GATE-LENGTH BIASING FOR DIGITAL CIRCUIT OPTIMIZATION

Applicant: Tela Innovations, Inc., Los Gatos, CA (US)

Inventors: Puncet Gupta, Marina Del Ray, CA (US); Andrew B. Kahng, Del Mar, CA (US)

Assignee: Tela Innovations, Inc., Los Gatos, CA (US)

Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

Appl. No.: 14/245,852
Filed: Apr. 4, 2014

Prior Publication Data

Related U.S. Application Data
Continuation of application No. 13/620,669, filed on Sep. 14, 2012, now Pat. No. 8,949,768, which is a continuation of application No. 12/212,353, filed on Sep. 17, 2008, now Pat. No. 8,127,266, which is a continuation of application No. 11/145,025, filed on Jun. 3, 2005, now Pat. No. 7,441,211.

Provisional application No. 60/678,694, filed on May 6, 2005.

Int. Cl. G06F 17/50 (2006.01)

U.S. Cl. CPC ........ G06F 17/5081 (2013.01); G06F 17/5063 (2013.01); G06F 17/5068 (2013.01); G06F 2217/78 (2013.01)

Field of Classification Search
USPC ................. 716/103, 107–109, 131–134, 139
See application file for complete search history.

References Cited
U.S. PATENT DOCUMENTS
5,619,420 A 4/1997 Breid
6,009,248 A 12/1999 Sato et al.
6,091,845 A 7/2000 Pierret et al.
6,219,630 B1 4/2001 Yonezawa et al.

FOREIGN PATENT DOCUMENTS
JP 2002258463 A 9/2002

OTHER PUBLICATIONS

Primary Examiner — Nghiia Doan
Attorney, Agent, or Firm — Martine Penilla Group, LLP

ABSTRACT
Methods and apparatus for a gate-length biasing methodology for optimizing integrated digital circuits are described. The gate-length biasing methodology changes a nominal gate-length of a transistor to a biased gate-length, where the biased gate-length includes a bias length that is small compared to the nominal gate-length.

27 Claims, 14 Drawing Sheets
References Cited

U.S. PATENT DOCUMENTS

6,425,115 B1 7/2002 Risier et al.
6,567,972 B1 5/2003 Tanaka et al.
6,581,028 B1 6/2003 Hayashi
6,687,145 B2 2/2004 Houston
6,691,255 B1 2/2004 Burden et al.
6,734,521 B2 5/2004 Houston
6,826,738 B2 11/2004 Cadour
6,931,617 B2 8/2005 Sanie et al.
6,954,918 B2 10/2005 Houston ....... 716/122
6,968,527 B2 11/2005 Pierart
6,993,741 B2 1/2006 Liebmann et al.
6,996,787 B2 2/2006 Houston
7,032,194 B1 4/2006 Haseh
7,149,999 B2 12/2006 Kahng et al.
7,175,940 B2 2/2007 Ladig et al.
7,222,328 B2 5/2007 Hasumi et al.
7,441,211 B1 10/2008 Gupta et al.
7,480,880 B2 1/2009 Visweswariah et al.
7,716,612 B1 5/2010 Gupta et al.
8,103,981 B2 1/2012 Kahng et al.
8,127,266 B1 2/2012 Gupta et al.
8,159,865 B2 5/2012 Cui et al.
2009/0024974 A1 1/2009 Yamada

OTHER PUBLICATIONS

Y. Tsai, “Influence of Leakage Reduction Techniques on Delay/Leakage Uncertainty,” pp. 1-6, 2005, Dept. of Computer Science and Eng., Penn State University, PA, Proc. of the 18th Int’l Conf. on VLSI Design held with 4th Int’l Conf. on Embedded Systems Design (VLSID ’05); 1063-9677/05 $20 2005 IEEE.


http://www.kld-tencor.com; “Prolith” (Copyright 2010).


Kao, J. et al.; “MICMOS Hierarchical Sizing Based on Mutual Exclusive Discharge Patterns”; 1998; ACM/IEEE Design Automation Conference.


Cao, Y. et al.; “Design Sensitivities to Variability: Extrapolations and Assessments in Nanometer VLSI”, 2002; IEEE ASIC-SOC.

Carpenter, Wally; “International SEMATECH: A Focus on the Photomask Industry”; Autumn 2000; Yield Management Solutions.


References Cited

OTHER PUBLICATIONS


Rieger et al.; “Layout Design Methodologies for Sub-Wavelength Manufacturing” Jun. 18-22, 2001; ACM.


Sirisantana, N. et al.; “High-Performance Low-Power CMOS Circuits Using Multiple Channel Length and Multiple Oxide Thickness”; 2000; International Conference on Computer Design.

Sreedhar, A. et al. “Statistical Yield Modeling for Sub-Wavelength Lithography”, University of Massachusetts at Amherst, Department of Electrical & Computer Engineering, Test Conference, 2008; IEEE.


Tsai et al.; “Influence of Leakage Reduction Techniques on Delay/Leakage Uncertainty”, Proceedings of the 18th Int'l Conf. on VLSI Design held jointly with 4th Int'l Conf. on Embedded Systems Design (VLSID '05), Jan. 3-7, 2005, Penn State University, PA.


* cited by examiner
FIG. 1
FIG. 3

1. Begin
2. Library Generation
3. Design Optimization
4. End
Begin

Identify Cells To Bias

Determine Bias Length(s)

Design Biased Variants

End

FIG. 4A
FIG. 4B

1. Begin
2. Identify Design Goal
3. Choose Biasing Algorithm
4. Select Timing Analyzer
5. Optimize
6. End
procedure $L_{GateBiasing}$
1 Run SSTA to initialize $s_p \forall$ cell instances, $p$
2 $S \leftarrow \{\}$
3 forall cell instances, $p$
4 $P_p \leftarrow \text{ComputeSensitivity}(p)$
5 $S \leftarrow S \cup P_p$
6 do
7 $P^* \leftarrow \max(S)$
8 if($P^* \leq 0$)
9 exit
10 $S \leftarrow S - \{P^*\}$
11 $\text{SaveState()}$
12 $\text{Downsize cell instance } p^*$
13 $\text{EISTA}(p^*)$
14 if($\text{TimingViolated()}$)
15 $\text{RestoreState()}$
16 else
17 $N \leftarrow p^* \cup \text{fan-in and fan-out nodes of } p^*$
18 forall $q \in N$
19 if($P_q \in S$)
20 $P_q \leftarrow \text{ComputeSensitivity}(q)$
21 Update $P_q$ in $S$
22 while($|S| > 0$)

procedure $\text{ComputeSensitivity}(q)$
1 $\text{old_slack} \leftarrow \text{Slack on cell instance } q$
2 $\text{old_leakage} \leftarrow \text{Leakage of cell instance } q$
3 $\text{SaveState()}$
4 $\text{Downsize cell instance } q$
5 $\text{CISTA}(q)$
6 $\text{new_slack} \leftarrow \text{Slack on cell instance } q$
7 $\text{new_leakage} \leftarrow \text{Leakage of cell instance } q$
8 $\text{RestoreState()}$
9 return ($\text{old_leakage} - \text{new_leakage})/(\text{old_slack} - \text{new_slack})$
Begin

Identify Cells To Bias

Determine Bias Length

Design Biased Variants

End

FIG. 6
Begin

Identify Design Goal

Identify Optimization Goal

Manual or Automatic?

Manual

Analyze Cell Topology

Automatic

Enumerate Configurations

Evaluate Under Load

Choose Configuration

End

FIG. 7
FIG. 10
FIG. 11

1100 Nominal Layout

1105 Gate Length Bias Adjustment Module

1115 OPC Engine

1125 OPC Solution

1110 Annotated Layout/Bias Requirements
FIG. 14
1. GATE-LENGTH BIASING FOR DIGITAL CIRCUIT OPTIMIZATION

CLAIM OF PRIORITY TO RELATED APPLICATION


BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to optimization of digital integrated circuits, and more particularly, to small gate-length biasing of transistors to improve performance characteristics.

2. Description of the Related Art

Modern-day digital integrated circuits are complex devices that often must meet high performance standards. Due to their complexity, the design and simulation of integrated circuits is also a complex task. Furthermore, the modern-day manufacture of integrated circuits has now reached minimum feature sizes that are down into the nanometer scale. Each new technology generation brings ever-tightener requirements for manufacturing process control. As a result, there is a demand for approaches that can improve the performance characteristics of integrated circuits, preferably with minimal disruption to existing design and manufacturing processes.

For example, power consumption is one aspect of circuit performance. High power dissipation in integrated circuits unfortunately leads to reduced circuit performance and reliability, and has a large impact on packaging costs. Power in complementary metal oxide semiconductor (CMOS) circuits consists of a dynamic component and a static component, which is primarily due to leakage currents. While lowered supply voltages (and consequently lowered threshold-voltages) and aggressive clock gating can achieve dynamic power reduction, these techniques typically increase leakage power and therefore cause the share of total power to increase. Manufacturers face the additional challenge of leakage variability: recent data indicates that leakage of microprocessor chips from a single 180 nm wafer can vary by as much as 20%. Thus, leakage power has become an important design concern for the system-level chip designer since it is becoming an ever-increasing component of total dissipated power, with its contribution projected to increase from 18% at 130 nm to 54% at the 65 nm node.

Leakage current is generally composed of three major components: (1) subthreshold leakage, (2) gate leakage, and (3) reverse-biased drain-substrate and source-substrate junction band-to-band tunneling leakage. The reverse-biased diode junction leakage does not depend on gate-length (also called channel length), gate leakage is linearly proportional to gate-length, and subthreshold leakage has an exponential dependence on gate-length. Subthreshold leakage, which is also proportional to operating temperature, is usually the dominant contributor to total leakage at 130 nm. It is likely to remain so in the future. This is especially true since gate leakage, which has only a small dependence on temperature, is often much reduced compared to subthreshold leakage in technologies using thick gate insulator thicknesses or high dielectric constant insulators, which is likely the case for technology nodes less than 65 nm.

Another leakage source is gate induced drain leakage (GIDL), which is primarily due to minority carriers in drain depletion region. GIDL is important primarily for moderately doped drains, since lightly doped drain (LDD) regions do not have high enough electric fields to trigger GIDL. LDD regions should not narrow due to channel length increases. Additionally, GIDL is a strong function of channel width and oxide thickness but not channel length. GIDL largely depends on the gate-drain overlap region, which does not change with changes in channel length.

Proposed techniques for leakage power reduction generally include use of multiple supply ($V_{DD}$ and $V_{SS}$) and gate threshold ($V_T$) voltages; and the assignment of input values to inactive gates such that leakage is minimized. Such leakage reduction methodologies can be divided into two classes depending on whether they reduce standby leakage or runtime leakage. Standby techniques reduce leakage of devices that are known not to be in operation, while runtime techniques reduce leakage of active devices.

Several techniques have been proposed for standby leakage reduction. Body biasing or VTMOS-based approaches dynamically adjust the device $V_T$ by biasing the body terminal. This technique has also been used to reduce leakage of active devices. Multi-threshold CMOS (MTCMOS) techniques use high-$V_T$ CMOS (or NMOS or PMOS) devices to disconnect one or both of $V_{DD}$ or $V_{SS}$ from logic circuits implemented using low $V_T$ devices in standby mode. In source biasing, a positive bias is applied in standby state to source terminals of off devices. Other techniques include the use of transistor stacks and the use of input-vector control. Among the drawbacks of these techniques are increased logic design complexity, circuit layout area overhead, and the coarse-grained nature of the resulting power reductions.

