PSCML: Pseudo-Static Current Mode Logic

Yier Jin* and Yiorgos Makris†
*Department of Electrical Engineering, Yale University
†Department of Electrical Engineering, The University of Texas at Dallas
{yier.jin@yale.edu, yiorgos.makris@utdallas.edu}

***Abstract*—We introduce a new logic style called Pseudo-Static Current Mode Logic (PSCML), which aims to alleviate the power consumption and delay overhead concerns that have thwarted the wide-spread acceptance of a previously proposed Dynamic Current Mode Logic (DyCML) style. Different from DyCML, the proposed new logic style may be viewed by its environment as static, hence any PSCML-based gate/module can be readily embedded into static CMOS designs to construct CMOS/PSCML hybrid circuits. Simulation results show that, at the cost of some area increase, PSCML is faster and consumes less power than DyCML for most applications.**

I. Introduction

As low power consumption becomes increasingly important in contemporary circuit design [1], several methods have been proposed to address the problem. Most of these methods follow one of two trends. The first trend aims to lower the supply voltage. By reducing the voltage that drives the entire design, the dynamic and overall power expended by the circuit is reduced. These savings, however, come at the expense of reduced performance, hence such methods are mainly geared towards power-critical applications, such as mobile devices, smart cards, battery-powered devices, etc. The second trend aims to curtail unnecessary power expenditure through power gating. Unused circuit parts are temporarily shut off or switched to a low power-consuming standby mode in order to save power. However, a complicated power controller may be required, possibly offsetting the benefits.

Along an orthogonal direction to the above two trends, researchers are developing new circuit architectures which consume less power yet maintain high performance. Among them MCML (MOS Current Mode Logic [2]) is a good example which reduces dynamic power consumption at the expense of relatively large static power dissipation and design complexity. In [3], authors combined MCML with dynamic logic style to propose a high-performance DyCML (Dynamic Current Mode Logic) architecture, the advantages and disadvantages of which are detailed in Section II.

In order to overcome the limitations of DyCML, in this paper we present a static version of this architecture which we refer to as Pseudo-Static Current Mode Logic (PSCML). In most applications, this new PSCML architecture achieves lower power consumption and faster computation speed as compared to DyCML. Furthermore, its static interface to the environment enables PSCML blocks to serve as drop-in replacements for static CMOS equivalents, which is not possible with DyCML. The PSCML architecture can be widely used in power-constrained areas such as cell phones, wireless devices, etc. Furthermore, when implemented in cryptographic components such as encryption/decryption cores in smart cards, PSCML blocks can increase the robustness against differential power analysis (DPA) attacks [4].

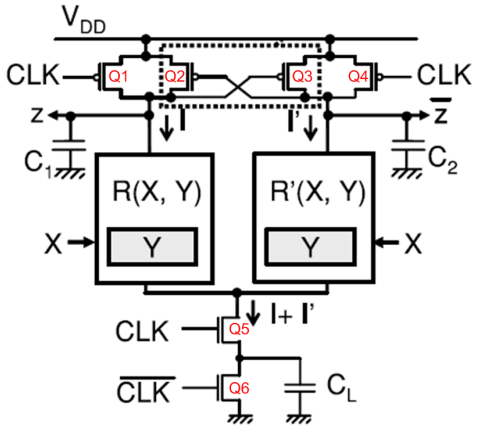

Fig. 1. Basic architecture of DyCML [5]

The rest of the paper is organized as follows: In Section II, we review the DyCML architecture and discuss its advantages and disadvantages. In Section III, we introduce the proposed PSCML architecture and we present its key characteristics. In Section IV, we report the results of comparing the DyCML and PSCML styles using two sample circuits which were designed using both technologies. Conclusions are drawn in Section V.

II. Dynamic Current Mode Logic (DyCML)

DyCML is a reduced swing logic style, which was first proposed in order to decrease propagation delay in high-performance circuit designs [3] and has, subsequently, been implemented successfully in various applications [5], [6]. Figure 1 shows the basic architecture of DyCML, which includes three parts: (i) a cross-coupled keeper (i.e., PMOS transistors Q1, Q2, Q3 and Q4), (ii) two NMOS/resistor branches of different equivalent resistances, $R(X,Y)$ and $R'(X,Y)$, and (iii) a power saving foot (i.e., NMOS transistors Q5 and Q6, and a capacitor C_L acting as virtual ground for power saving purposes). Q1 and Q4 are controlled directly by the clock signal `CLK`. Q5 is connected to `CLK` whereas Q6 is connected to the inverse clock signal $\overline{\texttt{CLK}}$. The NMOS/resistor branches constitute the functional logic of the circuit. Any structures can be used to construct these branches, as long as they provide appropriately differing resistances for each input combination.

