Journal of Circuits, Systems, and Computers
Vol. 19, No. 7 (2010) 1449–1464

DOI: 10.1142/S021812661000675X

World Scientific
www.worldscientific.com

LOOKUP TABLE-BASED ADAPTIVE BODY BIASING OF MULTIPLE MACROS FOR PROCESS VARIATION COMPENSATION AND LOW LEAKAGE*

BYUNGHEE CHOI and YOUNGSOO SHIN
Department of Electrical Engineering, KAIST,
Daejeon 305-701, Republic of Korea

Received 2 February 2008
Revised 19 April 2010

A reduced supply voltage must be accompanied by a reduced threshold voltage, which makes this approach to power saving susceptible to process variation in transistor parameters, as well as resulting in increased subthreshold leakage. While adaptive body biasing is efficient for both compensating process variation and suppressing leakage current, it suffers from a large overhead of control circuit. Most body biasing circuits target an entire chip, which causes excessive leakage of some blocks and misses the chance of fine grain control. We propose a new adaptive body biasing scheme, based on a lookup table for independent control of multiple functional blocks on a chip, which controls leakage and also compensates for process variation at the block level. An adaptive body bias is applied to blocks in active mode and a large reverse body bias is applied to blocks in standby mode. This is achieved by a central body bias controller, which has a low overhead in terms of area, delay, and power consumption. The problem of optimizing the required set of bias voltages is formulated and solved. A design methodology for semicustom design using standard-cell elements is developed and verified with benchmark circuits.

Keywords:

1. Introduction

The supply voltage of CMOS circuits keeps being reduced in step with technology scaling so as to manage their power consumption. This increases the circuit delay, and the threshold voltage is reduced to compensate. This leads to an exponential increase in subthreshold leakage, which is the main component of standby power consumption.[1] A reduced supply voltage has another implication in the design of circuits: process variations due to transistor parameters such as channel length and threshold voltage have a higher impact on speed and leakage current.[2] The spread in frequency and leakage distribution due to process variation can cause a 20 × variation in chip leakage and a 30% variation in chip frequency.[3] This wide variation in

*This paper was recommended by Regional Editor Krishna Shenai.

frequency and leakage affects the yield, since chips with excessive leakage and chips at too low a frequency have to be discarded.

In order to accommodate the process variation and to reduce the leakage current, body bias circuits are used to control body (or substrate) bias dynamically. The threshold voltage of an MOS transistor is a function of its body to source potential V_{sb}:

$$V_{\text{th}} = V_{\text{th0}} + \gamma(\sqrt{\psi + V_{sb}} - \sqrt{\psi}) \,, \tag{1}$$

where V_{th0} is the threshold voltage when the source is at the body potential (zero-bias threshold voltage), γ is the body effect coefficient, and ψ is the surface potential at threshold. The threshold voltage V_{th} can be modulated to achieve higher performance by a forward body bias (FBB), i.e., $V_{sb} < 0$. The switching power can be reduced by means of FBB, since it allows the same frequency to be achieved at a lower supply voltage.[4] A reverse body bias (RBB), i.e. $V_{sb} > 0$, uses a higher threshold voltage and further reduces standby leakage current: the leakage current of a circuit is monitored and a feedback controller adjusts the body voltage until the predetermined leakage target is met.[5] It is possible to utilize FBB and RBB together, and this is called adaptive body bias (ABB), which has been shown to be very effective for minimizing the impact of both die-to-die and within-die parameter variations on frequency and active leakage power.[6]

Although body biasing is efficient, the biasing circuits represent a large overhead in terms of area, power consumption, and the delay required to adjust the body bias. Thus, most circuit techniques for body biasing are targeted to an entire chip or several functional blocks, where the overhead of the biasing circuits is acceptable because of the scale of the circuits that they control, but the downside is that blocks are not controlled independently. Figure 1(a) illustrates the conventional chip-level body biasing, where two functional blocks, which we call *macros*, are to be controlled by the body biasing circuit. Assume that the first macro has a negative timing slack of -5 while the second macro has a positive slack of 5. We need to apply FBB (through the body biasing circuit) so that the slack of the first macro becomes 0, as shown in Fig. 1(b). The second macro, however, receives the same FBB, which causes

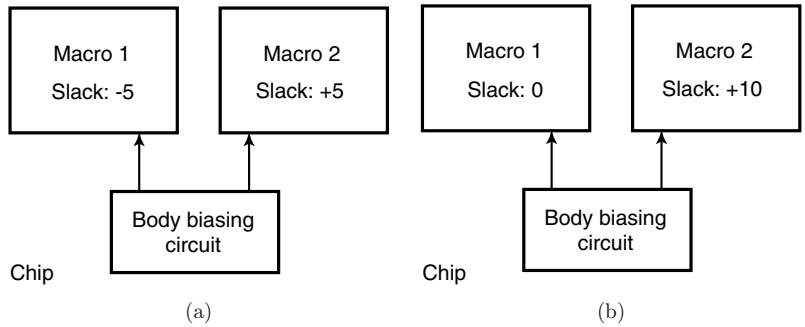


Fig. 1. Conventional chip-level body biasing: (a) before body biasing and (b) after body biasing.

September 21, 2010 3:00:11pm WSPC/123-JCSC 00675 ISSN: 0218-1266 FA1

its slack to increase, say up to 10. This results in excessive leakage current from the second macro, which on the other hand is unavoidable since both macros share the same body biasing circuit. Furthermore, when the first macro is idle, applying strong RBB can reduce its leakage, which is not possible if the second macro is active.