Currently, to the inventors' knowledge, the primary mainstream approach to runtime leakage reduction is the multi-$V_T$ manufacturing process. One drawback to this technique is the rise in process costs due to additional steps and masks. However, the increased costs have been outweighed by the resulting leakage reductions and multi-$V_T$ processes are common. One complication facing the multi-$V_T$ approach is the increased variability of $V_T$ for low-$V_T$ devices. This occurs in part due to random dopant fluctuations, as well as worsened drain induced barrier lowering (DIBL) and short-channel effects (SCE) in devices with lower channel doping. The larger variability in $V_T$ degrades the achievable leakage reductions of multi-$V_T$ approaches and worsens with continued MOS scaling. Moreover, multi-$V_T$ methodologies do not offer a smooth tradeoff between performance and leakage power. Devices with different $V_T$ typically have a large separation in terms of performance and leakage, for instance a 15% speed penalty with a 10x reduction in leakage for high-$V_T$ devices.

Gate-length ($L_{Gate}$) also affects device leakage currents. Large changes to gate-lengths, however, even in devices within non-critical gates, result in heavy delay and dynamic power penalties. Large changes would also necessitate large changes in design methodology, for example, potentially significant changes in design rules. In addition, cell layouts with large changes to gate-lengths are not layout-swappable with their nominal versions, resulting in substantial engineering change order (ECO) overheads during layout. Moreover, traditional sizers, which focus on width-sizing or multi-$V_T$ processes for optimization, perform poorly with gate-length sizing because it is fundamentally different than width sizing.
Thus, there is a need to improve digital circuits, for example, by reducing leakage current and thus leakage power, while minimally impacting delay performance and/or design and manufacturing processes.

SUMMARY OF THE INVENTION

The present invention overcomes the limitations of the prior art by providing small biasing of device gate-lengths, preferably in a manner that has low impact on existing design and/or manufacturing processes. For example, biasing of device gate-lengths affords an additional design space to reduce chip leakage power and its variability. Typically, leakage power decreases exponentially, and delay increases linearly, with increasing gate-length. Thus, it is possible to increase gate-length only marginally to take advantage of the exponential leakage reduction, while improving time delay performance only linearly. From a design flow standpoint, the use of only slight increases in gate-length can preserve pin- and layout-compatibility. Therefore, the technique can be applied, for example, as a post-layout enhancement step. Application of gate-length biasing, primarily to those devices that do not appear in critical paths, can achieve zero or negligible degradation in delay characteristics for the chip.

In an exemplary embodiment, the gate-length biasing methodology includes optimizing a circuit by adjusting a nominal gate-length of a transistor by a small bias length. The small bias length may be, for example, less than 10% of the nominal gate-length or less than a predefined fraction of the nominal gate-length. The bias length may be determined by evaluating a design tradeoff, such as leakage power versus circuit delay. The gate-length biasing methodology may be applied, for example, at a cell level or at a transistor level. The transistor may be part of a cell that is in a non-critical timing path, or part of a cell that has asymmetrical timing areas. The nominal cell to replace may be identified, for example, using a sensitivity-based downsizing approach, or a sensitivity-based upsizing approach, or a combination thereof.

Small gate-length biasing can also be used for various other purposes. For example, it may be used to reduce various types of power consumption (e.g., total power, static power or dynamic power), preferably in a manner that has minimal or zero impact on timing delays. Decreasing gate-lengths of certain devices can reduce time delays or increase the operating frequency of the chip. Gate-length biasing may also be used to increase the on-chip signal integrity. As a final example, gate-length biasing may be used for manufacturability purposes: for example, to increase the reliability or manufacturability of the chip. The amount and sign of the biasing and the specific transistors to be biased will depend in part on the purpose of the biasing. Biasing can be positive (to longer gate-lengths) or negative (to shorter gate-lengths), depending on the application.

In an exemplary embodiment, the gate-length biasing methodology may be implemented, for example, by generating an enhanced library including a biased variant(s) of a nominal cell, where a biased transistor in the biased variant corresponds to a nominal transistor in the nominal cell. The biased transistor includes a biased gate-length where the bias length is small compared to a nominal gate-length of the nominal transistor. In an exemplary embodiment, the biased variant remains pin-compatible with the nominal cell.

In another exemplary embodiment, the gate-length biasing methodology may be implemented using optical proximity correction to apply the small bias length to particular transistors in a nominal layout. The implementation may include one or more of, for example, shifting a required error tolerance, applying a starting edge offset, using a maximum error tolerance, accounting for iso-dense layout effects, or other techniques.

In another exemplary embodiment, the gate-length biasing methodology may be implemented using an electronic design automation (EDA) tool, for example, a design rule checker, to generate a biased layout including the small bias length applied to one or more transistors.

Other aspects of the invention include devices and systems corresponding to the methods and embodiments described above, and digital circuits produced by these methods.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention has other advantages and features which will be more readily apparent from the following detailed description of the invention and the appended claims, when taken in conjunction with the accompanying drawings, in which:

FIG. 1 is a graph of the variation of delay and leakage with gate-length for an industrial 130 nm process.

FIGS. 2A and 2B are a conceptual representation of a gate-length biasing methodology in accordance with the present invention.

FIG. 3 is a flow chart of an exemplary embodiment of an \( L_{\text{Gate}} \) biasing methodology in accordance with the present invention.

FIG. 4A is a flow chart of an exemplary CLLB embodiment of the library generation step of the method of FIG. 3.

FIG. 4B is a flow chart of an exemplary CLLB embodiment of the design optimization step of the CLLB method of FIG. 3.

FIG. 5 is the pseudocode for an exemplary embodiment of a leakage optimization implementation.

FIG. 6 is a flow chart of an exemplary TLLB embodiment of the library generation step of the method of FIG. 3.

FIG. 7 is a flow chart of an exemplary embodiment of a biasing methodology to design biased variants for the library generation method of FIG. 6.

FIG. 8 is a schematic diagram of a simple NAND cell biased using the TLLB method.

FIG. 9 is an image of a cell layout of the generic AND2X6 cell with simulated printed gate-lengths for all devices in the cell.

FIG. 10 is a graph of the leakage distributions for the unbiased, technology-level selectively biased, and uniformly biased scenarios for a representative test case.

FIG. 11 is a flowchart of an exemplary embodiment of an OPC implementation of the gate-length biasing methodology of the present invention.

FIG. 12 is a flowchart of an exemplary embodiment of a cell library implementation of the gate-length biasing methodology of the present invention.

FIG. 13 is a flowchart of an exemplary embodiment of an EDA tool implementation of the gate-length biasing methodology of the present invention.

FIG. 14 is a flowchart of an exemplary embodiment of a bias requirement communication methodology of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Gate-Length Biasing Methodology

Novel approaches using small gate-length (\( L_{\text{Gate}} \)) biasing, usually (but not always) increases in \( L_{\text{Gate}} \) for device and circuit optimization, and preferably having minimal impact
on circuit performance and manufacturing costs, are described. The terms “gate-length biasing” and “$L_{\text{gate biasing}}$” are used interchangeably to refer to the proposed technique. The phrase “biasing a device” implies adjusting the gate-length of the device slightly. The term “nominal gate-length” refers to the gate-length of an unbiased device. The methodology may be used, for example, to optimize (e.g., reduce) runtime leakage power. The following examples are based on the goal of reducing leakage power (and, as a result, the gate-length biasing usually results in an increase in gate-lengths), but it should be recognized that gate-length biasing is not limited to this particular application or to increases in gate-length.

One advantage of this technique is that small increases in gate-length can have significant effects on device characteristics such as leakage current reduction since they take significant advantage of the SCE and incur only small penalties in drive current and input capacitance. Typically, leakage power decreases exponentially, and delay increases linearly, with increasing gate-length. For example, FIG. 1 is a graph of the variation of delay and leakage with gate-length for an industrial 130 nm process. FIG. 1 shows the advantage to be gained, in particular, leakage reduction for minimal delay penalty, by slight increases in gate-length. Leakage current flattens out with gate-length beyond 140 nm, making $L_{\text{gate biasing}}$ less desirable in that range.

FIGS. 2A and 2B are a conceptual representation of a gate-length biasing methodology in accordance with the present invention. FIG. 2A is a conceptual representation of an unbiased circuit 210. Unbiased circuit 210 includes transistors A, B, C, and D, where transistor D has width W and length L. FIG. 2B is a conceptual representation of a biased circuit 210*, which is circuit 210 of FIG. 2A after application of the gate-length biasing method of the present invention. Biased circuit 210* includes unbiased transistors A, B, C, and D of circuit 210. However, transistor D of unbiased circuit 210 has been replaced by transistor D' in biased circuit 210*, where transistor D' has width W and length L+ΔL.

The gate-length biasing technique of the present invention may be applied, for example, at the cell level or at the transistor level. At the cell level, in an exemplary embodiment, the approach enhances a standard cell library by adding $L_{\text{gate biasing}}$ biased variants to the library. For example, as shown in FIG. 2A, transistor D is part of cell Q in the unbiased circuit 210. In biased circuit 210* of FIG. 2B, however, cell Q has been replaced by cell Q', which is a biased variant of cell Q and includes transistor D'. In an exemplary embodiment, the gate-length biasing methodology is applied primarily to devices in non-critical timing paths, to optimize device characteristics. A critical timing path 215, which does not include transistors D or D', is shown in FIGS. 2A and 2B. For example, in the context of leakage reduction, a leakage optimization approach is used to incorporate slower, low-leakage cells into non-critical paths, while retaining faster, high-leakage cells in critical paths.

On the other hand, at the transistor level, since different transistors control different timing arcs of a cell, an exemplary embodiment of gate-length biasing at the transistor level includes individually modifying delays of different timing arcs. Asymmetry in timing criticality of different timing arcs of a cell instance in a circuit, and that of rise and fall transitions, can be used by transistor level gate-length biasing to yield significant leakage savings. For example, any of the transistors in FIG. 2A or 2B might be replaced with a corresponding biased variant as a result of applying the gate-length biasing methodology at the transistor level. Alternatively, different biased variants of cell Q may be generated based on different timing arcs, and then cell Q is replaced by the appropriate biased version Q' in circuit 210*.

The gate-length biasing methodology can provide several advantages and benefits. One potential benefit of the gate-length biasing methodology is that if a gate-length bias is less than the pitch of the layout grid, the biased design generally avoids design-rule violations. This allows optimization without costly redesign. Moreover, it implies that the biased and unbiased cell layouts are pin-compatible and hence layout-swappable. This allows gate-length biasing-based optimization to be possible at any point in a design flow, unlike sizing-based methods.

Other potential benefits of the gate-length biasing methodology in the context of leakage power optimization, are significant leakage reduction and minimal or zero delay penalty. Test cases showed that with a bias of 8 nm in a 130 nm process, leakage reductions of 24% to 58% were achieved for a set of the most commonly used cells with a delay penalty of under 10%. Using simple sizing techniques, other test cases showed 33% leakage savings with less than 3% dynamic power overhead and no delay penalty. These phenomena are not restricted, however, to the 130 nm node, and similar benefits are likely for other process nodes as well.

