Circuit Design Style for Energy Efficiency: LSDL and Compound Domino

Xiao Yan Yu[1,4], Robert Montoye[2], Kevin Nowka[3],
Bart Zeydel[4] and Vojin Oklobdzija[4]

[1] IBM Poughkeepsie
2455 South Rd
Poughkeepsie, NY 12601, USA
xianyu@us.ibm.com
[2] IBM T.J. Watson Research Center
1101 Kitchawan Rd, Route 134
Yorktown Heights, NY 10598, USA
montoye@us.ibm.com
[3] IBM Austin Research Laboratory
11501 Burnet Rd
Austin, TX 78758, USA
nowka@us.ibm.com
[4] ACSEL Laboratory
University of California
Davis, CA 95616, USA
{zeydel,vojin}@acsel-lab.com

Abstract. Introduction of sub-90nm technology has made a profound impact on circuit designs. Thus, it requires understanding of existing design styles for desired energy-efficiency. We compare adder designs in the energy-delay space, implemented with Limited Switch Dynamic Logic (LSDL) and Compound Domino Logic (CD) in a 65nm SOI technology. Evaluation results show that LSDL can provide more than 35% energy savings than CD with 25% switching activity at relaxed cycle times greater than 10.5 FO4.

1 Introduction

The emergence of sub-90nm poses several challenges to circuit designers. First, the interconnect delay is becoming more dominant compared to transistor delay as technology moves beyond 90nm. Traditionally, designs are created prior to physical implementation. Performance degradation due to interconnect parasitic has been considered as minor effect. Today, technology scaling provides higher levels of integration and smaller devices. As a result, delays due to interconnect parasitic can no longer be ignored. [7] Performance impact due to physical implementation must be analyzed at design time in order to guarantee optimality of the design.

As transistor length reduces beyond 90nm, gate oxide is reduced to a few atom layers. Sub-threshold and gate oxide leakage are increasing. This is especially true for low-V_t transistors which have been intensively used in high-performance designs. Hence, it is desirable to use low-V_t devices only on critical paths. As circuits approach the ultimate power limit, high performance application designs need to be energy-efficient. Therefore, it is very valuable to use energy efficient circuit design styles, especially when performance is not pushed to the limit. We need to know which circuit family can provide the desired speed while not consuming more energy than allowed by the energy budget. In this paper, we analyze two representative energy efficient logic families, namely, Limited Switch Dynamic Logic (LSDL) [3] and Compound Domino Logic (CD) [4]. Several adder topologies are implemented using these two families and are compared for their energy efficiencies. The evaluation of these implementations is based on delay characterizations at the nominal supply voltage. The rest of the paper is organized as follows. Section II briefly describes the operations of LSDL and CD. Section III provides details of the adder topologies used in the evaluation of these two logic families. Section IV discusses the evaluation results. Our goal here is show which of these two logic families is the best in the energy-delay space based on our analysis of adder implementations.

2 Preliminary

Based on the analysis of adder topologies in a 100nm technology, CD has been shown to be more energy efficient than conventional domino logic [1], [2]. Figure 1(a) shows a CD implementation of a function. It is composed of a dynamic gate followed by a static gate such as 2-input NOR or NAND instead of an inverter. When the clock signal is low, all dynamic nodes are pre-charged to logic "1" and all static nodes fall to logic "0".

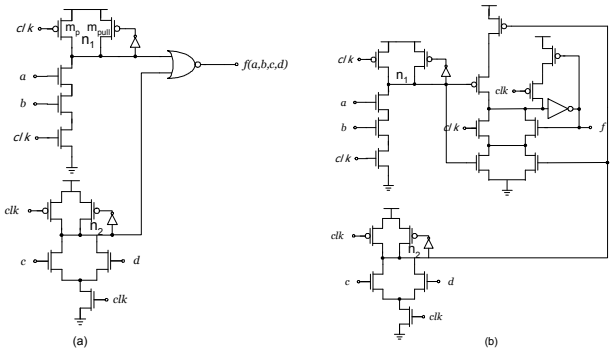

Figure 1(a) Compound Domino (b) Limited Switch Dynamic Logic

Figure 1(b) shows a LSDL implementation of the function, which is the complement of the function shown in Figure 1(a). It is composed of dynamic gates with a latch output. All dynamic nodes are pre-charged to logic "1" when the clock is low. Static

nodes, on the other hand, preserve the value from the previous evaluation cycle. To improve performance, some logic may be merged into the latch. Since the latch is capable of producing outputs of both polarities, it eliminates the need to have dual-rail circuitry in LSDL.