The working mechanism of DyCML can be divided into two phases: precharge and evaluation. When `CLK` is low, the whole DyCML circuit is in the precharging phase (i.e., Q1 and Q4 are ON and current flows from the power supply to the output nodes). At the same time, Q6 is ON to discharge electrons from the virtual ground node to the ground.

When CLK switches to high, the circuit is in the evaluation phase. In the beginning of this phase, all four PMOS transistors (Q1-Q4) are OFF to separate the output nodes from the power supply. During the entire evaluation phase, with Q6 OFF and Q5 ON, the virtual ground acts as the ground. As time elapses, both the output and the inverse output nodes are discharged, each through its own NMOS/resistor branch with different resistances. As soon as the voltage of the winning node drops below the PMOS transistor threshold voltage Vt, the PMOS transistors in the cross-coupled keeper (Q2 or Q3) will be turned on so that the winning output node will strengthen its leading position and will finally reach a low voltage, while the other output node remains at high voltage.

From the above description, it is apparent that the larger the resistance difference between the two branches, the better the performance that DyCML can achieve. We already know that NMOS shows higher carrier mobility than PMOS and therefore, less resistance for the current to come through. Also, NMOS transistors require smaller transistor widths, compared to PMOS, due to their higher mobility and consequently, the power and area savings become larger, compared to the case of using PMOS in the two branches. In the rest of the paper, we only consider NMOS transistor branches.

Advantages of DyCML include fast speed, noise immunity and robustness to supply voltage scaling. Compared to Domino and MCML, DyCML achieves the shortest delay and, by extension, the lowest Energy Delay Product (EDP, $delay^2 * power$) [3]. Nevertheless, the following shortcomings of DyCML have prevented its widespread use:

- DyCML family behaves well regarding power dissipation only when the switching activity of inputs is comparable or close to the clock's frequency. However, this is in general not the typical case in VLSI circuits and it may happen especially in datapaths and in modules implementing DSP algorithms or in arithmetic components.
- Distributing both CLK and $\overline{\text{CLK}}$ can be cumbersome and incur high overhead for large circuits and will also add to the overall power consumption.
- Integrating DyCML modules with static CMOS to form a DyCML/CMOS hybrid is not straightforward. In order to use one DyCML gate, we would have to convert the entire module to dynamic logic. Given the sparsity of dynamic logic intellectual property (IP) modules, DyCML becomes a less attractive option for large circuit design.

III. Pseudo-Static Current Mode Logic (PSCML)

A. Basics of PSCML

In order to retain the advantages of DyCML but overcome its shortcomings, we developed PSCML as an improvement upon the DyCML architecture. The first shortcoming that PSCML aims to address is the excessive power consumption during the precharge phase of DyCML. For dynamic logic families (including DyCML), the time required to charge the output nodes is typically much shorter than the time required to stabilize the outputs during the evaluation phase. Therefore, if we implement unbalanced clock signals (i.e. clock signals with less than 50% duty cycle, as shown in Figure 2), as opposed to the balanced clock signals currently employed, we can reduce the power consumption of the entire circuit and provide a longer evaluation phase.

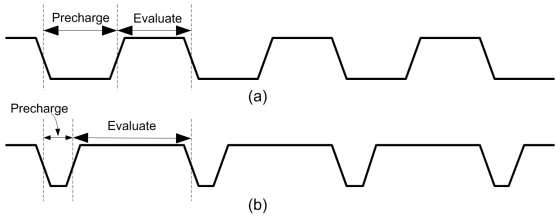

Fig. 2. (a) Balanced-clock signal, (b) Unbalanced-clock signal

However, the cost of generating and distributing unbalanced clock signals is much higher and will offset their benefit in lowering power consumption. Instead, our approach is to generate unbalanced clock signals locally, i.e., to equip every gate with a pulse generator providing an unbalanced clock pulse to charge the output nodes. Combining the above considerations results in a pseudo-static version of current mode logic, which we named PSCML. Figure 3 shows the architecture of the proposed new logic, which contains two main parts: a pulse generator and a DyCML block. Input signals feed into both the pulse generator and the DyCML block. Pulses, acting as the internal unbalanced clock signal, are generated if and only if one or more input signals are switched. If no input signals are changed, no pulse is generated on the internal clock signal, hence the output of the PSCML structure is retained and no dynamic power is consumed. The overall operating procedure of PSCML is similar to that of static logic which grants PSCML a static interface.