Therefore, when we have multiple, especially many, macros on the same chip, chip-level body biasing is not efficient in terms of compensating process variation and suppressing leakage current. In order to achieve fine-grain control of leakage and to compensate for intra-die process variation, it is important to be able to control several macros on the same chip independently, which is only possible if biasing circuits with very low overheads can be used.

In this paper, we propose a new ABB scheme in which multiple macros are controlled independently, depending on their mode of operation. ABB is used to compensate for the process variation in the performance of a macro when it is in active mode and RBB is used to reduce its leakage current in standby mode. The salient feature of the proposed scheme is a *lookup table* that holds a binary code for each macro corresponding to its active mode body bias voltage. The binary code is fetched by a power management unit, and then the corresponding body bias voltage is generated by the controller. The code length and the set of bias voltages must be designed carefully to ensure maximum process compensation while keeping the overhead of bias controller tolerable; we show how this problem can be formulated and solved. We also propose a design methodology for applying our scheme to designs based on standard-cell elements.

The remainder of this paper is organized as follows: in the next section, we describe our lookup table-based adaptive body bias scheme, and cover the overall operation of the circuit, the body bias generator, and various issues that arise in implementing the proposed scheme using standard-cell elements. In Sec. 3, the optimization of body bias voltages is discussed. Experimental results are presented in Sec. 4, and we draw conclusions in Sec. 5.

2. Lookup Table-Based Adaptive Body Biasing

2.1. *Overall operation*

Figure 2 outlines the way in which a lookup table can be used for adaptive body biasing. Suppose we have n independent macro functional blocks (macros for brevity) on a chip. A power management unit (PMU) detects[a] a state change of a macro. When a macro changes its state from standby to active mode, the PMU fetches a codeword from the lookup table. The codeword is input to the adaptive body bias

[a]There are many alternative power management interfaces and the full range of possibilities is beyond the scope of this paper. For example, a macro may have internal logic that detects its own standby state and sends a standby request to the PMU. The PMU may then acknowledge the request, depending on the configuration of the whole system. The same logic can be used to detect the wakeup condition and to interface with the PMU to achieve a return to active mode.

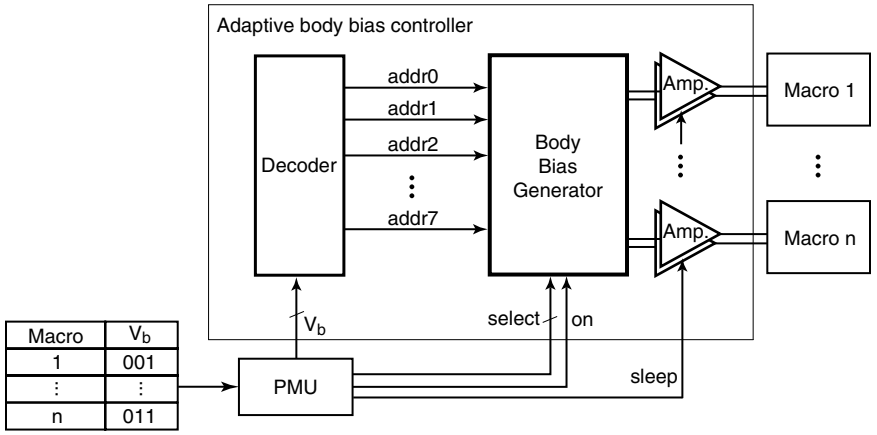


Fig. 2. Adaptive body biasing using a lookup table.

controller, which is marked as a block in Fig. 2. The controller then generates a pair of active-mode body bias voltages for the macro (one for nMOS and the other for pMOS transistors). When a macro changes its state from active to standby mode, a predetermined large reverse body bias is directly generated by the controller without using the lookup table.

The lookup table holds a codeword for each macro corresponding to the active mode body bias of that macro. The number of bits in each codeword determines the number of available bias voltages for compensating for process variations. Obviously, more bias voltages allow finer compensation for compensating process variation, but more bits means a larger lookup table and a larger overhead for the adaptive body bias controller. Thus, the length of the codeword needs to be determined carefully; this topic will be discussed in more detail in the next section. The values of the lookup table entries are determined and programmed after fabrication. The delay of each macro (in particular, the delay of critical path replica) is monitored while trying each codeword one by one (i.e., applying different body bias), and the code that allows the macro to meet its delay target is selected. The selected code then can be programmed in the lookup table, such as programmable ROM.

The proposed architecture allows multiple macros, each of which operates in more than one modes, to be controlled independently. In active mode, either FBB or RBB is used for process compensation, depending on the process variation of the macro. In standby mode, a large RBB is used to suppress the leakage current.

2.2. *Body bias controller*

Once the PMU has fetched a codeword for a macro, the decoder shown in Fig. 2 generates an address which has one bit at 1 for each combination of values in the codeword. This address is then used by the body bias generator to generate the body biases.

