

## Chapter 2

# OVERVIEW OF THE FACTORS AFFECTING THE POWER CONSUMPTION

David Chinnery, Kurt Keutzer

*Department of Electrical Engineering and Computer Sciences*

*University of California at Berkeley*

*Berkeley, CA 94720, USA*

We investigate differences in power between application-specific integrated circuits (ASICs) and custom integrated circuits, with examples from 0.6 $\mu$ m to 0.13 $\mu$ m CMOS. A variety of factors cause synthesizable designs to consume 3 to 7 $\times$  more power. We discuss the shortcomings of typical synthesis flows, and changes to tools and standard cell libraries needed to reduce power. Using these methods, we believe that the power gap between ASICs and custom circuits can be closed to within 2.6 $\times$  at a tight performance constraint for a typical ASIC design.

## 2.1 INTRODUCTION

In the same technology generation, custom designs can achieve 3 to 8 $\times$  higher clock frequency than ASICs [18]. Custom techniques that are used to achieve high speed can also be used to achieve low power [62]. Custom designers can optimize the individual logic cells, the layout and wiring between the cells, and other aspects of the design. In contrast, ASIC designers generally focus on optimization at the RTL level, relying on EDA tools to map RTL to cells in a standard cell library and then automatically place and route the design. Automation reduces the design time, but the resulting circuitry may not be optimal.

Low power consumption is essential for embedded applications. Power affects battery life and the heat dissipated by hand-held applications must be limited. Passive cooling is often required, as using a heat sink and/or fan is larger and more expensive.

Power is also becoming a design constraint for high-end applications due to reliability, and costs for electricity usage and cooling. As technology scales, power density has increased with transistor density, and leakage power is

becoming a significant issue even for high end processors. Power consumption is now a major problem even for high end microprocessors. Intel canceled the next generation Tejas Pentium 4 chips due to power consumption issues [100].

In this chapter, we will discuss the impact of manual and automated design on the power consumption, and also the impact of process technology and process variation. Our aim is to quantify the influence of individual design factors on the power gap. Thus, we begin by discussing a process technology independent delay metric in Section 2.2. Section 2.3 discusses the contribution to a chip's power consumption from memory, control and datapath logic, and clocking, and also provides an overview of dynamic and leakage power.

In Section 2.4, we compare full custom and synthesizable ARM processors and a digital signal processor (DSP) functional unit. We show that ASICs range from 3 to 7 $\times$  higher power than custom designs for a similar performance target. To date the contribution of various factors to this gap has been unclear. While automated design flows are often blamed for poor performance and poor energy efficiency, process technology is also significant. Section 2.5 outlines factors contributing to the power gap. We then examine each factor, describing the differences between custom and ASIC design methodologies, and account for its impact on the power gap. Finally, we detail approaches that can reduce this power gap. We summarize our analysis in Section 2.6.

## 2.2 PROCESS TECHNOLOGY INDEPENDENT FO4 DELAY METRIC

At times we will discuss delay in terms of FO4 delays. It is a useful metric for normalizing out process technology dependent scaling of the delay of circuit elements.

The fanout-of-4 inverter delay is the delay of an inverter driving a load capacitance that has four times the inverter's input capacitance [38]. This is shown in Figure 2.1. The FO4 metric is not substantially changed by process technology or operating conditions. In terms of FO4 delays, other fanout-of-4 gates have at most 30% range in delay over a wide variety of process and operating conditions, for both static logic and domino logic [38].

If it has not been simulated in SPICE or tested silicon, the FO4 delay in a given process technology can be estimated from the channel length. Based on the effective gate length  $L_{eff}$ , the rule of thumb for FO4 delay is [39].

$$360 \times L_{eff} \text{ ps for typical operating and typical process conditions} \quad (2.1)$$

$$500 \times L_{eff} \text{ ps for worst case operating and typical process conditions} \quad (2.2)$$

where the effective gate length  $L_{eff}$  has units of micrometers. Typical process conditions give high yield, but are not overly pessimistic. Worst case operating conditions are lower supply voltage and higher temperature than typical operating conditions. Typical operating conditions for ASICs may assume a temperature of 25°C, which is optimistic for most applications. Equation (2.2) can be used to estimate the FO4 delay in silicon for realistic operating conditions [39].