This PSCML approach yields the benefits listed below:

- **Faster speed:** The delay of PSCML is the sum of the delays of the pulse generator and the DyCML structure. The DyCML delay can also be divided into precharging pulse width and evaluation delay. With a sophisticated design of the pulse generator (see following subsection), we can make the sum of the pulse generator delay plus the pulse width smaller than the traditional precharging phase using balanced clocks. The evaluation delay remains unchanged as in DyCML, hence the orerall delay of PSCML is expected to be shorter that DyCML.
- **Lower power consumption:** Because a shorter precharging phase reduces the dynamic power consumption and because, in most applications, input signals change less frequently than the clock signal, PSCML consumes much less power by providing a precharging pulse only when an input changes.
- **Static interface:** With a self-contained pulse generator, PSCML is now fully compatible with static CMOS logic. A PSCML module can replace its static CMOS equivalent without touching clock signals or the surrounding logic.

B. Pulse Generator

From the above description, it becomes evident that the overall performance of PSCML is closely related to the design

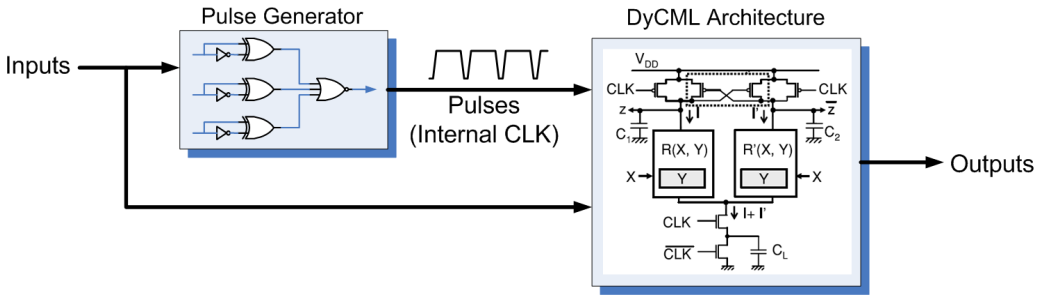

Fig. 3. Basic architecture of PSCML

of the pulse generator and the generated internal clock signal. Constraints in designing the pulse generator include:

- **Appropriate pulse width:** The internal clock signal should be low long enough to fully precharge the DyCML output nodes, yet as short as possible so that power is saved over the DyCML approach.
- **Pulse generation delay:** Since the pulse generation delay is part of the total delay of PSCML, the design of the generator should quickly covert input changes to pulses.
- **Low power consumption:** In order to keep PSCML as a competitive alternative not only to DyCML but also to other static design approaches, the power consumed for pulse generation should be minimal.
- **Robustness:** Unwanted pulses (glitches) should be avoided to prevent undesired power dissipation.

Based on above mentioned constraints, a pulse generator whose gate level architecture is shown in Figure 3, is carefully designed and fine-tuned.

IV. Simulation Results

In order to demonstrate the improvements that PSCML offers over DyCML, we designed a 2-bit adder in both logic styles and simulated it using Cadence Spectre and the TSMC $.13\mu m$ process [7]. In addition, in order to demonstrate the trend of these improvements as circuit complexity increases, we also implemented a 3-bit adder in both logic styles using the same technology.

Given the similar characteristics between PSCML and DyCML, the crucial parameters to compare include area, power consumption and computation speed. Table I lists the area comparison of 2-bit and 3-bit adders. As can be observed, the area of the PSCML-based design is around 25% larger than that of the DyCML-based design because of the insertion of pulse generators. Note, however, that this area comparison is rather inflated: in our comparison, we assume the worst case wherein each logic gate has its own pulse generator. In reality, we can share a pulse generator among adjacent gates and, thereby, lower the area overhead incurred by PSCML-based designs over DyCML-based designs, possibly at the expense of some computational speed. We should also be aware that the global clock routing area is not considered when calculating the total area for DyCML-based designs.