September 21, 2010 3:00:12pm WSPC/123-JCSC 00675 ISSN: 0218-1266 FA1

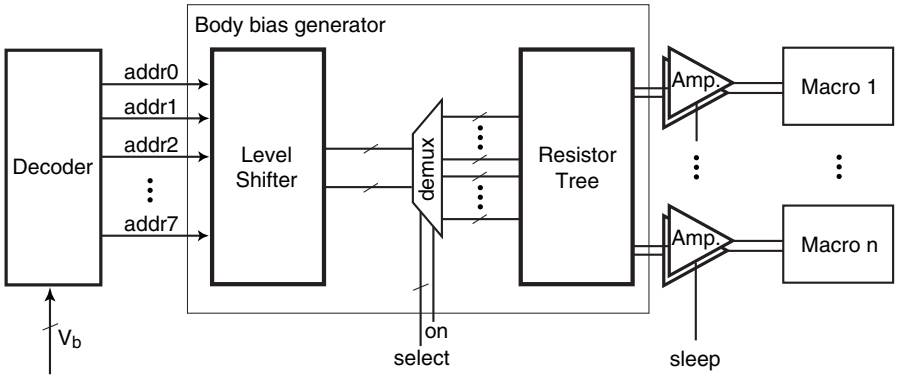


Fig. 3. Body bias generator.

Figure 3 shows the body bias generator in detail. It consists of a level shifter, a demultiplexer (DEMUX), and a resistor tree. The resistor tree requires voltages of V_{ddh}(higher than V_{dd}) and V_{ddl}(lower than V_{ss}), instead of V_{dd} and V_{ss}. A level shifter is employed to convert the address from the decoder, which uses V_{dd} as logic `1` and V_{ss} as logic `0`, to a new pair of addresses: one for the pMOS switches in the resistor tree and the other for the nMOS switches. The address for the pMOS switches uses the levels V_{ddh} and V_{ss}; the address for the nMOS switches uses V_{dd} and V_{ddl}. The details will be explained in the next subsection.

After generation, the addresses are routed to the resistor tree through the DEMUX. Note that the resistor tree requires a pair of addresses for each macro, and so there are $2n$ addresses between the DEMUX and the resistor tree. The `select` signal, which is $\lceil \log_2 n \rceil$ bits wide, selects the macro to which level-shifted addresses are routed. The `on` signal, which turns on the DEMUX, is important in the operation of the body bias generator. Normally the DEMUX is turned off by de-asserting the `on` signal, decoupling the resistor tree from the level shifter. When the PMU wants to apply the active body bias to a particular macro, the corresponding values appear on the `select` lines. However, it takes time for the decoder and the level shifter to generate the required signals. Thus, the `on` signal must only be asserted after the delay for decoding and level shifting, so that the selected macro receives the correctly decoded and level-shifted addresses. Once the DEMUX has transferred the required addresses, `on` is de-asserted again, turning off the DEMUX.

2.2.1. *Resistor tree*

In order to generate the active-mode body bias voltage, we use a resistor tree, as shown in Fig. 4. This tree consists of N equal transistors connected in series, which divide the potential difference between V_{ddh} and V_{ddl} into N intermediate potentials. A set of predetermined bias voltages can then be obtained by connecting switches

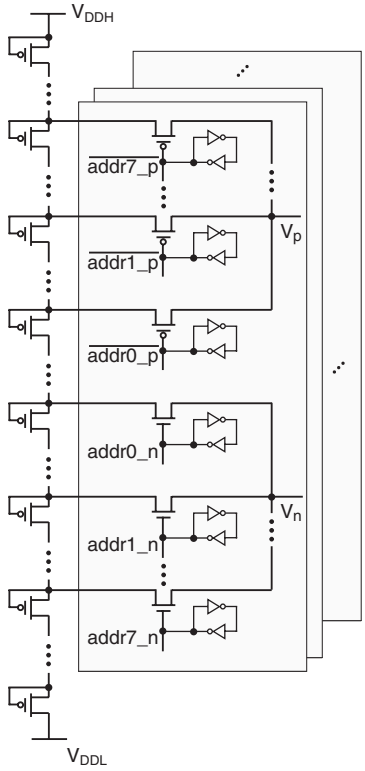


Fig. 4. Resistor tree for generating active mode body bias voltages.

where needed. The choice of bias voltages and the number of transistors in the resistor tree will be discussed in the next section.

We use a pMOS switch to obtain the pMOS body bias voltage V_p, since the bias voltage for the pMOS body is around V_{dd}, although it will be higher than V_{dd} for reverse body biasing. We therefore apply V_{ddh} to the gate of any pMOS switches that need to be turned off. Similarly, an nMOS switch is used to produce the nMOS body bias voltage, and we apply V_{ddl} to the gates of switches that are to be turned off. For instance, suppose that `macro 1` in Fig. 2 makes the transition from standby to active mode. The PMU fetches the codeword `001`, which is then decoded to yield `01000000`. The logic level is shifted (see Fig. 3) so that, if the address is to be used for pMOS switches (see Fig. 4), `addr1` corresponds to V_{ddh} while the remaining bits correspond to V_{ss}; but if the address is destined for nMOS switches, `addr1` corresponds to V_{dd} while the remaining bits correspond to V_{ddl}.

The body of each pMOS device in the resistor tree is biased to its own source, meaning that the n-well of each device needs to be isolated. This represents an area overhead, but frees the pMOS devices from the body effect. It also guarantees the

September 21, 2010 3:00:13pm WSPC/123-JCSC 00675 ISSN: 0218-1266 FA1

stability of bias voltages generated by the resistor tree, even if V_{th} changes. In other words, the bias voltages are determined only by the number of serially connected pMOS devices, and are not affected by process variations. This is an important property of a body bias controller.

Since we use the same resistor tree to bias all n macros, each macro uses a dedicated switch, as shown in Fig. 4. When the resistor tree is used to bias one of the macros, the status of the switches for all the other macros must be maintained, and this is achieved by latches at the gate input of all switches.

