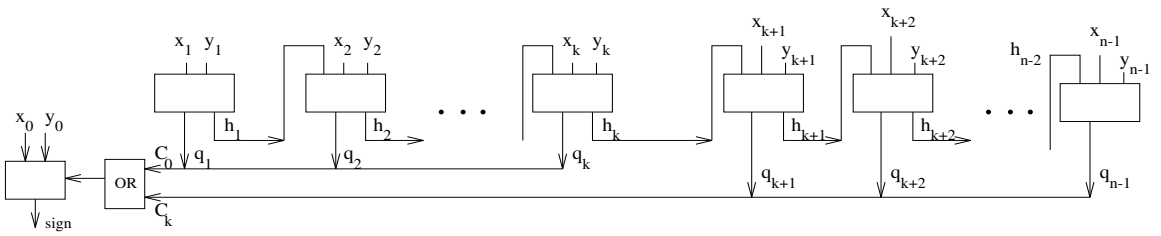

Figure 5: Improved iterative scheme: partitioned C_0 bus.

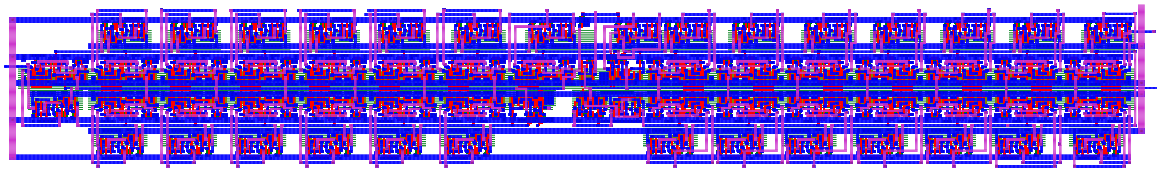

Figure 6: Layout of the improved iterative scheme (partitioned C_0 bus).

Although we only presented gate-level simulation of the improved tree type algorithm, the reduction in transition count and corresponding reduction in power dissipation is evident.

Acknowledgments. This research has been supported in part by the NSF Grant MIP-9314172 "Arithmetic Algorithms and Structures for Low-Power Systems."

References

[1] M.D. Ercegovac and T. Lang "Reducing Transition Counts in Arithmetic Circuits", *IEEE Symposium on Low Power Electronics*, pp.64-65, 1994.

[2] M.D. Ercegovac and T. Lang "Sign Detection and Comparison Networks with a Small Number of Transitions", UCLA Computer Science Department Technical Report, 1995.

[3] M. Horowitz, T. Indermaur and R. Gonzalez "Low-Power Digital Design", *IEEE Symposium on Low Power Electronics*, pp.8-11, 1994.

[4] K Keutzer and P. Vanbekbergen "The Impact of CAD on the Design of Low Power Digital Circuits", *IEEE Symposium on Low Power Electronics*, pp.42-45, 1994.

[5] S.R. Powell and P.M. Chau, "Estimating Power Dissipation of VLSI Signal Processing Chips: the PFA Technique," *VLSI Signal Processing, IV*, Eds. H.S. Moscovitz, K. Yao, and R. Jain, IEEE Press, 1990, pp.250-259.

[6] "TSIM Manual" UCLA Computer Science Department Technical Report, 1995.

	Standard		Improved	
	c_{i-1}	C_{switch}	h_i	C_{switch}
Bit 1	.36	67.4	.25	64.1
Bit 2	.35	67.3	.24	96.4
Bit 3	.35	67.6	.16	76.4
Bit 4	.36	67.8	.10	56.1
Bit 5	.36	61.8	.06	35.0
Bit 6	.36	64.0	.03	29.2
Bit 7	.36	67.3	.02	25.7
Bit 8	.36	62.9	.01	24.8
Bit 9	.36	61.8	.01	23.2
Bit 10	.36	67.8	.00	22.3
. . .				
Bit 22	.36	67.8	.00	21.3
Bit 23	.36	67.3	.00	21.3
Bit 24	.36	66.6	.00	21.3
Bit 25	.36	61.3	.00	21.3
Bit 26	.35	67.1	.00	21.3
Bit 27	.35	66.6	.00	21.4
Bit 28	.34	59.5	.00	21.3
Bit 29	.32	59.7	.00	21.3
Bit 30	.26	50.4	.00	21.3
Bit 31	.19	20.1	–	0.0
C_0 bus	–	–	.29	510.0
Total	–	1946.0	–	1406.5

Table 3: Average transitions, switching capacitance per input vector for 32-bit iterative sign detection circuits.

