Optimizing the Power-Delay Product of a Linear Pipeline by Opportunistic Time Borrowing

Mohammad Ghasemazar, and Massoud Pedram, *Fellow, IEEE*

Abstract— **in this paper we present and solve the problem of power-delay optimal soft linear pipeline design. The key idea is to use soft-edge flip-flops to allow time borrowing between consecutive stages of the pipeline in order to provide the timing-critical stages with more time to complete their computations. We formulate the problem of optimally designing the soft-edge flip-flops and setting the clock frequency and supply voltage so as to minimize the power-delay metric of a pipeline under scenarios using deterministic or statistical delay models. In the first problem formulation, timing violations are avoided by respecting the deterministic worst case path delays. Next, the same problem is formulated for a scenario where stage delays are assumed to be random variables, and we minimize power-delay product while limiting the probability of timing violations in pipeline. The soft-edge flip flops are equipped with dynamic error detection (and correction) circuitry to detect and fix the errors that might arise from over-clocking. Although the system is capable of recovering from the errors, there is a trade-off between performance and power saving, which is exploited to further minimize the power-delay product of the pipeline circuit in our third proposed algorithm. Experimental results show the efficacy of our proposed solution techniques for each scenario.**

Index Terms— **Power optimal pipeline design, soft edge flip flops, time borrowing, soft pipeline, power-delay product.**

I. Introduction

With the increase in demand for battery-operated personal computing devices and wireless communication equipment, the need for power-efficient design has increased. In addition, rising levels of power dissipation and the resulting thermal problems have become key limiting factors to processor performance. Due to their high utilization, pipelined data path in a modern processor is a major contributor to power consumption of the processor, and consequently, one of the main sources of heat generation on the chip [1]. Many techniques have been proposed to reduce power consumption of a microprocessor's pipeline such as pipeline gating [1], clock gating [3], and voltage scaling [4].

In this paper we present the problem of power-delay optimal pipeline design in a synchronous linear pipeline by means of applying voltage scaling and appropriately designing the flip flops. We propose mathematical solutions to this problem in both deterministic and probabilistic frameworks. Our technique is based on the idea of utilizing *soft-edge flip-flops* (SEFF) for slack passing and decreasing the error rate in the pipeline stages. The linear pipeline composed of soft-edge flip flops is called a *soft pipeline*.

Soft-edge flip-flops have a small transparency window which allows time borrowing across pipeline stages and is beneficial for reducing the effect of clock uncertainty. Soft-edge flip-flops have been used for minimizing the effect of clock skew on circuit performance [7][8] and minimize the effect of process variation on parametric yield [9]. In this work, SEFF is used for compensating for unbalanced pipeline stage delays by means of time borrowing. It is observed that this imbalance of path delays of different pipeline stages is very common in pipelined circuits [6].

In this work, we describe a unified methodology for optimally selecting the transparency window of the SEFF's in a linear pipeline so as to achieve the minimum power-delay product for the pipeline by means of opportunistic time borrowing. We take on three power-delay optimization problems as explained next. In the first problem formulation, timing violations are avoided by respecting the worst case path delays (calculated by static timing analysis and treated as deterministic values) for every stage in a pipeline. Next we formulate the same problem for a scenario where stage delays are assumed to be random variables, and find the solution with minimum power-delay product while ensuring that the probability of timing violations due to increased operation frequency of pipeline is lower than a threshold. Thirdly, we allow timing violations to take place but then implementing a mechanism to detect and fix the generated errors while accounting for the corresponding power and delay penalties for error correction.

Preliminary versions of this research appeared in [10][11]. This paper substantially extends previous works in several directions:

i) Three general problem formulations are presented, along with one special case of the third formulation that is similar to problem presented in [10]. The first formulation is similar to the one presented in [11], but with major modifications.

ii) This paper proposes to use the power-delay product metric as the objective function in optimizations. Also, the timing constraints of time borrowing are redefined.

iii) Designs of a number of SEFF circuits are included.

iv) Experimental results have been redone and extended to reflect the aforesaid changes.

v) Mathematical proofs for optimality of solutions and convexity of problems are provided.

The remainder of this paper is organized as follows. In section II we provide some background on pipeline design. Soft-edge flip-flops and their characteristics are introduced in section III. Section IV describes our proposed techniques for optimizing power-delay in a soft pipeline in different frameworks. Sections V and VI are dedicated to experimental results and brief summary of related work, respectively, while section VII concludes the paper.

II. Background

A. Timing Constraints in a pipeline

A simple (synchronous) 2-stage linear pipeline circuit is depicted in Fig. 1. A linear pipeline is defined as a pipeline with the following properties: (i) processing stages are linearly connected, with no feedback loops (ii) it performs a fixed function, and (iii) stages are separated by flip-flops which are clocked with the same *clk* signal. We call the set of flip-flops that separate consecutive pipeline stages as a *FF-set*, e.g., FF_0 … FF_2 in Fig. 1 are FF-sets.

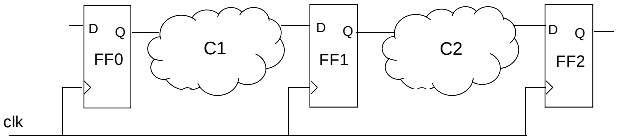

Fig. 1. A simple linear pipeline.

Clearly, delay of combinational circuit and interconnect[1] depend on the supply voltage of pipeline (see eq. (3) and (4)); so are the timing characteristics of the flip-flops, such as setup time, hold time and clock-to-Q delay (and D-to-Q delay; see section III.A). Let's assume the pipeline is operating under voltage level v_j (any variable with subscript *j* in the following equations denotes its value under supply voltage *j*). To guarantee the correct operation of the pipeline, the following timing constraints must be satisfied in all stages of pipeline:

$$d_{ij} \le T_{clk,j} - t_{cq,(i-1)j} - t_{s,ij} \quad \forall i{:}\, 1 \le i \le N \tag{1}$$

$$\delta_{ij} \ge t_{h,ij} - t_{cq,(i-1)j} \quad \forall i{:}\, 1 \le i \le N \tag{2}$$

where d_i and δ_i denote the maximum and minimum delays of combinational logic in stage *i*, T_{clk} denotes the clock cycle time, $t_{s,i}$ and $t_{h,i}$ are setup and hold times of flip-flops in the i^{th} FF-set whereas $t_{cq,i-1}$ denotes clock-to-Q delay of flip-flops in *i*-1st FF-set. *N* denotes the number of pipeline stages.

Inequality (1) gives the constraint set on the maximum delays of combinational logic and flip-flop timing characteristics to prevent setup time violations. Conversely, inequality (2) specifies the constraint set on the minimum delay of pipeline stages in order to prevent short path data race hazards. Notice that to account for the effect of clock skew, t_{skew}, we can simply add t_{skew} to the left side of inequality (1) and subtract it from the left side of inequality (2).

B. Combinational Logic Block Modeling

When the supply voltage of a combinational logic is changed, its delay can be obtained from alpha-power law [8]:

$$d_{ij} = d_i(v_j) = \lambda_j \left(\frac{V_0 - V_t}{v_j - V_t}\right)^{\alpha} d_i(V_0) \tag{3}$$

$$\delta_{ij} = \delta_i(v_j) = \lambda_j \left(\frac{V_0 - V_t}{v_j - V_t}\right)^{\alpha} \delta_i(V_0) \tag{4}$$

where α is a technology parameter which is around 2 for long channel devices and 1.3 for short channel devices, and V_t denotes the magnitude of the threshold voltage of transistors. Coefficient λ_j captures the effect of temperature increase (due to power consumption) on delay, and is defined as (5).

$$\lambda_j = (1 + \left.\frac{\partial d}{\partial \theta}\right|_{v_j} \Delta\theta(v_j)) \tag{5}$$

In the above equation $\Delta\theta(v_j)$ is the increase in steady state temperature of circuit under voltage level v_j with respect to temperature at V_0, and $\partial d/\partial \theta$ is the voltage-dependent slope of delay-temperature curve at voltage level v_j (which captures *inverted temperature dependence* effect, too [12]). We assume the only source of temperature increase is the circuit's power consumption (based on circuit's thermal models [13]), which is itself a function of voltage as given in (6). Hence, the steady state temperature of a circuit can be calculated for a voltage v_j.

Note that equations (3) and (4) are used to calculate worst-case delays under the assumption that V_t does not vary (no process variation). For the scenarios that consider V_t variations, such as section IV.B of this work, it is precise to use profiled d_{ij} and δ_{ij} at any voltage.

Additionally, the total power consumption of combinational logic, P_{Comb}, changes as follows due to voltage scaling[2]:

$$P_{Comb}(v_j, T_{clk}) = \left(\frac{v_j}{V_0}\right)^2 E_{dyn} \frac{1}{T_{clk}} + \left(\frac{v_j}{V_0}\right)^3 P_{leak} \tag{6}$$

where E_{dyn} and P_{leak} are total dynamic energy dissipation and leakage power consumption of the combinational logic at nominal supply voltage V_0.

C. Effect of Delay Variations

As technology scales, process, voltage, and temperature (PVT) variations are becoming critical design concerns due to their effect on logic and interconnect delay [14]. Process variations such as random dopant fluctuations, and gate-oxide thickness variations modulate MOSFET characteristics and parasitic components, causing variation in the switching delays of identical gates [15][16].

The random maximum and minimum stage delays are described by probability distribution functions (PDF) and cumulative distribution functions (CDF) with corresponding mean, μ, and variance, σ. This distribution has been assumed to be a Gaussian (Normal) distribution [17] in some works such as [18][19]. However, precise statistical timing analysis schemes have proposed non-Gaussian distribution models due to nonlinearity of max/min operations on delays of gates and paths and their correlation [20][21][22].

In order to account for the random variations (Gaussian or non-Gaussian) of the path delays in equations (1)-(2), one should express the probability of violating the setup or hold conditions as a function of delay variations. The probability of *satisfying* setup time constraint in pipeline stage *i* with voltage

[1] In the entire work, the interconnect delay would be integrated in the combinational logic's delay, and where we refer to combinational delay, it also includes the interconnect delay.

