POWER REDUCTION IN PIPELINE DESIGNS

F. Grassert F. Sill D. Timmermann
Institute of Applied Microelectronics and Computer Science
University of Rostock
Richard-Wagner-Str. 31, D-18119 Rostock, Germany
Tel.: 0049 381 498 - 3534; Fax: - 3601
{frank.grassert, frank.sill, dirk.timmermann}@etechnik.uni-rostock.de

Abstract. *The realization of fast datapaths in signal processing environments requires fastest logic styles with synchronous behavior. This paper presents a systematic method which efficiently combines improvements on algorithm and logic level. Thus, the design of power efficient, fast and synchronous pipelines is possible. To reduce the power consumption of dynamic logic, we show methods for single-rail structures utilizing redundant number systems. Therefore, we discuss the realization of self-timed structures with single-rail logic and redundant number systems and we present a new scheme which eases the understanding and construction of such self-timed structures. First simulations for a horizontal redundant adder slice show area and power savings of 40% and 30% compared to complementary Domino logic.*

1 Introduction

The realization of data-paths with highest performance, i.e., high throughput or low latency, is a main issue in the area of signal processing. Applications, e.g. in wireless environments, emphasize power consumption as an important criterion for comparisons. On the way to more efficient algorithms, redundant number systems like Carry-Save reduce evaluation time by avoiding carry propagation as long as we stay in the redundant number system. Beneath the algorithmic level, there are several possibilities on the logic level to speedup these data-paths. The use of dynamic logic like Domino [1] ensures fastest evaluation because only N-transistors realize the logic function (explained in section 2). The question and the main scope of this paper is how to realize such fast algorithms in dynamic logic in a power efficient way.

As a realization for fastest evaluation with dynamic logic, Horowitz et al presented in [2] a self-timed scheme for a ring divider. In [3], the authors presented AC-TSPC logic, an integration methodology for self-timed schemes in synchronous designs. Sechen discusses in [4] a scheme for single-rail dynamic logic. The advantages of redundant calculation are shown in papers like [5]. However, the efficient combination of algorithm and logic level remains an open question.

In this paper, a combination of methods from algorithmic and logic level is presented which results in a scheme to design power efficient pipelines. The advantages of redundant

number systems are combined with new ideas on logic level to realize inverting functions in dynamic, single-rail logic styles and to allow single-rail, self-timed structures. This use of single-rail, self-timed logic reduces power consumption in comparison to dual-rail logic structures and does not require netlists with only non-inverting functions. Section 2 describes some basics for understanding the issues of dynamic logic. Section 3 describes the use of the self-timed structures in synchronous environments. Section 4 deals with possibilities for the combination of single-rail logic with self-timed schemes when implementing redundant number systems. A completion detection scheme is shown where redundant numbers are used in an advantageous way. Section 5 shows simulation results of a redundant adder row example and section 6 finishes with the summary.

2 Basics

Comprehension of the pros and cons of dynamic logic, distinction between single- and dual-rail and the resulting inference for clocking schemes is fundamental and will be discussed briefly. Figure 1 shows Domino logic [1] and True Single Phase Clock (TSPC) logic [6]. The dynamic principle is based on two phases controlled by a clock signal and can be explained for Domino as follows: during precharge phase (clock is low), the dynamic node is precharged to high and the output is reset to low. During evaluation phase (clock is high), the output node can be discharged to ground, depending on the input values of the logic tree. Because only N-transistors realize the logic function the evaluation is faster than in static CMOS. Again, because of merely N-transistors in the following logic the output load is less than in static CMOS.