3 Design Style Evaluation

We chose to evaluate the effectness of LSDL and CD on a class of parallel carry look-ahead schemes [9]. These schemes are extended from the idea of carry look-ahead computation and rely on recursive carry computation as follows:

$$(g,p)\bullet(g',p')=(g+p\cdot g',p\cdot p') \tag{1}$$

Thus,

$$(G_i,P_i)=\begin{cases}(g_0,p_0),if\ \ i=0\\(g_i,p_i)\bullet(G_{i-1},P_{i-1}),if\ \ 1\le i\le n-1\end{cases} \tag{2}$$

These adders are also known as recurrence solver adders. There are two properties of this recursive formulation:

$C_i = G_i \forall$ i

The operator "•" is associative

Adders implemented using these techniques are favorable due to their regular layout and fixed fan-out.

In this paper, we will be analyzing three representative adder topologies suitable for performance-demanding applications, namely, full tree prefix-4 carry look-ahead adder, sparse tree Ling adder with sparseness of 3 and sparse tree pseudo carry look-ahead adder with sparseness of 2.

3.1 Full Tree Prefix-4 Carry Look-ahead Adder (Park Adder)

Park *et al* have proposed a 64-bit carry look-ahead and Prefix-4 Kogge-Stone hybrid adder in [11]. Figure 2 shows the prefix diagram of this adder. It has been implemented using CD. It has the shortest number of stages on the critical path among all previously published designs.

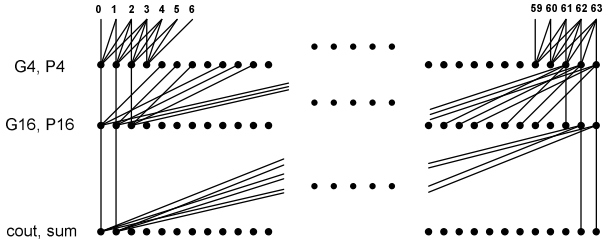

Figure 2 Prefix diagram of Park adder

The advantages of this design over others are:

- Group-4 generate and propagate terms are produced directly from primary inputs
- Final carry generation at each bit is combined with sum generation

This scheme simplifies the conventional group generate and propagate term into:

$$\begin{aligned} G_{4_i} &= \left(g_i + g_{i+1} + g_{i+2} + p_{i+2}g_{i+3}\right)\cdot\left(g_i + p_i p_{i+1}\right) \\ P_{4_i} &= \left(g_i + g_{i+1} + g_{i+2} + p_{i+2}p_{i+3}\right)\cdot\left(g_i + p_i p_{i+1}\right) \end{aligned} \tag{3}$$

by using the following properties:

$$\begin{cases} g_i = p_i \cdot g_i \\ p_i = p_i + g_i \end{cases} \tag{4}$$

where $p_i = a_i + bi$ and $gi = ai \cdot bi$. With the above mentioned advantages, a design with three stages of CDL gates on its critical path is produced.

3.2 Sparse Tree Ling adder (Ling662 Adder)

Figure 3 shows the prefix diagram of a sparse tree Ling adder with sparseness of 3. This scheme is a variant of a Ling design that is proposed in [16], [17]. It has three stages on its critical path like the Park adder described previously. Group-6 pseudo generate (H6) and propagate (I6) bits are generated every three bits using Ling's carry scheme [10] in its first stage.

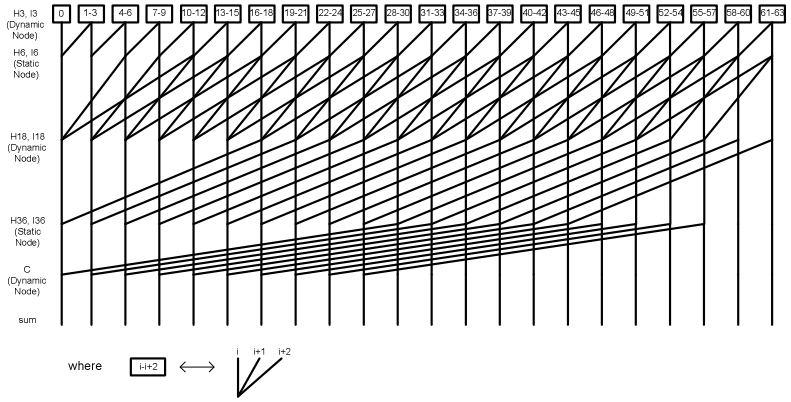

Figure 3 Prefix diagram of sparse tree Ling 662 adder

The equations below show formulations of Ling's group-6 pseudo carry and propagate terms using primary inputs.

