Adaptive Clock Scheduling for Pipelined Structures

Ben Kuiper and Sorin Cotofana
Computer Engineering Lab, Delft University of Technology,The Netherlands

Abstract— **This paper introduces a technique called Adaptive Inverter Chain Based Clock Scheduling, which can observe and compensate delay variations in pipelined structures. The main idea behind the method is to expose the data and the clock to the same variations such that the register data sampling process is not disturbed by variations. The proposed scheme also includes a mechanism to detect time failures and to take counter actions to recover from such situations. When compared with other state of the art proposals, which require the augmentation of the registers, our proposal requires a relative smaller area overhead due to the fact that it is focussed on clock and not on data. Moreover our simulations indicate that, depending on the specific delay variations and pipeline logic delay sensitivity to input data patterns, it can enable an up to 46% performance improvement.**

I. Introduction

The clock frequency of synchronized circuitry is determined by the worst-case delay of the critical path in the combinational logic plus augmented with the register overhead and a certain safety margin. Due to larger delay variations determined by the ongoing downscaling of CMOS technology [1], the gap between the critical path worst-case delay and the average delay is becoming bigger. This implies that in practical designs the clock frequency is not scaling with the mean critical path delay made possible by the advances in technology.

Currently some systematic approaches have been proposed that can make the clock period less than the worst-case critical path delay and consequently improve performance, e.g., [2], which require additional circuitry at the flip-flop level. Another systematic proposal for design for variability is Casta DIVA [3], which proposes an architectural template for a System On Chip (SOC) that allows circuit and system designers to seamlessly incorporate design for variability in their designs. The proposed architecture is organized into three layers and the lowest layer (local agents) deals with variations in combinational logic.

In this paper we propose a possible local agent implementation. We introduce an Adaptive Inverter Chain Based Pipeline (AICBP) structure, which can observe and compensate delay variations. The main idea behind the method is to mold into the clock propagation the data propagation profile such that data and clock are exposed to the same variations and the register data sampling process is not disturbed by variations. This can be partially achieved by the utilization of an inverter chain [4] that lets the clock propagate along with the data. This results in a substantial reduction of the over design and in principle the clock period can be made equal to the critical path mean delay. However, given that the logic delay also depends on input data and that this simultaneous exposure to variations cannot be 100% accurate the synchronization cannot be fully achieved all the time. In view of that our proposal also includes a mechanism to detect time failures and to take counter actions to recover from such situations. Our simulations indicate that, in comparison with a traditional pipeline, our proposal can induce an up to 46% performance improvement. We note here that the actual improvement depends on the specific delay variations and on the combinational logic properties, e.g., logic delay sensitivity to input data patterns.

II. Adaptive Inverter Chain Based Pipeline

As depicted in Figure 1 the clock inputs of the registers in an AICBP are connected via inverter chains. The purpose of the inverter chain is to have a delay that is just as long as the delay of the critical path in the combinational logic plus the setup time of the flip-flops. More specifically, each inverter chain belonging to a stage should equal the delay of the critical path of that particular stage. If the inverter chain is embedded into to the corresponding combinational logic and laid out next to the critical path, it is exposed to the same voltage, temperature, fabrication process parameters, and other factors that may influence the data delay. In this way the clock delay has a strong correlation with both the static and dynamic variations of the logic delay. By adjusting the size and number of inverters in the chain the clock delay can be made equal with the delay of the critical path regardless the circumstances. Such an approach guarantees that the register data sampling process is adapted to the delay of the logic.

Due to the adaptation mechanism the clock frequency can be higher (mean delay) than the one corresponding to the same design build around a standard pipeline (worst-case delay). However, given that in practice the actual stage delay can be larger than the mean delay, an AICBP has to embed specific circuitry to prevent, detect, and resolve time faults that may occur due to the utilization of a clock frequency higher than the one determined by the worst case delay. These subsystems enable performance improvement and constitute the main difference between AICBP and [4].

A. Contamination Detection

The delay of the inverter chain, and thus also the critical path in the combinational logic, is most likely different for each stage, and it also varies in time. The consequence of these delay variations is that the data sampling cannot occur at the same time at all the pipeline registers. Due to this asynchronous register clocking two sets of data might be simultaneously present in the same pipeline stage, i.e., the

978-1-4244-4958-3/09/ 25.00

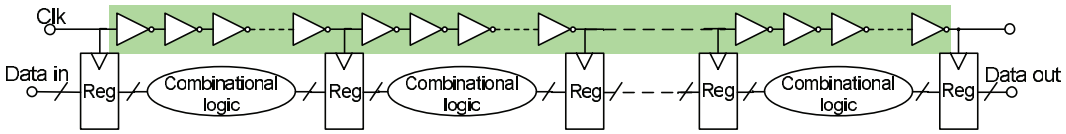

Fig. 1. AICBP Clock Signal Distribution.