$L_{eff}$  is often assumed to be about 0.7 of the drawn gate length for a process technology – for example, 0.13μm for a 0.18μm process technology. However, many foundries are aggressively scaling the channel length to increase the speed. Thus, the FO4 delay should be calculated from the effective gate length, if it is known, rather than from the process technology generation.

From previous analysis [18], typical process conditions are between 17% and 28% faster than worst case process conditions. Derating worst case process conditions by a factor of 1.2× gives

$$600 \times L_{eff} \text{ ps for worst case operating and worst case process conditions (2.3)}$$

Equation (2.3) was used for estimating the FO4 delays of synthesized ASICs, which have been characterized for worst case operating and worst case process conditions. This allows analysis of the delay per pipeline stage, independent of the process technology, and independent of the process and operating conditions.

**Note:** these rules of thumb give approximate values for the FO4 delay in a technology. They may be inaccurate by as much as 50% compared to simulated or measured FO4 delays in silicon. These equations do not accurately account for operating conditions. Speed-binning and process improvements that do not affect the effective channel length are not accounted for. Accurate analysis with FO4 delays requires proper calibration of the metric: simulating or measuring the actual FO4 delays for the given process and operating conditions.

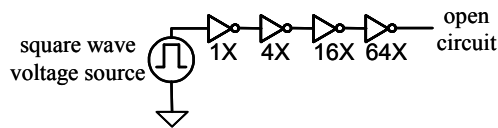


Figure 2.1 This illustrates a circuit to measure FO4 delays. The delay of the 4X drive strength inverter gives the FO4 delay. The other inverters are required to appropriately shape the input waveform to the 4X inverter and reduce the switching time of the 16X inverter, which affect the delay of the 4X inverter [38].

## 2.3 COMPONENTS OF POWER CONSUMPTION

Designers typically focus on reducing both the total power when a circuit is active and its standby power. There is usually a minimum performance target, for example 30 frames/s for MPEG. When performance is less important, the energy per operation to perform a given task can be minimized.

Active power includes both dynamic power consumption, when the logic evaluates or the clock transitions, and current leakage when logic is not switching. There is no computation in logic in standby, the clock must be gated to prevent it switching, and leakage is the dominant source of power consumption in standby.

The major sources of power consumption in circuitry are the clock tree and registers, control and datapath logic, and memory. The breakdown of power consumption between these is very application and design dependent. The power consumption of the clock tree and registers ranged from 18% to 36% of the total power for some typical embedded processors and microprocessors (see Section 3.2.4). In custom cores for discrete cosine transform (DCT) and its inverse (IDCT), contributions to the total power were 5% to 10% from control logic, about 40% from the clock tree and clock buffers, and about 40% from datapath logic [101][102]. Memory can also account for a substantial portion of the power consumption. For example, in the StrongARM caches consume 43% of the power [62].

### 2.3.1 Dynamic power

Dynamic power is due to switching capacitances and short circuit power when there is a current path from supply to ground.

The switching power is proportional to  $\alpha f C V_{dd}^2$ , where  $\alpha$  is the switching activity per clock cycle,  $f$  is the clock frequency,  $C$  is the capacitance that is (dis)charged, and  $V_{dd}$  is the voltage swing. The switching activity is increased by glitches, which typically cause 15% to 20% of the activity in complementary static CMOS logic [77].

Short circuit power typically contributes less than 10% of the total dynamic power [14], and increases with increasing  $V_{dd}$ , and with decreasing  $V_{th}$ . Short circuit power can be reduced by matching input and output rise and fall times [96].

As the dynamic power depends quadratically on  $V_{dd}$ , methods for reducing active power often focus on reducing  $V_{dd}$ . Reducing the capacitance by downsizing gates and reducing wire lengths is also important.

### 2.3.2 Leakage power

In today's processes, leakage can account for 10% to 30% of the total power when a chip is active. Leakage can contribute a large portion of the average power consumption for low performance applications, particularly when a chip has long idle modes without being fully off.