[2] This super-linear dependency of leakage power on supply voltage is due to combined effect of drain induced barrier lowering and off-state leakage equation (V_{dd}×I_{OFF}). Its cubic form was empirically observed in SPICE simulations.

v_j for a given cycle time $T_{clk,j}$, denoted by $p_{setup,ij}$, can be written as probability of the maximum delay of combinational logic in that stage, d_i, being less than the available slack time:

$$\begin{aligned} p_{setup,ij} &= P\{d_{ij} \le T_{clk,j} - t_{s,ij} - t_{cq,(i-1)j}\} \\ &= F_{ij}^{d}(T_{clk,j} - t_{s,ij} - t_{cq,(i-1)j}) \end{aligned} \quad (7)$$

where F_{ij}^{d} denotes the CDF of delay of pipeline stage i under voltage setting j. The probability of a setup time constraint *violation* in pipeline stage i is thus calculated as:

$$\begin{aligned} q_{setup,ij} &= P\{d_{ij} > T_{clk,j} - t_{s,ij} - t_{cq,(i-1)j}\} \\ &= 1 - F_{ij}^{d}\left(T_{clk,j} - t_{s,ij} - t_{cq,(i-1)j}\right) = 1 - p_{setup,ij} \end{aligned} \quad (8)$$

Similarly, given the CDF of minimum delay of stage i under voltage setting j, F_{ij}^{δ}, probability of violating ($q_{hold,ij}$) the hold time constraint of stage i may be calculated as:

$$\begin{aligned} q_{hold,ij} &= P\{\delta_{ij} < t_{h,ij} - t_{cq,(i-1)j}\ \} \\ &= F_{ij}^{\delta}\left(t_{h,ij} - t_{cq,(i-1)j}\right) \end{aligned} \quad (9)$$

Note that we ignore the effect of variability on flip-flop timing characteristics and only focus on the effect of variability on the combinational logic delays. To a first order, the clock-to-Q and setup-time of input and output flip-flops are much smaller than the maximum delay of combinational logic, and hence, we can ignore variations of flip-flop characteristics compared to the logic. This is however not true with respect to the hold-time and the minimum delay of logic. Therefore, we insert an adequate number of delay elements (see section IV) to alleviate the hold time violation in the worst case value of (minimum) hold time of flip-flop.

The CDF of maximum and minimum delays of stage i under voltage setting j (denoted by F_{ij}^{d} and F_{ij}^{δ}, respectively) can be in the form of any distribution function. These functions are provided by the extensive statistical timing analysis of the circuit [23] (which is performed prior to our proposed algorithms). Let $\mu_{d,ij}$ and $\mu_{\delta,ij}$ denote mean values of the maximum stage delay and the minimum stage delay of i^{th} logic stage under j^{th} voltage setting, respectively, while $\sigma_{d,ij}$ and $\sigma_{\delta,ij}$ are standard deviations of corresponding delay distributions.

D. *Pipeline Delay Model*

Average pipeline delay, denoted by D, is the primary performance metric in a pipeline. It is defined as the average time it takes to process one data/instruction unit and produce a valid output. In other words, pipeline delay can be interpreted as the inverse of its effective throughput.

$$D = T_{clk} \times \frac{\text{clock cycle count}}{\text{number of valid output data}} \quad (10)$$

We assume that the pipeline can process at most one data/instruction unit if it does not encounter timing violations, hence, $T_{clk} \le D$. In a pipeline that processes each data in one cycle, its average delay is equal to the clock period, T_{clk} (that is determined by the slowest pipeline stage; see equation (1).) However, if the pipeline stalls or gets flushed, for any reason, the average processing time of data/instruction increases. In other words, the delay is not simply the inverse of the clock frequency, rather it also probabilistically accounts for the overhead of correcting potential setup time problems in an over-clocked pipeline.

III. Soft-Edge Flip-Flops (SEFF)

The key design idea of a soft-edge flip-flop (SEFF) [5] is to create a *transparency window* right after (or before in case of backward time borrowing) the clock edge, during which the data can still be captured. This allows passing of timing slacks between adjacent pipeline stages [11]. Some SEFF designs are derived by applying modifications to conventional hard-edge counterparts. We focus on some of the most widely used flip-flop circuits in state-of-the-art processors [25]. SEFF designs based on master-slave FF (MSFF), hybrid latch FF (HLFF) and monostable-based FF (MBFF) are studied in this work.

Fig. 2 illustrates the design of master-slave SEFF, used in IBM Power PC 603 processor. The key modification in the SEFF version is that by delaying the clock of the master latch, both master and slave latches are ON for the duration of transparency window. Fig. 2 (b) illustrates the timing diagram for key signals of a master-slave SEFF. The dashed square highlights the transparency window which is the overlap of *clk* and its delayed version, *clkd*. If the overlap between edge of *clk* and the latching edge of *clkd* is larger than the delay through the master latch, the master–slave pair is transparent to the input during the window after the edge of main clock, *clk*. The delayed clock and its reverse-polarity can be produced locally for each FF-set (or multiple FF-sets that have equal transparency window size) by utilizing some inverter chain, appropriately sizing them and changing chain length in order to achieve the desired transparency window size.

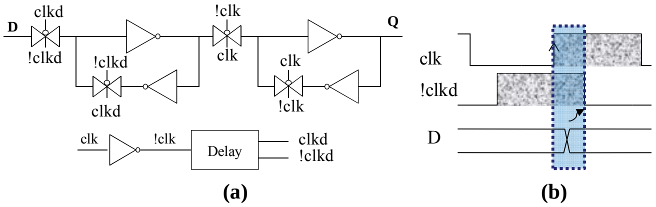

Fig. 2. Positive-edge triggered master SEFF (a) circuit (b) Timing

The hybrid latch flip-flop [5], is shown in Fig. 3, which is originally a soft-edge flip-flop; here, our purpose is to adjust size of its transparency window as required. Fig. 3 also illustrates the timing waveforms corresponding to operation of HLFF. In this figure, the shaded area denotes the transparency window, which is generated by overlap of *clk* and *!clkd*. During the time interval when both of these signals are high, both stacks of transistors act as inverter gates to transfer D to $\overline{\text{S}}$ and then to Q. In order to increase the transparency window size in the HLFF, delay of inverter chain, I_1 to I_3 in Fig. 3(a), should be decreased by the desired amount.

HLFF is one of the fastest SEFF designs used in industrial designs, such as AMD K6 processor [25] for its advantages of high performance and relatively small area. Large power consumption, the glitch activity, and somewhat complex implementation are its drawbacks [25]. Note that this architecture has its transparency window before clock edge. Hence, it is suitable for backward time borrowing schemes.

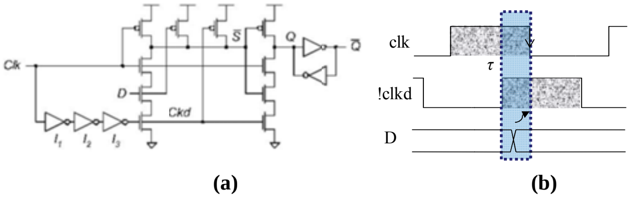

Fig. 3. Negative-edge triggered HLFF (a) circuit (b) timing diagram

Monostable-based flip flop is another industrial neg-edge flip flop that we convert it to SEFF. MBFF suffers from large area and high power consumption [25].

In order to modify MBFF's circuit to admit an adjustable transparency window size, a delay element is inserted in its design, as illustrated in Fig. 4(a). In this design, the first stage of the flip flop generates a short pulse on nodes *S* or *R* to trigger the S-R latch. The delay element essentially extends this pulse width, providing longer time for *D* to arrive and get captured in the SR latch. Fig. 4(b) demonstrates timing waveform of this SEFF for *D*=1 (for *D*=0, the pulse applies to *R*). The triggering pulse can be de-asserted as early as a t_1 delay after the negative-edge of *clk* and is asserted exactly after a t_2 delay after the negative edge of *!clkd*.

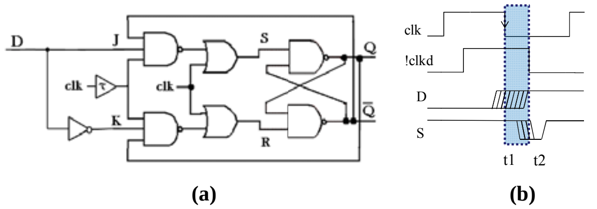

Fig. 4. Monostable-based SEFF (a) circuit (b) timing of operation

Due to the practical advantages of Master Slave based SEFF we will focus on this design for the rest of this paper to derive equations and use it design problems. Similar equations and discussions hold for other SEFF designs.

A. SEFF Timing Characteristics

To *optimally* select the transparency window of the SEFF's, we must accurately account for the effect of the transparency window on SEFF's power consumption and its timing characteristics, i.e., setup time, hold time, clock-to-Q delay and D-to-Q delay. The setup time, t_s, and hold time, t_h of a SEFF may be modeled as linear functions of the transparency window size:

$$\begin{cases} t_s(w) = a_1 w + a_0 \\ t_h(w) = b_1 w + b_0 \end{cases} \tag{11}$$

where *w* denotes the transparency window size and a_0 through b_1 are technology- and design-specific coefficients. Experimental SEFF characterization data provided in Fig. 5 confirm the linear model for SEFF timing characteristics.

The *clock-to-Q delay*, t_{cq}, of SEFF is practically independent of the transparency window width. It is defined as the delay between the positive edge of clock and the time that output is valid when input data arrives before the transparency window.

We define the term *D-to-Q delay* of a SEFF, t_{dq}, to denote the input to output propagation delay of data when it is transparent. t_{dq} is also independent of transparency window width (see Fig. 5.)

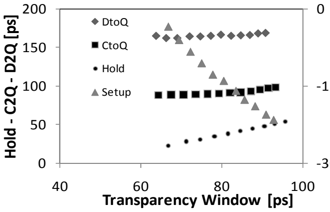

Fig. 5. Timing characteristics of SEFF – hspice simulations at 90nm

If the supply voltage of the flip-flop can be adjusted to a new voltage level, v_j, then the coefficients of linear models of setup and hold time as well as values of t_{cq} and t_{dq} will become voltage-dependent parameters, i.e.,

$$\begin{cases} t_{s,ij} = t_s(w_i, v_j) = a_1(v_j)w + a_0(v_j) \\ t_{h,ij} = t_h(w_i, v_j) = b_1(v_j)w + b_0(v_j) \\ t_{cq,j} = t_{cq}(v_j)\,, \qquad t_{dq,j} = t_{dq}(v_j) \end{cases} \tag{12}$$

Timing characteristics of SEFF are measured by HSIPCE simulations (sweeping voltage) to determine voltage dependent values and coefficients through linear regression. Fig. 6 shows SPICE simulations of setup and hold time as linear functions of transparency window size and voltage level for SEFF of Fig. 2.

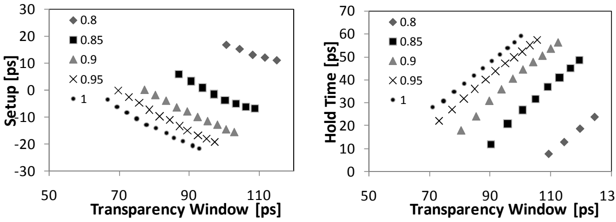

Fig. 6. Setup time (left) and hold time (right) as functions of the supply voltage level and the transparency window width

B. Soft-Pipeline Timing Constraints

Introduction of a transparency window to a flip-flop not only modifies the timing characteristics of a SEFF, but also changes the timing constraints imposed on the pipeline due to implementation of time borrowing. Inequality (1) for the setup time constraint ignores the time borrowing effect between stages. However, hold time constraint does not change in case of time borrowing; note the $t_{cq,(i-1)}$ is in fact the window independent $t_{cq,j}$ for all of the stages.