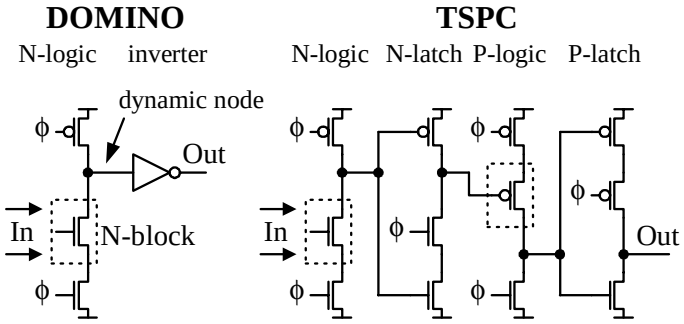

Figure 1: Domino and TSPC logic; Domino needs more than one clock phase, TSPC has alternating N-logic, N-latches and P-logic, P-latches.

However, there are two main disadvantages of dynamic logic styles: realization of inverting logic functions is difficult and clock load is extreme as a clock signal is essential for the function of every single gate. In Domino logic, no inverting functions are possible at all (always an inverting dynamic stage with a following inverter) and, therefore, a netlist with only non-inverting functions is mandatory. In TSPC logic, an inverting function can be realized with a modified structure of the P-part, called N2-part. Mostly, dynamic logic is build up as a differential style like DCVSL. Here, both the inverted and the non-inverted output is generated independently. This results in two logic blocks thus often doubling the area. Furthermore, for an inversion inverted inputs are always necessary.
In pipelined designs, clocking structure severely affects clock load. TSPC logic results in extreme pipelining with highest clock load because every logic gate realizes also a register

function. This ensures highest throughput and clock frequency. For low latency, self-timed structures are better. Here, the evaluation time of the structure can be reduced to only the sum of the single gate evaluation times without additional delays through latches. As a compromise, the authors presented in [3] AC-TSPC logic which integrates short chains of self-timed logic into a synchronous design. This approach results in lower clock load and, consequently, lower power consumption. Furthermore, a latch-free structure is possible which reduces evaluation time and, again, power consumption. However, those advantageous methods are still only usable for dual-rail or differential dynamic logic styles, since completion detection is necessary. The generation of a completion signal in dual-rail structures is simple, because the outputs of all gates are on the same level during precharge phase. It is only during evaluation phase that the outputs of the independent complementary logic blocks exhibit different values. The evaluation is complete when this change is detected.

3 Using self-timed structures in synchronous pipeline-designs

3.1 Basics of self-timed structures

There are two main ways to build up a self-timed scheme for dynamic logic: the gate outputs control the clocking of the preceding or the following gate. The main advantage of the first structure is a simple implementation with minimum evaluation time. If the evaluated outputs of a gate have settled, a completion signal is generated and this sets the previous gate in the precharge phase (inputs are processed – start precharge). Precharged outputs set the previous gate in the evaluation phase (outputs are precharged – start next evaluation, inputs can be processed). Because the evaluation of the outputs starts only with valid inputs, each gate is waiting for valid inputs during the evaluation phase. Therefore, the evaluation time of the critical path is only the sum of the gate evaluation times without additional delays.
To clarify this behavior, figure 2 shows two consecutive dynamic dual-rail gates and their timing diagram. As clock signals, the ready signals are used. In the starting point, all ready signals are high and, therefore, all gates are in evaluation phase and wait for valid inputs. A valid information on gate 1 is immediately processed and leads to valid outputs. Also, all consecutive gates process the data without delay. Therefore, in the signal diagram, the rising edges of I_1, I_2 and O_2 follow directly one after another. As shown in figure 2, the falling edge of data signal I_2 directly depends on the signal $ready_2$, because $ready_2$ sets gate 1 in the precharge phase, thus resets the outputs.