Optimally choosing  $V_{dd}$  and  $V_{th}$  to minimize the total power consumption for a range of delay constraints in 0.13 $\mu$ m technology, the leakage varied from 8% to 21% of the total power consumption in combinational logic, as discussed later in Section 4.6.1. However, the possible  $V_{dd}$  and  $V_{th}$  values depend on the particular process technology and standard cell libraries available. For example for a delay constraint of 1.2 $\times$  the minimum delay, the best library choice had  $V_{dd}$  of 0.8V and  $V_{th}$  of 0.08V (see Table 7.7 with 0.8V input drivers), and leakage contributed on average 40% of total power.

Leakage power in complementary static CMOS logic in bulk CMOS is primarily due to subthreshold leakage and gate leakage. Subthreshold leakage increases exponentially with decrease in  $V_{th}$  and increase in temperature. It can also be strongly dependent on transistor channel length in short channel devices. Gate leakage has increased exponentially with reduction in gate oxide thickness. There is also substrate leakage. Leakage has become increasingly significant in deep submicron process technologies.

## 2.4 ASIC AND CUSTOM POWER COMPARISON

To illustrate the power gap, we examine custom and ASIC implementations of ARM processors and dedicated hardware to implement discrete cosine transform (DCT) and its inverse (IDCT). ARM processors are general purpose processors for embedded applications. ASICs often have dedicated functional blocks to achieve low power and high performance on specific applications – for example, media processing. JPEG and MPEG compression and decompression of pictures and video use DCT and IDCT. There is a similar power gap between ASIC and custom for the ARM processors and for DCT and IDCT blocks.

### 2.4.1 ARM processors from 0.6 to 0.13 $\mu$ m

We compare chips with full custom ARM processors, soft, and hard ARM cores. Soft macros of RTL code may be sold as individual IP (intellectual property) blocks and are portable between fabrication processes. In a hard macro, the standard cell logic used, layout and wiring have been specified and optimized then fixed for a particular fabrication process. A hard macro may be custom, or it may be “hardened” from a soft core. A complete chip includes additional memory, I/O logic, and so forth.

Table 2.1 Full custom and hard macro ARM<sup>s</sup> [11][31][32][43][70]. The highlighted full custom chips have 2 to 3× MIPS/mW.

Processor	Technology (um)	Voltage (V)	Frequency (MHz)	MIPS	Power (mW)	MIPS/mW
ARM710	0.60	5.0	40	36	424	0.08
Burd	0.60	1.2	5	6	3	1.85
Burd	0.60	3.8	80	85	476	0.18
ARM810	0.50	3.3	72	86	500	0.17
ARM910T	0.35	3.3	120	133	600	0.22
StrongARM	0.35	1.5	175	210	334	0.63
StrongARM	0.35	2.0	233	360	950	0.38
ARM920T	0.25	2.5	200	220	560	0.39
ARM1020E	0.18	1.5	400	500	400	1.25
XScale	0.18	1.0	400	510	150	3.40
XScale	0.18	1.8	1000	1250	1600	0.78
ARM1020E	0.13	1.1	400	500	240	2.08

Table 2.2 The highlighted ARM7TDMI hard macros have 1.3 to 1.4× MIPS/mW versus the synthesizable ARM7TDMI-S cores [5].

ARM Core	Technology (um)	Frequency (MHz)	Power (mW)	MIPS/mW
ARM7TDMI	0.25	66	51	1.17
ARM7TDMI-S	0.25	60	66	0.83
ARM7TDMI	0.18	100	30	3.00
ARM7TDMI-S	0.18	90	35	2.28
ARM7TDMI	0.13	130	10	11.06
ARM7TDMI-S	0.13	120	13	8.33

To quantify the power gap between ASIC and custom, we first examined hard macro and full custom ARM<sup>s</sup>, listed in Table 2.1. Compared to the other designs, the three full custom chips in bold achieved 2 to 3× millions of instructions per second per milliwatt (MIPS/mW) at similar MIPS, as shown in Figure 2.2. The inverse of this metric, mW/MIPS, is the energy per operation. The Dhrystone 2.1 MIPS benchmark is the performance metric [98]. It fits in the cache of these designs, so there are no performance hits for cache misses or additional power to read off-chip memory.