Fig. 7 illustrates setup time constraint fundamentals of a time borrowing operation among three consecutive stages, in which stage *i* uses the timing slack of stage *i*+1, and stage *i*+1 uses that of stage *i*+2. In this figure, D_i and Q_i represent the input and output of FF-sets of stage *i,* respectively. In this case, the following timing constraint sets should be met, which establishes the condition for time borrowing between stages *i* and *i*+1 [26]:

$$\begin{cases} d_i \le T_{clk} - t_{cq} - t_{s,i} & 1 \le i \le N \quad (13) \\ d_i + d_{i+1} \le 2T_{clk} - t_{cq} - t_{dq} - t_{s,i+1} & 1 \le i \le N \quad (14) \end{cases}$$

Inequality (13) is in fact the same setup time constraint as (1) for a single stage which ensures that delay of *i*-th stage is able to meet the setup time of its destination SEFF with time borrowing enabled. Inequality (14) assumes that stage *i* may borrow time from stage *i*+1, but the accumulated delay of these two stages (plus setup time and clock-to-Q of SEFF's) should not exceed two clock periods. Note that in inequality (14), in the SEFF-set *i*, data arrives within the transparency window and propagates to the output only after a delay of t_{dq}.

In general, setup time constraints corresponding to an *N*-stage soft-pipeline under voltage state *j* can be written as:

$$\begin{cases} (m+1)T_{clk} - t_{cq,j} - m \cdot t_{dq,j} - t_{s,(i+m)j} \ge \sum_{x=i}^{i+m} d_{xj} \\ \qquad\qquad\qquad 0 \le m \le N-i, 1 \le i \le N \end{cases} \quad (15)$$

Inequality set (15) covers setup time constraints applied to single stages and multiple stages involved in time borrowing. The parameter *m* denotes the depth of time borrowing in this equation. If *m*=0, the inequality represents the setup time constraint within a single pipeline stage, and larger values of *m* produce the setup timing condition on accumulative delays of multiple consecutive pipeline stages. Also in the statistical framework, setup constraint violation probability may be written as:

$$\begin{cases} q_{setup,ij,m} = P\left\{(m+1)T_{clk,j} - t_{cq,j} - mt_{dq,j} - t_{s,(i+m)j} \le \sum_{x=i}^{i+m} d_{xj}\right\} \\ \qquad\qquad\qquad 0 \le m \le N-i, 1 \le i \le N \end{cases} \quad (16)$$

$$q_{setup,ij} = 1 - \prod_{m=0}^{N-i}\left(1 - q_{setup,ij,m}\right) \quad (17)$$

As mentioned in section II.C, the effect of variability on the flip-flop timing characteristics is negligible, and the only random variables in (16) are d_{ij}'s, which are correlated [20][27]. Let ρ_{ik} denote the correlation between the maximum stage delays of stage *i* and *k*. Given the CDF of all d_{ij}'s and ρ_{ik}, we can estimate the CDF of summation of d_{ij}'s, by assuming that it follows the same form as any of d_{ij}'s, with corresponding mean and variance. The mean and variance of any summation of correlated d_{ij}'s may be calculated as:

$$\begin{aligned} \mu &= E\left(\sum d_{ij}\right) = \sum \mu_{d,ij} \\ \sigma^2 &= var\left(\sum d_{ij}\right) = \sum \sigma_{d,ij}^2 + \sum_{i \ne k} \sigma_{d,ij}\sigma_{d,kj}\rho_{ik} \end{aligned} \quad (18)$$

Note that we assume the circuits that our proposed algorithms optimizes are fully synthesized and mapped circuits and standard SSTA tools have been used to perform timing analysis on each stage of the pipeline. Such tools do account for various sources of variability specified for them and certainly consider the effect of spatial process variations and/or reconvergent fanout paths in their calculations.

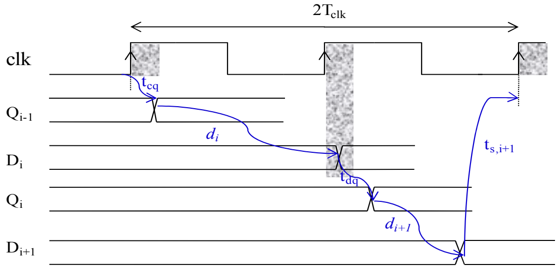

Fig. 7. Time borrowing between two stages of a soft pipeline.

C. SEFF Power Consumption Model

Power consumption of a SEFF is generally an increasing function of window size, *w*. This is due to the fact that increasing the window size is performed by resizing and/or increasing the number of inverters in delayed clock path; both methods result in an increase in the dynamic and leakage power consumption of the SEFF. Fig. 8 illustrates the total power consumption of a master-slave SEFF as a function of its window size, for a fixed clock period and two different voltage values. The discontinuities (jumps) in the curve are due to a change in the number of inverters in delay path.

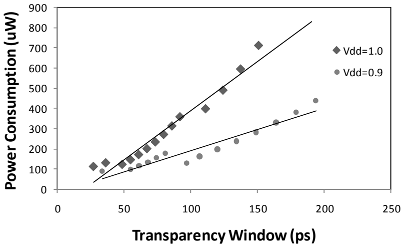

Fig. 8. Power consumption as a function of window size of SEFF.

From Fig. 8, one can conclude that power dissipation of the SEFF may be approximated as a linear function of the transparency window width, for a fixed clock period. To approximate effect of both dynamic and leakage power consumption for any window size and any clock period in the SEFF circuit, its power consumption may be calculated as:

$$P_{SEFF} = k_3(v)\frac{w}{T_{clk}} + k_2(v) \cdot w + k_1(v)\frac{1}{T_{clk}} + k_0(v) \quad (19)$$

where *v* denotes the supply voltage level, and $k_0(v)$ through $k_3(v)$ are voltage- and technology-dependent coefficients which can be determined through HSPICE circuit simulation. In equation (19), the two terms with inverse of T_{clk} correspond to dynamic power consumption while the other terms correspond to leakage power.

IV. Power-Delay Optimization in a Pipeline

Due to significance of both performance and power efficiency in pipelined systems, we chose Power-Delay product as the cost metric to optimize the design of such systems. Note that in the Power-Delay product, delay is not simply the inverse of the clock frequency, rather it also probabilistically accounts for the overhead of correcting potential setup time problems in an over-clocked pipeline. In this way, we are able to find values of our optimization variables so that the increase in setup time violation and corresponding timing overhead is compensated by the decrease in the power dissipation. Consequently, an optimum power-delay operating point for a linearly pipelined design with time borrowing and error detection/correction capability is found.

In this section, we solve the problem of power-delay optimization in a linear pipeline using SEFF. We formulate the problem for three scenarios:

(i) The stage delay is captured by worst case delay estimates,

(ii) Statistical timing analysis is used to model the stage delay, and no timing violation is allowed,

(iii) The stage delay is still computed by statistical timing models, but timing failures are allowed to exist and automatically be detected and fixed.

In scenario (i), we deal with worst case combinational circuit delays as deterministic values. The worst case delay is the maximum observed value of combinational circuit's delay, over all possible inputs combinations under any possible operating conditions (different PVT corners.) Satisfying the timing constraints of (1) and (2) for these conservative delay values results in error-free operation of the pipeline. On the other hand, in scenario (ii), we will consider the path delays as random variables and will use statistical timing equations. Under scenario (iii), we allow a few timing violations to occur and adopt an error detection mechanism to guarantee correct functionality of pipeline. This framework can aggressively scale pipeline frequency to improve delay, while the error detection and correction imposes power and delay penalties. Our solution considers the trade-off between delay reduction and penalties caused by errors.

The key motivation for using SEFF's in a pipeline circuit is that some positive slack may be available in one or more stages of the pipeline. Utilizing SEFF allows passing this slack to more timing critical stages of the pipeline to provide them with more freedom in power optimization by voltage scaling.

An Illustrative Example

As an example, consider the three stage pipelined circuit of Fig. 9 operating at a supply voltage level of V_{DD}. The per-stage maximum logic delays are shown in the figure. Let's assume the setup time, hold time, and the clock-to-Q delay of all (hard-edge) FF's are 25ps each. Assuming fixed and uniform time allocation across the three pipeline stages, from equation (1), the minimum clock period is 500ps, and no slack is available to the first stage of the pipeline. However, if FF1 is replaced with a SEFF with a transparency window of 50ps, the available slack at the second stage is passed to the first stage, providing the first stage with 50ps of borrowed time. Now since positive slacks are available in all stages of the pipeline, the circuit can be operated at higher clock frequency and/or a smaller supply voltage in order to reduce the power consumption, and possibly the power-delay metric (ideally, V_{DD} may be reduced by approximately 10%, resulting in roughly 19% power saving).

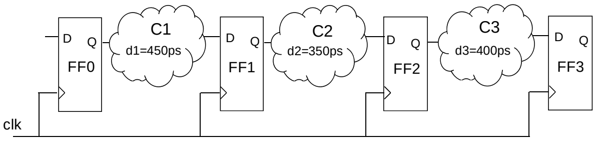

Fig. 9. Example of slack passing

Delay Elements

From equation (2), one can see that increasing the transparency window of the i^{th} soft-edge FF-set puts a more stringent constraint on the hold time condition for the i^{th} stage of the pipeline. Therefore, if needed, delay elements may be utilized in the minimum-delay path(s) to alleviate the hold time constraint violation. Insertion of a delay element with a delay magnitude of z_i would change equation (9) as follows:

$$\begin{aligned} q_{hold,ij} &= P\{\delta_{ij} < t_{h,ij} - t_{cq,(i-1)j} - z_i\} \\ &= F_{ij}^{\delta}\left(t_{h,ij} - t_{cq,(i-1)j} - z_i\right) \end{aligned} \tag{20}$$

Delay elements are indeed created by utilizing some inverters and appropriately sizing them in order to meet the desired delay lower bound while incurring minimum power loss. The power overhead of a delay element is denoted as:

$$P_{DE}(z,v) = h_2(v)\cdot z + h_1(v)\frac{z}{T_{clk}} \tag{21}$$

where z is the desired delay and $h_2(v)$ and $h_1(v)$ are voltage dependent parameters, to be determined by HSPICE simulations. Fig. 10 illustrates the linear model fitting on the measured data. Note that the delay elements are produced by means of a chain of multiple buffers; to get larger delay, more buffers are needed. This causes power dissipation increase with increased delay as shown in Fig. 10, with discontinuity points due to change in the number of buffers.