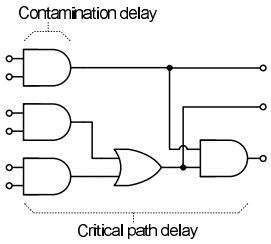

Fig. 2. Contamination and Critical Path Delay.

register at the beginning of a stage has already sampled its data, while the register at the end of the stage has not sampled yet. In such a case the just sampled data might affect the inputs of the register at the end of the stage resulting in data corruption. Such a situation is defined as a contamination fault. To cope with contamination faults AICBP embeds a contamination detection system, which detects contaminations and takes appropriate measures in order to always have the correct result at the end of the pipeline. These measures introduce a certain overhead but due to the statistical nature of delay variations contaminations are not occurring very often. Therefore the performance penalty induced by contamination is not significant when compared with the performance improvement achieved due to the smaller clock period.

The contamination fault occurrence is determined by the contamination delay, which is defined as the time it takes for a change at the combinational circuit input to affect its output. The contamination delay is data dependent but in our case we are interested in the contamination delay caused by the smallest path through the combinational logic. If data sets are spaced apart from each other with more than the contamination delay, a contamination fault can never happen.

The contamination delay also varies but we can certainly assume that it is a fraction of the critical path delay. To clarify this let us assume the case depicted in Figure 2 where the contamination delay is $\frac{1}{3}$ of the critical path delay. Because the gate which determines the contamination delay is located close to the other gates it is exposed to the same factors which influence delay. Consequently, the contamination delay is always a $\frac{1}{3}$ of the critical path delay regardless the conditions. Based on this kind of reasoning we can determine the contamination delay as a certain fraction of the critical path delay for any combinatorial logic.

Given that the contamination delay is a fraction of the critical path it is also a fraction of the inverter chain length. Thus, when a rising (falling) clock edge propagating through the inverter chain arrives at this point in the inverter chain it signals that its associated data set might affect the inputs of the register at the end of the stage. This property is used by the contamination detection system in Figure 3, which checks if the in-front data are already clocked when the rising edge arrives at the mentioned fraction. It is thus important to place the clocking point of the flip-flop in Figure 3 at a fraction of the chain length which equals the contamination and critical path delay ratio. This guarantees the detection of all contamination faults.

The delay of the inverter chain in front of the contamination detection logic in Figure 3 must have a very strong correlation with the delay of the logic which determines the smallest contamination delay. Thus one needs to properly layout the circuit in such a way that the inverter chain is placed next to this logic along with the circuitry in Figure 3.

B. Contamination Faults Recovery Procedure

When a contamination fault occurs the pipeline cannot continue the calculations because at some point invalid data will come out of the pipeline. The fault must actually be recovered, but there is no way to recover, because the intermediate pipeline results are not saved. Thus the recovery must be done by flushing the pipeline content, after which the data, which was thrown away, are re-inserted in the pipeline. This time the logic may have slightly different delays, because for example the supply voltage and perhaps also the temperature may be slightly different. According to [5] certainly 32% of the total delay variance is caused by mechanisms which influence the delay in the time domain of a clock period. Therefore the chance to have again a contamination fault for the re-executed data is rather low. This mechanism preserves the original execution order thus the same data will finally come out of the pipeline as would have been the case with no fault.

Figure 4 depicts a four stage pipeline augmented with a time fault recovery mechanism. The figure is a very basic one, because its intention is only to make clear the different components that are used for re-execution. The FIFO memory contains all data sets currently in the pipeline such that they can be injected again in the pipeline in case of a contamination.

The control logic in Figure 4 detects whether there is a contamination fault, and if so, it makes sure data from the FIFO are reinserted. The control logic resets to all stages in case of a contamination fault in order to be sure that all the systems in every chain are reset properly. The control logic can also communicate with surrounding systems, and it can be seen as the island agent in the Casta DIVA platform.

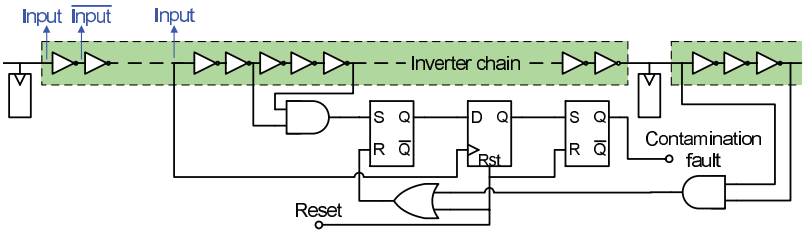

Fig. 3. Contamination Detection System.

