A Fast-Carry Adder with CMOS Transmission Gates

P. M. FENWICK

Department of Computer Science, The University of Auckland, Private Bag, Auckland, New Zealand

A parallel adder due to Kilburn uses a carry path with cascaded switches instead of more conventional logic gates. It is shown that a similar adder using CMOS transmission gates in the carry path combines high performance with a simple and economical design which should be suitable for integration.

1. INTRODUCTION

A traditional and important problem of computer hardware involves designing a fast parallel adder to minimise the effect of the worst-case carry propagation. The method in most general use is Carry Look-Ahead, in which the carry at one stage is obtained in terms of Carry-Generate and Carry-Propagate functions from lower-order adder stages. A full Carry Look-Ahead adder is seldom implemented because, in an adder of width n bits: (i) the amount of carry logic is proportional to n^2; (ii) the gates in the higher-order stages have a fan-in of up to n; and (iii) the lower-order stages may require a fanout which also approaches n. For these reasons Carry-Lookahead is usually applied first within blocks of a few bits, and then repeated on the blocks which are regarded as stages in a higher-radix adder. Further levels of Look-Ahead may be necessary in large adders. In any case the extensive carry-path logic of the CLA adder may pose problems in terms of interconnections, logic volume (i.e. chip area) and regularity. These matters are discussed by Brent and Kung.[1]

There are several other adder designs which are important where long sequences of additions are expected, such as during multiplication. The most important of these is the Carry Save adder, which saves the intermediate carries in a register for incorporation on the next addition. It avoids all need for carry propagation during the sequence of additions, but usually needs a CLA adder for final carry assimilation. Other techniques by Fenwick[3] and Hsu et al.[5] arise from noting that even when the carry does propagate through the whole adder the next addition does not have to await carry completion, but may be started as soon as the low-order sum is complete. The result is a pipelined adder in which the carry and control signals propagate in synchronism along the adder.

Yet another approach, by Kilburn et al.,[6] relies on technology rather than logic to achieve high speed. The carry path of the Kilburn adder contains no normal logic at all, but instead has switches which propagate the carry or force it to a 0 or 1, depending on the summand digits, as shown in Fig. 1. (A switch is closed if the adjoining logical condition is TRUE). Even though the switch set-up time may be relatively large, the switch setting may be done in parallel for all stages of the adder; the setting delay is not compounded by the adder length. The carry path itself, however, depends only on conduction through established connections. The conduction delay through each switch is much less than a normal logic delay, and is sufficiently small that even the propagation delay over the entire carry path is not excessive. The result is a high-performance adder of a very simple design. An extensive discussion of the performance of the Kilburn adder, its performance with more modern transistors, and its amenability to integration, may be found in the paper by Gosling,[4] while a version which does employ carry-path logic, but with a unique logic circuit, is given in Ref. 2.

2. AN IMPROVED KILBURN ADDER

The original Kilburn adder used saturated, nearly symmetrical, transistors as the carry-path switches. Modern switching transistors are definitely not suitable for this application because of the excessive V_{ce} of the saturated device and their high degree of asymmetry. Furthermore, it is doubtful that suitably fast symmetrical transistors would be amenable to integration (see Gosling,[4] mentioned above). There is however one modern circuit element which seems ideally suited to the Kilburn adder, and that is the CMOS Transmission Gate. While there is certainly a logic delay in opening and closing the gate as needed, the transmission path through the gate itself is essentially resistive, with no offset voltage and relatively little delay. It should be possible to cascade even the 16 or 32 transmission gates for a full carry path without too much trouble from d.c. signal attenuation provided that there is no resistive loading, i.e. that CMOS or MOS circuits only are driven from the carry path. Some delay arises from the shunt capacitive loading and the series resistance of the gates, but it will be seen that the final performance is quite acceptable.

