A Comparison of CMOS Implementations of an Asynchronous Circuits Primitive: the C-Element

Maitham Shams
Department of Electrical Engineering

Jo C. Ebergen
Department of Computer Science

Mohamed I. Elmasry
Department of Electrical Engineering

University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

Abstract

One of the most frequently used primitives in asynchronous control circuits is the C-element. The three most popular CMOS implementations of the C-element are compared with respect to energy-efficiency, delay, and area, with an emphasis on energy. The three implementations have been introduced by Sutherland, Martin, and Van Berkel. We show that in a typical environment, Van Berkel's implementation is superior to the other implementations with respect to energy consumption and area, for the same delay.

1 Introduction

Various applications have demonstrated that asynchronous circuits have great potential for low-power and high-performance design. One of the most frequently used primitives in asynchronous control circuits is the C-element. The purpose of the paper is to compare the three most popular CMOS implementations of the C-element. One implementation of the C-element has been introduced by I.E. Sutherland [1] and is used often in high-performance micropipelines; a second implementation has been introduced by A.J. Martin and has been used in the Caltech asynchronous microprocessor [2] and an asynchronous, low-power version of the ARM developed at Manchester University [3]; a third implementation has been introduced by K. Van Berkel and is being used in the TANGRAM silicon compiler for low-power design at Philips Research Laboratories [5]. The different implementations are examined in two different setups. For each setup, we measure the delay and energy dissipation through HSPICE simulations.

2 The C-element

The C-element has been introduced by D. E. Muller [6] and is therefore also called the "Muller C-element." A C-element has two inputs a and b and one output c. Traditionally, its logical behaviour has been described as follows. If both inputs are 0 (1) then the output becomes 0 (1); otherwise the output remains the same. For the proper operation of the C-element, it is also assumed that once both inputs become 0 (1), they will not change until the output changes. A state diagram is given in Figure 1 along with a commonly used schematic. The

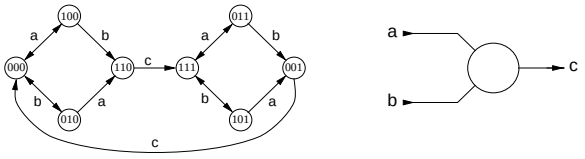

Figure 1: State diagram and schematic of the C-element

behavior of the output, c, of a C-element can be expressed in terms of the inputs a and b and the previous state of the output, $\hat{c}$, by the following Boolean function

$$c = \hat{c} \cdot (a + b) + a \cdot b \qquad (1)$$

3 Measurement Setup

Two testing environments are considered: one to evaluate the effect of the input capacitance and driving ability of an individual C-element gate, and another one to evaluate the performance of a series of cross-coupled C-element gates, where the performance of individual elements are mutually dependent. Figure 2 illustrates

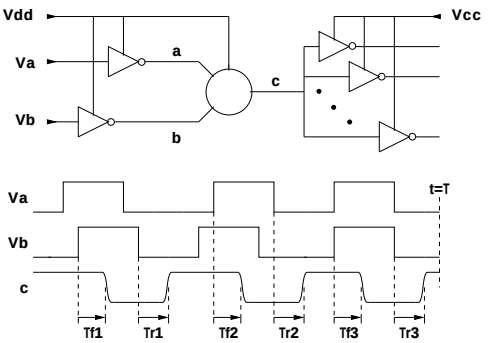

Figure 2: First measurement setup: testing for optimal sizing for known fan-out

the first measurement setup. The C-element is driven by two inverters which are fed by the ideal sources V_a and

ISLPED 1996 Monterey CA USA
0-7803-3571-8/96/$5.00 © 1996

V_b. At the output, the C-element is driving a number of inverters similar to the input inverters in size. The NMOS and PMOS transistors of the inverters are 4 and 10 microns wide, respectively. The delay through the C-element depends on the order and arriving times of the input signals. Exhaustive simulations are needed to find the worst-case delay by altering the arriving interval between the input signals. A slightly less accurate, but faster technique is to be only concerned with the arriving order of the input signals. As shown in the figure, using this method, six cases are considered and the respective delays produced are measured. The worst-case delay D is obtained by the following.

$$D = \max(T_{f1}, T_{f2}, T_{f3}, T_{r1}, T_{r2}, T_{r3}) \quad (2)$$

The energy consumption of the C-element per output transition also depends on the arriving times and order of the inputs. We measure the average energy consumption per output cycle E, corresponding to the illustrated waveforms, from the following.

$$E = P(V_{dd})\ \frac{T}{3} \quad (3)$$

where $P(V_{dd})$ is the average power extracted from V_{dd}, and T is the total period of the output waveform. The HSPICE simulations for this setup were performed at a frequency of 40 MHz.