2.2.2. *Amplifier*

The pMOS devices in the resistor tree operate in the subthreshold region. Therefore, the current that they draw is the subthreshold leakage current, which is very small and inadequate to drive the body of a macro. An amplifier, as shown in Fig. 5, is therefore required to boost the weak current from the resistor tree for nMOS body biasing.[b]

A simple two-stage amplifier is used: the first gain stage is a differential-input single-ended output stage, and the second is a common-source stage. Figure 6(a) shows the control signals applied to the amplifier for the transition from active to standby mode. The `amp_on` signal is de-asserted first, which turns off the transistors

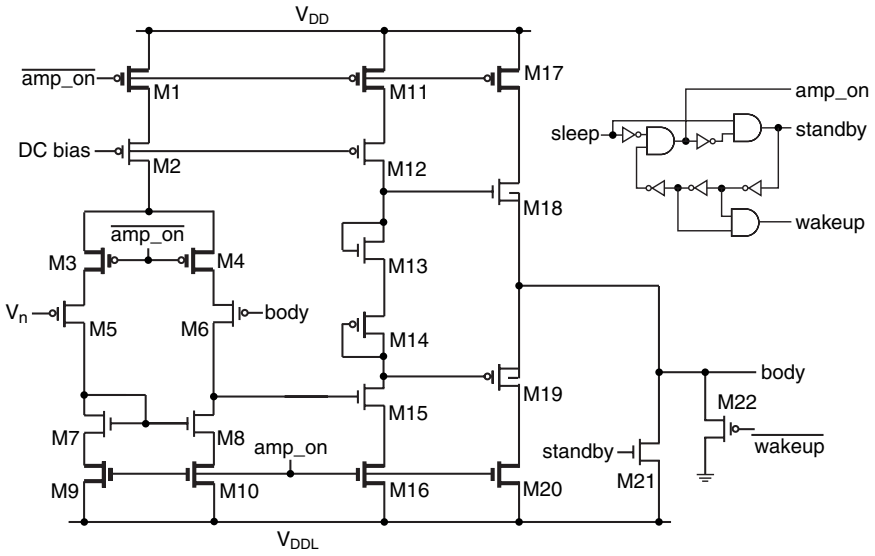


Fig. 5. Body bias amplifier.

[b]A similar amplifier is used for pMOS body biasing. The polarity of all transistors is inverted and the control signals (`amp_on`, `standby`, and `wakeup`) are complemented. The supplies are V_{ss} and V_{ddh} instead of V_{dd} and V_{ddl}.

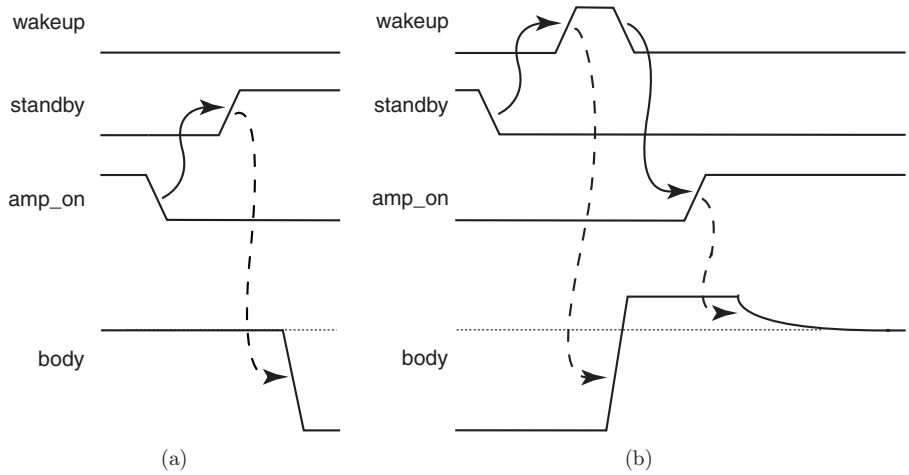


Fig. 6. Mode transition: (a) active to standby and (b) standby to active.

highlighted in Fig. 5, so as to reduce the overall power consumption of the amplifier during standby mode. This is followed by asserting the `standby` signal, which turns on M21. This transistor then applies the predetermined large reverse body bias (V_{ddl}) to the bodies of the nMOS devices in a macro. Note that M22 remains turned off by the de-asserted `wakeup` signal. The presence of M3 and M4 is important for the safe operation of the amplifier. Since the gate of M6 is connected to the bodies of the nMOS devices in a macro, a large reverse body bias applied through M21 might reduce V_n, the output of the amplifier, at the gate input to M5. This would affect the potential of the resistor tree in the opposite direction, which might in turn affect the body bias of other macros in active mode, since the one resistor tree is shared among all macros. This potential problem can be avoided by turning off M3 and M4, which cuts the path from M6 to M5.

Figure 6(b) shows the transition from standby to active mode. The `standby` signal is de-asserted, which turns off M21. M22 is then turned on by `wakeup`, and the body potential of nMOS devices quickly goes up from V_{ddl} to V_{ss}. Once the body is stable at V_{ss}, M22 is turned off, and the amplifier is subsequently turned on by the `amp_on` signal. The bodies of the nMOS devices gradually settle down to the potential that is required to compensate for the process variation of their macro. The presence of M22 is also important in the transition from standby to active mode. If we switch directly from a large reverse body bias to an active-mode body bias, which is around V_{ss} for nMOS devices, the potential at the gate of M6 can affect the gate potential of M5. We alleviate this problem by using M22 to boost the body potential from V_{ddl} to V_{ss}, and then turn on the amplifier by means of the `amp_on` signal.