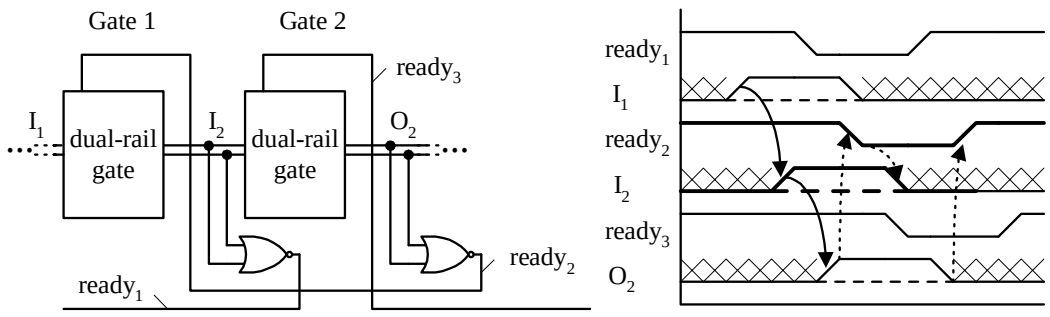

Figure 2: Basic self-timed structure and timing diagram; because the data signals I_1, I_2, O_2 are complementary, X means the precharged, not valid state of data signals.

The dependencies of the ready signals are shown in figure 2 for the signal $ready_2$, as an example. $Ready_2$ depends directly on the data outputs O_2: valid outputs result in the falling edge and precharged outputs result in the rising edge of $ready_2$. Furthermore, because $ready_2$ is the clock signal of gate 1, $ready_2$ has influence on the data signal I_2. The result of this scheme is that all ready signals depend directly on the data outputs of a gate and the *precharge* of data signals depends directly on the clock signals. In contrast, the *evaluation* of data signals depends on both the clock signal and valid inputs.
AC-TSPC [3] is a scheme to include such dual-rail self-timed structures in a global clock system. Hereby, it is possible to keep the critical path delay at minimum without additional delays through latches. Therefore, the evaluation time of the critical path is the minimum clock period. However, the structure must be carefully designed and a calculation of the timing behavior is advantageous. Furthermore, such structures reduce the sensibility against clock skew and, consequently, no additional delays for an ensured function are necessary.

3.2 Integration in a global clock scheme

Because a synchronous design of pipeline structures is essential, we present a modified scheme for the integration of short self-timed chains in a synchronous clocking scheme. Because each single dynamic gate requires a clocking signal, the reduction of the very high clock load is advantageously. Therefore, the application of a self-timed scheme which is integrated in a synchronous clock results in power reduction. For this, we present new modifications of the AC-TSPC logic [3]. A simple self-timed scheme of a few gates (3 to about 10) is realized (figure 3) and forms a synchronous block by controlling the last gate with the global clock.

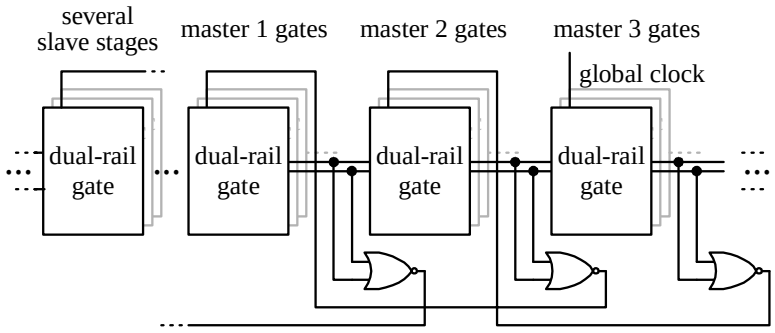

Figure 3: Structure of a synchronous block with an internal dual-rail, self-timed scheme.

In contrast to the old scheme, the new scheme is completely synchronous from an external point of view. Furthermore, the split into master and slave stages eases the comprehension. The last three gates are called master gates, and all others are called slave gates (see figure 3). Notice, that the distinction between master and slave gate does not mean a different structure of these gates. All gates are of dynamic dual-rail logic. The main clock controls directly each last gate of a chain (third master gates). Indirectly, the clock controls then the behavior of the other two master gates through the self-timed structure. That means, that the precharge and evaluate phases of master gate two depend on the outputs of master gate three. Only when the global clock sets the master gate three in evaluation or precharge, a change of the phases of master gate two can be forced. A similar dependency obtains for master gate one which is controlled by master gate two. In contrast, the phases of all slave gates switch completely independent from the global clock.