Another potential benefit of the gate-length biasing methodology is a choice of biasing strategies. Exemplary embodiments of the gate-length biasing methodology include a cell-level biasing methodology and a transistor-level methodology. Further optimization may be obtained by using transistor-level gate-length biasing in conjunction with cell-level gate-length biasing. In a comparison of gate-length biasing at the cell-level and at the transistor-level in the context of leakage reduction, test cases showed transistor-level gate-length biasing can further reduce leakage by up to 10% but requires a potentially significantly larger library. As a result of this tradeoff, in one approach, transistor-level biasing is done for only the most frequently used cells such as inverters, buffers, NAND, and NOR gates. Fortunately, the most frequently used cells have one or two inputs and hence only a small number of variants need be characterized for them. To further reduce library size, only one of the cell variants in which different logically equivalent inputs are fast may be retained, and pin-swapping techniques can be used during leakage optimization.

Another potential benefit of the gate-length biasing methodology is that the devices with biased gate-length may be more manufacturable and may have a larger process margin than the nominal devices. Biasing typically does not require extra process steps, unlike multiple-threshold based leakage optimization methods.

Another potential benefit is that gate-length biasing can lead to more process-insensitive designs, for example, leakage variability reduction. Since the sensitivity of leakage to gate-length reduces with increased gate-length, a fixed level of variability in gate-length translates to reduced variability in leakage. Leakage variability may occur, for example, due to dopant variation, voltage supply variation, and temperature variation. Dopant variation results in threshold variation, which can cause not only leakage variation, but also timing changes. Dopant fluctuation is inversely proportional to the square root of the gate area. Increasing the gate-length slightly results in a slightly increased gate area and thus reduced threshold and leakage variation. Additionally, multi-gate-length designs, e.g., a design including nominal and biased gate-lengths, track supply voltage variation better than multi-threshold voltage designs, also leading to reduced leakage variation. In particular, test cases showed that gate-length biased designs can have 41% less leakage worst-case vari-
ability in presence of inter-die variations as compared to nominal gate-length designs. In presence of both inter- and intra-die critical dimension variations, this implies selective gate-length biasing may yield designs less sensitive to variations.

Other potential benefits of the gate-length biasing methodology include ease of use and potential for further optimization by using the methodology in conjunction with other techniques for further benefit. The use of more than two gate-lengths, for example, for the most commonly used cells, along with improved sizing techniques, or other common optimization techniques, is likely to yield further optimization, such as better leakage savings. In particular, in the context of leakage reduction, further reductions in leakage are possible by applying gate-length biasing after first applying the multi-threshold voltage technique, which is widely used for power reduction.

As an example of the benefits available from applying the gate-length biasing methodology to a process with multi-$V_{th}$ an advanced low power process may yield products that have an on current ($I_{on}$) for high-$V_{th}$ devices of less than 20 pA/μm, with a portion of $I_{ds}$ due to junction leakage that is not affected by gate-length biasing, and three orders of magnitude delta between active and standby currents. An $I_{ds}$ of 20 pA/μm indicates that the leakage power budget is quite small and that leakage is always a concern. Also, the three orders of magnitude difference is quite large, indicating room for further optimization. Thus, even though the design starts with a multi-$V_{th}$ library first, gate-length biasing can still play a significant role in the low power design flow.

The gate-length biasing methodology could exploit the design spaces offered by considering the overall distribution of power consumption (e.g., including transponder power) and sensitivities to both standby and total power by, for example, constraining the biasing so as to not increase total power, etc. In particular, the gate-length biasing methodology may apply an optical proximity correction (OPC) guidance “knob” (discussed in more detail infra), not available in standard multi-$V_{th}$ processes, to achieve improved robustness and silicon quality. Also, the gate-length biasing methodology provides a “granularity” win. In particular, conventional critical dimension (CD) biases on nominal or low-$V_{th}$ devices are inserted to gain speed but they also push leakage up substantially beyond the 20 pA/μm value. Instead of discretely jumping from 20 pA/μm to nA leakage levels with the corresponding 50% speed boost, a conventional low power process could include a larger $I_{Gate}$ and nominal- or low-$V_{th}$ combination to create a finer tradeoff of speed versus leakage. Lastly, no matter how small the high-$V_{th}$ device leakage is, the gate-length biasing methodology typically provides further reduction in leakage. Gate-length biasing of high-$V_{th}$ devices can be valuable because there are often many high-$V_{th}$ devices because the sizer begins with high-$V_{th}$, and a large percentage of high-$V_{th}$ devices can be biased because of large slack on many paths even after $V_{th}$ assignment. For these reasons, significant leakage savings are possible. This highlights the value of the gate-length biasing technique, since further reductions in leakage are realizable using both optimization methods in conjunction with each other.

The gate-length biasing methodology may also yield further benefit to a process that already includes multiple gate-lengths. For example, the gate-length biasing methodology may provide one or more of: essentially continuous gate-length sizing without increased mask layers, topology based transistor level biasing, OPC error biasing, biasing with awareness of iso-density effects, and other benefits. Each of these benefits can lead to further optimization than is available with a multiple gate-length process alone.

The following sections include further descriptions of exemplary approaches to small increases in gate-length (e.g., referred to simply as $L_{Gate}$ biasing), including the following. Cell-level and transistor level gate-length biasing methodologies, which may be based, for example, on less than 10% increases in drawn $L_{Gate}$ of devices and may be used to address leakage reduction, are discussed. Experiments and results showing benefits of cell-level and transistor-level $L_{Gate}$ biasing methodologies in different design scenarios such as dual-$V_{th}$ are also discussed. An analysis of potential benefits and caveats of such biasing methodologies, including some possible implications for lithography and process variability, is presented. Lastly, various implementation methods for gate-length biasing are presented.

As discussed above, gate-length biasing may be implemented at the cell level. FIG. 4A is a flow chart of an exemplary cell-level $L_{Gate}$ biasing (CLLB) embodiment of the library generation 305 step of method 300. A library may include hundreds of cells designed for a particular technology node. To enhance the library, in accordance with the CLLB approach, cells are identified 405 for biasing. For example, all cells in the library may be targeted for biasing, or some subset of cells, such as the most commonly used cells, may be identified.

As shown in FIG. 4A, library generation 305 also includes determining 410 a bias length for the targeted cells. In one embodiment, a single bias length is chosen. However, in alternative embodiments, multiple bias lengths may be chosen. The tradeoff is in the final size of the library generated by adding biased cell variants. The CLLB approach generally relies on the $V_{th}$ roll-off curve for a given technology. The
roll-off curve affects the feasibility of the approach and also affects the chosen bias length, i.e., how much to increase the nominal gate-length.

One embodiment of the CLLB library generation focuses on less than 10% biasing. However, alternative embodiments may include biasing of less than 10%. Bias lengths less than 10% of the nominal gate-length are advantageous for several reasons. First, the nominal gate-length of the technology is usually very close to or beyond the “knee” of the leakage vs. L\textsubscript{gate} curve which arises due to SCE. For large bias (i.e., significant increases in L\textsubscript{gate}), the advantage of super-linear dependence of leakage on gate-length is lost. Moreover, dynamic power and delay both increase almost linearly with gate-length. Therefore, small biases give more “bang for the buck.” Second, from a manufacturability point of view, having two prevalent pitches, which are relatively distinct, in the design can harm printability properties (i.e., the size of the process window). Cells with biases of less than about 10% can often retain the same poly-pitch as the unbiased version of the cell. There is a small decrease in spacing between gate-poly geometries, but minimum spacing rules typically are not violated even when the unbiased polys are at minimum spacing, since the biases are within the tolerance margins. Since design rule check (DRC) tools first snap to grid, biases of under about 10% typically are considered acceptable due to margins in design rules. Third, an increase in drawn dimension that is less than the layout grid resolution (e.g., typically 10 nm for 130 nm technology) ensures pin-compatibility with the nominal version of the cell. This is important to ensure that multi-L\textsubscript{gate} optimizations can be done post-placement or even after detailed-routing, without ECOs. In this way, the layout transparency that has made multi-V\textsubscript{th} optimization so adoptable within chip implementation flows is retained. Biases smaller than the layout grid-pitch typically will also ensure design-rule correctness for the biased cell layout, provided that the unbiased version is design-rule correct.

As shown in FIG. 4A, library generation also entails designing the biased variants. In particular, the biased L\textsubscript{gate} library is laid out and characterized. Since small biases to the gate-length are usually selected, the layout of the biased library cell usually does not need to change except for a simple automatic scaling of dimensions. Moreover, if the bias is smaller than the minimum layout grid pitch, design rule violations typically do not occur. After the slight modifications to the layout, the biased versions of the cell are put through the standard extraction and power/timing characterization process.

As an example, seven test cases were chosen to provide verification of the CLLB approach. For library generation, the test cases were first synthesized with the complete Artisan TSMC 130 nm library to identify the most frequently used cells. A restricted library was generated composed of variants of the 25 most commonly used cells in the test cases. A biased variant, in which all devices had the biased gate-length, was added for each cell.

The SPICE models for unbiased cells had a nominal gate-length of 130 nm for all transistors. All transistors in a biased variant of a cell had a gate-length of 138 nm. Choosing 138 nm as the biased gate-length places the delay of the low-V\textsubscript{th}-biased variant between the low-V\textsubscript{th}-nominal gate-length variant and the nominal-V\textsubscript{th}-nominal gate-length variant. Larger bias can lead to larger per-cell leakage savings at a higher performance cost. However, in a resizing setup (described below) with a delay constraint, the leakage benefit over the whole design can decrease as the number of instances that can be replaced by their biased version is reduced. Larger or smaller biases may produce larger leakage reductions for some designs. Libraries, however, are usually not design specific and a biased gate-length that produces good leakage reductions for most designs is desirable. The approach for determining the biased gate-length described above will generally work well for most typical designs. Those of skill in the art will appreciate that the value of 138 nm is highly process specific and is not intended to reflect the best biased gate-length for all 130 nm processes. Alternative embodiments may use biasing at lower levels of granularity, for example, having multiple biased gate-lengths and/or independently biasing devices within a cell.

As discussed above, in an exemplary embodiment of method 300, design optimization 310 entails leakage reduction. FIG. 4B is a flow chart of an exemplary CLLB embodiment of the design optimization step of method 300. An exemplary embodiment of design optimization 310 includes identifying a design goal. In one embodiment, the design goal may be minimizing delay, which is often a primary design goal in circuit design. Alternative embodiments, however, may be designed to achieve other design goals, such as other types of power reduction, improved manufacturability or reliability or yield, or reducing timing delays or increasing the operating frequency of a chip. In the case of delay as a design goal, a circuit designer performs gate-width sizing to achieve the minimum possible delay. Such sizing may be performed, for example, prior to gate-length biasing.

In the example of FIG. 4B, design optimization 310 also includes choosing a biasing algorithm. Different algorithms will be apparent. For large optimizations, an iterative approach may be desirable. For small cases, exhaustive search may be adequate. In one embodiment, the biasing algorithm may be a downsizing algorithm for leakage reduction. In an alternative embodiment, an upsizing algorithm for leakage reduction may be used. In further alternative embodiments, a combination of downsizing and upsizing, or other algorithms may be used. A sensitivity-based, downsizing algorithm for leakage optimization begins with all nominal cells and replaces cells on non-critical paths with biased variants. An upsizing algorithm begins with all biased variants in the circuit and replaces critical cells with their nominal-L\textsubscript{gate} variants. In many cases, downsizing appears to be more effective at leakage reduction than upsizing irrespective of the delay constraints. An intuitive rationale is that upsizing approaches have dual objectives of delay and leakage reduction during cell selection for upsizing. Downsizing approaches, on the other hand, only downsize cells that do not cause timing violations and have the sole objective of leakage minimization. An upsizing approach, however, may be faster when loose delay constraints are to be met since fewer transistors are upsized. Delay is almost always the primary design goal, however, and loose delay constraints are rare.