$$
\begin{aligned}
H_{3_i} &= a_i \cdot b_i + a_{i+1} \cdot b_{i+1} + (a_{i+1} + b_{i+1}) \cdot (a_{i+2} \cdot b_{i+2}) \\
I_{3_i} &= (a_{i+1} + b_{i+1}) \cdot (a_{i+2} + b_{i+2}) \cdot (a_{i+3} + b_{i+3}) \\
&\textit{and} \\
H_{6_i} &= H_{3_i} + I_{3_i} \cdot H_{3_{i+3}} \\
I_{6_i} &= I_{3_i} \cdot I_{i+3}
\end{aligned}
\tag{5}
$$

At end of the second stage, group-36 pseudo generate (H36) and propagate (I36) bits are generated. Final pseudo carry and its conversion to physical carry and sum generation are done in the third stage. This configuration has reduces input loading and wire complexity compared with other designs.

3.3 Sparse Tree Pseudo Carry Look-ahead Adder (PCLA842)

Figure 4 shows the prefix diagram of a sparse tree pseudo carry look-ahead adder with sparseness of 2. It has the same number of stages on its critical path as other adders in this paper.

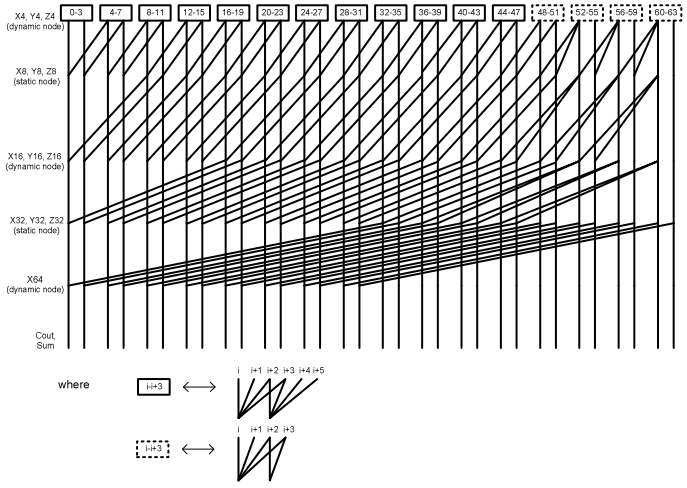

Figure 4 Prefix diagram of sparse tree PCLA842

There are two key differences between this adder and others. First of all, it transmits pseudo carry (X), propagate (Y) and complementary (Z) terms so that the group-4 pseudo carry (X4) and propagate (Y4) terms can be generated from primary inputs using footed dynamic gates with the maximum transistor stack height of 4. If Ling's recursion is used, it would result in an implementation consisting of footed dynamic

gates with the stack height of 5. Secondly, non-uniform prefix scheme is used in order to reduce branching at nodes on its critical path. Group-8 pseudo generation (X8), propagate (Y8) and complementary (Z8) bits are generated at every other bit in the static gate of the first stage. Final pseudo carry (X64) is generated in the dynamic gate in the third stage which is then used to select the correct sum. This approach reduces the complexity of the wiring and the loading caused by wires at later stages. Thus, it is more efficient than other published adder designs for performance-demanding applications.

4 Evaluation results

We have implemented Park adder, Ling662 and PCLA842 using both CD and LSDL. Transistors with nominal threshold voltage are used for each design. All dynamic gates have maximum NMOS stack height of 4 and are footed to reduce leakage current. Each design is optimized for delay using Logical Effort [1, 2, 8] in a 65nm SOI CMOS technology with a nominal supply voltage at 85°C. The optimization points of each design are obtained by varying the transistor sizes of the input drivers. The wire lengths are estimated using 4 μm bit pitch. The bit pitch and bit stack arrangements are often determined during floorplanning after iterations of simulating critical path and logic repartition [18]. In a custom data-path design, the wires between two stages are generally placed orthogonal to the data signals. The length of these wires is then a function of how many bit positions they have to cross in order to reach their destination nodes. The bit pitch of a data-path is usually determined during floorplanning. Hence, the length of each wire inside the data-path is simply the product of the bit pitch and the number of bit positions it has to cross [6]. Wire widths are allowed to increase in order to minimize the RC delay. The resultant wire capacitances are then calculated based on the characteristic of the technology. The outputs of all designs are loaded with the equivalent of a 50 μm gate. This load simulates the actual loopback bus in a pipeline stage.