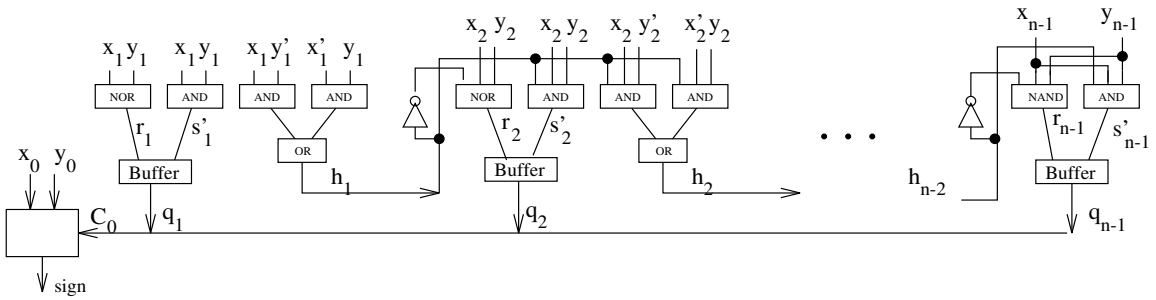

Figure 3: Improved iterative scheme, utilizing three-state buffers.

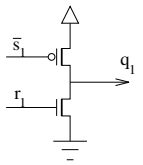

Figure 4: Three-state buffer.

inputs			outputs			
h_{i-1}	x_i	y_i	r_i	$\overline{s_i}$	h_i	q_i
0	X	X	0	1	0	X
1	0	0	1	1	0	0
1	0	1	0	1	1	X
1	1	0	0	1	1	X
1	1	1	0	0	0	1

Table 2: Truth table for stage i of the improved iterative sign detection circuit.

each h_i of the improved circuit. The switching capacitance (units of fF/input vector), broken down by input bit, are also shown. From the table, we observe that for the standard iterative circuit, the expected switching capacitance is nearly identical for each bit i, where $1 < i < 26$. (The variations between different bit positions are a result of irregularities in the layout's interconnect wiring.) Using the data in the table we estimate that for any precision $n > 5$, the expected switching capacitance would be $C_{\text{switch}} \approx 65.0(n-1) - 70.6$ fF/input.

The results demonstrate that for the improved circuit, the switching capacitance for each bit decreases to the lower limit of switching about 21.3 fF/input. (The circuits for Bits 1 and 31 are simpler than the other bits explaining the lower switching capacitance of these bits.) This lower limit could be attributed almost entirely to the two inverters which generate the $\overline{x_i}$ and $\overline{y_i}$ signals. However, the improved scheme must pay a price for switching the large capacitance of the C_0 bus, over 1/3 of the total switching capacitance for this circuit. Assuming the capacitance of the C_0 bus grows linearly with the precision n, we can estimate the expected switching capacitance for an improved circuit with precision $n > 5$ as $C_{\text{switch}} \approx (21.3 + 16.5)(n-1) + 234.7$ fF/input. For $n \geq 12$, the expected switching capacitance would be lower for the improved scheme.

The total capacitance associated with the C_0 bus can be reduced by partitioning it into sections. When we partitioned the C_0 bus into two nearly equal sized partitions ($k = 16$ in Figure 5) the combined switching capacitance associated with generating the C_0 signal was reduced by 1/3. The total switching capacitance of the circuit was decreased to 1247 fF/input vector. Note that there is very little activity on the C_k bus; further partitioning of the C_0 bus would likely continue to reduce the switching capacitance associated with this signal.

Conclusion

Utilizing an inhibit signal to avoid unnecessary signal transitions, we wer able to reduce the expected power dissipation of the sign detection circuit with only a small increase in the worst-case delay through the circuit.

Many variations to the inhibit signal scheme may be employed in reducing transitions. We demonstrate recursive inhibit signals (were h_i is a function of h_{i-1} as in the improved iterative sign-detection scheme), and fixed inhibit signals (were h_i is a function of a fixed subset of the circuit's inputs as in the improved recursive scheme). For the case of the sign-detection circuit, many alternatives may be used to reduce the worst-case propagation delay, analogous to the tricks played to improve adder circuits. For example, an "inhibit-skip" (much like a carry-skip) may be used to quickly propagate the inhibit signal to the lesser significant bit positions. Similarly, there some modifications may be made to reduce the capacitance of the C_0 bus. We demonstrated one such modification, a simple partition of the bus into two equal-sized sections.