The circuit that generates the control signals (`wakeup`, `standby` and `amp_on`) from the `sleep` signal received from the PMU is also shown in Fig. 5.

September 21, 2010 3:00:15pm WSPC/123-JCSC 00675 ISSN: 0218-1266 FA1

2.3. *Design methodology for cell-based semicustom design*

In order to validate the proposed lookup table-based adaptive body biasing in semicustom designs using standard-cell elements, we developed a custom cell library and associated layout methodology. We took 21 cells (four inverters, three 2-input NAND gates, one 3-input NAND gate, one 4-input NAND gate, one 2-input NOR gate, one tri-state buffer, six flip-flops, and four latches) from commercial 180-nm cell library, removed the body contacts, optimized the layout, and then re-characterized the devices using SPICE simulation. By optimizing the layout, we were able to reduce the height of each cell by 11%, which achieves a saving of area.

Our layout methodology is shown in Fig. 7, where double-back layout pattern is employed to reduce area. A new tap cell[7] was designed to deliver the body biases, supplied by the adaptive body bias controller, to the n-well and p-well, i.e., V_p and V_n coming from the controller (see Fig. 5) are connected to V_p and V_n terminals of the tap cells, respectively, which then supply potentials to n- and p-wells, as shown in Fig. 7. The tap cells are inserted in a regular fashion; they are fixed in their locations, and then the logic cells are placed and routed automatically. The columns of the tap cells are separated by 50 μm,[7] which are determined by well impedance. The layout of a tap cell and of a 2-input NAND gate are also shown in the figure. Figure 8 compares two layouts of the same circuit: the layout using conventional standard-cells is shown

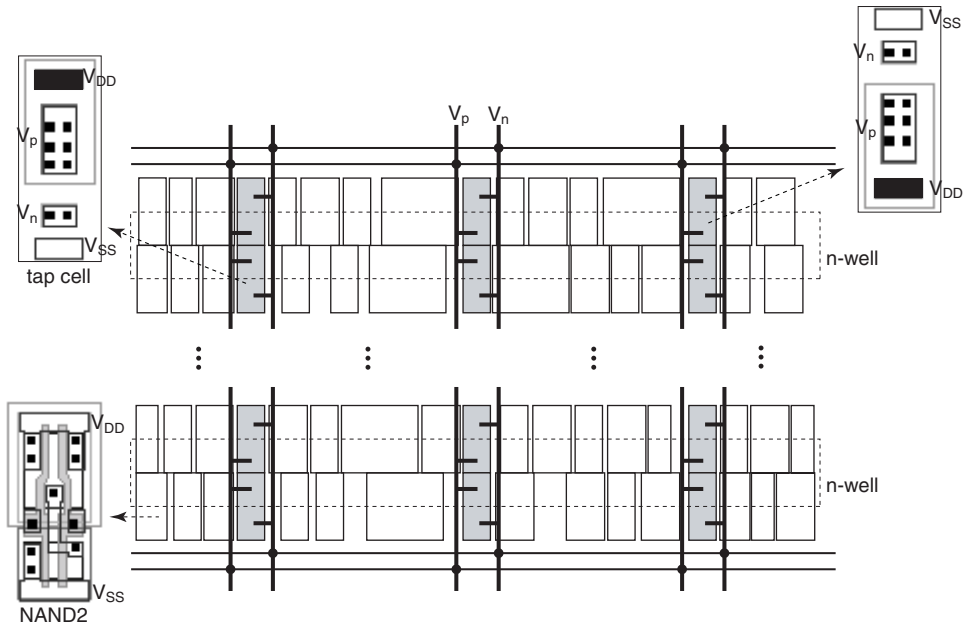
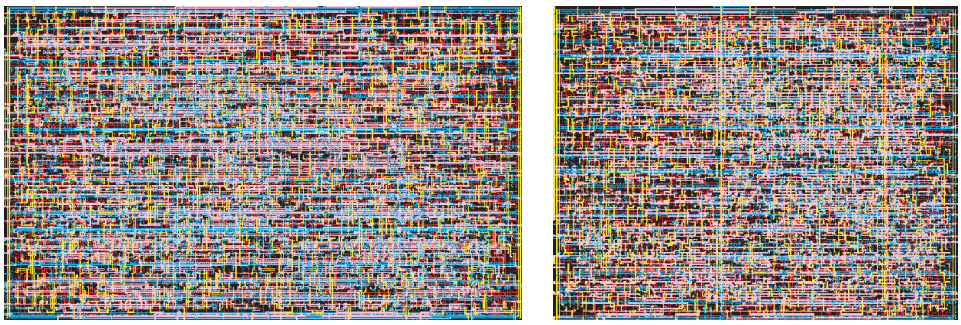


Fig. 7. Layout methodology.



(a) (b)

Fig. 8. Comparison of layouts using (a) conventional standard-cells and (b) our custom cells without body taps.

in Fig. 8(a) while the layout using our custom cells is shown in Fig. 8(b), which shows clear advantage in terms of area.