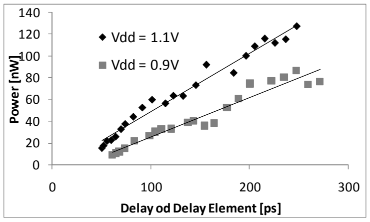

Fig. 10. Power vs. Delay in Delay Element

A. Power-Delay Optimal Soft Pipeline (OSP)

The problem of power-delay optimal soft pipeline (OSP) design is defined as that of finding optimal values of the global supply voltage level, pipeline clock period, and the transparency windows of the individual soft-edge FF-sets in the design so as to minimize the total power-delay product of an N-stage pipeline circuit subject to setup and hold time constraints. From (19), (6) and (21), total power consumption of pipeline is:

$$P_{total} = P_{Comb,j} + \sum_{i=1}^{N-1} P_{SEFF,ij} + \sum_{i=1}^{N} P_{DE,ij} \tag{22}$$

$$= P_{leak,j} + \frac{E_{dyn,j}}{T_{clk}} + \sum_{i=1}^{N-1}\left(k_{3j}\frac{w_i}{T_{clk}} + k_{2j}\cdot w_i + \frac{k_{1j}}{T_{clk}} + k_{0j}\right) + \sum_{i=1}^{N}\left(h_{2j}\cdot z_i + h_{1j}\frac{z_i}{T_{clk}}\right)$$

where all terms with subscript j correspond to their value under supply voltage v_j, i.e. k_{3j}=$k_3(v_j)$ and so on.

Delay of the pipeline (system delay) on the other hand is calculated by (10). Since no errors are allowed in the pipeline, the delay is equal to the pipeline clock period (and thus, it is the pipeline energy dissipation in this case.) Hence, the problem of power-delay optimal soft pipeline (OSP) may be formulated as:

$$\left\{\begin{array}{ll} \multicolumn{2}{l}{Minimize\ P_{total}\cdot D} \\ \multicolumn{2}{l}{= T_{clk}\left(P_{Comb,j} + \sum_{i=1}^{N-1} P_{SEFF,ij} + \sum_{i=1}^{N} P_{DE,ij}\right)} \\ \text{such that:} & \\ (m+1)T_{clk} - t_{cq,j} - m.t_{dq,j} - t_{s,(i+m)j} \ge \sum_{x=i}^{i+m} d_{xj} & \\ \qquad\qquad 0 \le m \le N-i, 1 \le i \le N & \\ t_{h,ij} - t_{cq,j} - z_{ij} \le \delta_{ij} & 1 \le i \le N \\ w_{min} \le w_i \le w_{max} & 1 \le i \le N-1 \\ 1 \le j \le S \quad (v \in \{V_1, \dots, V_S\}) & \end{array}\right. \tag{23}$$

The first and second sets of inequalities in (23) are respectively the setup and hold time constraints in the pipeline stages, the third set of inequality constraints imposes an upper bound and a lower bound on the transparency window of the flip-flop imposed by the library or design rules (typically, $w_{min} \ge 0$ and $w_{max} < ½T_{clk}$). Finally, the last statement in (23) enforces the supply voltage of the pipeline to be from the set of available voltages $\{V_1, \dots, V_S\}$, where $V_0=V_1> \dots > V_S$ (V_0 is the nominal supply voltage). Note that problem formulation (23) has $2N+1$ optimization variables corresponding to N-1 transparency window sizes, w_i, for the N-1 soft-edge FF-sets in the linear pipeline, N delay element values, z_i, for the N stages of the pipeline, one supply voltage variable setting, v, and one clock period variable, T_{clk}.

Referring back to Fig. 1, for the sake of consistency with the input and output environments and to avoid imposing constraints on the sender or receiver of data for the linear pipeline circuit in question, we impose the *boundary condition* that the first and last FF-sets in the pipeline are composed of hard-edge FF's whereas intervening FF-sets may be SEFF's.

To solve the problem stated in (23) efficiently, we enumerate all possible values for v, and for each fixed v we solve a quadratic program (i.e., we minimize a quadratic cost function subject to linear inequality constraints), which can be solved optimally in polynomial time. In the fixed supply voltage OSP problem formulation, $P_{leak,i}$ term drops out of the cost function, the last constraint disappears, and all others become only dependent on w_i, z_i and T_{clk} variables. We refer to this version of the problem as OSP-FV, OSP with fixed voltage:

$$\left\{\begin{array}{ll} \multicolumn{2}{l}{Minimize\ \Big(T_{clk}P_{leak,j} + E_{dyn,j} + \sum_{i=1}^{N}\left(h_{2j}T_{clk}\cdot z_i + h_{1j}\cdot z_i\right)} \\ \multicolumn{2}{l}{\qquad + \sum_{i=1}^{N-1}\left(k_{3j}\cdot w_i + k_{2j}\cdot w_i\cdot T_{clk} + k_{1j} + k_{0j}\cdot T_{clk}\right)\Big)} \\ \text{such that:} & \\ (m+1)T_{clk} - t_{cq,j} - m.t_{dq,j} - t_{s,(i+m)j} \ge \sum_{x=i}^{i+m} d_{xj} & \\ \qquad\qquad 0 \le m \le N-i, 1 \le i \le N & \\ t_{h,ij} - t_{cq,j} - z_{ij} \le \delta_{ij} & 1 \le i \le N \\ w_{min} \le w_i \le w_{max} & 1 \le i \le N-1 \end{array}\right. \tag{24}$$

Note that in OSP-FV problem, all the voltage-dependent coefficients, i.e., k_3-k_0 in P_{SEFF} and h_2, h_1 in P_{DE} equation, as well as the coefficients in $t_{s,i}$, $t_{h,i}$, t_{cq}, and t_{dq} are recalculated for the voltage under test. Also, E_{dyn}, P_{leak}, d_i and δ_i are given window-size-independent inputs (generated by profiling or given by (4)-(6)) for each voltage.

Lemma 1: In the optimal solution of OSP-FV design problem, the transparency window of the i^{th} SEFF-set is equal to the time borrowed by combinational logic in the i^{th} stage.

Proof: According to the discussion in section II.A and Fig. 8, the power consumption of a SEFF is a monotonically increasing function of the transparency window size while its setup time is a decreasing function of the same. Now, from the OSP-FV problem formulation of equation (23), a minimum decrease in the setup time of the i^{th} SEFF-set $t_{s,i}$ which meets the long-path constraint in the i^{th} stage of the pipeline, will produce the minimum increase in the power dissipation of the i^{th} SEFF-set $P_{SEFF,i}$. Therefore, the optimal solution is achieved by utilizing the smallest possible window sizes which prevent setup time violation. ■

Lemma 2: In the optimal solution of OSP-FV design problem, the delay element inserted in the i^{th} stage of the pipeline is equal to the minimum extra time needed to meet the hold time constraint at the i^{th} soft-edge FF-set.

Proof: According to the discussion in section III, the power consumption of a delay element is a monotonically increasing function of the target delay value while the hold time of a SEFF is an increasing function of the same. Now, from the second inequality (hold time condition) in the OSP-FV problem formulation of (23), a minimum delay value z_i added to the i^{th} stage of the linear pipeline which meets the short-path constraint for that stage, will produce the minimum increase in the power of the combinational logic in the i^{th} P_{DE} (z_i, v). Hence, the optimal solution is achieved by utilizing smallest possible delay elements which prevent hold time violations. ■

Theorem 1: The optimal solution to OSP design problem is obtained by solving the OSP-FV design problem S times for each distinct voltage level and selecting the voltage level v^* and the corresponding w_i^*, z_i^* and T^*_{clk} values that minimize the total power dissipation for v^*.

Proof: This follows from the observation that solution of the OSP-FV problem produces w_i's, z_i's and $T^*_{clk,i}$ for each possible v and we enumerate over all v's to get the global optimum solution in an exhaustive manner. ■

Note that although SEFF's are custom-designed and their transparency windows are set only once at design time, implementing the optimal transparency window of SEFF's may not be practical. Because, for instance, device (transistor) size and hence delay of window generation circuitry of SEFF cannot be any arbitrary value. Therefore, we round off the optimal sizing solution to its closest larger-sized match that is implementable. Since this realized SEFF will have minimally larger transparency window size, it will not violate any setup time constraints, while increasing the power consumption as minimum as possible. However, if the hold time constraints are violated by this adjustment, then adding delay elements may be used in violating short paths to solve the problem, with negligible impact on power-delay metric of pipeline.

The pseudo-code presented in Fig. 11 summarizes the steps in OSP algorithm.

```
Determine $P_{leak,j}$, $E_{dyn,j,}$ $d_{ij}$ and $\delta_{ij}$ and voltage-dependent coefficients $a_{1j}$, $a_{0j}$, $b_{1j}$, $b_{0j}$, $t_{cq,j}$, $t_{dq,j}$, $k_{3j}$, $k_{2j}$, $k_{1j}$, $k_{0j}$, $h_{2j}$, $h_{1j}$ for all voltages
for ($v = V_j$, $j$++, $Vj \in \{V_1, \dots, V_S\}$) {
   $PD_j$ = Solution to OSP-FV($v$)
}
$v$*= ArgMin $PD_j$ for $1 \leq j \leq S$
Set $w_i$*'s and $z_i$*'s as the solution of OSP-FV($v$*)
Round-off $w_i$*'s and $z_i$* to closest upper feasible match
```

Fig. 11. Pseudo-code of OSP algorithm

B. Statistical Power-Delay Optimal Soft Pipeline (SOSP)

In section A, we followed the conventional static timing analysis framework in which deterministic values of worst case circuit delays are used to specify the circuit timing. However, due to process and environmental variations in integrated circuits, the path delays may vary from one die to next and from one operating condition to the other. Consequently, the path delays may be modeled by random variables [15]. Therefore, we will replace the deterministic timing constraints with the probability of timing violations in a pipeline as given by equations (8) and (9).

The problem of statistical power-delay optimal soft pipeline (SOSP) design is defined as that of finding optimal values of the operating voltage and frequency and the transparency window sizes of the individual soft-edge FF-sets in the pipeline so as to minimize the total power-delay metric in a soft pipeline circuit with N pipeline stages and S voltage states. As mentioned earlier, SEFF enables opportunistic time borrowing across adjacent stages of the pipeline in order to provide timing-critical stages with more time to complete their computations and thereby, reduces the probability of timing errors at a particular frequency.

Let $q_{setup,ij}$ and $q_{hold,ij}$ denote probabilities of setup time and hold time *violations* at stage i of the pipeline under supply voltage v_j, as given in equations (17) and (20). Assuming that the probability of encountering an error in a specific combinational circuit stage is independent of other stages, the probability of having a timing error in the entire pipeline, $q_{pipeline,j}$ is calculated by (25). This probability should be limited to an extremely small value, ε, (e.g. 10e-12) to make failure of the pipeline virtually impossible.

$$q_{pipeline,j} = 1 - \prod_{i=1}^{N} \Big(\big(1 - q_{setup,ij}\big)\big(1 - q_{hold,ij}\big) \Big) \qquad (25)$$

Now then, SOSP can be formulated as (26). It minimizes the power-delay product of the pipeline, subject to an upper-bound on the error probability, denoted by ε.

$$\begin{cases} Minimize\ (T_{clk} \left(P_{Comb,j} + \sum_{i=1}^{N-1} P_{SEFF,ij} + \sum_{i=1}^{N} P_{DE,ij} \right)) \\ \text{such that:} \\ \quad q_{pipeline,j} \leq \varepsilon \\ \quad w_{min} \leq w_i \leq w_{max} \qquad\qquad 1 \leq i \leq N-1 \\ \quad 1 \leq j \leq S \ (v \ \in \{V_1, \dots, V_S\}) \end{cases} \qquad (26)$$

Note that even though the circuit delay is modeled as a random variable due to process variations, the power consumption is not. It is known that the effect of V_t or L_{eff} variation on dynamic power consumption is negligible [28]. On the other hand, since we do not make any modifications to the combinational circuit part (e.g. do not perform gate sizing or logic re-synthesis) leakage power of logic gates is not affected by our optimization. So we set these leakage values to any fixed amount; we consider the maximum values (worst case) of leakage power consumption of combinational circuit.