In order to compare power and speed between PSCML and DyCML, we first identify the highest clock frequency that the DyCML structure can achieve. Then, we sweep the frequency downwards in a step-wise fashion. For every step, we experiment with a range of frequencies for the input signals (in most applications, the inputs of a module will not change every clock cycle but at a much slower frequency). The high end of this range is the clock frequency and the low end of this range is ten times smaller than the clock frequency. By performing a comparison for this entire range, we can clearly delineate the scenarios where PSCML outperforms DyCML and assess its effectiveness for a given application.

TABLE I
Area Comparison

	DyCML	PSCML
2-bit adder	$210\mu m^2$	$280\mu m^2$
3-bit adder	$320\mu m^2$	$400\mu m^2$

Figure 4 shows the PSCML vs. DyCML comparison results for the 2-bit adder. As we mentioned in Section III.A, the delay of DyCML includes two parts: precharging phase and evaluation delay. The power consumption of DyCML is also a summation of power consumed in precharging phase and evaluation phase. The way to measure delay and power for PSCML is the same as that for static circuit [8]. The highest frequency in which the DyCML circuit operates correctly is 1GHz. Therefore, as we explained in the previous paragraph, we start our comparison assuming that the clock frequency is 1GHz and the highest input signal switching frequency is also 1GHz, for both the DyCML and the PSCML circuits. Figure 4(a) shows the power consumption and delay comparison results in this case, as we sweep the input switching frequency, with the solid line representing the DyCML-based 2-bit adder and the dashed line representing the PSCML-based 2-bit adder.

In the DyCML case, the input switching frequency has no effect on the computation speed and the power consumption for DyCML, which for `CLK=1GHz`, have values of 0.6ns and $14\mu W$ per operation, respectively. This is expected, since in DyCML pre-charging and evaluation will be performed in every clock cycle, independent of whether the inputs switch or not. In PSCML, however, performance is highly related to the input signal switching frequency because the internal pulse is generated only if one or more input signals are switched. When the input switching frequency drops from 1GHz to 100MHz, the power consumption decreases from $43\mu W$ to $5\mu W$ and the delay increases from 0.52ns to 0.59ns. Figure 4(a) plots this trend, which shows that PSCML outperforms DyCML both in terms of power consumption and in terms of computation speed for applications with input switching frequency in the

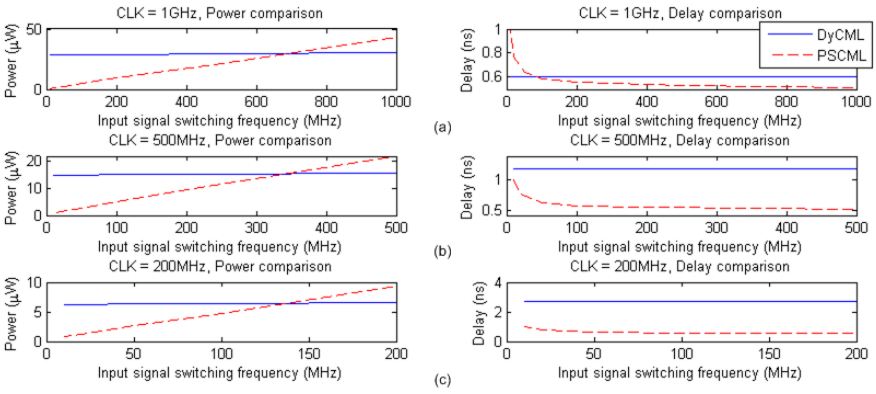

Fig. 4. Comparison between DyCML and PSCML (2-bit adder)