The basic design for one stage of the adder is shown in Fig. 2, with a suggested implementation using a 4-input Multiplexer in Fig. 3. It is most important to note that this Multiplexer is an Analogue Multiplexer (a coordinated set of transmission gates) and not a 'conventional' logical or digital Multiplexer. While the digital device is in many ways equivalent to the analogue
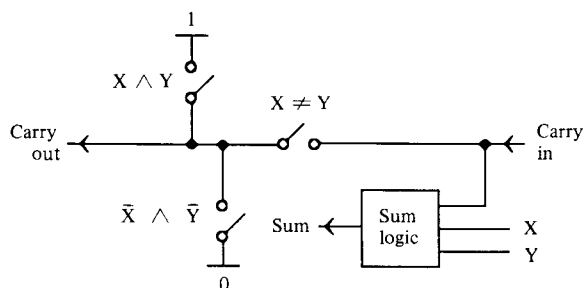
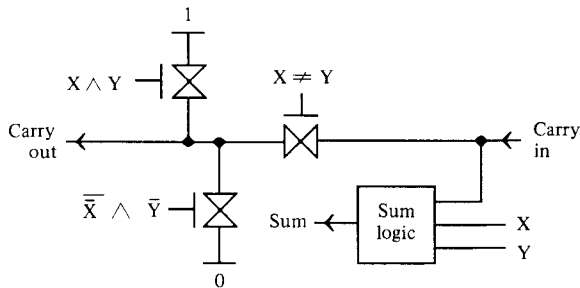**Fig. 1. Basic Kilburn adder**

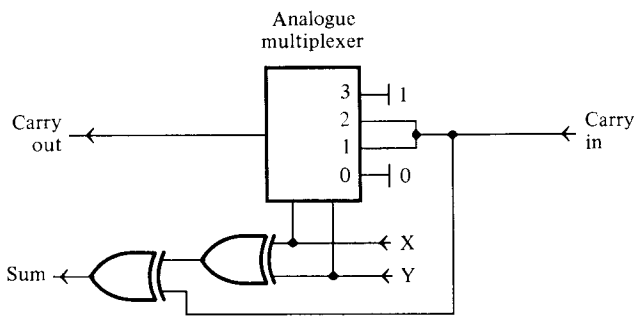Fig. 2. Kilburn adder with transmission gates

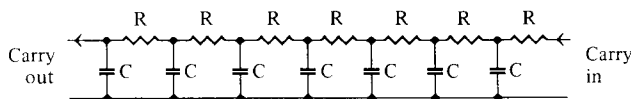Fig. 3. An implementation of the Kilburn adder

Fig. 4. Equivalent circuit of part of carry path

multiplexer, it is characterised by a logic switching delay from input to output, rather than the much shorter conduction delay of a transmission gate. It may be worth noting here that although the two inputs of a CMOS Exclusive-OR are logically equivalent, they are not necessarily equivalent electrically. Some implementations have a much higher gain from one input, and it is probably this input which should connect to the carry path to minimise the effects of slow signal transitions.

3. CARRY-PATH PERFORMANCE

The main question about the proposed design is the performance of a cascade of 16, 32 or even more transmission gates, and the distortion and delays suffered by a signal which traverses the cascade. A 28 stage carry path was simulated by cascading CMOS 4066 transmission gates on a breadboard test configuration. The gates are of a relatively old and slow technology which, combined with the capacitances and loadings of the breadboard layout, leads to a relatively low performance. A design with more modern CMOS technology should lead to a much better final performance. As expected, there is no problem with d.c. carry attenuation – the final carry logic levels are maintained through the whole carry path. The problems arise with the transient behaviour of the circuit.

Simple theory expects each transmission gate to behave as a series resistance. In conjunction with the stray capacitances and the input capacitances of the Sum Exclusive-OR gate, the electrical equivalent of the carry

path is an r.c. transmission line, as shown in Fig. 4. The line has no attenuation when feeding into a load with zero conductance (infinite resistance), but has an overall delay which is proportional to the square of the line length. The signal rise time also increases with distance along the line. Mead and Conway (Ref 8, pp. 22,23) discuss the transient behaviour of an r.c. transmission line, and the use of inverters in the line to reduce the a.c. signal degradation and consequent delays. Gosling also mentions the need for 'restandardisation' of the carry-path signal, but to overcome d.c. signal-level shifts rather than a.c. degradation.