Next we approximate $q_{pipeline,j}$ which is given by (25) with a convex function to simplify the problem statements. Result of expanding equation (25) becomes a summation of all $q_{setup,ij}$ and $q_{hold,ij}$'s and their mutual product of second and higher order. Since all error probabilities, i.e. $q_{setup,ij}$ and $q_{hold,ij}$'s, are relatively small values (e.g. in the order of 1e-3 or 1e-4) the product of any two (or more) of such functions are negligible compared to the summation of first order terms and could be ignored. The resulting equation for $q_{pipeline,j}$ would be a simple summation of $q_{setup,ij}$ and $q_{hold,ij}$'s:

$$q_{pipeline,j} \cong \sum_{i=1}^{N} \big(q_{setup,ij} + q_{hold,ij} \big) \qquad (27)$$

Furthermore, to conveniently formulate the problems as quadratic programs, we approximate $q_{setup,ij}$ and $q_{hold,ij}$ as first order polynomial functions of SEFF characteristics and T_{clk}:

$$q_{setup,ij} \cong qsT_j \cdot T_{clk} + \sum_{m=0}^{N-i} qsw_{mj} \cdot w_{i+m} + qs_j(i) \qquad (28)$$

$$q_{hold,ij} \cong qhd_j \cdot z_i + qhw_j \cdot w_i + qh_j(i) \qquad (29)$$

where qsT_j, qsw_j, qhd_j, qhw_j are coefficients (of T_{clk}, window size, delay element and window size in $q_{setup,ij}$ and $q_{hold,ij}$ respectively) corresponding to voltage setting j, and $qs_j(i)$ and $qh_j(i)$ are voltage and stage-delay dependent fixed terms. As a preprocessing step, we linearize the CDF of any max (min) stage delay around its μ+3σ (μ-3σ) point, i.e. for any x within a boundary around such point, F_{ij} (x) ≈ $\alpha_{ij}.x+\beta_{ij}$. Hence equations (16) and (20) can be approximated as follows, and all coefficients, q^{*}_{j}, be determined accordingly:

$$\begin{aligned} q_{setup,ij} &= F_{ij}\big(T_{clk,j} - t_{s,ij} - t_{cq,(i-1)j}\big) \\ &\cong \alpha_{ij} \cdot T_{clk,j} - \alpha_{ij} \cdot a_{1j} \cdot w_i + \beta_{ij} - \alpha_{ij} a_0 - \alpha_{ij} t_{cq,j} \end{aligned} \tag{30}$$

$$\begin{aligned} q_{setup,ijm} \cong\; & \alpha_{ij}(m+1)T_{clk,j} - \alpha_{ij} a_{1j} w_i + \beta_{ij} \\ & - \alpha_{ij} a_0 - \alpha_{ij} t_{cq,j} - \alpha_{ij} \cdot m \cdot t_{dq,j} \end{aligned} \tag{31}$$

$$q_{hold,ij} \cong \alpha_{ij} b_{j1} w_i - \alpha_{ij} z_i + \beta_{ij} - \alpha_{ij} t_{cq,j} - \alpha_{ij} b_{j0} \tag{32}$$

Again, using Theorem 1, we conclude similar algorithm to solve the SOSP problem presented in (26), to enumerate all possible values of *v*, and we solve a quadratic program for each *v*. We refer to this version as SOSP-FV, SOSP with fixed voltage, in which, variables are only transparency window sizes, pipeline clock period, and delay elements.

$$\begin{cases} Minimize\ (T_{clk,j}\left(P_{Comb,j} + \sum_{i=1}^{N-1} P_{SEFF,ij} + \sum_{i=1}^{N} P_{DE,ij}\right)) \\ \text{such that:} \\ \quad q_{pipeline,j} \le \epsilon \\ \quad w_{min} \le w_i \le w_{max} \qquad\qquad 1 \le i \le N-1 \end{cases} \tag{33}$$

Theorem 2: The SOSP-FV problem is a convex problem, and the optimal solution to it (if the feasible region is not empty), minimizes the objective function.

Proof: In general, the product or ratio of two convex functions are not convex [29], and hence we used the additive approximation in (27) for $q_{pipeline,j}$ instead of (25). Therefore, the objective function of SOSP-FV problem is a quadratic function of its variables (the transparency window sizes, delay elements, and clock period) while the constraints are linear. ■

Now then, the convex optimization problem of SOSP-FV is efficiently solvable by using any commercial mathematical optimization tools. Of course, when a solution is obtained we must verify the condition for approximations, but this has always been the case in our experimental results.

C. Error-Tolerant Statistical Power-Delay Optimal Soft Pipeline (ESOSP)

The problem formulations presented in section A and B conservatively calculate the pipeline operation clock frequency to avoid timing violations causing pipeline errors. However, only for some specific combination of inputs is the critical path sensitized, and therefore, the aforesaid formulations result in a pessimistic clock period. Instead, error-tolerant statistical power-delay optimal soft pipeline (ESOSP) algorithm aggressively scales down the pipeline clock period to improve performance, while implementing a mechanism to capture and fix any possible timing violations due to this over-clocking. The proposed algorithm explores the trade-off between delay improvement and increase in power as well as the power and delay penalties caused by timing errors.

An error handling mechanism is incorporated in our design to guarantee correct functionality under all conditions. Error detection and correction can be fully implemented in the flip flop circuit, as described in Appendix (See VII.B). In another method, error detection is built in the flip flop circuit while error correction mechanism is supported by pipeline architecture itself (through data/instruction flushing and replaying the same data/instructions this time under a transitory operating condition which is more conservative, e.g. lower frequency) (See VII.A). If the error rate is relatively low, area and power overhead of FF design with built-in error detection circuit will be negligible, compared to FF with built-in error correction circuit.

For simplicity, we focus on the fixed voltage version of ESOSP problem, and generate the solution to original problem of ESOSP by combining the solutions to multiple instances of ESOSP-FV based on Theorem 1. Let P_j denote the average total power consumption of pipeline under supply voltage v_j, and $P_{p,j}$ denote the average power overhead when encountering an error at same voltage v_j (this overhead includes the power consumed for computing erroneous data as well as flushing it and its following data units). Also, let γ denote the average delay (in clock cycles) corresponding to error detection and correction, such as flushing. Given an error probability of q_j under some voltage v_j, the expected value of power-delay objective function may be written as:

$$\Phi = \big(1 - q_j\big)P_j T_{clk,j} + q_j\big(P_j + P_{p,j}\big)\gamma T_{clk,j} \tag{34}$$

In fact, error probability, q_j, is a decreasing function of T_{clk}. This is the source of trade-off between power-delay metric of error-free and erroneous operation of pipeline. Decreasing T_{clk} reduces the power-delay for error-free operation (the first term in (34)), but increases q_j and as a result, the error correction overhead (the second term in (34).

Implementation of time borrowing across adjacent stages of the pipeline effectively reduces the probability of error due to timing errors, q_j, and avoids the subsequent power and delay penalties of error correction step for any T_{clk}. Increasing transparency window size, however, increases total power consumption. Fortunately, gained power saving tends to more than compensate for it.

Remember P_j in equation (34) denotes the sum of power consumptions of the combinational logic blocks (that also includes delay elements and hard edge FF's) and SEFF's, without encountering an error. P_j is a function of voltage, SEFF's window sizes and delay elements, and equation (22) can be rearranged as,

$$P_j = A_j + \frac{B_j}{T_{clk}} + \sum_{i=1}^{N-1}\left(k_{3j}\frac{w_i}{T_{clk}} + k_{2j}w_i\right) + \sum_{i=1}^{N}\left(h_{2j}z_i + h_{1j}\frac{z_i}{T_{clk}}\right) \tag{35}$$

with A_j and B_j representing all the terms corresponding to constant values and coefficients of $1/T_{clk}$, respectively. For simplicity, let's assume the power overhead of error correction is β times that of only producing a data value without encountering an error, i.e. $P_{p,j} = \beta . P_j$ (Value of the β parameter is obtained from micro-architectural and circuit simulations).

The ESOSP-FV problem is defined as finding optimum w_i*'s*, z_i's and T_{clk} in the following formulation:

$$\begin{cases} Minimize\ (1-q_j)P_jT_{clk} + q_jP_j(1+\beta)\gamma T_{clk} \\ \text{such that:} \\ w_{min} \le w_i \le w_{max} \qquad 1 \le i \le N-1 \\ T_{min} \le T_{clk} \le T_{max} \\ q_j \cong q_{pipeline,j} = \sum_{i=1}^{N} (q_{setup,ij} + q_{hold,ij}) \end{cases} \tag{36}$$

Note that the objective function of (36) is a third order polynomial with proposed linear approximations for q_j, which can be solved using general convex optimization tools [30][31]. In section E, we introduce another constraint which bounds the undetected error probability, and should be added to (36).

D. ESOSP for Profiled Operation

Dynamic Voltage and Frequency Scaling (DVFS) is widely used to minimize the power consumption in microprocessors. The entire pipeline should meet timing constraints in every *circuit state* (also known as DVFS setting). A circuit state is uniquely identified by a supply voltage level which is simultaneously applied to all stages of the pipeline. Changing the voltage to bring about a new circuit state affects the power consumption of pipeline as well as combinational path delay and time budget of combinational circuit.

Consider a scenario whereby based on the system-level power management policy, it has been determined that the circuit will operate in each of its circuit states according to some probability distribution. We present another formulation to minimize the average expected power-delay product over all DVFS circuit states. More precisely, given the probability values for being in various circuit states during the active mode of pipeline operation, we attempt to minimize the power-delay product averaged over all such states.