3. Optimization of Bias Voltages

The resistor tree generates a fixed number of predetermined body bias voltages. Therefore, if a certain macro needs a bias voltage that is not provided by the resistor tree, the voltage that is closest to the required one (but larger than that in case of nMOS, or smaller than that in case of pMOS) has to be used. For example, suppose a macro is too slow than expected due to process variations, but compensation can be achieved by applying 12 mV to the body of its nMOS devices. Suppose also that the resistor tree generates 10 mV and 20 mV among its bias voltages. We need to apply 20 mV since 10 mV would not meet the delay target of the macro because it would still be too slow. But at 20 mV, the macro is too leaky and unnecessarily fast. Thus we see that it is important to determine a set of body bias voltages such that the overall excessive active leakage can be minimized.

We model the change in threshold voltage due to process variations using random variable x that follows a normal distribution[3]:

$$x \sim N(\mu, \sigma^2)\,, \tag{2}$$

where μ is the mean (i.e., the threshold voltage corresponding to a perfect process) and σ denotes the standard deviation. Let $x_1, x_2, \ldots, x_n$ be n zero-bias threshold voltages and $V_1, V_2, \ldots, V_n$ be n body bias voltages, such that the threshold voltage of nMOS devices with x_i become μ when V_i is applied to the substrate. This implies that V_i is negative (reverse body bias) if $x_i < \mu$ (device is too fast), and positive (forward body bias) otherwise. A similar assumption can be made to pMOS devices; we will focus on nMOS in this section.

September 21, 2010 3:00:21pm WSPC/123-JCSC 00675 ISSN: 0218-1266 FA1

Note that V_i is a function of x_i, i.e., for any $x_i \neq \mu$, we can determine how much body bias we have to apply so that the threshold voltage becomes μ. Therefore, once we determine the value of x_is (zero-bias threshold voltages that can be perfectly compensated), we can accordingly determine V_is, which are then used to design resistor tree (see Fig. 4). The objective is to minimize the expected value of excessive (active-mode) leakage when x (which is not one of x_is) is compensated.

The subthreshold leakage of a turned-off nMOS device in a CMOS inverter can be approximated by

$$I_{\text{sub}} \propto e^{-V_{\text{th}}/S}\,, \tag{3}$$

where S denotes the subthreshold slope. As a measure of over-compensation (i.e., the amount of excessive leakage), we introduce the quantity

$$Q = \int_{-\infty}^{+\infty} [e^{-V_{\text{th}}(x)/S} - e^{-\mu/S}] f_x(x) dx\,, \tag{4}$$

where $V_{\text{th}}(x)$ is a threshold voltage V_{th} after one of V_is is applied to compensate zero-bias threshold voltage x, and $f_x(x)$, as shown in Fig. 9, corresponds to the probability density function of x, which follows Gaussian distribution.

Now suppose we have an nMOS device with a zero-bias threshold voltage between x_2 and x_3, for example. We have to apply V_3, since V_2 would still makes the device too slow. Since the device is now too fast, its leakage is excessive. The extent of this leakage is determined by the new threshold voltage resulting from the application of V_3, which cannot be expressed by a closed-form expression. We instead approximate this new threshold voltage by $\mu - (x_3 - x)$, i.e., the offset from the perfect compensation (μ) is assumed to be linear provided that $x_3 - x$ is a small quantity. Therefore, in general,

$$V_{\text{th}}(x) = \mu - (x_i - x)\,, \quad x_{i-1} < x < x_i\,. \tag{5}$$

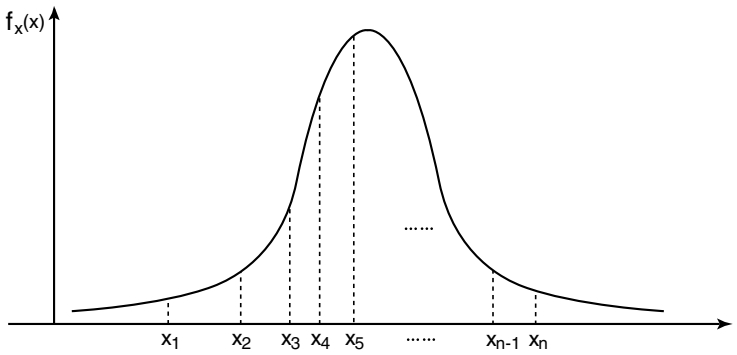


Fig. 9. Probability density function of x that models the variation of zero-bias threshold voltage.

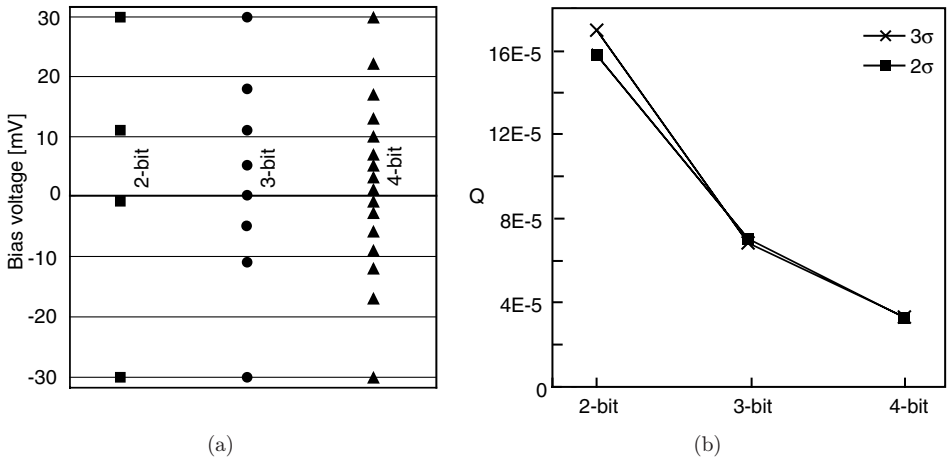
Substituting Eq. (5) into Eq. (4) yields

$$Q = e^{-\mu/S}\int_{-\infty}^{x_n}[e^{(x_i-x)/S}-1]f_x(x)dx$$
$$\approx e^{-\mu/S}\left[\int_{-\infty}^{x_n}e^{(x_i-x)/S}f_x(x)dx-1\right]. \quad (6)$$

Note that the upper limit of the integration is x_n but not ∞, since it is not possible to compensate for threshold voltages larger than x_n. This is not a problem in practice because we can use a large value, such as 3σ, for x_n, which enables us to compensate for all likely threshold variations.