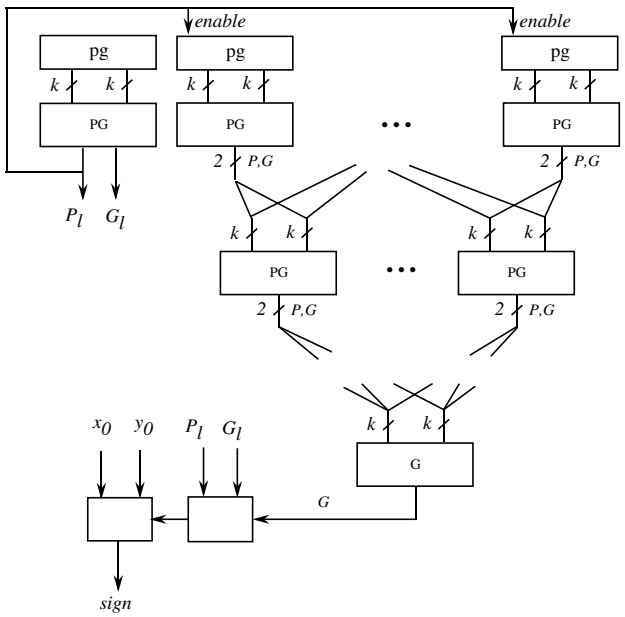

Figure 2: Tree (lookahead) network with reduced transitions.

In [1], analysis was given to show the expected number of transitions would approach a constant, independent of the precision of the circuit n.

Tree Scheme

Tree type algorithms exist with a worst-case delay $\mathcal{O}(\log n)$, opposed to $\mathcal{O}(n)$ for the iterative algorithms. We extend our approach to such implementation. As before we reduce the number of transitions by eliminating unecessary transitions. In the implementation of Figure 2, the generation of p, g signals is inhibited for all but the most significant k bits when P_l (the P signal from the left-most PG module) is 0.

Experimental Results

We performed gate-level simulation of the sign detection circuits from the preceding discussion using a large number of randomly generated input vectors (about 40K) and TSIM ([6]). Our simulations assume coincident input signal arrivals for the circuit, and uniform input delays for each gate. We do not assume that the complements of the input signals, $\overline{x_i}$ and $\overline{y_i}$, are externally available; the circuit must generate them if needed. For the improved tree scheme we used a module size of $k = 4$. The results for 32 and 64-bit circuits are summarized in Table 1. The idealized column shows the average number of signals which change state per input vector. This number indicates the best possible expected number of signal transitions for a given algorithm; in practice the number would be larger due to spurious transitions and hazards. The unit delay column shows the average number of transitions per input vector, assuming unit delay for each logic gate. For the standard iterative algorithm, the carry and output signals may transition a multiple number of times per input vector as the signals propagate along the carry chain; this is reflected in the figures of this column. Optimizations, such as delay-balancing along different signal paths, may be performed to bring the transition counts closer to the experimental ideal values given. The data demonstrates that the proposed improvements will reduce the expected number of transitions.

Circuit	Idealized	Unit delay
Standard iterative 32	63.4	74.8
Improved iterative 32	35.8	37.7
Standard iterative 64	131.4	156.4
Improved iterative 64	68.4	70.2
Standard tree 32	94.3	114.2
Improved tree 32, $k = 4$	54.7	79.9
Standard tree 64	196.2	236.3
Improved tree 64, $k = 4$	102.6	153.3

Table 1: Average transition counts per input vector for 32-bit sign detection circuits.

CMOS Implementation

We implemented 32-bit versions of the improved iterative sign-detection circuit in MOSIS 0.6 micron CMOS technology. For comparison, we also implemented the standard sign-detection circuit in the same technology. Both circuits were implemented using a standard-cell approach, with no special optimizations performed at the macro level. For the improved iterative circuit, we use three-state buffers to drive the C_0 bus. This particular design avoids the static power dissipation of the pull-down logic required for a wired-OR as well as provides stronger drive capabilities than standard transmission gate three-state buffers. (Figures 3 and 4.) We also deviate slightly from the proposed scheme by not providing circuitry to initialize $h_i = 0$ prior to application of each input vector, increasing the expected number of transitions. Like the TSIM results, we assume that the complements of the input signals, $\overline{x_i}$ and $\overline{y_i}$, are not available, and must be generated by the circuit. Each bit i, for $2 \leq i \leq n-1$, of our implemention follows the truth table in Table 2.