Let π_j denote the probability of being in circuit state s_j (characterized for a given voltage level v_j). Then, the weighted cost function is defined as:

$$\tilde{\Phi} = \sum_{j=1}^{S} \pi_j \Phi(s_j) \tag{37}$$

The ESOSP-Profiled problem is thus formulated as:

$$\begin{cases} Minimize\ \sum_{j=1}^{S} \pi_j(1-q_j)P_jT_{clk,j} + q_jP_j(1+\beta)\gamma T_{clk,j} \\ \text{such that:} \\ w_{min} \le w_i \le w_{max} \qquad 1 \le i \le N-1 \\ T_{min} \le T_{clk,j} \le T_{max} \qquad 1 \le j \le S \\ q_j \cong \sum_{i=1}^{N} (q_{setup,ij} + q_{hold,ij}) \end{cases} \tag{38}$$

Now then, ESOSP tries to minimize the power-delay product of the pipeline, and find the optimum set of frequencies, $T_{clk,j}$ (j=1, …, S) under each circuit state, and a set of optimum window sizes, w_i (i=1, ..., N-1), for each FF-set, and the optimum delay elements of each stage, z_i (i=1, ..., N). Hence, for S circuit states and N pipeline stages, there are S+2N-1 optimization variables; in each circuit state, we apply the calculated optimum frequency to all stages of the pipeline. Notice optimum window size for each soft-edge FF-set (recall that the first and last FF-sets use always hard-edge FF's), as well as delay elements are design time decisions and these size assignments are independent of circuit state.

E. Bounding the Probability of Undetected Errors

An undetected error in the pipeline can occur due to a very long path that violates internal timing of SEFF. Normally, in a SEFF with built-in error handling mechanisms, the input data is re-sampled at a later time by utilizing a *phase-shifted* global clock signal, PS (see section VII.A). The undetected error probability is the probability of data arriving after T_{clk}+PS which is calculated by (39) – notice that this equation is similar to (8) except that we have replaced T_{clk} with T_{clk}+PS because an undetected error occurs only when the arrival time of the correct data is later than the triggering edges of the PS Clock in the current cycle. Consequently, given the CDF for max stage delays, the probability of an undetected error in pipeline stage i and supply voltage v_j is:

$$\varepsilon_{undetected,ij} = 1 - F_{ij}^{d}(T_{clk} + PS - t_{s,ij} - t_{cq,j}) \tag{39}$$

The overall rate of undetected errors for all voltage levels is:

$$\varepsilon_{undetected} = 1 - \prod_{j=1}^{S}(1 - \prod_{i=1}^{N}(1 - \varepsilon_{undetected,ij})) \tag{40}$$

To impose an upper bound on undetected-error probability, we include PS as a new variable of optimization to problem formulations with error detection technique enabled, along with the following constraint where $\varepsilon_{UpperBound}$ is user provided (typically in the same order as ε in (33), e.g. 1e-6 to 1e-10).

$$\varepsilon_{undetected} < \varepsilon_{UpperBound} \tag{41}$$

V. Experimental Results

A. Simulation Setup

To extract the parameters used in the optimization problem, we performed transistor-level simulations on soft-edge flip-flops by using HSPICE [32]. We used 90nm technology model [33] with nominal supply voltage of 1.2V. Simulations have been conducted at die temperature of 85°C. In all experiments, the set of available voltage levels is {0.8V, 0.9V, 1V, 1.1V, 1.2V}. We synthesized a number of linear pipelines, including some modified ISCAS89s benchmarks (denoted by TBx) and datapath and processor circuits to construct a set of benchmarks. SIS [34] and Synopsys Design Compiler packages were used for synthesizing benchmarks. We then performed timing simulations and used Synopsys PrimeTime to extract the static value of longest and shortest path delays of each pipeline stage under each voltage setting.

Next, we considered max and min stage delays of a pipeline to have probability density functions. For this, we run Monte Carlo simulations on fully synthesized and mapped logic circuits to generate the max/min stage delay distributions by monitoring the top 100 critical paths of each stage (identified using Synopsys PrimeTime timing analysis tool) affected by

Table 1. Power-Delay improvement by OSP

Test-bench	Stage delays at nominal voltage (max, min) [ps]					Baseline	Base+VS		OSP		%PDP Saving	
						T_{clk}*[ps]	V_{dd}*[V]	T_{clk}*	V_{dd}*	T_{clk}*	Base	Base+VS
tb1	(353,140)	(214,112)	(254,107)	(217,110)		458.5	0.8	707.7	1.0	471.5	38.2	12.4
tb2	(646,192)	(670,232)	(550,158)	(648,192)	(583,189)	786.1	0.8	1206.9	0.9	1028.5	42.4	13.9
tb3	(334,108)	(280,98)	(219,80)			397.3	0.9	534.6	1.0	467.1	44.4	17.8
tb4	(250,96)	(254,96)	(251,95)	(253,96)		329.4	1.0	380.8	1.0	384.9	14.9	-3.0
TROY proc. [ns]	(1270,320)	(2188,429)	(4759,150)	(4788,315)	(1279,230)	4893	0.9	6986.7	0.9	6408.5	26.7	8.7
Openrisc1200[ns]	(2172,280)	(2514,359)	(7738,351)	(6862,436)	(1739,487)	7843	1.0	9487.9	0.9	12288.5	28.2	11.6
Viterbi decoder	(817,175)	(858,164)	(926,215)	(773,183)		1055.3	0.8	1608.5	0.8	1584.1	33.6	12.1

variations. We assumed a σ/μ ratio of 5% for sources of variation, i.e. threshold voltage and channel length, similar to [21], and applied it to circuit simulations. We also assumed ρ=0.5 for correlation of stage delays. Then we use the linear approximation of (32)-(34) for any stage delay distribution around its μ+3σ, (or μ-3σ for min stage delay).

Finally, we formulate different algorithms given all the coefficients and parameters needed. To solve the mathematical problems developed in this paper, MATLAB [30] and TOMLAB toolbox [31] have been used. The algorithms calculate the optimal values of the operating supply voltage and frequency and the transparency window sizes of the individual soft-edge FF-sets in the design that minimized the total power-delay in the soft pipeline circuit.

B. Linear approximation of general stage delay CDF

Given the delay distribution of all stages of pipeline, we apply the linear approximation of (32)-(34) where the error rate is below %5. Fig. 12 illustrates the linear and piecewise linear estimates of sample CDF. The overall mean square relative error of the linear model was 1e-4 and that of piecewise linear approximation was 4.5e-6. In our simulations, we used piecewise linear approximation with two regions intersecting at 99 percentile of CDF; T_{clk} determines the region of estimation for each stage. For estimating multistage delays, we use the average of coefficients of linear models of involved stages delays. For all testbenches, the error of this linear approximation (single stage and multistage) remained below 2e-4 for linear model and under 1e-5 for piecewise linear model, which is acceptable, and does not have a high impact on the results of our solutions.

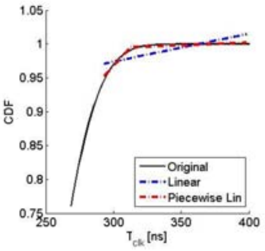

Fig. 12. Accuracy of linearly approximating stage delay CDF

C. OSP Simulation Results

In order to evaluate the performance of the proposed OSP algorithm, we assumed two conventional FF based approaches as the baselines for comparison: *Baseline* implements a conventional pipeline (which contains only conventional hard-edge FFs) and always runs at nominal voltage of 1.2V. The second method is *Base+VS* which adds the support for voltage scaling to *Baseline*. Both baselines were operating at the minimum clock cycle time for the pipeline circuit. This clock frequency was calculated for each of the test linear pipeline circuits listed in Table 1 using standard timing equations of (1) and (2) (for regular FF's) and next the power dissipation of pipeline was subsequently computed. Next, OSP was run on each circuit, exploiting time borrowing across different stages, and thus, power saving. Percentage improvement of Power-Delay product by OSP with respect to *Baseline* and *Base+VS* are reported in on these benchmarks are provided in Table 1. The first entry in this table is the name of benchmark. Specifications of benchmark, i.e., the max and min delays of each pipeline stage at nominal voltage are reported in the second through sixth columns of table. The next five columns report the optimum supply voltage (V*) and clock period (T^*_{clk}) for *Baseline* (runs at nominal voltage), *Base+VS*, and *OSP*. The last two columns show the percentage of reduction in power-delay achieved by OSP (compared to *Baseline* and *Base+VS* algorithms) which are also depicted in Fig. 13.

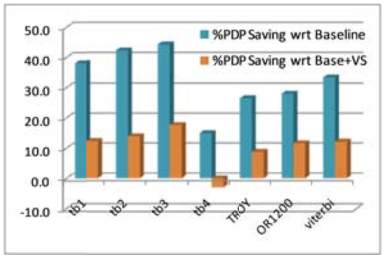

Fig. 13. Power-Delay reduction by OSP

As it is illustrated in Fig. 13, average power-delay saving of 32% and 10% is achieved by OSP compared to *Baseline* (by applying voltage scaling and time borrowing) and *Base+VS* (by only time borrowing), respectively. In case of *tb4*, the saving is negative compared to *Base+VS*, since it has the same logic circuit duplicated in each pipeline stage (balanced stages). As expected, there is no room for time borrowing in it; hence the power overhead of added circuitry causes PDP loss. Note that by balanced, we refer to (nearly) equal stage delays.

An interesting observation in the results of Table 1 is that the optimum clock period calculated by OSP or *Base+VS* is much larger than the one used by *Baseline.* This is because the objective of these two algorithms is the Power-Delay Product (PDP), and in many cases, the PDP is reduced when the supply voltage is reduced, and subsequently, T_{clk} is increased. However, if the operating frequency of circuit is the important design criterion, a minimum frequency limit, f_{min}, may be imposed by adding a linear constraint in the form of $T_{clk}<1/f_{\min}$ to the OSP problem formulations (and the other ones.) For instance, we enforced f_{min} to be higher than 85% of the *Baseline* frequency, for *tb2* and *tb4*. In case of tb4, the result did not change since the result is already in the range. However, in case of *tb2*, the PDP saving of OSP (compared to *Baseline*) reduced to about 38% while its optimum operating voltage and clock period were found to be 1V and 914ps, respectively. Here, by limiting the minimum frequency of circuit, the benefit of voltage scaling is reduced, but time borrowing is still useful in minimizing the clock cycle time.

To provide more insight into the results, we studied how using SEFF's is done in a soft pipeline by solving OSP-FV. In this set of experiments, the supply voltage of each pipeline was set at the nominal value and OSP-FV has been invoked to find the minimum values of T_{clk}. Table 2 shows the optimum clock period of *Base+VS* and OSP along with the SEFF window sizes for each test circuit under nominal voltage. For example, in the optimum soft pipeline of first benchmark, tb1, the window sizes are such that the first stage borrows time from its next stage, while others do not. Note that in soft pipeline of TROY and OR1200, some window sizes are set to the maximum allowed size (300ps in this case).