The phrase “downsizing a cell instance” (or node) implies replacing the cell or node by its biased variant in the circuit. In an exemplary embodiment of a sensitivity-based downsizing algorithm, the sensitivity associated with cell instance p represents the sensitivity associated with cell instance p in the circuit. In an exemplary embodiment of a sensitivity-based downsizing algorithm, the sensitivity associated with cell instance p represents the sensitivity associated with cell instance p in the circuit.
As shown in FIG. 4B, design optimization 310 also includes selecting 460 a timing analyzer. A timing analyzer is a useful component of a delay-aware power optimization approach. It is used to compute delay sensitivity to biasing of cell instances in the design. For an accurate yet scalable implementation, various embodiments of the design optimization 310 may choose from, for example, three types of timers that vary in speed and accuracy: standard static timing analysis (SSTA), exact incremental STA (ESTA) or constrained incremental STA (CISTA).

Under SSTA, slacks and actual arrival times (AATs) are propagated forward after a topological ordering of the circuit. Required arrival times (RATs) are back-propagated and slack is then computed. Under ESTA, timing analysis begins with the fan-in nodes of the node that has been modified. From all these nodes, slacks and AATs are propagated in the forward direction until the values stop changing. RATs are back-propagated from only those nodes for which the slew, AAT or RAT has changed. Under CISTA, sensitivity computation involves temporary modifications to a cell to find change in its slack and leakage. To make this step faster, the incremental timing calculation can be restricted to only one stage before and one stage after the modified gate. The next stage is affected by slew changes and the previous stage is affected by the pin capacitance change of the modified gate. The ripple effect on other stages farther away from the gate, primarily due to slew changes but potentially also due to coupling induced delay as the arrival time windows can change, may be neglected since high accuracy is not critical for sensitivity computation.

As shown in FIG. 4B, design optimization 310 also includes optimizing 465. In an exemplary embodiment of a CLIB approach, design optimization 310 entails leakage reduction. Thus, an exemplary embodiment of optimizing 465 includes performance of leakage optimization.

FIG. 5 is sample pseudocode for an exemplary embodiment of a leakage optimization implementation. The algorithm begins with SSTA and initializes slack values $S_p$ in Line 1. Sensitivities $P_p$ are computed for all cell instances $p$ and put into a set $S$ in Lines 2-5. The largest sensitivity $P_p$ is selected and removed from the set $S$, and the algorithm function if $P_p$ is 0. In Line 11, the function SaveState saves the gate-length of all transistors in the circuit as well as the delay, slew, and slack values. The cell instance $p$ is downsized and ESTA is run from it to update the delay, slew, and slack values in Lines 12-13. The timing libraries capture the effect of biasing on slew as well as input capacitance, and the static timing analyzer efficiently and accurately updates the design to reflect the changes in delay, capacitance, and slew due to the downsizing move. If there is no timing violation (e.g., negative slack on any timing arc) then the move is accepted, otherwise the saved state is restored. If the move is accepted, sensitivities of node $P_p$, its fan-in nodes, and its fan-out nodes are updated in Lines 17-21. The algorithm continues until the largest sensitivity becomes negative or the size of the set $S$ becomes zero. Function ComputeSensitivity($q$) temporarily downsizes cell instance $q$ and finds its slack using CISTA. Since high accuracy is not critical for sensitivity computation CISTA, which is faster but less accurate than ESTA, may be used for timing analysis.

As an example of the CLIB design optimization 310, minimum delay was identified 450 as the design goal. Gate-

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Leakage (mW)</th>
<th>CPU (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s9234</td>
<td>0.0712</td>
<td>4.86</td>
</tr>
<tr>
<td>c5135</td>
<td>0.3317</td>
<td>24.18</td>
</tr>
<tr>
<td>c5552</td>
<td>0.6284</td>
<td>55.56</td>
</tr>
<tr>
<td>s13207</td>
<td>0.1230</td>
<td>33.43</td>
</tr>
<tr>
<td>c6288</td>
<td>1.8730</td>
<td>508.86</td>
</tr>
<tr>
<td>ali128</td>
<td>0.4687</td>
<td>1122.89</td>
</tr>
<tr>
<td>s34417</td>
<td>0.4584</td>
<td>1331.40</td>
</tr>
</tbody>
</table>

Transistor-Level Gate-Length Biasing

The term “timing arc” indicates an intra-cell path from an input transition to a resulting rise (or fall) output transition. Generally, for an $n$-input gate, there are $2n$ timing arcs; however, there may be four timing arcs corresponding to non-unique inputs (e.g., select input of MUX). Due to different parasitics as well as PMOS/NMOS asymmetries, these timing arcs can have different delay values associated with them. For instance, Table 2 shows the delay values for the same input slew and load capacitance pair for different timing arcs of a NAND2X2 cell from the Artisan TSMC 130 nm library. The asymmetry in delays of various timing arcs within the NAND2X2 cell is clear. Pin swapping is a common post-synthesis timing optimization step to make use of the asymmetry in delays of different input pins. To make use of asymmetry in rise/fall delays, techniques such as P/N ratio perturbations, for example, may decrease circuit delay.

<table>
<thead>
<tr>
<th>Timing Arc</th>
<th>Propagation Delay (ps)</th>
<th>Transition Delay (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A → Y</td>
<td>99.05</td>
<td>104.31</td>
</tr>
<tr>
<td>A → Y</td>
<td>73.67</td>
<td>79.12</td>
</tr>
<tr>
<td>B → Y</td>
<td>107.20</td>
<td>112.98</td>
</tr>
<tr>
<td>B → Y</td>
<td>70.65</td>
<td>76.37</td>
</tr>
</tbody>
</table>

The gate-length biasing methodology can also exploit the asymmetries in delay values using transistor-level gate-length biasing (TLIB). Since different transistors control different timing arcs of a cell, TLIB can individually modify delays of different timing arcs. For example, TLIB can yield
leakage optimization by "recovering" leakage from cell instances in which: (1) not all timing arcs are timing-critical, and/or (2) rise and fall transitions are not both timing-critical at the same time.

As with CLLB described above, TLLB uses libraries of cells, which are used by system-level chip designers to design particular chips. Each cell includes one or more transistors configured to perform some function, such as a NAND cell, an inverter, a buffer, or other circuit. The current-voltage characteristics of the transistors in a cell are determined by the device's properties, which are generally optimized for a particular technology node; for example, a 130 nm gate-length technology node. Further perturbations, however, made to some of these properties, for example, gate-length biasing, can be used to further optimize performance for particular designs.

FIG. 6 is a flow chart of an exemplary TLLB embodiment of the library generation step of the method of FIG. 3. Similarly to FIG. 4A, cells are identified for biasing and bias lengths are determined 410.

As shown in FIG. 6, library generation 305 also entails designing 715 the biased variants. For each cell, the library may contain variants corresponding to all subsets of the set of timing arcs. A gate with n inputs has 2^n timing arcs and therefore 2^n variants, including the original cell. Given a set of critical timing arcs, the goal is to assign a biased I_{Gate} to some transistors in the cell and nominal I_{Gate} to the remaining transistors.

FIG. 7 is a flow chart of an exemplary embodiment of a biasing method 750 to design 715 biased variants. Under biasing method 750, a design goal is identified 755, for example, minimum delay. An optimization goal is also identified 760, for example, leakage reduction. Given the design goal and the optimization goal, the task is to design biased variants such that, for example, (1) critical timing arcs have a delay penalty of less than 1% with respect to the original unbiased cell, and (2) cell leakage power is minimized. In one embodiment, assignment of a biased I_{Gate} to transistors in a cell, given a set of critical timing arcs, can be done manually 765 by analyzing 770 the cell topology for simple cells. In an alternative embodiment, however, the process can also be automated 765. In an exemplary automatic process, all configurations for each cell in which nominal I_{Gate} is assigned to some transistors and biased I_{Gate} to the others are enumerated 775. For each configuration, the delay and leakage are determined 780 under a canonical output load, for example, using SPICE simulations with an inverter (INVX1) as a load. For each possible subset of timing arcs that can be simultaneously critical, one biasing configuration is chosen 785 based on the two criteria given earlier.

As an example, FIG. 8 is a schematic diagram of a simple NAND cell biased using the TLLB method. The biasing scheme shown in FIG. 8 shows I_{Gate} biasing of the transistors in the simplest NAND cell (NAND2X1) when only the rise and fall timing arcs from input A to the output Out are critical. In this case only the PMOS device 805 with B as its input can be slowed without penalizing the critical timing arcs.

Referring to FIG. 3, the TLLB method also entails design optimization 310. As discussed above, in an exemplary embodiment of the TLLB method, design optimization 310 entails leakage reduction, in which leakage optimization is performed. The exemplary embodiment of the design optimization 310 method of FIG. 4B, discussed above with respect to the CLLB method, may also be applied for the TLLB method. For example, a sensitivity-based downsizing approach that is similar to the one described above with reference to FIGS. 4B and 5 and the CLLB method can be used to optimize for leakage reduction. The method keeps track of the slack on every timing arc and computes sensitivity for each timing arc.

A particular benefit of the TLLB method is that it can be performed in conjunction with other optimization techniques, including the CLLB method discussed above. For example, in one embodiment, to limit the runtime and memory requirements, optimization may occur first at the cell level, using the CLLB method 300, and then at the transistor level, using the TLLB method 600. Further savings can be achieved, for example, by optimizing only the unbiased cells in the circuit. In another embodiment, the TLLB method may be implemented to further optimize a process flow that already includes a multi-V_{th} approach or even a multi-gate-length approach to provide optimization with a device-level granularity to optimize based on critical timing arcs.