Both the standard iterative sign-detection and the improved circuit exhibit a worst case delay between 25.0 and 30.0ns. We performed switch level simulation of the resulting circuits using Irsim-Cap with 64K random, evenly distributed input vectors. Table 3 shows the average number of transitions per input vector for each c_{i-1} of the standard iterative circuit and

On Reducing Transition Counts in Sign Detection Circuits

Miloš D. Ercegovac
milos@cs.ucla.edu

Charles Fabian
fabian@cs.ucla.edu

Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90024

Tomás Lang
tomas@ece.uci.edu

Dept. of Elec. and Comp. Eng.
University of California, Irvine
Irvine, CA 92717

Abstract

An approach to reducing the average number of signal transitions in the networks for sign-detection and comparison of magnitudes is evaluated at the gate level. We implemented the circuits in 0.6 micron CMOS technology, and performed simulations to substantiate the claims of reducing total power consumption, with little penalty on speed or size. The simulation verifies that the approach significantly reduces the average number of transitions.

Summary

Increasingly, power consumption is as an important parameter as speed and area in the design of digital integrated circuits. This trend is being driven by both the increase in portable applications as well as problems of heat dissipation associated with high integration circuits [4].

Charging and discharging capacitors generally consumes most of the power in a CMOS circuit. For each node of a circuit, the dynamic power of a circuit is

$$P = \sum C_i V^2 \alpha_i F$$

over all nodes i of the circuit, where C_i is the capacitance of node i and α_i is the activity ratio of node i [3]. The product $C_i \alpha_i F$, the switching capacitance, can be measured to estimate the dynamic power. Assuming all nodes have comparable capacitance, the dynamic power can be directly related to the number of signal transitions [5]. In [1], [2] an approach is proposed to reduce the average number transitions expected to perform sign detection which is useful in magnitude comparison. In this paper we present several designs and experimental results to support the claim that the proposed method does in fact significantly reduce the expected transition counts.

We compute the sign of $x + y$, where x and y are fractions in two's complement representation. This operation is significant as it is used in the magnitude comparison of two numbers, performed by finding the sign of their difference. The fractions are represented by the n-bit vectors $X = x_0 x_1 ... x_i ... x_n$ and $Y = y_0 y_1 ... y_i ... y_n$ so that $x = -x_0 + \sum_{i=1}^{n} x_i 2^{-i}$ and $y = -y_0 + \sum_{i=1}^{n} y_i 2^{-i}$. The sign of $x + y$ is obtained by the expression $sign = x_0 \oplus y_0 \oplus c_0$ where c_0 is the carry-out of bit position 1 when $x + y$ is performed.

We consider iterative (ripple) and tree (lookahead) implementations of sign-detection networks.

Iterative Scheme

The standard iterative algorithm for sign detection uses the carry recurrence $c_{i-1} = p_i c_i + g_i$ where p_i may be implemented either as $p_i = x_i \oplus y_i$ or as $p_i = x_i + y_i$ and $g_i = x_i y_i$.

Since only the most significant carry-propagation chain affects c_0, we modify the iterative algorithm so that the transitions in the other chains are avoided. We produce an inhibit signal (Figure 1) that propagates from most significant to least significant bit to determine j, the position of the most-significant bit with $p_j = 0$. Then $c_0 = g_j$. This value can be placed on a bus or propagated back to the most-significant position.

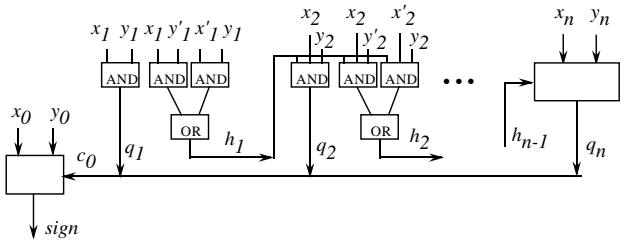

Figure 1: Iterative (ripple) scheme with reduced transition count.

Let $h_i = 1$ indicate that the most-significant propagation chain includes the bit i. The proposed algorithm is

0 Initialize all $h_i = 0$ for $1 \leq i \leq n-1$

0 $h_1 = x_1 \oplus y_1$

0 $h_{i+1} = h_i x_{i+1} y'_{i+1} + h_i x'_{i+1} y_{x+1}$ for $i = 1, ...n-2$

0 $c_0 = x_1 y_1 + \sum_{i=1}^{n-1} (h_i x_{i+1} y_{i+1})$