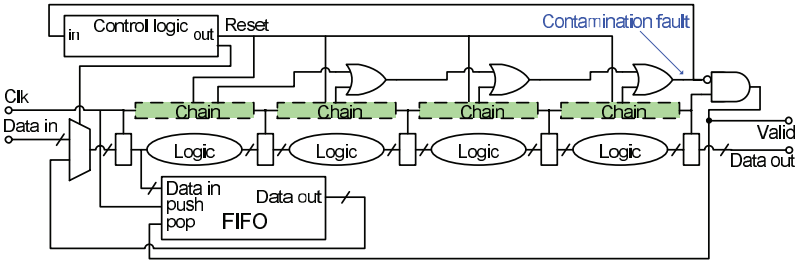

Fig. 4. Overview of the circuitry needed for fault treatment.

C. Contamination Prevention System

Re-execution is costly and has a negative impact on performance thus contamination prevention may be very useful. Preventing contaminations completely may not be of interest however as this may have a negative effect on overall performance. In view of that, we let the data sets be spaced apart from each other in such a way that contaminations are not likely to occur. This can be achieved by delaying rising edges that are too close to the previous rising edge. Delaying a rising edge can be achieved with the circuitry presented in Figure 5. This holds the rising edge at its position until the rising edge in front reaches the release point. In this way it is guaranteed that rising edges are spaced apart from each other with the *hold distance* introduced in Figure 5.

The choice for the *hold distance* value is essential for performance as: (i) if too short it causes too many contamination faults; (ii) if a rising edges is hold longer at its place, upcoming data might contaminate the data belonging to the edge which is hold.

III. Experimental Results

To get a preliminary evaluation of the potential performance we carried on VHDL simulations of an AICBP based array multiplier. A logic delay variance of $\sigma/\mu = 0.1$ (μ = mean delay critical path, σ = variance critical path) is chosen. This should be a realistic delay, and according to [5] and [6] this might already be too severe. The results are presented in Figures 6 and 7 for a simulation run of 100,000 cycles. The contamination/propagation ratio is $\frac{5}{9}$, and all the values in these graphs are percentages of the traditional pipeline clock period. It is assumed that this clock period is equal to $\mu + 3\sigma$. The horizontal axis of the figures specifies the clock frequency and the vertical axis the average frequency (over 100,000 cycles) at which the pipeline outputs data. Due

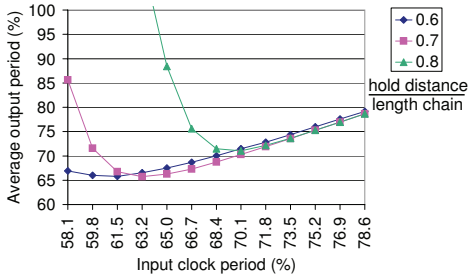

Fig. 6. Performance of a 4 stage AICBP (100,000 clock cycles)

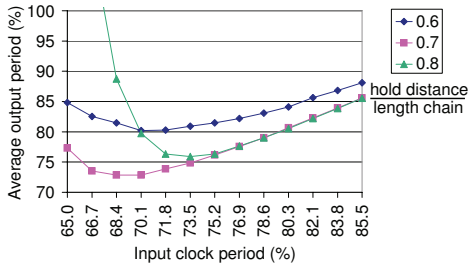

Fig. 7. Performance of an 8 stage AICBP (100,000 clock cycles)

to contamination faults the AICBP will not regularly output data, and therefore the average output frequency over 100.000 cycles is presented.

It can be observed that there is an optimal clock frequency, and at the optimal clock frequency the performance improvement is 35% and 28% for a 4 stage and 8 stage AICBP, respectively. As expected, the position of the hold system is affecting the performance. Placing the hold system earlier in the chain, like *hold distance/length chain = 0.6*, worsens performance, because the occurrence of small contamination

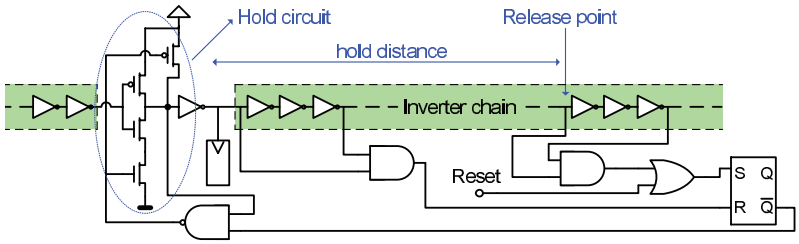

Fig. 5. The Hold System.