Once we have μ and σ for the process variation and S, which is the subthreshold slope, we can minimize Eq. (6), which gives us values for $x_1, x_2, \ldots, x_n$. This in turn yields a set of body bias voltages $(V_1, V_2, \ldots, V_n)$, which we can use to design our resistor tree. As an example, for a commercial 180-nm CMOS process, $\mu = 400\,\text{mV}$ and the 3σ variation is 30 mV for nMOS devices. We vary the number of bits in the codewords in the lookup table (see Fig. 2) from 2 to 4, giving 4, 8, or 16 body bias voltages. In all cases, we fix the values of x_1 and x_n to -3σ and 3σ, which corresponds to minimizing Eq. (6) for $2^n - 2$ bias voltages with an n-bit codeword.

We used Maple to minimize Eq. (6), and Fig. 10(b) shows the results at room temperature (Q varies with S, which is a function of temperature). The figure clearly shows that, as we increase the number of bits in the codeword, the overall excessive leakage goes down, as we expect, since the increased number of available bias voltages allows finer control of body bias. However, the difference in leakage between 3-bit and 4-bit codewords is not significant, and it would therefore be reasonable to choose a 3-bit codeword because using 4 bits increases the complexity of the



(a) (b)

Fig. 10. (a) Optimized body bias voltages and (b) variation of excessive leakage with number of bits in the codeword.

controller significantly (see Figs. 2 and 3). Figure 10(a) shows the bias voltages that were obtained as a result of this optimization process. We repeated this experiment, with $x_1 = -2\sigma$ and $x_n = 2\sigma$. The results of this test are also shown in Fig. 10(b), but the difference between 2σ and 3σ is not significant. Additionally, we repeated the same experiment for different temperatures, but the results were virtually unchanged, implying that Q in our definition is not a strong function of temperature.

4. Experimental Results

We performed experiments on a set of four circuits taken from the ISCAS'89 benchmarks. Table 1 gives the characteristics of the original circuits. Each circuit was mapped on to a commercial 180-nm triple-well, 1.8 V gate library. Using the same 21 gates from the library, we were able to compare the original circuit with the one that is mapped to our custom library. Each circuit was placed and routed, and used the area shown in the third column. The transistor-level netlist is then extracted from the layout and simulated to determine the standby leakage current (fourth column) and the active-mode circuit delay (fifth column).

The sixth column of Table 1 shows the area of each circuit when mapped on to our custom cell library, as explained in Sec. 2. Compared to the original circuit, the use of custom cells gives us area savings of between 7% and 11% even including tap cells, due to the reduced cell height.

The layout of the adaptive body bias controller is shown in Fig. 11. The controller occupies an area of $70\,\mu\text{m} \times 105\,\mu\text{m}$, of which 57% is taken up by the resistor tree. The size of this proportion is due to the well isolation required by pMOS devices. The resistor tree consists of 96 pMOS devices, $V_{ddh} = 3.3\,\text{V}$, and $V_{ddl} = -1.5\,\text{V}$, so that bias voltages between V_{ddl} and V_{ddh} can be generated in steps of 50 mV. The negative voltage of V_{ddl} could be provided from out of the chip or could be generated by using a charge pump, which is beyond the scope of this paper. The codeword consists of 3 bits which gives good process compensation, as explained in the previous section. The body bias generator (see Fig. 3) draws 935 nA, and the amplifier draws $70\,\mu\text{A}$ when turned on (see Figs. 5 and 6), but negligible current when turned off. The current of body bias generator is comparable to the leakage of example circuits shown in column 4 of Table 1, but we took very small circuits as shown in column 2 for the sake

Table 1. Experimental result on ISCAS benchmark circuits at room temperature, for $V_{\text{dd}} = 1.8\,\text{V}$.

	Original circuit				Test circuit				
Circuits	Gates	Area (μm^2)	Leak. (nA)	Delay (ns)	Area (μm^2)	ΔV_{th} (mV)	Leak. (nA)	Comp. delay (ns)	V_n (V)/V_p (V)
c3540	1669	120×105	512	2.304	107×109	−30	4.12	2.291	+0.2/+1.6
c6288	2416	315×115	910	0.813	291×112	−10	13.43	0.812	+0.1/+1.7
s1423	731	121×105	170	3.135	110×107	10	3.76	3.134	−0.05/+1.85
s9234	5808	315×70	1001	0.763	291×71	30	24.59	0.759	−0.15/+1.95

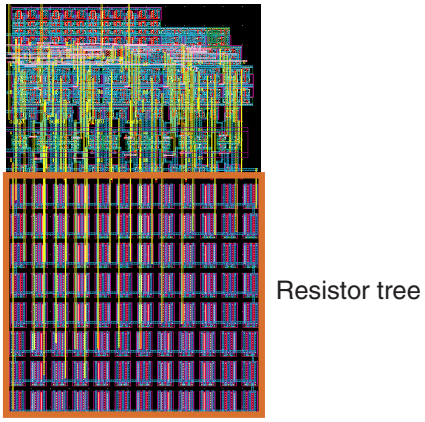


Fig. 11. The layout of the adaptive body bias controller.

of run time of circuit simulation; in large practical circuits, therefore, this current can be assumed to be negligible. The amplifier draws quite a lot current (but only during active mode), which, however, may be small compared to active current of circuits themselves.