**EXPERIMENTAL EXAMPLES**

A test flow for validation of the I_{Gate} biasing methodology was implemented in the context of leakage reduction. Seven test cases were chosen for investigation. Details of the test cases used in the experiments are given in Table 3. For each test case, Table 3 shows the source of the test case, the number of cells in the circuit, delay, leakage power, and dynamic power. Sequential test cases (e.g., those beginning with "s") were handled by converting them to combinational circuits by treating all flip-flops as primary inputs and primary outputs. The test flow was designed to validate an I_{Gate} biasing methodology in which CLLB was performed first followed by TLLB to show further reductions in leakage. Thus, while library generation and design optimization are discussed primarily with respect to the CLLB method, the discussion applies as well to the TLLB method.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Source</th>
<th>#Cells</th>
<th>Delay (ns)</th>
<th>Leakage (mW)</th>
<th>Dynamic (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a9234</td>
<td>ISCAS'89</td>
<td>861</td>
<td>0.437</td>
<td>0.7074</td>
<td>0.5907</td>
</tr>
<tr>
<td>c5315</td>
<td>ISCAS'85</td>
<td>1442</td>
<td>0.1556</td>
<td>1.4413</td>
<td>1.5345</td>
</tr>
<tr>
<td>c7552</td>
<td>ISCAS'85</td>
<td>1902</td>
<td>0.4885</td>
<td>1.8328</td>
<td>2.0813</td>
</tr>
<tr>
<td>e13207</td>
<td>ISCAS'89</td>
<td>1957</td>
<td>0.904</td>
<td>1.3934</td>
<td>0.6296</td>
</tr>
<tr>
<td>e2688</td>
<td>ISCAS'85</td>
<td>4289</td>
<td>2.118</td>
<td>3.5994</td>
<td>8.0216</td>
</tr>
<tr>
<td>e35417</td>
<td>ISCAS'89</td>
<td>7828</td>
<td>0.892</td>
<td>4.9381</td>
<td>4.2069</td>
</tr>
</tbody>
</table>

In this example, to identify the cells to bias, the test cases were synthesized with the Artisan TSMC 130 nm library using Synopsys Design Compiler v2003.06-SP1 with low-V_{th} cells only. To limit library characterization runtime, the library was restricted to variants of the following 25 most frequently used cells: CLKINVX1, INVX12, INVX1, INVX3, INVX4, INVX8, INVXL, MX12X1, MX12X4, NAND2X4, NAND2X1, NAND2X2, NAND2X4, NAND2X6, NAND2X8, NAND2X1, NOR2X1, NOR2X2, NOR2X4, NOR2X6, NOR2X8, OAI2X1, XNOR2X1, XNOR2X4, XOR2X4. To identify the most frequently used cells, the test cases were synthesized with the complete library and the 25 most frequently used cells were selected. The delay constraint was kept tight so that the post-synthesis delay was close to the minimum achievable delay. The enhanced library was generated and the circuit designs optimized as described previously.

This example focused on up to two gate-lengths (nominal and biased) and two threshold voltages. Experiments were performed for the following scenarios: (1) Single-V_{th}, single-
TABLE 4

<table>
<thead>
<tr>
<th>Cell</th>
<th>Leakage Reduction (%)</th>
<th>Delay Penalty (%)</th>
<th>Leakage Reduction (%)</th>
<th>Delay Penalty (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLKINVX1</td>
<td>30.02</td>
<td>5.59</td>
<td>34.12</td>
<td>5.54</td>
</tr>
<tr>
<td>INVX2</td>
<td>30.28</td>
<td>4.70</td>
<td>36.27</td>
<td>6.87</td>
</tr>
<tr>
<td>INVX3</td>
<td>29.45</td>
<td>5.08</td>
<td>33.63</td>
<td>5.12</td>
</tr>
<tr>
<td>INVX4</td>
<td>30.72</td>
<td>5.68</td>
<td>35.67</td>
<td>5.52</td>
</tr>
<tr>
<td>INVX8</td>
<td>30.01</td>
<td>5.36</td>
<td>35.38</td>
<td>6.28</td>
</tr>
<tr>
<td>ORX5</td>
<td>29.97</td>
<td>6.75</td>
<td>35.73</td>
<td>5.25</td>
</tr>
</tbody>
</table>

To assess the maximum impact of biasing, the power-performance envelope obtained by replacing every device in the design by its device-level biased variant was explored. The leakage optimization approach was then applied to selectively bias cells on non-critical paths. Table 5 shows the impact of gate-length biasing on power for single threshold-voltage designs. In particular, Table 5 shows leakage reduction, dynamic power penalty, and total power reduction for the test cases when \( V_{bi} \) was applied without \( V_{bi} \) assignment. The delay penalty constraint was set to 0%, 2.5%, and 5% for each of the test cases. Note that the delay penalty for SVT-SGL was always set to 0% due to the non-viability of \( V_{bi} \) and \( L_{GATE} \) knobs. SVT-DGL was slower than SVT-SGL for delay penalties of 2.5% and 5%.

<table>
<thead>
<tr>
<th>Test</th>
<th>SVT-SGL</th>
<th>SVT-DGL</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SVT-SGL</td>
<td>SVT-DGL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Leakage (mW)</td>
<td>Dynamic (mW)</td>
<td>Total (mW)</td>
</tr>
<tr>
<td>s2324</td>
<td>0.437</td>
<td>0.7047</td>
<td>0.3907</td>
</tr>
<tr>
<td>n13207</td>
<td>0.904</td>
<td>1.3934</td>
<td>0.6296</td>
</tr>
<tr>
<td>c5315</td>
<td>0.556</td>
<td>1.4413</td>
<td>1.5345</td>
</tr>
<tr>
<td>c7552</td>
<td>0.485</td>
<td>1.8328</td>
<td>2.0813</td>
</tr>
<tr>
<td>c6288</td>
<td>0.590</td>
<td>1.8328</td>
<td>2.0813</td>
</tr>
<tr>
<td>aln128</td>
<td>2.363</td>
<td>5.1571</td>
<td>4.4177</td>
</tr>
<tr>
<td>s38417</td>
<td>0.692</td>
<td>4.9381</td>
<td>4.2069</td>
</tr>
<tr>
<td>0.710</td>
<td>4.9381</td>
<td>4.2069</td>
<td>9.1450</td>
</tr>
<tr>
<td>0.727</td>
<td>4.9381</td>
<td>4.2069</td>
<td>9.1450</td>
</tr>
</tbody>
</table>
Table 6 shows the impact of gate-length biasing on power when $V_{th}$ biasing was applied together with the dual-$V_{th}$ approach (i.e., for double threshold-voltage designs). The delay penalty constraint was set to 0%, 2.5%, and 5% for each of the test cases. Tables 5 and 6 also show the delay and CPU runtime consumed.

<table>
<thead>
<tr>
<th>TEST</th>
<th>DVT-SGL</th>
<th>DVT-DGL</th>
<th>REDUCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay (ns)</td>
<td>Leakage (mW)</td>
<td>Dynamic (mW)</td>
</tr>
<tr>
<td>s9234</td>
<td>0.437</td>
<td>0.0894</td>
<td>0.3697</td>
</tr>
<tr>
<td></td>
<td>0.447</td>
<td>0.0914</td>
<td>0.3691</td>
</tr>
<tr>
<td>c5315</td>
<td>0.458</td>
<td>0.0873</td>
<td>0.3676</td>
</tr>
<tr>
<td></td>
<td>0.356</td>
<td>0.0722</td>
<td>0.1429</td>
</tr>
<tr>
<td></td>
<td>0.750</td>
<td>0.2871</td>
<td>1.4193</td>
</tr>
<tr>
<td></td>
<td>0.854</td>
<td>0.2401</td>
<td>1.4119</td>
</tr>
<tr>
<td></td>
<td>0.255</td>
<td>0.6798</td>
<td>1.9332</td>
</tr>
<tr>
<td></td>
<td>0.497</td>
<td>0.4089</td>
<td>1.9114</td>
</tr>
<tr>
<td></td>
<td>0.560</td>
<td>0.3647</td>
<td>1.8904</td>
</tr>
<tr>
<td></td>
<td>0.904</td>
<td>0.1735</td>
<td>0.5930</td>
</tr>
<tr>
<td></td>
<td>0.927</td>
<td>0.1561</td>
<td>0.5920</td>
</tr>
<tr>
<td></td>
<td>0.949</td>
<td>0.1536</td>
<td>0.5919</td>
</tr>
<tr>
<td>c6288</td>
<td>2.111</td>
<td>0.9733</td>
<td>7.7472</td>
</tr>
<tr>
<td></td>
<td>2.171</td>
<td>1.2258</td>
<td>7.5399</td>
</tr>
<tr>
<td></td>
<td>2.224</td>
<td>0.8446</td>
<td>7.4160</td>
</tr>
<tr>
<td>relu128</td>
<td>2.306</td>
<td>0.6847</td>
<td>3.9800</td>
</tr>
<tr>
<td></td>
<td>2.363</td>
<td>0.6515</td>
<td>3.9837</td>
</tr>
<tr>
<td></td>
<td>2.421</td>
<td>0.5965</td>
<td>3.9817</td>
</tr>
<tr>
<td>s38417</td>
<td>0.692</td>
<td>0.5862</td>
<td>3.8324</td>
</tr>
<tr>
<td></td>
<td>0.710</td>
<td>0.5637</td>
<td>3.8309</td>
</tr>
<tr>
<td></td>
<td>0.727</td>
<td>0.5504</td>
<td>3.8306</td>
</tr>
</tbody>
</table>

As shown in Tables 5 and 6, in some examples (e.g., s9234, s13207, s38417), selective gate-length biasing at the circuit level reduced circuit leakage by up to 30% with no delay penalty (i.e., delay penalty constraint set to 0%). The results of Tables 5 and 6 also indicate that the leakage reductions primarily depend on the slack profile of the circuit. If many of the paths have near-zero slack, then the leakage reductions are smaller. As the delay penalty increases, more slack is introduced on paths and larger leakage reductions are seen. The results also show leakage reductions were smaller when a circuit was previously optimized using dual-$V_{th}$ assignment. This is expected because dual-$V_{th}$ assignment consumes slack on non-critical paths reducing the slack available for $L_{Gate}$ optimization. Larger leakage reductions were also observed in sequential circuits. This is most likely because circuit delay is determined primarily by the slowest pipeline stage and the percentage of non-critical paths is typically higher in sequential circuits. In particular, the two circuits for which less leakage reductions were seen (c6288, c5552) have very few non-critical paths and a very small percentage of cells could be biased. For these test cases, even $V_{th}$ assignment does not achieve as much as it does on other test cases. For these test cases, leakage savings due to gate-length biasing are less for DVT than for SVT because fewer cells are left on non-critical paths after $V_{th}$ assignment. A greater fraction of cells are expected to lie on non-critical paths for larger sequential circuits, making the gate-length biasing methodology especially attractive for such circuits.

The leakage models in these experiments did not include gate leakage, which can marginally increase due to gate-length biasing. Gate leakage is composed of gate-length independent and dependent components. The gate-length independent component includes edge direct tunneling ($I_{ed}$), while the gate-length dependent component includes gate-to-channel ($I_{gc}$) and gate-to-body ($I_{gb}$) tunneling. The gate-length independent component, which stems from the gate-drain and gate-source overlap regions, is not affected by gate-length biasing.

To assess the change in gate-length dependent components due to biasing, SPICE simulations were performed to report the gate-to-channel leakage for nominal and biased devices. Since the gate-to-body component is generally two orders of magnitude smaller than the gate-to-channel component, it was therefore excluded from the analysis. The analysis used 90 nm BSIM4 device models from a leading foundry that model all five components of gate leakage described in BSIM v4.4.0.