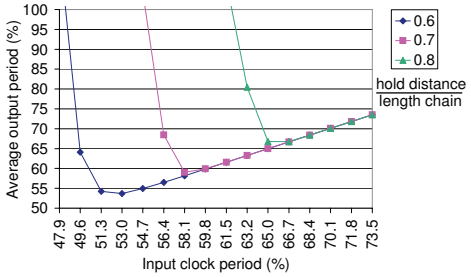

Fig. 8. Performance of a 4 stage ICBP with a dynamic delay distribution of $\sigma/\mu = 0.06$ (100,000 clock cycles)

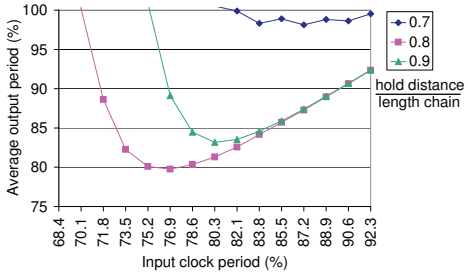

Fig. 9. Performance of a 4 stage ICBP with a contamination/critical path ratio of $\frac{1}{3}$ (100,000 clock cycles)

delays increases the number of contaminations. Holding the data and rising edge too long has a negative effect though as it can be observed for *hold distance/length chain = 0.8*.

The simulations in Figures 6 and 7 assume only dynamic delay variation, i.e., the logic and the inverter chain get a new delay with a distribution of $\sigma/\mu = 0.1$ every time a new rising edge arrives at the stage. These conditions are actually too severe in comparison with reality, according to [5] only about 35% of the total delay variance is caused by mechanisms which influence delay in such a small time domain. Actually, for a more realistic case, the simulation must be set up such that the delays have a part of the $\sigma/\mu = 0.1$ fixed during simulation, and another part varies each cycle, so that this is a dynamic delay variation. It is quite difficult to make figures like Figure 6 for every possible fixed delay variation in each stage. In order to still get an impression of the performance possibilities of the AICBP, the fixed delay variations are assumed to be zero. This might be the case in a real chip as well, although this is certainly not the case for most chips. The dynamic variations are assumed to have a variance of $\sigma/\mu = 0.06$, because this would be 35% for independent statistical processes. Of course the dynamic variations are not an independent statistical process, but this is just a worst-case assumption. The results for such a simulation are presented in Figure 8. It can be seen that the performance can be improved by 46% for the *hold distance/length chain = 0.6* case. The absence of very small contamination delays due to the statistical distribution are the major cause of this improvement.

To reduce the overhead the contamination/critical path ratio can be reduced at the expanse of performance degradation. Figure 9 presents the results for a contamination/critical path ratio of $\frac{1}{3}$ and a distribution of $\sigma/\mu = 0.1$.

IV. Conclusions

In this paper we proposed a technique to improve the pipeline throughput which can observe and compensate delay variations. The main idea behind the method is to expose the data and the clock to the same variations such that the register data sampling process is not disturbed by variations. When compared with other state of the art proposals, our proposal requires a relative smaller area overhead due to the fact that it is focussed on clock and not on data. Moreover our simulations indicate that, depending on the specific delay variations and pipeline logic delay sensitivity to input data patterns, it can enable an up to 46% performance improvement.

References

[1] S.R. Nassif, "Delay variability: sources, impacts and trends", *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 368–369, February 2000.

[2] Dan Ernst et al., "Razor: A low-power pipeline based on circuit-level timing speculation", *IEEE Proc. 36th International Symp. Microarchitecture, 2003*, pp. 7–18, December 2003.

[3] Sorin Cotofana and Cor Meenderinck, "Casta DIVA - a design for variability platform", *International Semiconductor Conference, 2008. CAS 2008.*, vol. 2, pp. 373–376, October 2008.

[4] D. Andrade et al., "A new compensation mechanism for environmental parameter fluctuations in CMOS digital ICs", *Microelectronics Journal*, vol. 40, no. 6, pp. 952–957, June 2009.

[5] Sani Nassif et al., "High Performance CMOS Variability in the 65nm Regime and Beyond", *IEEE International Electron Devices Meeting, 2007.*, pp. 569–571, December 2007.

[6] Liang-Teck Pang and Borivoje Nikoli, "Measurement and Analysis of Variability in 45nm Strained-Si CMOS Technology", *Custom Integrated Circuits Conference, 2008*, pp. 129–132, September 2008.