Table 2. OSP-FV's result – T_{clk} and window sizes

Test Bench	T_{base} [ps]	T_{clk}* [ps]	W* [ps]				%PDP Saving
tb1	458.5	393.2	77	0	0		20.5
tb2	786.1	749.8	14.1	13.7	0	21	5.9
tb3	397.3	394	40.5	55			9.4
tb4	314.9	387.5	0	0	0		-2.7
TROY	5057.9	4774	0	0	300	300	5.8
OR1200	8215.6	7781	0	0	300	0	5.2
Viterbi	1055.3	952.9	0	0	124.8		9.4

D. SOSP Simulation Results

Next, we considered randomness and variability of longest and shortest delays of pipeline stages (calculated as described in section A. We then set up SOSP, as the quadratic program presented in (26) with the mentioned linear approximation for $q_{i,pipeline}$, and solved it using TOMLAB optimization tool. It calculated the optimal values of the operating supply voltage and frequency and the transparency window sizes of the individual soft-edge FF-sets in the design that minimized the total power-delay in the soft pipeline circuit. By setting ε equal to inverse of total number of critical paths, we avoid violation of timing constraints.

For purpose of performance comparison, we used two baseline methods similar to the case of OSP, i.e. *Baseline* is limited to operation in nominal voltage while *Base+VS* can also set the supply voltage. The baselines determined the maximum clock frequency of the circuits based on a statistical analysis similar to SOSP, except for utilizing hard-edge FF's in the pipeline circuit. Table 3 reports the simulation results of applying SOSP to the benchmarks of Table 1 (with statistical specifications). The first entry of Table 3 denotes the test circuit; the second entry shows the maximum frequency determined by *Baseline* under nominal supply voltages. The third and fourth columns show the optimum operating voltages and frequencies obtained by *Baseline*, and the next two columns are the optimum ones calculated by SOSP, and finally the last two columns show the percentage of power-delay improvement compared to the two baselines.

Table 3. SOSP performance in Power- Delay reduction

Test-bench	Base T*[ps]	Base+VS		SOSP		%SOSP PDP Saving	
		V_{dd}	T* [ps]	V_{dd}	T* [ps]	Base	Base+VS
tb1	441.7	0.8	675.5	0.8	625.3	46.7	20.0
tb2	774.4	0.8	1193.3	0.9	1012.6	40.3	10.9
tb3	402.0	0.9	411.3	0.8	644.5	52.8	22.6
tb4	371.8	0.8	575.2	0.8	587.2	28.5	-6.2
TROY	4702	1.0	5612.9	1.1	5231.9	24.3	8.3
OR1200	7792	0.9	10197	1.0	9155.0	31.8	6.8
Viterbi	1022.6	1.1	1086.4	1.1	1012.7	22.3	12.8

E. ESOSP Simulation Results

Next we measured the error penalties of error detection and correction in a pipeline by micro-architectural simulations. Then we set up and solved ESOSP problem as formulated in (38), and next compared to *Baseline* described in section D, which calculates the optimum frequency of a conventional pipeline (composed of hard-edge FF's) under nominal voltage. Since ESOSP benefits from voltage scaling, time borrowing (denoted by TB) and error tolerance, we studied the portion of total expected power-delay saving due to each of these techniques in the statistical framework. Table 4 summarizes percentage of power-delay improvements of three techniques compared to the *Baseline* algorithm described in section D. The first one is *Base+VS* algorithm, that implements only voltage scaling (denoted by VS). The second algorithm is our proposed SOSP which combines voltage scaling and time borrowing (denoted by VS+TB). The third algorithm is ESOSP that adds error tolerance to SOSP. Table 4 gives the optimum voltage and clock periods for the testbenches as well as the optimum overall error rate of pipeline, q_{total}. Fig. 14 illustrates the share of each technique in the overall power-delay improvement with respect to *Baseline*.

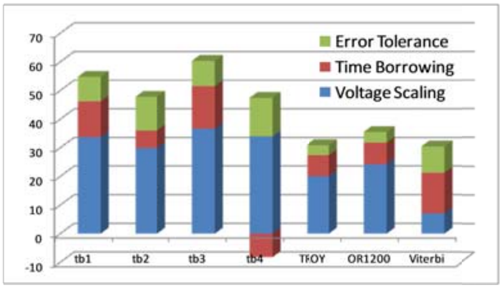

Fig. 14. Power-Delay reduction by OSP

Table 4 also reports the details of optimum operating point of the soft pipeline along with the total error rate of pipeline. Form this table, it can be referred that if the power and delay penalties of error correction are high for a circuit (for example if the stage that usually causes the error is placed in later

Table 4. ESOSP performance and comparison to baseline

Test Bench	%PDP Reduction vs. Base			ESOSP		
	VS	VS+TB	ESOSP	V_{dd}*	T*[ns]	q_{total}
tb1	33.7	46.2	54.8	0.8	533.8	2.11
tb2	30.0	36.0	47.8	0.9	852.9	1.71
tb3	36.7	51.5	60.3	0.8	520.7	1.35
tb4	33.9	25.8	39.2	0.8	493.6	1.86
TROY	20.1	27.4	30.9	1.1	4658.3	1.05
OR1200	24.2	31.8	35.5	1.0	8461.9	0.95
Viterbi	7.1	21.2	30.5	1.1	844.3	2.20

stages of pipeline, i.e. distant from input) then the optimum error rate tends to be small for such circuits, as expected.

Finally, we compared our ESOSP algorithm to an advanced baseline, *Base+CS*, which adopts the useful clock skew technique on top of *Baseline*. In this method, the pipeline stages are made balanced (by up to five FO4 inverter delays) by means of adjusting skew of clock for each individual stage. In contrast, ESOSP reduces the imbalance of pipeline by means of time borrowing. The results of this comparison show an average PDP saving of 38.2% for ESOSP for all testbenches. Compared to the 42.7% of average PDP saving of ESOSP with respect to *Baseline*, one can conclude that the share of PDP saving that was due to time borrowing reduces about 4.5%. The reason is that these two methods have almost the same effect on balancing the stage delays, and hence, clock period reduction gained by using SEFFs with respect to *Base+CS* is lower. However, using SEFF's enables dynamic (variable) time borrowing while the clock skew is a static (fixed) method for path delay balancing across different pipeline stages.

As far as the overhead of our proposed techniques (including OSP, SOSP, and ESOSP) is concerned, the area overhead of a soft pipeline is very small compared to normal pipeline. Because the circuit structure of the SEFF's is different from that of conventional FF's only in that SEFF's use an additional delay element (e.g., chain of inverters). The area overhead of this delay element is small compared to the area of the original FF. In addition, compared to the size of rest of circuit, the area overhead of added internal circuitry of SEFF's is miniscule. Finally, as far as the runtimes of our proposed algorithms are concerned, for all benchmarks, it takes less than two seconds on a 2.4GHz Xeon Pentium-4 PC (with 2GB of memory) to run any of these algorithms in MATLAB/TOMLAB toolbox.

VI. Related Work

Soft-Edge Flip Flops – Soft-edge flip-flops have been used for minimizing the effect of clock skew on static and dynamic circuits [6, 7]. Recently, authors of [9] proposed an interesting approach to utilize soft-edge flip-flops in sequential circuits in order to minimize the effect of process variation on yield. They formulated the problem of statistically aware SEFF assignment which maximizes the gain in timing yield as an integer linear program (ILP) and proposed a heuristic algorithm to solve the problem.

In [35], SEFF is utilized in the heart of proposed low-overhead solution to tackle the delay increase caused by Negative Bias Temperature Instability (NBTI), as the most critical reliability issue in sub-90nm technology nodes. SEFF has also application in reducing combinational circuit's Soft Error Rate (SER) [36] by leveraging the effect of temporal masking caused by introduction of transparency window to SEFF circuit design. It is more delay and power efficient compared to circuit redundancy based techniques [36].

Time borrowing – Authors of [37] proposed an architectural framework, called ReCycle, which adopts clock skew based time borrowing to compensate for process variation in a pipeline latching elements. It solves a linear program to determine optimum clock skews of pipeline stages that improve maximum attainable frequency. It enables the pipeline to tolerate process variation, after fabrication.

In a recent work, [38], authors have optimized pipeline clock frequency by means of replacing the flip-flops with pulsed latches to enable time borrowing, as well as skewing clock. Introduction of clock skew to an edge-triggered flip-flop has an effect similar to the circuit retiming in VLSI timing optimization- movement of the flip-flops across combinational logic module boundaries [39]. Although it achieves time borrowing as SEFF does, but it makes physical design flow more complex, and in some cases, the standard tools require modification to support clock skew technique. Furthermore, useful clock skew assignment technique is a static solution and cannot account for circuit variability and other sources of uncertainty in the input data or environment. It has been shown to be ineffective for addressing process variation and circuit imbalance [9]. Moreover, in useful clock skew assignment method, the triggering edges of all FF's get delayed to the amount of critical path. In contrast SEFF provide a dynamic time borrowing structure since they allow time to be borrowed across different stages up to a maximum limit but only as much as needed. In other words, SEFF can pass data anytime during its transparency window, while a FF with skewed clock passes the data only at the shifted edge of clock. Obviously, adjusting clock for each individual flip-flop lifts this limitation at the cost of a complex design effort.

Integrated error handling mechanisms – Although the off-line determined voltage-frequency configurations are effective, they have been proved to be quite conservative. Razor flip-flop design [5] obtains an significant power reduction by adopting an smart opportunistic voltage scaling scheme. It only reduces voltage upon detection of timing errors in pipeline. It equips a pipeline with delay error detection capability as well as error correction mechanism.

In a later work, authors of [40], propose two local tuning mechanisms in the context of Razor dynamic voltage scaling: a per-stage voltage controlling and per-stage clock skew adjustment. Its drawbacks are rather complex to provide separate voltage supplies for each pipeline stage in physical implementation, plus the disadvantages of clock skewing technique mentioned earlier. In a recent work, Razor architecture has been revisited and Razor II has been proposed that provides both low-power operation and SER tolerance

[41]. Its power saving is achieved by performing only error detection in the FF, while correction is performed through architectural replay. This allows significant reduction in the complexity and size of the FF, too. Our work efficiently combines the power saving integrated error handling mechanism of Razor, with the performance enhancer time borrowing technique. Similar to Razor, MicroFix architecture [42] takes the delay errors as the indicator to required DVFS action. It handles errors in a prediction based manner [42].

VII. Conclusion

We presented and solved the problem of minimizing power-delay product metric in a linear pipeline by utilizing soft-edge flip-flops to perform time borrowing between consecutive stages of the pipeline. We formulated the problem of optimally selecting the transparency window sizes of the SEFF's and the clock frequency of pipeline so as to optimize the power-delay product of entire pipeline, in three different scenarios that assume deterministic worst case path delays or probabilistic random delays for pipeline stage delays. Also, by over-clocking the pipeline and allowing timing violations to occur and then being recovering the errors, our proposed ESOSP algorithm exploits the trade-off between performance and power saving to further minimize the expected power-delay product of a pipeline. The SEFF's are equipped with dynamic error detection and correction mechanism, to fix the generated errors. Our experimental results demonstrated that the proposed technique is quite effective in reducing the expected power-delay of a pipeline.