The branching effort of each node is calculated iteratively to assure correctness. We assume that both polarities of sum output are allowed. This is reasonable since conversion of polarity can be achieved within bus drivers. A two-phase non-overlapping clocking methodology is a preferred style in LSDL implementations. In order to maintain the fairness of this comparison, the same clocking methodology is applied to both LSDL and CD designs. All delays are normalized to the fanout-of-4 (FO4) inverter delay, which does not vary over process, temperature and supply voltage [5].
We use the equivalent transistor width to estimate the energy value at a performance point. Hence, the comparisons in transistor width-delay space are equivalent to comparisons in energy-delay space. To calculate the equivalent transistor width, we assume 100% clock activity rate for both LSDL and CD. Our experiences show that the data activity rate for CD gates is about 50% and the data activity rate for LSDL gates can be in the range of 10% to 25%. Our intention here is not to justify the data activity rate for our LSDL analysis but to provide a way to compare LSDL against CD. Thus,

we have performed analysis on these LSDL implementations with data activity rate at both its lower and upper bounds, which corresponds to rates of 10% and 25%. For the simplicity of our analysis, we ignore any overheads in generating the clock phases. Figure 5 and Figure 6 show the transistor width vs. performance results with activity rate for LSDL to be 10% and 25% respectively. Solid curves show the results of CD implementations and dotted curves show the results of LSDL implementations. At the data activity rate of 10% for LSDL, it is evident that more than 50% energy saving is possible when using LSDL compared to CD. This is at the cost of having a stronger input driving source to achieve the same performance point as CD.

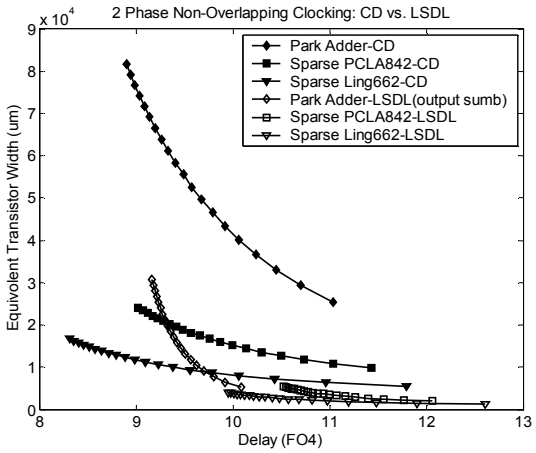

Figure 5 Results assuming 10% data activity rate for LSDL

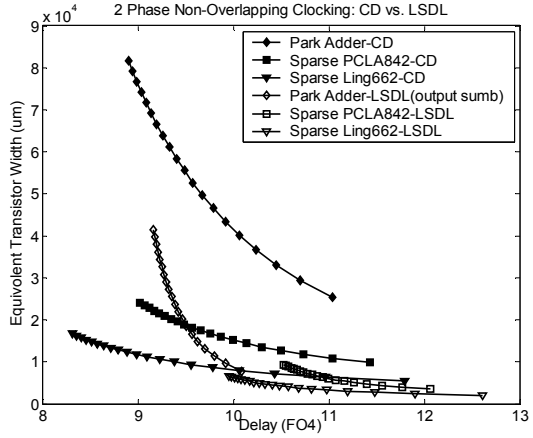

Figure 6 Results assuming 25% data activity rate for LSDL

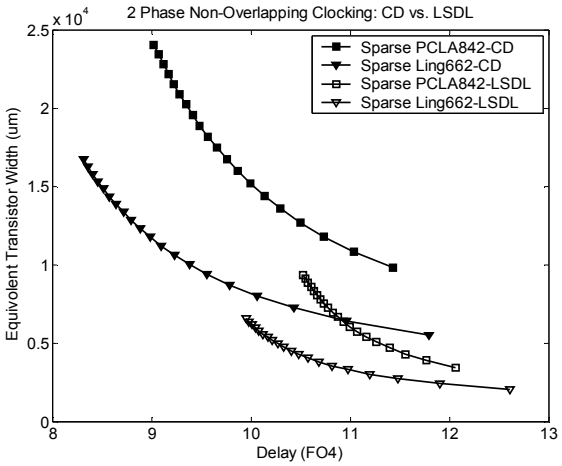

Figure 7 Results of PCLA and Ling with 25% data rate for LSDL

At 25% data activity rate, LSDL implementations of PCLA842 and Ling662 are more advantageous than their CD counterparts at speeds slower than 10 FO4 and 10.5 FO4 respectively. Figure 7 shows results of PCLA842 and Ling662 at upper bound data rate of for LSDL. It clearly demonstrates that significant energy savings can be achieved using LSDL.