Lower power was achieved in several ways. The DEC StrongARM used clock-gating and cache sub-banking to substantially reduce the dynamic power [62]. The Intel XScale and DEC StrongARM used high speed logic styles to reduce critical path delay, at the price of higher power consumption on these paths. To reduce pipeline register delay, the StrongARM used pulse-triggered flip-flops [62] and the XScale used clock pulsed latches [22]. Shorter critical paths allow the same performance to be achieved with a lower supply voltage (V<sub>dd</sub>), which can lower the total power consumption. Longer channel lengths were used in the StrongARM caches to reduce the

leakage power, as the two 16kB caches occupy 90% of the chip area [62]. The XScale used substrate biasing to reduce the leakage [24].

For the same technology and similar performance (MIPS), the Vdd of the full custom chips is lower than that of the hard macros – reducing Vdd gives a quadratic reduction in dynamic power. The StrongARM can operate at up to 233MHz at 2.0V and the XScale can operate at up to 1GHz at 1.65V [43]. If operating at higher performance was not required, it is likely that even higher MIPS/mW could have been achieved.

Energy efficiency can be improved substantially if performance is sacrificed. Burd’s 0.6um ARM8 had software controlled dynamic voltage scaling based on the processor load. It scaled from 0.18MIPS/mW at 80MHz and 3.8V, to 2.14MIPS/mW at 5MHz and 1.2V [11]. Voltage scaling increased the energy efficiency by 1.1× for MPEG decompression which required an average clock frequency of 50MHz, and increased the energy efficiency by 4.5× for audio processing which required a clock frequency of only 17MHz [12].

There is an additional factor of 1.3 to 1.4× between hard macro and synthesizable ARM7 soft cores, as shown in Table 2.2. These MIPS/mW are higher than those in Table 2.1, as they exclude caches and other essential units. The ARM7TDMI cores are also lower performance, and thus can achieve higher energy efficiency.

Overall, there is a factor of 3 to 4× between synthesizable ARMs and the best full custom ARM implementations.

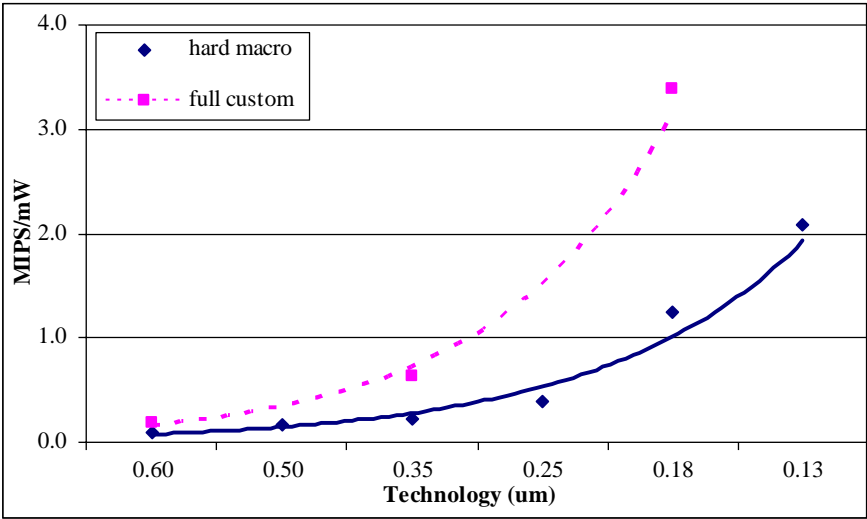


Figure 2.2 This graph compares MIPS/mW of custom and hard macro ARMs in Table 2.1.

### 2.4.1.1 Other full custom ARM implementations

There are two other noteworthy higher performance full custom ARMs, though they are less energy efficient than the 0.18 $\mu$ m XScale.

Samsung's Halla is a full custom 0.13 $\mu$ m implementation of the ARM1020E with power consumption from 0.26W at 400MHz and Vdd of 0.7V to 1.8W at 1200MHz and Vdd of 1.1V [50]. Achieving 1480MIPS at 1200MHz clock frequency, the energy efficiency ranged from 0.82MIPS/mW at 1200MHz to 1.90MIPS/mW at 400MHz. Differential cascode voltage switch logic (DCVSL) was used for high performance, but DCVSL has substantial power consumption compared to complementary static CMOS logic that is used in ASICs. Sense amplifiers were used with the low voltage swing dual rail bus to detect voltage swings of less than 200mV, achieving high bus speeds at lower power consumption [60]. The die area of the Halla was 74% more than ARM's 0.13 $\mu$ m ARM1020E.