Table 7 shows the impact of gate-length biasing on sub-threshold leakage and gate tunneling leakage for biased and unbiased, nominal $V_{th}$, 90 nm NMOS and PMOS devices of width at 25°C and 125°C C. The reductions in subthreshold and gate leakage as well as the total leakage reduction are shown. As shown in Table 7, although the subthreshold leakage itself increases significantly with temperature, the percentage reduction in it due to gate-length biasing does not change much. The results of Table 7 indicate total leakage reductions were high even when gate leakage was considered. Based on these results, the increase in gate leakage due to gate-length biasing appears negligible for these cases. Furthermore, since gate-length biasing is a runtime leakage reduction approach, the operating temperature is likely to be hotter than room temperature. At typical operating temperatures, gate leakage is not a major portion of total leakage, since it could be more than five times less than subthreshold leakage. Thus, when the operating temperature is elevated, the reduction in total leakage is approximately equal to the reduction in subthreshold leakage and total leakage reductions similar to the results presented in Tables 5 and 6 are expected.
Gate leakage is predicted to increase with technology scaling. Technologies under 65 nm, however, are likely to adopt high-k gate dielectrics which could significantly reduce gate leakage. If this trend continues, in terms of scalability, subthreshold leakage will likely remain the dominant problem at high operating temperatures. Thus, the gate-length biasing method discussed is likely to be of increasing benefit at smaller technology nodes. Even if gate leakage were to approach subthreshold leakage, which would be a suboptimal device engineering solution point, gate-length biasing results in exponential savings in subthreshold current for a linear penalty in gate leakage, which, while not optimal, may still be preferable. One remote issue is where the minimum gate-length is set at the peak of the "hump" often seen in $V_{th}$ vs. L curves due to the reverse short channel effect and caused by halo implants. If the minimum gate-length is set at the peak of the hump gate leakage may actually worsen with increasing gate-length. Typically, this is not the case, however, since such a minimum gate-length is usually not the best design point. For example, with a typical commercial 90 nm low-$V_{th}$ device, the 80 nm drawn gate-length can be biased by over 10 nm before reaching the start of the hump.

In addition, because vertical electric fields do not increase due to gate-length biasing, another possible benefit of the gate-length biasing methodology is that negative-bias thermal instability (NBTI) is not expected to increase with gate-length biasing. Should NBTI correlate with gate-length, possibly due to weak process dependent variations in NBTI due to channel length, $I_{gate}$ biasing of PMOS transistors may be constrained, since NBTI primarily affects PMOS devices while NMOS devices are more strongly affected by subthreshold leakage.

Manufacturability and process variability are important considerations of the $I_{gate}$ biasing approach. As the gate-length biasing method relies primarily on biasing of drawn gate-length, it is important to correlate this with actual printed gate-length on the wafer. This is important as the bias induced in gate-length is of the same order as the typical critical dimension (CD) tolerances in manufacturing processes. Moreover, experimental observations are consistent with expecting larger gate-lengths to have better printability properties leading to less CD—and hence leakage—variability. To validate the multiple gate-length approach in a post-manufacturing setup, a reticle enhancement technology (RET) and process simulation flow for an example cell master are followed.

In this experiment, model-based optical proximity correction (OPC) is performed on the layout of a generic AND2X6 cell using Calibre v9.3_2.5, using annular optical illumination with $\lambda=248$ nm and NA=0.7. The printed image of the cell is then calculated using dense simulation in Calibre. FIG. 9 is an image of a cell layout 900 of the generic AND2X6 cell with simulated printed gate-lengths 905 for all devices in the cell. Gate-length $I_{gate}$ is measured for every device in the cell, for both biased and unbiased versions. Table 8 shows a comparison of the printed gate-lengths for biased and unbiased versions of the seven NMOS and PMOS devices labeled in FIG. 9. The unbiased ideal gate-length is 130 nm while the biased ideal is 138 nm. As expected, biased and unbiased gate-lengths are highly correlated and track each other well. There are some outliers that may be due to the relative simplicity of the OPC model used. High correlation between printed dimensions of biased and unbiased versions of the cells implies that the benefits of gate-length biasing estimated using drawn dimensions will not be lost after RET application and the manufacturing process.

### TABLE 8

<table>
<thead>
<tr>
<th>Device</th>
<th>PMOS</th>
<th>NMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Unbiased</td>
<td>Biased</td>
</tr>
<tr>
<td>1</td>
<td>128</td>
<td>135</td>
</tr>
<tr>
<td>2</td>
<td>127</td>
<td>131</td>
</tr>
<tr>
<td>3</td>
<td>127</td>
<td>131</td>
</tr>
<tr>
<td>4</td>
<td>124</td>
<td>131</td>
</tr>
<tr>
<td>5</td>
<td>124</td>
<td>131</td>
</tr>
<tr>
<td>6</td>
<td>124</td>
<td>132</td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>135</td>
</tr>
</tbody>
</table>

Another potentially valuable benefit of slightly larger gate-lengths is the possibility of improved printability. Minimum poly spacing is larger than poly gate-length, so that the process window, which is constrained by the minimum resolvable dimension, tends to be larger as gate-length increases even though poly spacing decreases. Table 9 shows the process window improvement with gate-length biasing. In particular, the depths of focus for various values of exposure latitude (ELAT), with the same illumination system as above, for 130 nm and 138 nm lines are shown. The data in Table 9 was obtained using process simulation performed with ProLith v8.1.2, using a CD tolerance of 13 nm.

### TABLE 9

<table>
<thead>
<tr>
<th>Defocus (µm)</th>
<th>ELAT (%) for 130 nm</th>
<th>ELAT (%) for 138 nm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>4.93</td>
<td>5.30</td>
</tr>
<tr>
<td>0.2</td>
<td>6.75</td>
<td>7.26</td>
</tr>
<tr>
<td>0.2</td>
<td>5.69</td>
<td>6.24</td>
</tr>
</tbody>
</table>

A number of sources of variation can cause fluctuations in gate-length, and hence in performance and leakage. Up to 20x variation in leakage has been reported in production microprocessors. For leakage, the reduction in variation post-gate-length biasing is likely to be substantial as the larger gate-length is closer to the "flatter" region of the $V_{th}$ vs. $I_{gate}$ curve. To validate this intuition, the impact of gate-length variation on leakage and performance, both pre- and post-biasing, was studied using a simple worst-case approach. The CD variation budget was assumed to be ±10 nm. The performance and leakage of the test case circuits were measured at...
the worst-case (WC), nominal (NOM), and best-case (BC) process corners, which focus on gate-length variation. This was done for the DVT-DGL approach in which biasing was done along with dual-Vth assignment.

Table 10 shows the reduction in performance (e.g., circuit delay) and leakage power uncertainty with biased gate-length in presence of inter-die variations. The uncertainty spread is specified as a percentage of nominal. The results are given for dual-Vth, with a gate-length bias of 8 nm, and show significant reduction in leakage variability. For the seven test cases, up to a 41% reduction in leakage power uncertainty caused by linewidth variation was observed. In general, such large reductions in uncertainty may lead to substantial improvements in manufacturing yield and product cost, potentially outweighing benefits of alternative leakage control techniques. Note that the corner case analysis only models the inter-die component of variation, which typically constitutes roughly half of the total CD variation.

### TABLE 10

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Spread Reduction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unbiased (DVT-SGL)</td>
<td>Biased (DVT-DGL)</td>
</tr>
<tr>
<td>BC</td>
<td>WC</td>
</tr>
<tr>
<td>s2324</td>
<td>0.504</td>
</tr>
<tr>
<td>c515</td>
<td>0.642</td>
</tr>
<tr>
<td>c7552</td>
<td>0.559</td>
</tr>
<tr>
<td>a1207</td>
<td>1.029</td>
</tr>
<tr>
<td>c2288</td>
<td>2.411</td>
</tr>
<tr>
<td>a1218</td>
<td>2.631</td>
</tr>
<tr>
<td>a38417</td>
<td>0.793</td>
</tr>
<tr>
<td>Leakage (mW)</td>
<td></td>
</tr>
<tr>
<td>s2324</td>
<td>0.6591</td>
</tr>
<tr>
<td>c515</td>
<td>0.2558</td>
</tr>
<tr>
<td>c7552</td>
<td>0.4291</td>
</tr>
<tr>
<td>a1207</td>
<td>0.3603</td>
</tr>
<tr>
<td>c2288</td>
<td>1.8477</td>
</tr>
<tr>
<td>a1218</td>
<td>0.3827</td>
</tr>
<tr>
<td>a38417</td>
<td>0.3526</td>
</tr>
</tbody>
</table>

To assess the impact of both within-die (WID) and die-to-die (DID) components of variation, 10,000 Monte-Carlo simulations with \( \sigma_{WID} = \sigma_{DID} = 3.35 \) nm were run. The variations were assumed to follow a Gaussian distribution with no correlations. The results for three dual-Vth scenarios were compared: unbiased (DVT-SGL), biased (DVT-DGL), and uniformly biased (when gate-lengths of all transistors in the design were biased by 8 nm). Fig. 10 is a graph of the leakage distributions for the unbiased, technology-level selectively biased, and uniformly biased scenarios for a representative test case. As shown in Fig. 10, the distributions exhibit a "left-shift" with the introduction of biased devices in the design. Also, for uniform biasing, all devices are biased and the circuit delay no longer meets timing.

Table 11 presents the leakage power reductions from TLLB over CLLB. The test cases show up to a 50% reduction in leakage power for TLLB over CLLB. Since TLLB primarily biases devices of unbiased cells, it performs well over CLLB particularly when CLLB does not perform well (i.e., when CLLB leaves many cells unbiased). The leakage savings from TLLB, however, come usually at the cost of increased library size. As described above, the library for TLLB can be composed of all 2\(^n\) variants of each n-input cell. For the 25 cells in this test case, the library for TLLB was composed of a total of 920 variants. From the small leakage savings at the cost of significantly increased library size, TLLB is primarily more advantageous for single- and double-input cells that are frequently used.

### TABLE 11

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Delay (ns)</th>
<th>CLLB (mW)</th>
<th>TLLB (mW)</th>
<th>Reduction (%)</th>
<th>CLLB (s)</th>
<th>TLLB (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s2324</td>
<td>0.437</td>
<td>0.0722</td>
<td>0.0712</td>
<td>1.41</td>
<td>1.86</td>
<td>2.75</td>
</tr>
<tr>
<td>c4774</td>
<td>0.447</td>
<td>0.0650</td>
<td>0.0628</td>
<td>3.39</td>
<td>1.89</td>
<td>2.38</td>
</tr>
<tr>
<td>c515</td>
<td>0.458</td>
<td>0.0650</td>
<td>0.0596</td>
<td>2.28</td>
<td>1.83</td>
<td>2.31</td>
</tr>
<tr>
<td>c7552</td>
<td>0.485</td>
<td>0.0655</td>
<td>0.0636</td>
<td>3.42</td>
<td>6.14</td>
<td>13.44</td>
</tr>
<tr>
<td>e3207</td>
<td>0.497</td>
<td>0.0478</td>
<td>0.0438</td>
<td>8.92</td>
<td>10.51</td>
<td>43.22</td>
</tr>
<tr>
<td>e3288</td>
<td>0.509</td>
<td>0.0318</td>
<td>0.0293</td>
<td>6.02</td>
<td>10.55</td>
<td>58.98</td>
</tr>
<tr>
<td>e3343</td>
<td>0.504</td>
<td>0.1247</td>
<td>0.1228</td>
<td>1.58</td>
<td>11.59</td>
<td>17.15</td>
</tr>
<tr>
<td>e38417</td>
<td>0.297</td>
<td>0.1066</td>
<td>0.1055</td>
<td>1.08</td>
<td>11.73</td>
<td>15.62</td>
</tr>
<tr>
<td>e4212</td>
<td>0.194</td>
<td>0.1027</td>
<td>0.1021</td>
<td>0.61</td>
<td>11.76</td>
<td>14.28</td>
</tr>
<tr>
<td>e4288</td>
<td>2.118</td>
<td>1.9517</td>
<td>1.9157</td>
<td>1.84</td>
<td>79.25</td>
<td>305.09</td>
</tr>
<tr>
<td>e4372</td>
<td>2.171</td>
<td>1.1980</td>
<td>1.1555</td>
<td>2.74</td>
<td>29.46</td>
<td>280.56</td>
</tr>
<tr>
<td>e4421</td>
<td>2.224</td>
<td>0.8203</td>
<td>0.8203</td>
<td>0.00</td>
<td>77.28</td>
<td>291.44</td>
</tr>
<tr>
<td>a1218</td>
<td>2.306</td>
<td>0.5184</td>
<td>0.4857</td>
<td>6.31</td>
<td>240.09</td>
<td>544.75</td>
</tr>
<tr>
<td>a1228</td>
<td>2.363</td>
<td>0.4970</td>
<td>0.4942</td>
<td>0.96</td>
<td>262.37</td>
<td>609.13</td>
</tr>
<tr>
<td>a1234</td>
<td>2.421</td>
<td>0.4907</td>
<td>0.4848</td>
<td>6.95</td>
<td>277.99</td>
<td>554.68</td>
</tr>
<tr>
<td>a38417</td>
<td>0.692</td>
<td>0.4838</td>
<td>0.4847</td>
<td>7.76</td>
<td>238.62</td>
<td>746.79</td>
</tr>
<tr>
<td>e4841</td>
<td>0.710</td>
<td>0.4189</td>
<td>0.3992</td>
<td>4.93</td>
<td>238.99</td>
<td>507.62</td>
</tr>
<tr>
<td>e4841</td>
<td>0.727</td>
<td>0.4067</td>
<td>0.3765</td>
<td>7.42</td>
<td>234.94</td>
<td>525.06</td>
</tr>
</tbody>
</table>