The CD implementations of PCLA842 and Ling662 are capable of operating at speeds faster than 10 FO4. It is difficult for LSDL to reach this speed range without a strong input driving source. Hence, CD is more suitable for the critical paths in a design.

5 Conclusion

In this paper, LSDL and CD are compared in a 65nm SOI technology. The evaluation is based on several adder topologies targeted at performance-demanding designs. Design points in the energy-delay space are obtained using a nominal supply voltage at 85°C. Estimations show that for all implemented topologies the LSDL is desirable at cycle times greater than 10.5 FO4, which is suitable for low power applications. CD can be used in timing-critical paths. However, careful selection of paths is required in order to keep power within budget. As technology continues to scale, power management will be more critical in a design than speed improvement. Careful implementation of circuit and placement may be required to provide an energy-efficient solution for future technologies.

Acknowledgement

This work is supported by IBM Academic Learning Assistance Program.

References

1. V. G. Oklobdzija, et. al, “Energy-Delay Estimation Technique for High-Performance Microprocessor VLSI Adders”, Proceedings of the International Symposium on Computer Arithmetic, ARITH-16, Santiago de Compostela, SPAIN, June 15-18, 2003.
2. V. G. Oklobdzija, et. al, "Comparison of High-Performance VLSI Adders in Energy-Delay Space", IEEE Transaction on VLSI Systems, Volume 13, Issue 6, pp. 754-758, June 2005.
3. R. Montoye, et. al., “A Double Precision Floating Point Multiply”, Digest of Technical Papers, 2003 IEEE International Solid-State Circuits Conference, San Francisco, Feb. 2003.
4. T. W. Houston, et. al., “Compound Domino CMOS Circuit”, U.S. Patent No. 5,015,882, Issued: May 14, 1991.
5. Horowitz, M. "VLSI Scaling for Architects", Presentation slides, Computer Systems Laboratory, Stanford University.
6. Kevin J. Nowka, “Issues in High-Performance Processor Design”, book chapter in The Computer Engineering Handbook, Vojin G. Oklobdzija (Ed.), CRC Press, Inc., 2002.
7. “Interconnect”, International Technology Roadmap for Semiconductors (ITRS) 2005.
8. I.E. Sutherland and R.F. Sproull, “Logical Effort: Designing for Speed on the Back of an Envelope”, in C.H. Sequin, Ed., Advanced Research in VLSI. Cambridge, MA: MIT Press, 1991.
9. P.M. Kogge and H.S. Stone, “A parallel algorithm for the efficient solution of a general class of recurrence equations”, IEEE Trans. Computers, Vol. C-22, No. 8, 1973, pp.786-793.
10. H. Ling, “High-Speed Binary Adder”, IBM J. Res. Dev., vol.25, p.156-66, 1981.
11. J. Park, et. al, “470ps 64-bit Parallel Binary Adder”, Digest of Technical Papers, of 2000 Symposium on VLSI Circuits, 2000.
12. S. K. Mathew, et. al, “Sub-500-ps 64-b ALUs in 0.18-μm SOI/bulk CMOS: design and scaling trends”, IEEE Journal of Solid-State Circuits, Volume 11, Nov. 2001.
13. S. Mathew, et. al, “A 4-GHZ 130-nm address generation unit with 32-bit sparse-tree adder core”, IEEE Journal of Solid-State Circuits, Volume 38, Issue 5, May, 2003.
14. S. Mathew, et. al, “A 4GHZ 300mW 64b Integer Execution ALU with Dual Supply Voltage in 90nm CMOS”, Digest of Technical Papers of 2004 IEEE International Solid-State Circuit Conference, San Francisco, February, 2004.
15. Y. Shimazaki, et. al, “ A shared-well dual-supply-voltage 64-bit ALU”, IEEE Journal of Solid-State Circuits, Vol. 39, Issue 3, March, 2004.
16. Bart R. Zeydel et. al, “Efficient Mapping of Addition Recurrence Algorithms in CMOS”, 17th IEEE Symposium on Computer Arithmetic, June 27-29, 2005.
17. G. Dimitrakopoulos and D. Nikolos "High Speed Parallel-Prefix VLSI Ling Adders", IEEE Transactions on Computers, Feb. 2005.
18. R. M. Averill, et. al, “Chip Integration Methodology for the IBM S/390 G5 and G6 Custom Microprocessors”, IBM Journal of Research and Development, Vol. 43, No. 5/6, 1999.