The second measurement setup is shown in Figure 3. The C-elements in the chain are all of the same size. This setup differs form the first setup in that the performance of the C-elements are now mutually dependent, which affects the overall performance of the chain. A bubble at the input of a C-element schematic means that an inverted version of that input must be used. This, of course, can be implemented using an inverter. The chain of the C-elements shown, without the inverters at the two ends, form the control circuit of an n-stage micropipeline. The inverters are added to make the micropipeline self-driven and oscillating. The signals at the nodes indicated by $r(i)$, where i represents the stage number, can be interpreted as "request for the next stage". Initially, all nodes are set to low, and the only possible event is the rising of $r(0)$. This logic "one" then propagates through all the request nodes. Meanwhile, other transitions are produced at $r(0)$ which propagate toward the end, in turn. If not interfered, the oscillation of the nodes continues forever. A much simplified waveform is shown in the same figure. The parameters of interest in this test are the latency L, throughput, and energy per throughput. The frequency of oscillation of the micropipeline F, which is half its throughput, is the inverse of Φ. The energy dissipation thus, is

$$E = \Phi\ P(V_{dd}) \quad (4)$$

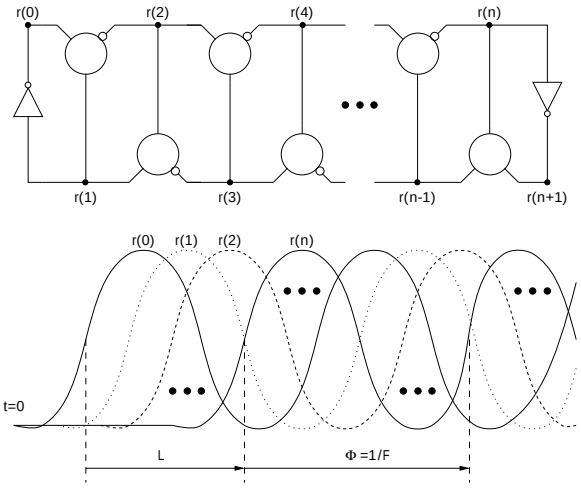

Figure 3: Second measurement setup: testing for optimal sizing of a chain structure in presence of feed-back

where $P(V_{dd})$ is the average power extracted from V_{dd}.

The results obtained through the first setup are useful when we want to insert a C-element in a spot where the input drive and output capacitance are known. The second setup, however, can give suggestions for the implementation of a subsystem in which the C-elements are mutually dependent. The technology file of a 0.8 μm BICMOS process has been used in our HSPICE simulations. In this technology, the threshold voltage of NMOS devices is 0.81 V and that of PMOS devices is -0.90 V. Voltage Power supply of 3 V is assumed in all simulations.

4 C-Element Implementations

A conventional pull-up pull-down realization of the function is shown in Figure 4. This circuit has been presented in [1] by Sutherland. This implementation is ratioless, i.e. it does not impose any restrictions on the sizes of the transistors. From the operation of

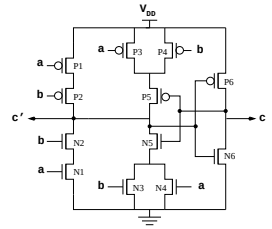

Figure 4: Conventional pull-up pull-down CMOS implementation of the C-element [1]

the circuit, we conclude that N1, N2, and N6 are the main pull-down transistors which contribute to output switching; they are of size W. Whereas N3, N4, and N5

only provide the necessary feed-back to hold the state of the output when values of the inputs do not match; hence, they are made as small as possible to reduce their loading effect. Similarly, the feed-back transistors P3, P4, and P5 have minimum width, while P1 and P2, the normal pull-up transistors, have widths $W_p = 2.5W$.