Appendix

A. Soft-Edge Flip-Flops with Built-in Error Detection

We have adopted an error detection mechanism in the design of SEFF to guarantee correct computation in the pipeline. More precisely, we have utilized a *multi-sampling technique* in the pipeline registers similar to Razor FF [5] (however, Razor integrates error correction circuitries, too, that increases flip-flop delay). Usually, flip flops with built-in error detection are intended to operate under a condition with low error rate; this would make the amortized performance and power overheads of micro-architectural correction negligible, while error correction is much faster but with high amortized overheads in built-in correction mechanisms (see B).

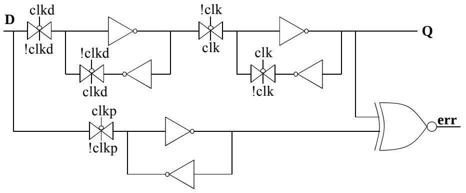

Fig. 15. Positive edge SEFF with built-in error detection

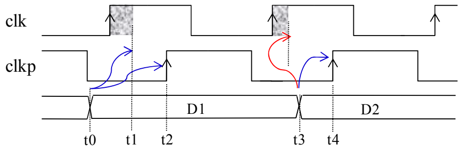

Fig. 16. Timing waveform of error detection in SEFF

In a SEFF with built-in error detection, a secondary latch, called *shadow latch*, is added to each conventional flip-flop. This shadow latch re-samples the input data at a later time by utilizing a *phase-shifted* global clock signal, *clkp*. Hence, the input will be double sampled at the triggering edges of the normal clock and the delayed clock. If there is a setup time violation in the pipeline stage, comparison of these two data values would detect the error. Fig. 15 shows the internal architecture of a master-slave SEFF with built-in error detection mechanism. Fig. 16 illustrates the operation of error detection circuitry. In this figure, data unit *D1* arrives early enough to get correctly latched in the FF at time *t1*. The error detection unit samples it at *t2* as the correct data. On the other hand, data *D2* misses the latching window (indicated by the red arrow in the figure) and cannot be latched at time *t3*. Instead *D1* or an invalid data is stored. However, at time *t4*, the error detection unit re-samples the data and captures *D2;* the result of XNOR of two sampled data indicates an error.

Introduction of the phase-shifted clock, *PS*, to design requires an additional timing constraint to avoid undetected errors or short path violations in the following scenarios. First, if the maximum delay of the preceding logic block is so large that the signal misses the triggering edges of both the main and PS clock edges. Second, as shown in Fig. 17, if the minimum delay of the combinational logic circuit succeeding a flip flop is too short, new data *D3* overwrites the valid one, *D2*, at *PS* clock edge and mistakenly marked as an error. We impose another timing constraint to address these scenarios:

$$t_{s,ij} + d_{ij}^{max} + t_{cq,j} - T_{clk} \le PS \le \delta_{ij}^{min} + t_{cq,j} - t_{h,ij} \quad 1 \le i \le N \qquad (42)$$

where *PS* denotes delay of PS-Clk relative to the main clock.

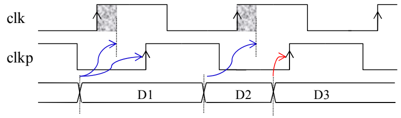

Fig. 17. Timing waveforms for the SEFF

B. Soft-Edge Flip-Flops with Built-in Error Correction

Similar to error detection, an error correction mechanism can be integrated in the flip flop circuit (see Razor FF [5]). As illustrated in Fig. 18, a multiplexer is integrated in the SEFF which selects between the data sampled at main clock edge and the one sampled at PS clock edge, which is the corrected data in case of any error. Compared to micro-architecture based error correction mechanisms (e.g. flushing), this approach has less performance overhead, but higher power dissipation and area overheads because of internal multiplexer gate. The timing constraint of (42) applies also to this SEFF.

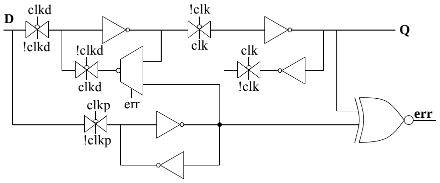

Fig. 18. Positive edge SEFF with built-in error correction

REFERENCES

[1] S. Manne, A. Klauser, and D. Grunwald, "Pipeline gating: speculation control for energy reduction," *Proc. Int'l Sym. Computer Architecture*, 1998,

[2] H. M. Jacobson, "Improved clock-gating through transparent pipelining," *Proc. Int'l Sym. on Low Power Electronics and Design*, 2004.

[3] H. Jacobson*, et al.* "Stretching the limits of clock-gating efficiency in server-class processors," *High-Performance Computer Architecture*, 2005.

[4] H. Partovi, *et al.*, "Flow-through latch and edge-triggered flip-flop hybrid elements," *Proc. Solid-State Circuits Conf.*, 1996.

[5] D. Ernst, *et al.*, "Razor: a low-power pipeline based on circuit-level timing speculation," *Proc. Int'l Sym. on Microarchitecture*, 2003.

[6] S. Das, *et al.*, "A self-tuning DVS processor using delay-error detection and correction", *IEEE Journal of Solid-State Circuits*, 2006.

[7] K. Choi, R. Soma, and M. Pedram, "Fine-grained dynamic voltage and frequency scaling for precise energy and performance trade-off based on the ratio of off-chip access to on-chip computation times." *IEEE Trans. on Computer Aided Design,* Vol. 24, No. 1, 2005, pp.18-28

[8] D. Harris and M. A. Horowitz, "Skew-tolerant domino circuits," *IEEE Journal of Solid*-State *Circuits*, 1997.

[9] V. Joshi, D. Blaauw, and D. Sylvester, "Soft-edge flip-flops for improved timing yield: design and optimization," *Proc. Int'l Conference on Computer-Aided Design*, 2007.

[10] M. Ghasemazar**,** M. Pedram, "Minimizing energy cost of throughput in a linear pipeline by opportunistic time borrowing," *Proc. Int'l Conf. Computer Aided Design, 20*08.

[11] M. Ghasemazar, B. Amelifard, and M. Pedram, "A mathematical solution to power optimal pipeline design by utilizing soft-edge flip-flops," *Proc. Int'l Symp. on Low Power Electronics and Design*, 2008.

[12] A. Dasdan, I. Hom, "Handling Inverted Temperature Dependence in Static timing Analysis," *ACM Trans. on Design Automation of Electronic Systems*, Vol. 11, No. 2, Apr. 2006

[13] M. Pedram, and S. Nazarian, "Thermal Modeling, Analysis and Management in VLSI Circuits: Principles and Methods," *Proc. of IEEE, Special Issue on Thermal Analysis of ULSI*, Vol. 94, 2006, pp. 1487-1501.

[14] K. Bernstein, *et al.*, "High-performance CMOS variability in the 65-nm regime and beyond," *IBM J. Res. & Dev.*, vol. 50, no. 4/5, Jul., 2006.

[15] Y. Ye, *et al.*, "Statistical modeling and simulation of threshold variation under dopant fluctuations and line-edge roughness," *Proc. Design Automation Conference*, 2008.

[16] S. R. Nassif, "Modeling and analysis of manufacturing variations", *Proc. IEEE Custom Integrated Circuits Conference*, 2001.

[17] S. Ross, *Introduction to probability models*, 9th edition, Academic Press, USA 2007.

[18] S. Choi, B. C. Paul, and K. Roy, "Novel sizing algorithm for yield improvement under process variation in nanometer technology," *Proc. Design Automation Conference*, 2004.

[19] M. Orshansky, A. Bandyopadhyay, "Fast Statistical Timing Analysis Handling Arbitrary Delay Correlations," *Design Automation Conf.*, 2004.

[20] Y Zhan, *et al.*, "Correlation-aware statistical timing analysis with non-gaussian delay distributions," in *Proc. Design Automation Conference*, 2005.

[21] J. Singh, S. Sapatnekar, "Statistical timing analysis with correlated non-Gaussian parameters using independent component analysis," in *Proc. Design Automation Conference*, 2006, pp. 155–160.

[22] L. Zhang, *et. al*, "Statistical static timing analysis with conditional linear MAX/MIN approximation and extended canonical timing model," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.,* vol. 25, 2006.

[23] D. Blaauw, *et al.*, "Statistical timing analysis: from basic principles to state of the art," *IEEE Trans. Computer-Aided Design*, vol 27, 2008.

[24] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* New York: Dover, 1972

[25] V. G. Oklobdzija, R. K. Krishnamurthy, *High-Performance Energy-Efficient Microprocessor Design* (Series on Integrated Circuits and Systems), 1st Ed. Springer, 2006

[26] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun, "Check Tc and min Tc: Timing verification and optimal clocking of synchronous digital circuits," *Proc. of Intl Conf. on Computer Aided Design*, November 1990.

[27] J. Le, X. Li, L. T. Pileggi, "STAC: Statistical Timing Analysis with Correlation", *Proc. of Design Automation Conference*, pp. 343-348, 2004.

[28] M. Mani, A. Devgan, and M. Orshansky, "An Efficient Algorithm for Statistical Minimization of Total Power under Timing Yield Constraints," *Proc. of Design Automation Conference,* 2005.

[29] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, UK: Cambridge University Press, 2003.

[30] MATLAB Optimization, http://www.mathworks.com

[31] Tomlab Optimization [Online] http://tomopt.com/tomlab/

[32] HSPICE: gold standard for accurate circuit simulation, http://www.synopsys.com/products/mixedsignal/hspice/hspice.htm

[33] Predictive Technology Model, http://ptm.asu.edu/

[34] E. M. Sentovich, *et al.*, "SIS: A System for Sequential Circuit Synthesis," University of California, Berkeley, Report M92/41, May 1992.

[35] K Duraisami, E Macii, M Poncino, "Using soft-edge flip-flops to compensate NBTI-induced delay degradation," *Great Lakes Sym. VLSI,* 2009.

[36] V. Joshi, R. R. Rao, D. Blaauw, D. Sylvester, "Logic SER reduction through flipflop redesign," *Int'l Sym.Quality Electronic Design*, 2006.

[37] A. Tiwari, S. R. Sarangi, J. Torrellas, "ReCycle: pipeline adaptation tolerate process variation," *Proc. Int'l Sym. Computer Architecture,* 2007.

[38] H. Lee, S. Paik, Y. Shin, "Pulse width allocation with clock skew scheduling for optimizing pulsed latch-based sequential circuits," *Proc. of Int'l. Conf. on Computer-Aided Design*, 2008.

[39] R. B. Deokar, and S. S. Sapatnekar, "A fresh look at retiming via clock skew optimization," *Proc. Design Automation Conference*, 1995.

[40] S. Lee, *et al.*, "Reducing pipeline energy demands with local DVS and dynamic retiming," *Int'l Sym. on Low Power Electronics and Design*, 2004.

[41] D. Blaauw, *et al.*, "Razor II: In-situ error detection and correction for PVT and SER tolerance," *Proc. Int'l Solid-State Circuits Conference*, 2008.

[42] G. Yan, *et* al. "MicroFix: exploiting path-grained timing adaptability for improving power-performance efficiency," *Proc. Int'l Sym. on Low Power Electronics and Design*, 2009.