In order to simulate the effects of process variation, we assumed that each circuit has a threshold voltage which differs from its standard value as shown in the seventh column (ΔV_{th}) of Table 1. In the eighth column is the standby leakage current of each circuit. Compared to the original circuit, the leakage is cut by a factor of between 40 and 124, due to the large reverse body bias that we use in standby mode.

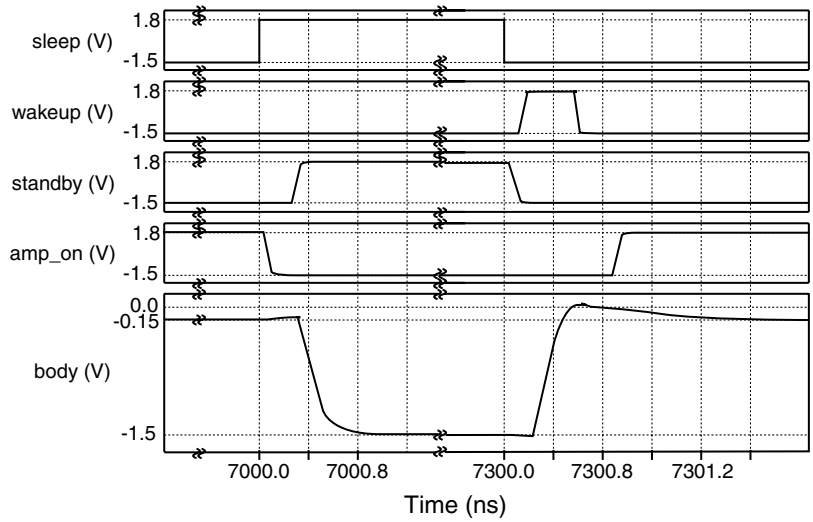


Fig. 12. Simulation of mode change from s9234: (a) active to standby and (b) standby to active.

The ninth column shows the delay in each circuit when active mode body bias is applied; the amount of bias is shown in the last column of the table. In contrast with the delays in the original circuit, all the circuits are now compensated.

Figure 12 is a SPICE simulation result of a mode change from s9234. The `sleep` signal received from the PMU is shown at the top of the figure, while the three subsequent plots show the control signals that are generated from it. The body potential of NMOS devices in s9234 is also shown in the figure. It is readily seen that both transitions take about 1 ns, which is short enough for fast mode change.

5. Conclusion

An adaptive body biasing has been used to compensate for process variation and to reduce subthreshold leakage current. The overhead of biasing circuits has limited its use to chip-level. In this paper, we have proposed a new adaptive body biasing scheme that can be used in block by block basis. The proposed scheme uses a lookup table that holds a codeword corresponding to the active mode body bias of each block on a chip, which, when applied, can compensate for process variation. A predetermined reverse body bias is used to reduce subthreshold leakage in standby mode. Since a fixed number of predetermined bias voltages are used, it is important to design them in efficient way, which we formulated and solved in a numerical fashion. We have presented the layout methodology for applying the proposed scheme to semi-custom designs using standard-cell elements. We performed an experiment with benchmark circuits, and have demonstrated that, through the use of proposed scheme, process variations can be compensated for and standby leakage current is reduced significantly.

Acknowledgments

This work was supported by Samsung Electronics.

References

1. K. Roy, S. Mukhopadhyay and H. Mahmoodi-Meimand, Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits, *Proc. IEEE* **91** (2003) 305−327.
2. T. Kobayashi and T. Sakurai, Self-adjusting threshold-voltage scheme (SATS) for low-voltage high-speed operation, *Proc. Custom Integrated Circuits Conf.*, May 1994, pp. 271−274.
3. S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi and V. De, Parameter variations and impact on circuit and microarchitecture, *Proc. Design Automat. Conf.*, June 2003, pp. 338−342.
4. J. W. Tschanz, S. G. Narendra, Y. Ye, B. A. Bloechel, S. Borkar and V. De, Dynamic sleep transistor and body bias for active leakage power control of microprocessors, *IEEE J. Solid-State Circuits* **38** (2003) 1838−1845.

5. T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, K. Suzuki, F. Sano, M. Norishima, M. Murota, M. Kako, M. Kinugawa, M. Kakumu and T. Sakurai, A 0.9-V, 150-MHz, 10-mW, 4 mm^2, 2-D discrete cosine transform core processor with variable threshold-voltage (VT) scheme, *IEEE J. Solid-State Circuits* **31** (1996) 1770−1779.
6. J. W. Tschanz, J. T. Kao, S. G. Narendra, R. Nair, D. A. Antoniadis, A. P. Chandrakasan and V. De, Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage, *IEEE J. Solid-State Circuits* **37** (2002) 1396−1402.
7. L. T. Clark, M. Morrow and W. Brown, Reverse-body bias and supply collapse for low effective standby power, *IEEE Trans. VLSI Syst.* **12** (2004) 947−956.