The C-element circuit illustrated in Figure 5 has been presented by Martin [7]. This circuit utilizes an inverter latch to maintain the state of the output when the inputs do not have a similar logic level. The circuit suffers from a race problem at node $\bar{c}$. There is an inherent re-

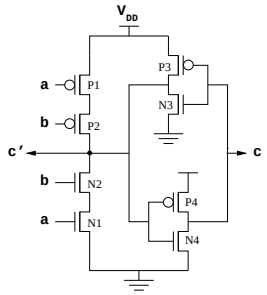

Figure 5: CMOS implementation of C-element using inverter latch (a) [7]

sistance to switching the state of the latch that can be reduced, but can not be eliminated. For a proper operation of the circuit, certain size ratios must be imposed on the transistors. The feed-back inverter should be a weak one to allow changes in the state of the latch. Accordingly, the following must hold:

$$\begin{cases} \left(\frac{W}{L}\right)_{\text{P1\&P2}} > 2r\left(\frac{W}{L}\right)_{\text{N3}} \\ \left(\frac{W}{L}\right)_{\text{N1\&N2}} > \frac{2}{r}\left(\frac{W}{L}\right)_{\text{P3}} \end{cases} \quad (5)$$

Minimum size transistors are chosen for the feed-back inverter to reduce the race problem.

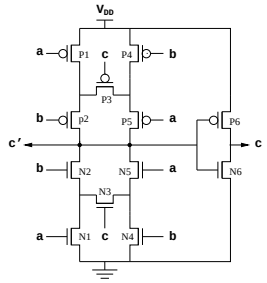

Figure 6: CMOS implementation of C-element by Van Berkel [8]

The next C-element implementation by Van Berkel [8] is shown in Figure 6. In this circuit, the output state is maintained through a feed-back conducting path of three transistors in the pull-up tree or the pull-down tree. Similar to Sutherland's circuit, this circuit is also ratioless. An advantage of this implementation is that it is symmetrical with respect to the inputs. For the circuit to have the same pull-up and pull-down resistances (when switching) as the previous two implementations, the normal N-tree and P-tree transistors, except those of the output inverter, must be made half the size. The feed-back transistors N3 and P3 are, as usual, of minimum size, and N6 and P6 have a normal size to achieve load driving capability of the previous circuits.

5 Results of the Tests

Figure 7 shows the energy dissipation versus the propagation delay for the three C-element implementations with a fan-out of 3 inverters under the first test. The size of the C-element gate increases for each curve from the right hand side of the graph toward the top of the graph. Thus, one might get two different energy readings for the same delay and implementation, but they correspond to two different sizes. Point "M" in the figure shows

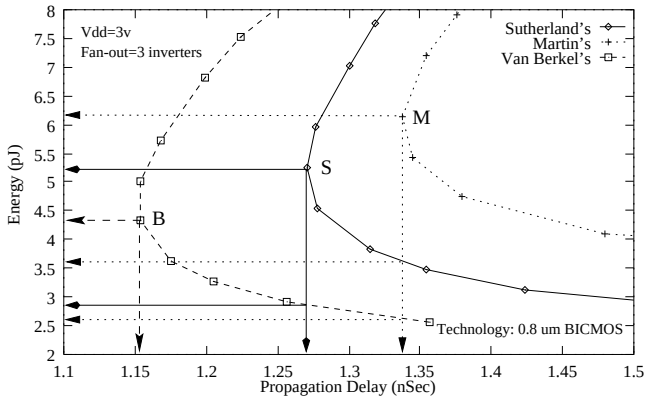

Figure 7: HSPICE simulation results, energy versus delay, for the C-element gates in the first test setup

the minimum delay achievable using the second (Martin's) implementation of the C-element. For the same delay, if we chose the first (Sutherland's) implementation we would be saving 42% in energy. Furthermore, if we chose the third (Van Berkel's) implementation, we would be saving 58% in energy over Martin's circuit. Similarly, point "S" shows the minimum obtainable delay using Sutherland's implementation. Van Berkel's circuit obtains the same delay for 45% less energy . Notice also that delays between 1.15 nSec to 1.27 nSec can be achieved by Van Berkel's circuit but not by the other two circuits. Although the minimum delays of the circuits are all within a 10% difference, Van Berkel's circuit achieves a significant energy savings of about 50% for the same delay.

The results of the simulations for the second test environment are depicted in the form of an energy-frequency (E-F) graph in Figure 8. The solid lines crossing the curves show the total gate area of the individual C-

elements used to form the micropipelines. Similar to the E-D graph, in an E-F graph the outermost curve (here furthest from the origin) is the best alternative. Assume