Apart from the expected behaviour of the r.c. line, the delay was found to be quite sensitive to the driving waveform. Many logic sources do switch quickly to a voltage level somewhat past the switching threshold for gates of that family, but this initial transient is followed by a very slow approach to the final value. This slow 'tail' to the response decreases the effective drive into the carry path and slows the response of the whole system. The driver output impedance must be low over the entire voltage transition (i.e. comparable to the series impedance of the transmission gate or less) for the driver to supply enough current into the line to charge the stray shunt capacitances. The result otherwise is a further increase in the carry-path delay. From this discussion, it is clear that circuits which drive the carry path must be able to supply a clean voltage step from a relatively low impedance if the overall performance is not to suffer.

Carry-path delays were found to be 800 ns for the full 28-stage path, and 300 ns for a shorter 16-stage path. This speed must be compared with that of comparable CMOS logic. A 16-bit adder using the CMOS 4057 4-bit ALU has an expected carry propagation time of 700 ns. A similar unit using the CMOS 4008 4-bit adder (which includes full carry look-ahead and may be assumed to have the carry-path logic designed for speed) has a carry delay of 180 ns (all delays are for a 10 V supply). The test carry path therefore has a performance which is superior to that of the simple CMOS adder, although somewhat inferior to another adder with full carry look-ahead. An adder with transmission gates designed specifically for the purpose should therefore be at least competitive in speed with more conventional designs.

One fast adder technique which may be appropriate to the transmission gate adder is 'Carry Skip'.[7] The carry skip adder includes logic to allow the carry to bypass or skip over blocks through which it would otherwise propagate. It resembles the carry look-ahead adder, but does not have the full internal carry prediction logic within each block. The worst-case carry propagation distance for an n-stage adder with single-level carry skip is reduced from $n-1$ to approximately $3\sqrt{n}-4$. Given a reasonably fast carry path, as we have here, it allows a moderate increase in the worst-case speed, without needing much extra logic.

4. APPLICATION TO INTEGRATED CIRCUITRY

A feature of the proposed adder is its simplicity and regularity, combined with probable high speed. It has none of the extensive inter-stage logic which characterises carry look-ahead adders, and should be easy to integrate and efficient in chip area. The carry-path transmission

gates may have to be relatively large in comparison with other logic elements to achieve the low impedance which is necessary for low transmission delays, but it is extremely unlikely that they would use more space than the logic for a carry look-ahead design. The Kilburn Adder with transmission gates is therefore presented as a possible design for CMOS integration.

A recent paper by Shively et al.[9] describes the use of transmission gate logic in a parallel multiplier which achieved a multiply rate exceeding 40 MHz with 1.5 μm NMOS technology. Their main emphasis was to reduce the delay of a Wallace-tree multiplier; their design still uses a conventional carry look-ahead adder for final carry assimilation. The present author believes that transmission gate technology should be relevant to the adder as well as to the multiplier. However, their results provide a conclusive demonstration of the relevance of transmission gates to VLSI arithmetic units.

REFERENCES

1. R. P. Brent and H. T. Kung, A regular layout for parallel adders. *IEEE Trans. Comp.* **C-31,** 260–264 (1982).
2. J. B. Earnshaw and P. M. Fenwick, Design for a parallel binary adder, *Electronic Engineering,* 794–796 (1966).
3. P. M. Fenwick, Binary multiplication with overlapped addition cycles, *IEEE Trans. Comp.* **C-18,** 71–74 (1969).
4. J. B. Gosling, Review of high-speed addition techniques, *Proc. IEE* **118,** 29–35 (1971).
5. P. Y. T. Hsu, J. T. Rahmeh, E. S. Davidson and J. A. Abraham, TIDBITS: Speedup via time-delay bit-slicing in ALU design for VLSI technology. 12*th Annual Int. Symp. on Comp. Arch.,* 28–35 (1985).
6. T. Kilburn, D. B. G. Edwards and D. Aspinall, A parallel arithmetic unit using a saturated-transistor fast-carry circuit. *Proc. IEE* **107** (B), 573–584 (1960).
7. M. Lehman and N. Burla, Skip techniques for high speed carry propagation in binary arithmetic units. *IRE Trans Elec. Comp.* **EC-14,** 691–698 (1961).
8. C. Mead and L. Conway, *Introduction to VLSI Systems.* Addison-Wesley, (1980).
9. R. R. Shively, W. V. Robinson and D. E. Orton, Cascading transmission gates to enhance multiplier performance. *IEEE Trans. Comput.* **C-33,** 677–679, (1984).