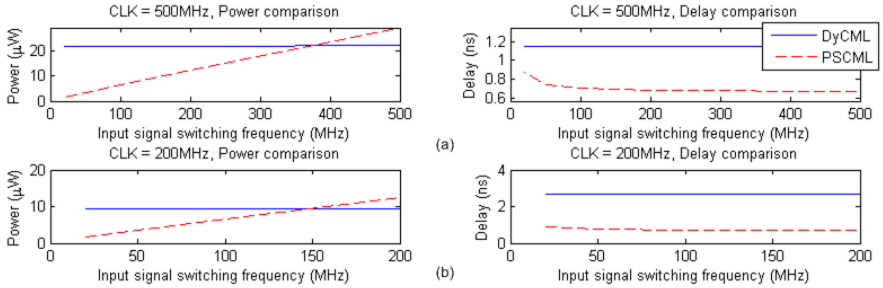

Fig. 5. Comparison between DyCML and PSCML (3-bit adder)

range between 100MHz and 700MHz. When inputs switch faster than 700MHz, PSCML consumes more power but is even faster. Using EDP as our metric, PSCML is better than DyCML for almost all input switching frequencies.

When the DyCML clock frequency is lowered to 500MHz, the benefits of PSCML become even higher, as shown in Figure 4(b). PSCML is faster than DyCML irrespective of the input switching frequency and power consumption is lower for input switching frequencies below 350MHz (which, again, is at the 70% mark of the DyCML clock frequency). Evidently, the gains in power consumption become larger as the input switching frequency decreases. Finally, as shown in Figure 4(c), if the DyCML clock frequency is lowered to 200MHz, the same conjectures can be drawn, with the additional observation that the delay improvement becomes even larger.

Figure 5 shows the comparison results for the 3-bit adder. In this case, the highest operating frequency for DyCML is 500MHz, so the upper bound of the input switching frequency is also set to this value. Comparing Figure 5 to Figure 4, we observe that, as the circuit becomes more complicated, the trend that PSCML is faster than DyCML almost for every input signal switching frequency is preserved. Furthermore, the conjecture that power consumption of PSCML is lower than than of DyCML for input switching frequencies up to 70% of the DyCML clock frequency also remains valid.

V. Conclusions

The new PSCML logic style introduced in this paper leverages the characteristics of the previously proposed DyCML style, yet encapsulates it within a pseudo-static "wrapper" in order to make it compatible with static logic. Careful design of this wrapper also addresses the power and performance limitations of DyCML. Simulation results showed that PSCML is faster than DyCML and consumes less power, as long as the input signal switching frequency is lower than 70% of the DyCML clock signal (which is typically the case in most applications). As a next step, we will construct libraries of standard-cells built on the PSCML architecture, so that automated synthesis and place-and-route tools can use these libraries and designers using the RTL-to-GDSII flow can work with these libraries to improve the performance of their designs. More complex blocks with cascaded, consecutive logic levels will also be added to the libraries.

References

[1] A. Chandrakasan and R. W. Brodersen, *Low Power Diginal CMOS Design*. MA: Kluwer, 1995.

[2] M. Yamashina and H. Yamada, "An MOS current mode logic (MCML) circuit for low-power sub-ghz processors," *IEICE Trans. on Electronics*, vol. E75-C, no. 10, pp. 1181–1187, 1992.

[3] M. Allam and M. Elmasry, "Dynamic current mode logic (DyCML): a new low-power high-performance logic style," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 3, pp. 550–558, 2001.

[4] I. Hassoune, F. Mace, D. Flandre, and J.-D. Legat, "Low-swing current mode logic (LSCML): A new logic style for secure and robust smart cards against power analysis attacks," *Microelectronics Journal*, vol. 37, no. 9, pp. 997–1006, 2006.

[5] A. Mochizuki, H. Kimura, M. Ibuki, and T. Hanyu, "TMR-based logic-in-memory circuit for low-power VLSI," *IEICE Trans. Fundamentals*, vol. E88-A, no. 6, pp. 1408–1415, 2005.

[6] S. Matsunagan, J. Hayakawa, S. Ikeda, L. Miura, H. Hasegawa, T. Endoh, H. Ohno, and T. Hanyu, "Fabrication of a nonvolatile full adder based on logic-in-memory architecture using magnetic tunnel junctions," *Applied Physics Express*, vol. 1, no. 9, pp. 091 301–1–091 301–3, 2008.

[7] *http://www.mosis.com/products/fab/vendors/tsmc/tsmc013-cl*.

[8] N. H. E. Weste and D. Harris, *CMOS VLSI DESIGN: A Circuits and Systems Perspective*. Addison-Wesley, 2005.