Implementations of Gate-Length Biasing Methodology

The general gate-length biasing methodology of the present invention can be applied to a circuit design in many different ways. For example, the mask maker or integrated circuit (IC) fab can implement gate-length biasing via optical proximity correction (OPC). Alternately, the provider of the cell library can offer an enhanced library containing gate-length biased variants of standard cells. As another example, electronic design automation (EDA) tool vendors may implement some or all of gate-length biasing as part of their software design tools (e.g., as part of a design rule checker).

An OPC tool-based implementation of the gate-length biasing methodology allows biasing to occur after a layout has been designed. Although the layout is changed after the circuit is designed, the circuit designer preferably uses the optimizations provided by gate-length biasing up front, for example, by using models based on the gate-length biased designs rather than on the nominal designs. OPC implementation of the gate-length biasing methodology allows optimization beyond that achievable by designers in traditional design flows through direct influence of the OPC process, which designers currently cannot do. For example, a biased gate-length of 93 nm, where the nominal gate-length is 90 nm, will reduce leakage. Unfortunately, the traditional combination of OPC and the process cannot guarantee CD control better than a best case/worst case gate-lengths of, for example, 88/98 nm (the "guardband"). An OPC implementation of the gate-length biasing methodology, aware of an optimization goal such as leakage reduction, will guide the OPC process such that it is extremely unlikely for the printed gate-length to be smaller than the 93 nm goal, if there is sufficient timing slack. One benefit of OPC implementation of the gate-length biasing methodology is that optimization is implemented without perturbing a foundry’s qualified OPC recipe.

Fig. 11 is a flowchart of an exemplary embodiment of an OPC implementation 1100 of the gate-length biasing methodology of the present invention. A nominal layout 1105 for
a circuit and an annotated layout or bias requirements 1110 for the layout are provided to a module 1115 that makes changes to the OPC process to implement gate-length biasing. The nominal layout 1105 describes the circuit before biasing, and information describing the bias is contained in the annotated layout or bias requirements 1110. The module 1115 may implement gate-length biasing in many different ways, as will be further described below. Results from the gate-length biasing module 1115 are provided to an OPC engine 1120, such as Calibre or Proteus, which yields an OPC solution 1125 for the gate-length biased layout.

The driver of the OPC engine 1120 of FIG. 11 preprocesses the runscript and the setup files of the OPC tool to enforce design-specific directives in the application of OPC. These directives are usually localized, e.g., on a per-cell or per-device basis. Access to an OPC implementation, such as implementations 1100, offers post-silicon benefits beyond those achievable by optimizations performed during the design flow (e.g., dual gate-length). The expected post-silicon benefit from using an OPC implementation depends on the design, the effort spent on OPC, and the quality of the OPC tool. Following are six embodiments, in order of expected increasing flow implementation effort, that an OPC implementation of the gate-length biasing methodology may provide without significant loss of timing or yield.

First, an exemplary embodiment of an OPC implementation of the gate-length biasing methodology provides a target CD tolerance “knob.” In this embodiment, the module 1115 can control the target CD tolerance that the OPC engine aims for, in a device- or feature-specific manner. Conventional OPC flows set a uniform, unsigned tolerance over the entire design. By contrast, this embodiment of the OPC gate-length biasing implementation sets signed, device-specific tolerances. For example, the channel length of a setup-critical gate may only be allowed to increase from nominal. Channel lengths of other gates with positive setup slack may be allowed only to increase from nominal, while maintaining timing-correctness, to reduce leakage.

Second, in another embodiment, an OPC implementation of the gate-length biasing methodology provides an OPC error “knob.” In this embodiment, the OPC implementation can control the direction of OPC error. In particular, the OPC implementation can drive OPC for a given gate such that the gate almost always prints with larger channel length than nominal, but still within the tolerance bounds. For example, if the tolerance on a gate-length is set to +/−5 nm, the OPC implementation will drive OPC such that the gate-length is almost always close to +5 nm rather than −5 nm, relative to nominal. The target CD tolerance and OPC error knobs together give an OPC implementation of the gate biasing methodology essentially a continuous range of gate-length variants, applicable on a device-specific basis, without requiring the full design process of cell layout, design rule checking, characterization of performance libraries, etc. For example, implementation of the gate-length biasing methodology at the OPC level avoids violation of across chip linewidth variation (ACLV) tolerances at the design level, which may maximize the optimization possible using OPC.

Third, in another embodiment, an OPC implementation of the gate-length biasing methodology provides an explicit biasing “knob.” In this embodiment, the OPC implementation performs explicit biasing of layout CDs. Bias, error magnitude, and error direction are co-optimized to achieve a much finer degree of control and hence favorable optimization, for example, a larger reduction in leakage.

Fourth, in another embodiment, an OPC implementation of the gate-length biasing methodology provides a sub-resolution assist feature (SRAF) insertion “knob.” With cooperation of the production OPC group, in this embodiment, the OPC implementation allows set up of alternative SRAF insertion recipes such that the implementation can use them to optimize yield. More precisely, the SRAF insertion can be optimized such that the design remains timing-correct and power is reduced through process variation, specifically, through focus variation. This optimization is generally applied in a device-specific manner.

Fifth, in another embodiment, an OPC implementation of the gate-length biasing methodology provides guardband reduction from knowledge of CD tolerance split. In this embodiment, the OPC implementation can optionally take, if available, the CD tolerance split, for example, from analysis of variance (ANOVA), as an input. This enables the OPC implementation to find out how much of the total technology-specific CD tolerance is attributable to OPC errors. As a result, the OPC implementation can reduce the guardband in its biasing optimization because it drives OPC along with biasing. This gives an OPC implementation more leeway in optimization while still being corner-case timing correct.

Sixth, in another embodiment, an OPC implementation of the gate-length biasing methodology uses out-of-focus process models. In this embodiment, an OPC implementation is aware of iso-dense layout pattern effects arising primarily from focus variation in lithography. Given certain abstractions of out-of-focus process models, if available, the OPC implementation uses its focus- or depth-of-focus (DOF)-aware timing and power analyses to further optimize yield. In particular, because the OPC implementation is layout and process aware, gate-lengths may be selectively biased taking iso-dense effects into account.

Alternatively, the gate-length biasing methodology may be implemented by generating an enhanced cell library available to designers, such that a designer can choose from among nominal cells and their biased variants. FIG. 12 is a flowchart of an exemplary embodiment of a cell library implementation 1200 of the gate-length biasing methodology of the present invention. The layout generator 1215 produces a biased version 1220 of a standard cell 1220. A gate-length bias 1205 is determined and provided to the layout generator 1215, which also receives a set of design rules 1210 for a particular technology. The layout generator 1215, for example, Prolific or Cadomus, generates the biased cell(s) 1220. Additionally, manual editing of one or more layouts may also occur. The resulting enhanced library may include standard (i.e., nominal) cell layouts in addition to standard cell layouts to which L\textsubscript{Gate} biasing has been applied (i.e., biased cells). Commercially available layout generators may be augmented with the functionality described previously (e.g., see FIG. 5) in order to automatically generate biased variants for a particular cell.

The gate-length biasing methodology may have an impact on layout design rules depending on the implementation methodology chosen. For example, cell libraries may have to plan for gate-length biasing by increasing contact to poly
spacing, depending on the use model. Making the post-layout to RET flow transparent to the designer may be accomplished, for example, by giving “hints” to the OPC implementation rather than explicit biases or as explicit changes in critical dimensions that are pre-qualified as layout-transparent. These options would generally occur after physical validation, and biases that are not pre-qualified as “safe” by foundry and library teams would generally not be implemented. If, on the other hand, gate-length biasing is done at the cell layout stage, then contact-to-poly spacing may need to be increased to be DRC correct. A mix of the two approaches (i.e., post- and pre-final layout) could potentially achieve even greater improvements because it is possible to bias OPC errors.

Alternatively, the gate-length biasing methodology may be implemented at the EDA tool level, for example, as part of a design rule checking tool. FIG. 13 is a flowchart of an exemplary embodiment of an EDA tool implementation 1300 of the gate-length biasing methodology of the present invention. An annotated layout or gate-length bias 1305 to be implemented for a subset of devices is provided to an EDA tool 1315, such as Calibre, Hercules, Diva, or Assurer. EDA tool 1315 also receives a set of design rules 1310 and generates a biased layout 1320 of the design, for example using the methodologies described previously.

As shown in FIG. 13, an annotated layout or gate-length bias 1305 is provided as part of EDA implementation 1300. Similarly, as shown in FIG. 11, an annotated layout or gate-length bias requirements 1110 are also provided as part of OPC implementations 1100. These bias requirements must be communicated to the OPC or EDA tools. FIG. 14 is a flowchart of an exemplary embodiment of a bias requirement communication methodology 1400, for example, for the OPC or EDA implementations 1100 and 1300 of FIGS. 11 and 13. An optimizer 1425 receives a variety of inputs, including, for example, a set of characterized models 1405, a set of design constraints 1410, a design layout 1415, and a design netlist 1420. The characterized models 1405 may include, for example, a timing library. The design constraints 1410 may include, for example, timing, power, yield, and other constraints. The optimizer 1425 uses the inputs to generate bias requirements 1430 for devices in the layout. The optimizer 1425 may include a nominal gate-length, while others may include a biased gate-length. In this example, the bias requirement communication methodology 1400 creates an annotation layer 1435 of shapes for every distinct bias solution. For example, the annotation layer may include bias of 2 nm overlaps for all devices or cells requiring a 2 nm bias. The result is an annotated layout 1440, which yields the bias implementation 1445 to provide to the OPC and EDA tools.