Intel's 90nm implementation of the XScale, codenamed Monahans, has 770mW dynamic power consumption at 1500MHz and Vdd of 1.5V with performance of 1200MIPS at this point [72]. The energy efficiency of Monahans is 1.56MIPS/mW at 1500MHz – data for improved energy efficiencies at lower Vdd has not been published. Clock pulsed latches were also used in this implementation of the XScale. The hold time for the clock gating enable signal was the duration of the clock pulse, and thus did not require latching. Domino logic was used for high performance in the shifter and cache tag NOR comparators. 75% of instruction cache tag accesses were avoided by checking if the instruction cache request line was the same as the previous one. Selective accesses and avoiding repeated accesses reduced power by 42% in the dynamic memory management unit [21].

### 2.4.2 Comparison of DCT/IDCT cores

Application-specific circuits can reduce power by an order of magnitude compared to using general purpose hardware [77]. Two 0.18 $\mu$ m ARM9 cores were required to decode 30 frames/s for MPEG2, consuming 15 $\times$  the power of a synthesizable DCT/IDCT design [28]. However, the synthesizable DCT/IDCT significantly lags its custom counterparts in energy efficiency.

*Table 2.3* Comparison of ASIC and custom DCT/IDCT core power consumption at 30 frames/s for MPEG2 [28][101][102].

Design	Technology ( $\mu$ m)	Voltage (V)	DCT (mW)	IDCT (mW)
ASIC	0.18	1.60	8.70	7.20
custom DCT	0.6 ( $L_{\text{eff}}$ 0.6)	1.56	4.38	
custom IDCT	0.7 ( $L_{\text{eff}}$ 0.5)	1.32		4.65



Fanucci and Saponara designed a low power synthesizable DCT/IDCT core, using similar techniques to prior custom designs. Despite being three technology generations ahead, the synthesizable core was 1.5 to 2.0× higher power [28]. Accounting for the technology difference by conservatively assuming power scales linearly with device dimensions [71], the gap is a factor of 4.3 to 6.6×. The data is shown in Table 2.3.

2.5 FACTORS CONTRIBUTING TO ASICS BEING HIGHER POWER

Various parts of the circuit design and fabrication process contribute to the gap between ASIC and custom power. Our analysis of the most significant design factors and their impact on the total power when a chip is active is outlined in Table 2.4. The “typical” column shows the maximum contribution of individual factors comparing a typical ASIC to a custom design. In total these factors can make power an order of magnitude worse. In practice, even the best custom designs can’t fully exploit all these factors simultaneously. Low power design techniques that can be incorporated within an EDA flow can reduce the impact of these factors in a carefully designed ASIC as per the “excellent” column in Table 2.4.

Most low power EDA tools focus on reducing the dynamic power in control logic, datapath logic, and the clock tree. The design cost for custom memory is low, because of the high regularity. Several companies provide custom memory for ASIC processes. Optimization of memory hierarchy, memory size, caching policies, and so forth is application dependent and beyond the scope of this book, though they have a substantial impact on the system-level performance and power consumption. We will focus on the power consumption in a processor core.

Table 2.4 Factors contributing to ASICs being higher power than custom. The excellent column is what ASICs may achieve using low power and high performance techniques. This table focuses on the total power when a circuit is active, so power gating and other standby leakage reduction techniques are omitted. The combined impact of these factors is not multiplicative – see discussion in Section 2.5.1.

