believe that the speed and simplicity of the circuit makes it an attractive alternative to the ones found in use.

2. An Observation About Addition

Let $A = (A_n,\ldots,A_1)$ and $B = (B_n,\ldots,B_1)$ be two n-bit numbers which are to be added. A_i denotes the ith lower order bit in A. Define a vector of bits M as follows:

$$M_i = \begin{cases} 1, \text{ if there exists } j,\ i > j \geq 1,\ A_j = B_j = 1 \text{ and for all } k,\ i > k > j \\ \qquad \text{either } A_k = 0 \text{ or } B_k = 0. \\ 0, \text{ otherwise} \end{cases}$$

M_i is the bit value (0 or 1) such that for some position j lower than i, both A_j, B_j are equal to M_i and j is the highest such position. Note

that if for some i, $A_j \neq B_j$ for every j lower than i, then $M_i = 0$.
In the foloowing definitions, we assume that $A_0 = B_0 = 0$. Let C denote the n-bit sum of A, B, i.e. any overflow bit is ignored. The full sum of A, B can be realized by letting the fictitious $A_{n+1} = B_{n+1} = 0$ and using an (n+1) bit adder.

2.1 Observation: $C_i = A_i + B_i + M_i$, where + denotes modulo 2 addition.

Let $M_i = A_j$, $i > j$.

If $M_i = 0$ then there is j lower than i such that $A_j = B_j = 0$. Hence any carry generated from bit positions lower than j will be "absorbed" at j; futhermore no carry can be generated at any bit position k, $i > k > j$, since either A_k or $B_k = 0$. Similarly if $M_i = 1$, any carry from positions lower than j will be "propagated" to i; otherwise a carry will be "generated" at the jth bit position and propagated to i.

3. Design of the Circuit

The addition circuit will involve computing M_i for every i and then computing $C_i = A_i + B_i + M_i$, in one parallel step for every bit position i. In the following, we design a circuit for computing M. Though it is possible to compute M in $\lceil \log_2 n \rceil$ parallel steps, for ease of description we show a circuit taking $\lceil \log_2 n \rceil + 1$ steps. Similarly, this simplified circuit uses $n(\lceil \log_2 n \rceil + 1)$ routers, though it is possible to eliminate n of these routers.

The idea behind the circuit is based on a recursive definition of M_i. Define $e_{it} \in \{0,1,d\}$ as the value of the first matching pair of bits A_j, B_j, within t bit positions lower than i. Formally for $1 \leq i \leq n$ and $1 \leq t \leq i$,

$$e_{it} = \begin{cases} A_j\ , & i > j \geq i - t,\ \text{if } A_j = B_j \text{ and for all } k,\ i > k > j, \\ & A_k \neq B_k \\ d\ , & \text{otherwise (if no such } j \text{ exists)} \end{cases}$$

For $t > i$, let $e_{it} = e_{ii}$, for every $1 \leq i \leq n$.

3.1 Ovservation

1. $e_{i,1} = \begin{cases} A_{i-1}\ , \text{ if } A_{i-1} = B_{i-1} \\ d\ , \text{ otherwise} \end{cases}$, $1 \leq i \leq n$.

2. $e_{i,2t} = \begin{cases} e_{it}, \text{if } e_{it} \neq d \\ e_{i-t,t}, \quad \text{otherwise} \end{cases}$, $1 \leq i \leq n$.

3. $M_i = e_{in}$

The circuit given below computes $(e_{i,2^j})$, for $j = 0,1,\ldots\lceil\log_2 n\rceil$, for every i. Hence from observation (3) above, M_i for every i gets computed after $\lceil\log_2 n\rceil + 1$ parallel steps.

A router element has at most 2 input lines and at most 2 output lines. Each input line receives data from $\{0,1,d\}$ and produces identical output on both output lines which are also from $\{0,1,d\}$. If a router has only one input line then the data received is simply routed to the output. If a router has 2 input lines then one of them is designated the "bottom" line and the other the "top" line. The data on bottom line is routed to output if it is not equal to d; otherwise the data on top line is routed.

The circuit for computing M is arranged in the form of a matrix. The matrix has n rows numbered 1,2,...n and $\lceil \log_2 n \rceil + 1$ columns numbered $(0,1,\ldots \lceil \log_2 n \rceil)$. A router element is placed at every matrix position. Let R_{ij} denote the router placed in the ith row and jth cloumn position. Intuitively, R_{ij} will produce $e_{i,2^j}$ as output. This determines the connection pattern among the routers according to the observation (3.1). For $j > 0$: R_{ij} receives inputs from $R_{i,j-1}$ (bottom line) and $R_{i-2^{j-1},j-1}$ (top line). For $j < \lceil \log_2 n \rceil$, $R_{i,j}$ has output lines to $R_{i,j+1}$ and $R_{i+2^j,j+1}$. If the corresponding routers are undefined, there is no line incident from or to them.

Input to the circuit is received by the routers $R_{i,0}$, $1 \leq i \leq n$, each of which has one input line. Output of the circuit is sent by the routers $R_{i,\lceil \log_2 n \rceil}$ $1 \leq i \leq n$, each of which has one output line. Each $R_{i,0}$ receives as input,

$$e_{i,1} = \begin{cases} A_{i-1} , & \text{if } A_{i-1} = B_{i-1} \\ d , & \text{otherwise} \end{cases}$$

Assuming that all R_{ij}, for a fixed j, can compute their outputs in one parallel step $\lceil \log_2 n \rceil + 1$ steps are required for computing M.

Theorem

Output of R_{ij} is $e_{i,2^j}$.

Proof: Proof is by induction on j.

j = 0: $R_{i,0}$, $1 \leq i \leq n$, has exactly one input line which receives $e_{i,1}$ as input and hence transmits $e_{i,1}$ as output.

Assuming that the theorem holds for all smaller values, R_{ij} receives $e_{i,2^{j-1}}$ on bottom line and $e_{i-2^{j-1},\ 2^{j-1}}$ on top line. Note that if $j \geq 1$, the bottom line exists. The result follows from the observation (3.1) about e_{ij} and the computation strategy of the router.

NOTES:

1. $R_{i,0}$, $1 \leq i \leq n$, simply transmit their inputs on their output lines. These routers can be eliminated by feeding $e_{i,1}$ to every $R_{i,1}$ and $R_{i+1,1}$. This reduces the number of routers by n and the time requirement by one unit.
2. The uniformity and simplicity of interconnections amony routers is one of the attractive features of this adder.
3. One possibility of encoding the data values in 2 binary bits is the following. 0,1,d are encoded by (10), (11) and either (01 or 00) respectively. Thus the first bit denotes whether or not the data equals d. The router may be built out of standard functions that map 2 input bits to one output bit as follows. Denoting the input bits on bottom line by (b_1,b_2) and on top line by (t_1,t_2) and the output bits by (f_1,f_2), it follows that,

$$f_1 = b_1 \vee t_1$$

$$f_2 = (b_1 \Rightarrow b_2) \wedge (\overline{b}_1 \Rightarrow t_2)$$

Hence a router will take 2 units of standard gate delay for its operation.

Inputs to $R_{i,0}$, $1 \leq i \leq n$, can be computed from A,B within one time unit.

4. The circuit for computing M can be used as a pipeline in which $\lceil \log_2 n \rceil + 1$ simultaneous computations may take place; one computation corresponding to each column of the matrix.

References

1. Anderson et al, Floating Point Execution Unit, IBM J. of Research and Development, Vol. 11, No. 1, Jan. 1967.