3.3 Starting point

For the following explanation, all slave gates are in evaluation phase but have no valid inputs at the starting point (see also figure 2). That means that the completion signals of the stages are still high, because there are no valid inputs and no valid outputs. Thus, the gates wait in evaluation for the next inputs. The first master gates are in precharge phase with precharged outputs, the second master gates are in evaluation phase with valid outputs. These gates hold the information of such self-timed block. The third master gates are controlled from the global clock and stay in precharge phase. Their outputs are precharged and so the completion signals for the second master gates stay in evaluation.

3.4 Evaluation of the slave gates

In the starting point, the self-timed block is waiting for valid inputs. If these inputs arrive, all first slave gates evaluate their outputs. Because every following slave gate is also waiting for valid inputs, all slave gates evaluate successively without delay (see figure 2). If the outputs of a slave stage are processed, the completion signal is generated and sets all gates of the previous stage to the precharge phase. Then, the precharged outputs set the completion signal back to evaluation and, therefore, all gates of the previous stage are set to evaluation phase and wait for valid inputs.

3.5 Evaluation of the master gates

With the rising clock edge, all third master gates evaluate their outputs using the valid inputs from the second master gates which store the information. If the outputs of the third master gates are valid, these information is processed by the following slave gates of the next block. Furthermore, these outputs generate a completion signal which sets all gates of the previous stage (second master gates) to precharge. After precharge, the second master gates produce the completion signal to set all first master gates to evaluation. At this point, the information from the slave part can be processed. The evaluation continues up to the second master stage. If the outputs at this stage settle, a whole cycle of the block has finished. The global clock must be back to low value before the new information arrives at the inputs of the third master stage to prevent a run-through condition.
There are constraints for an ensured function, e.g. the high phase of the global clock depends on the evaluation times of the first slave gates and signal connections between parallel blocks must be respected for the generation of the completion signals. We developed a software tool which automatically verifies a given netlist that all constraint are satisfied and which gives clues for further improvements of the netlist.

4 Using Redundant numbers for single-rail self-timed schemes

Because self-timed schemes are based on dual-rail structures to detect a complete evaluation, a method for completion detection in single-rail logic using redundant number representations is presented.

4.1 Redundant numbers for self-timed structures

The completion detection differentiates the states of the output nodes after precharge and after evaluation. Using a representation of a redundant number, it is possible to ensure a change of one or more bits which permits completion detection. This assumes that all

parallel gates evaluate in a small range of time Δt so that completion generation does not disturb slower or faster paths. This means, that the difference in evaluation time Δt is small enough so that delayed output signals are valid before the input signals are precharged. This assumption seems reasonable in most small blocks with limited logic depth, e.g. in datapaths. As an example, table 1 shows a representation of a radix 2 signed-digit number, but similar coding is possible for other redundant number representations also with higher radix. The redundant digits –1, 0, 1 are represented with two signals, i.e. two bit. The bit representation ensures that at least one bit changes its value during evaluation. Therefore, a completion can be detected if we take both bits into account. Figure 4 shows such a structure. Another approach for a generalization of such completion detection was presented in [7].

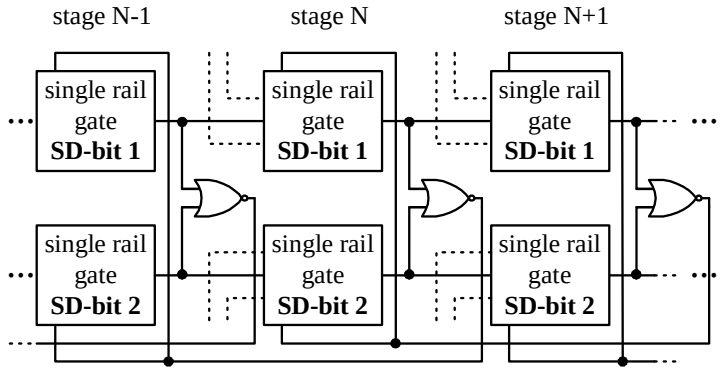

Figure 4: Self-timed structure with dynamic, single-rail logic implementing a redundant number representation.