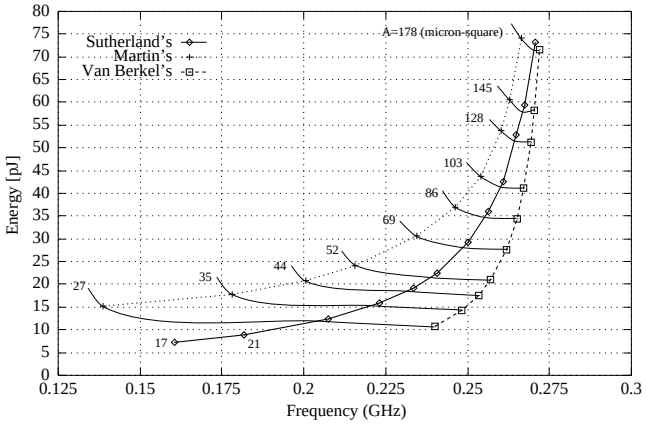

Figure 8: HSPICE simulation results, energy versus frequency, for the chains of C-element gates in the second test setup

that we are interested in designing a micropipeline control circuit to operate at a maximum frequency of 250 MHz (or a throughput of 500MHz). The graph shows that this is possible by using Martin's C-element with a gate area of 95 μm^2, which results in an energy consumption of 40 pJ per cycle. Alternatively, Sutherland's C-element with a gate area of 72 μm^2 could be adopted, dissipating 30 pJ, 25% less than the previous choice. However, if Van Berkel's C-element with a gate area of 35 μm^2 is used, the same throughput would be obtained at an energy of only 15 pJ. This choice reduces silicon gate area by a factor of 2.74 and energy by a factor of 2.67 compared to Martin's implementation. The savings over Sutherland's implementation is a factor of 2.09 in area and a factor of 2.0 in energy or power. The graph also shows that for higher frequencies, Van Berkel's circuit offers further savings in energy and area compared the other two implementations.

6 Discussion and Conclusions

A comparative study of CMOS implementations of the C-element in terms of energy, delay, and area has been presented. Three implementations from the literature have been considered: one by Sutherland [1], one by Martin [7], and one by Van Berkel [8]. The techniques we discussed can be extended to compare different implementations of other primitives. The energy-delay and energy-frequency graphs illustrated in this paper are a convenient way of presenting the trade-offs between energy, delay (or frequency), and area of a design all in one graph.

Although results may vary for other test environments, we can say that Van Berkel's C-element is definitely the right choice for low-power and high-performance applications. The superiority of Van Berkel's implementation can be explained intuitively as follows. Sutherland's implementation has six transistors dedicated to latching, which do not resist the output switching of the circuit. The second implementation has only two transistors dedicated to latching, but the topology is such that they do resist output switching. Van Berkel's implementation has the advantages of both: minimum overhead for latching with no resistance against output switching. Furthermore, Van Berkel's implementation has a nice symmetrical topology with respect to the inputs.

The results obtained through this study are not limited to the C-element and may be extended to similar circuits in which the state of their outputs must be latched. For example, this comparison demonstrated that minimizing the number of transistors dedicated to latching and avoiding topologies that resist output switching may result in significant energy savings.

References

[1] I. E. Sutherland, "Micropipelines," *Communications of the ACM*, vol. 32, pp. 720–738, June 1989.

[2] A. J. Martin, S. M. Burns, T. K. Lee, D. Borkovic, and P. J. Hazewindus, "The design of an asynchronous microprocessor," in *Advanced Research in VLSI: Proceedings of the Decennial Caltech Conference on VLSI* (C. L. Seitz, ed.), pp. 351–373, MIT Press, 1989.

[3] S. Furber, "Computing without clocks: Micropipelining the ARM processor," in *Proceedings Banff VIII Workshop: Asynchronous Digital Circuit Design* (G. Birtwistle and A. Davis, eds.), Workshops in Computing, Springer-Verlag, 1995.

[4] K. van Berkel, R. Burgess, J. Kessels, A. Peeters, M. Roncken, and F. Schalij, "A fully-asynchronous low-power error corrector for the DCC player," in *International Solid State Circuits Conference*, pp. 88–89, Feb. 1994.

[5] K. van Berkel, "VLSI programming of asynchronous circuits for low power," in *Proceedings Banff VIII Workshop: Asynchronous Digital Circuit Design* (G. Birtwistle and A. Davis, eds.), Workshops in Computing, Springer-Verlag, 1995.

[6] D. E. Muller and W. S. Bartky, "A theory of asynchronous circuits," in *Proceedings of an International Symposium on the Theory of Switching*, pp. 204–243, Harvard University Press, Apr. 1959.

[7] A. J. Martin, "Formal program transformations for VLSI circuit synthesis," in *Formal Development of Programs and Proofs* (E. W. Dijkstra, ed.), UT Year of Programming Series, pp. 59–80, Addison-Wesley, 1989.

[8] K. van Berkel, "Beware the isochronic fork," *Integration, the VLSI journal*, vol. 13, pp. 103–128, June 1992.