In alternate embodiments, the gate-length biasing methodology is implemented in computer hardware, firmware, software, and/or combinations thereof. Apparatus of the invention can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor; and method steps of the invention can be performed by a programmable processor executing a program of instructions to perform functions of the invention by operating on input data and generating output. The invention can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. Each computer program can be implemented in a high-level procedural or object-oriented programming language, or in an assembly or machine language if desired; and in any case, the language can be a compiled or interpreted language. Suitable processors include, by way of example, both general and special purpose microprocessors. Generally, a processor will receive instructions and data from a read-only memory and/or a random access memory. Generally, a computer will include one or more mass storage devices for storing data files; such devices include magnetic disks, such as internal hard disks and removable disks; magneto-optical disks; and optical disks. Storage devices suitable for tangibly embodying computer program instructions and data include all forms of nonvolatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM disks. Any of the foregoing can be supplemented by, or incorporated in, ASICs (application-specific integrated circuits) and other forms of hardware.

Although the detailed description contains many specifics, these should not be construed as limiting the scope of the invention but merely as illustrating different examples and aspects of the invention. It should be appreciated that the scope of the invention includes other embodiments not discussed in detail above. For example, while the descriptions above were given primarily in the context of increasing gate-lengths in order to reduce leakage power, neither increasing gate-length nor reducing leakage power is a requirement. For example, gate-length biasing can also be used to address other types of power consumption, manufacturability concerns and timing characteristics. In one application, gate-lengths on important timing paths may be reduced in order to reduce timing delays and/or allow an increase in operating frequency (e.g., clock frequency) for the chip.

As another example, positive and negative gate-length biasing can be combined to improve the clock frequency of a chip while still conserving (leakage) power. For example, the devices on the critical timing path can receive negative gate-length biases, thus achieving a speedup at the cost of increased leakage power. At the same time, other devices of the circuit can be slowed down (i.e., positive gate-length biases) to take advantage of the increased timing slack, resulting in an overall reduction in leakage power that more than compensates for the additional leakage power from the negatively-biased devices.

Various other modifications, changes and variations which will be apparent to those skilled in the art may be made in the arrangement, operation and details of the method and apparatus of the present invention disclosed herein without departing from the spirit and scope of the invention as defined in the appended claims. Therefore, the scope of the invention should be determined by the appended claims and their legal equivalents. Furthermore, no element, component, or method step is intended to be dedicated to the public regardless of whether the element, component, or method step is explicitly recited in the claims.

The invention claimed is:

1. A gate-length biasing method for modifying a nominal cell for an integrated digital circuit, the nominal cell containing one or more transistors, the method to be executed by a processor comprising:
   (a) selecting a trial set of one or more transistors in the nominal cell, each selected transistor having a nominal gate-length, the one or more transistors having gate widths defined as part of the nominal cell;
   (b) determining small bias lengths for the selected trial set of transistors, the small bias lengths all less than a pre-defined fraction of the nominal gate-length;
(e) adjusting the gate-lengths of the selected trial set of transistors by the small bias lengths to create a trial biased cell;
(d) comparing the trial biased cell to a current best biased cell with respect to a predefined goal including a tradeoff between reducing a leakage power and reducing an impact on timing delays for the integrated digital circuit; and
(e) updating the current best biased cell based on the comparison;
wherein gate-length biasing is not width-sizing of the gate widths;
wherein gate-length biasing is performed on the one or more transistors of the nominal cell post layout, for the integrated digital circuit;
wherein the nominal cell that has been gate length biased and the nominal cell that has not been gate length biased are pin-compatible and layout swapable.
2. The method of claim 1, wherein the integrated digital circuit is defined by a layout that includes a plurality of cells, the plurality of cells include nominal cells that have been gate-length biased and nominal cells that have not been gate-length biased.
3. The method of claim 1, wherein the adjusting the gate-lengths of the selected transistors comprises:
increasing the gate-lengths of each of the selected transistors by the small bias lengths.
4. The method of claim 1, wherein the adjusting the gate-lengths of the selected transistors comprises:
dereducing the gate-length of at least one of the selected transistors by one of the small bias lengths.
5. The method of claim 4, wherein the decreasing the gate-lengths of the transistors is based on reducing a timing delay for the integrated digital circuit.
6. The method of claim 4, wherein the decreasing the gate-lengths of the transistors is based on increasing an operating frequency for the integrated digital circuit.
7. The method of claim 1, wherein the selecting transistors and/or determining small bias lengths are based on reducing a leakage power for the biased cell.
8. The method of claim 7, wherein the selecting transistors and/or determining small bias lengths are further based on reducing an impact on timing delays for the integrated digital circuit.
9. The method of claim 8, wherein the selecting transistors and/or determining small bias lengths are required to have zero impact on timing delays for the integrated digital circuit.
10. The method of claim 1, wherein the selecting transistors and/or determining small bias lengths are based on reducing impact on timing delays for the integrated digital circuit.

11. The method of claim 10, wherein the selecting transistors and/or determining small bias lengths are required to have zero impact on timing delays for the integrated digital circuit.
12. The method of claim 1, wherein the selecting transistors and/or determining small bias lengths are based on reducing a total power for the integrated digital circuit.
13. The method of claim 1, wherein the selecting transistors and/or determining small bias lengths are based on reducing a static power for the integrated digital circuit.
14. The method of claim 1, wherein the selecting transistors and/or determining small bias lengths are based on reducing a dynamic power for the integrated digital circuit.
15. The method of claim 1, wherein the selecting transistors and/or determining bias lengths are based on reducing impact on timing delays for the integrated digital circuit taking into account timing asymmetries in timing arcs through the cell.
16. The method of claim 15, wherein reducing impact on timing delays for the integrated digital circuit takes into account timing asymmetries in rise and fall delays for timing arcs through the cell.
17. The method of claim 15, wherein reducing impact on timing delays for the integrated digital circuit takes into account timing asymmetries in input to output delays for the cell.
18. The method of claim 1, wherein the selecting transistors to bias comprises:
selecting transistors that are not in a critical timing path for the integrated digital circuit.
19. The method of claim 1, wherein the small bias lengths are all less than 10% of the nominal gate-lengths.
20. The method of claim 1, wherein the (a) through (e) are performed iteratively based on the predefined goal.
21. A gate-length biasing method for modifying a nominal cell for an integrated digital circuit, the nominal cell containing one or more transistors, the method to be executed by a processor comprising:
(a) selecting a trial set of one or more transistors in the nominal cell, each selected transistor having a nominal gate-length, the one or more transistors having gate widths defined as part of the nominal cell;
(b) determining small bias lengths for the selected trial set of transistors, the small bias lengths less than a predefined fraction of the nominal gate-length;
(c) adjusting the gate-lengths of the selected trial set of transistors by the small bias lengths to create a trial biased cell;
(d) comparing the trial biased cell to a current best biased cell with respect to a predefined goal including a tradeoff between reducing a leakage power and reducing an impact on timing delays for the integrated digital circuit; and
(e) updating the current best biased cell based on the comparison;
wherein gate-length biasing is not width-sizing of the gate widths;
wherein gate-length biasing is performed on the one or more transistors of the nominal cell post layout, for the integrated digital circuit;
wherein the adjusting the gate-lengths of the selected transistors comprises:
dereducing the gate-lengths of a set of the selected transistors by the small bias lengths,
wherein the increasing of the gate-lengths is performed without increasing a contact-to-gate spacing;
wherein the contact-to-gate spacing is a distance between gates of the transistors and contacts to source/drain defined at either side of the gates.
22. The method of claim 21, wherein the small bias length is less than a pitch of a layout grid of the nominal cell.
23. The method of claim 21, wherein the increasing the gate-lengths of the transistors is based on reducing a leakage power for the gate-length biased cell.
24. The method of claim 21, wherein the increasing the gate-lengths of the transistors is based on reducing a leakage power variability for the integrated digital circuit.
25. A gate-length biasing method for modifying a nominal cell for an integrated digital circuit, the nominal cell containing one or more transistors, the method to be executed by a processor comprising:
(a) selecting a trial set of one or more transistors in the nominal cell, each selected transistor having a nominal gate-length, the one or more transistors having gate widths defined as part of the nominal cell;
(b) determining small bias lengths for the selected trial set of transistors, the small bias lengths all less than a predefined fraction of the nominal gate-length;
(c) adjusting the gate-lengths of the selected trial set of transistors by the small bias lengths to create a trial biased cell;
(d) comparing the trial biased cell to a current best biased cell with respect to a predefined goal including a tradeoff between reducing a leakage power and reducing an impact on timing delays for the integrated digital circuit; and
(e) updating the current best biased cell based on the comparison;

wherein gate-length biasing is not width-sizing of the gate widths;
wherein gate-length biasing is performed on the one or more transistors of the nominal cell post layout, for the integrated digital circuit;

wherein in the integrated digital circuit is defined by a layout that includes a plurality of cells, the plurality of cells include nominal cells that have been gate-length biased and nominal cells that have not been gate-length biased, and wherein both nominal cells that have been gate-length biased and nominal cells that have not been gate-length biased are pin-compatible;

wherein being pin-compatible is associated with maintaining a same contact-to-gate spacing for both nominal cells that have been gate length biased and nominal cells that have not been gate length biased.

26. A gate-length biasing method for modifying a nominal cell for an integrated digital circuit, the nominal cell containing one or more transistors, the method to be executed by a processor comprising:

(a) selecting a trial set of one or more transistors in the nominal cell, each selected transistor having a nominal gate-length, the one or more transistors having gate widths defined as part of the nominal cell;
(b) determining small bias lengths for the selected trial set of transistors, the small bias lengths all less than a predefined fraction of the nominal gate-length;
(c) adjusting the gate-lengths of the selected trial set of transistors by the small bias lengths to create a trial biased cell;

(d) comparing the trial biased cell to a current best biased cell with respect to a predefined goal including a tradeoff between reducing a leakage power and reducing an impact on timing delays for the integrated digital circuit; and
(e) updating the current best biased cell based on the comparison;

wherein gate-length biasing is not width-sizing of the gate widths;
wherein gate-length biasing is performed on the one or more transistors of the nominal cell post layout, for the integrated digital circuit;

wherein gate-length biasing, performed post layout, requires that the gate-length biasing occur after the nominal cell has been included in a layout for manufacturing at least part of the integrated digital circuit, and the nominal cell includes the one or more transistors having the defined gate widths.

27. A gate-length biasing method for modifying a nominal cell for an integrated digital circuit, the nominal cell containing one or more transistors, the method to be executed by a processor comprising:

(a) selecting a trial set of one or more transistors in the nominal cell, each selected transistor having a nominal gate-length, the one or more transistors having gate widths defined as part of the nominal cell;
(b) determining small bias lengths for the selected trial set of transistors, the small bias lengths all less than a predefined fraction of the nominal gate-length;
(c) adjusting the gate-lengths of the selected trial set of transistors by the small bias lengths to create a trial biased cell;
(d) comparing the trial biased cell to a current best biased cell with respect to a predefined goal including a tradeoff between reducing a leakage power and reducing an impact on timing delays for the integrated digital circuit; and
(e) updating the current best biased cell based on the comparison;

wherein gate-length biasing is not width-sizing of the gate widths;

wherein gate-length biasing is performed on the one or more transistors of the nominal cell post layout, for the integrated digital circuit;

wherein gate-length biasing, performed post layout, requires that the gate-length biasing occur after the nominal cell has been included in a layout for manufacturing at least part of the integrated digital circuit, and the nominal cell includes the one or more transistors having the defined gate widths.

* * * * *