Table 1: SD-digit representation.

Digit	Self-timed representation
-1	10
0	01
1	11
free	00 – precharge

5 Example

As an example, we choose a horizontal adder slice for a redundant multiplier using a signed-digit representation. First results are shown in figure 5. For testing purposes, 2-bit adder structures with 6 stages in a synchronous block were simulated with a 0.6 µm AMS process with 3.3V. However, an automatic timing calculation of our self-timed netlists is already in test which permits the realization of larger structures. Adder rows for a 32bit multiplier are planned and comparisons with TSPC logic and static CMOS should be possible. First netlists were processed by the timing calculation tool and the application of different logic styles with these netlists is simple.

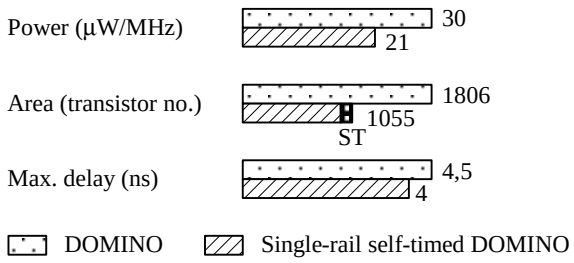

Figure 5: Comparison of the SD-adder structure in single-rail, self-timed Domino logic and complementary Domino; ST marks the area for self-timed logic.

6 Summary

This paper presents a comprehensive approach for combining methods from algorithmic and logic levels of circuit design: With redundant numbers, dynamic, single-rail, self-timed logic providing inverting functions becomes feasible. Therefore, the approach of redundant number systems to speedup the evaluation can be enhanced with fast and power efficient logic styles. The single-rail logic saves area and power. The integration of self-timed structures in synchronous designs lowers the clock load and, therefore, the power consumption. The speed is maintained through the latch-free scheme. The used self-timed scheme results in totally synchronous blocks from the external point of view. The presented ideas show a way to design fast and power efficient pipeline circuits. As a first example, a redundant number adder was simulated and shows the possibility of realizing single-rail, self-timed circuits in dynamic logic. In comparison to a dual-rail Domino realization, the power and area consumption was reduced to about 70% and 60%, respectively.

References

[1] Krambeck, R. H., Lee, C. M., Law, H.-F. S.: High-Speed Compact Circuits with CMOS. Journal of Solid-State Circuits, IEEE, Vol. SC-17, No. 3, June 1982.

[2] Williams, T. E., Horowitz, M. A.: A Zero-Overhead Self-Timed 160-ns 54-b CMOS Divider. IEEE Journal of Solid-State Circuits, Vol. 26, No. 11, November 1991.

[3] Grassert, F., Timmermann, D.: Dynamic Single Phase Logic with Self-timed Stages for Power Reduction in Pipeline Circuit Designs. IEEE International Symposium on Circuits and Systems (ISCAS), May 2001, pp. IV 144-147.

[4] Yee, G., Sechen, C.: Clock-Delayed Domino for Dynamic Circuit Design. IEEE Transactions on VLSI Systems, Vol. 8, No. 4 , August 2000.

[5] Kuninobu, S., Nishiyama, T., Edamatsu, H., Taniguchi, T., Takagi, N.: Design of High Speed MOS Multiplier and Divider Using Redundant Binary Representation. Proc. 8th. Symposium on Computer Arithmetic, New York, 1987, pp. 80-86.

[6] Yuan, J., Karlsson, I., Svensson, C.: A True Single Phase Clock Dynamic CMOS Circuit Technique. IEEE Journal of Solid-State Circuits, Vol. SC-22, 1987, pp. 899-901.

[7] Grassert, F., Timmermann, D.: Single-Rail Self-timed Logic Circuits in Synchronous Designs. IEEE MWSCAS Conference, Tulsa, August 2002.