Contributing Factor	Typical ASIC	Excellent ASIC
microarchitecture	5.1×	1.9×
clock gating	1.6×	1.0×
logic style	2.0×	2.0×
logic design	1.2×	1.0×
technology mapping	1.4×	1.0×
cell and wire sizing	1.6×	1.1×
voltage scaling	4.0×	1.0×
floorplanning and placement	1.5×	1.1×
process technology	1.6×	1.0×
process variation	2.0×	1.3×

Microarchitectural techniques such as pipelining and parallelism increase throughput, allowing timing slack for gate downsizing and voltage scaling. The microarchitecture also affects the average instructions per cycle (IPC), and hence energy efficiency. The power and delay overheads for microarchitectural techniques must be considered. With sufficient timing slack, reducing the supply voltage can greatly increase the energy efficiency. For example in Table 2.1, scaling the XScale from Vdd of 1.8V to 1.0V increases the efficiency from 0.78MIPS/mW to 3.40MIPS/mW, a factor of 4.4 $\times$ , but the performance decreases from 1250MIPS to 510MIPS.

Process technology can reduce leakage by more than an order of magnitude. It also has a large impact on dynamic power. Process variation results in a wide range of the leakage power for chips and some variation in the maximum operating clock frequency for a given supply voltage. For high yield, a higher supply voltage may be needed to ensure parts meet the desired performance target, resulting in a significant spread in power consumption. Limiting process variation and guard-banding for it without being overly conservative help reduce the power consumption.

Using a high speed logic style on critical paths can increase the speed by 1.5 $\times$  [18]. Circuitry using only slower complementary static CMOS logic at a tight performance constraint may be 2.0 $\times$  higher power than circuitry using a high speed logic style to provide timing slack for power reduction by voltage scaling and gate downsizing.

Other factors in Table 2.4 have smaller contributions to the power gap. We will discuss the combined impact of the factors and then look at the individual factors and low power techniques to reduce their impact.

### 2.5.1 Combined impact of the contributing factors

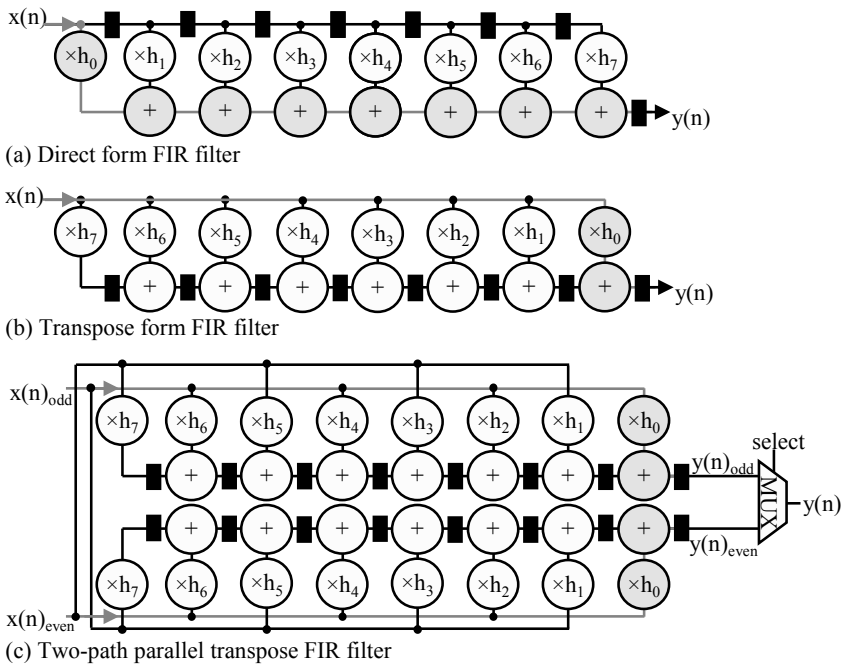
The combined impact of the factors is complicated. The estimate of the contribution from voltage scaling assumes that timing slack is provided by pipelining, so this portion is double counted. The timing slack depends on the tightness of the performance constraint, which has a large impact on the power gap. We assumed a tight performance constraint for both the typical ASIC and excellent ASIC for the contributions from microarchitecture, logic style, and voltage scaling in Table 2.4. If the performance constraint is relaxed, then the power gap is less. For example, from our model of pipelining to provide timing slack for voltage scaling and gate sizing, the power gap between a typical ASIC and custom decreases from 5.1 $\times$  at a tight performance constraint for the typical ASIC to 4.0 $\times$  if the constraint is relaxed by 7%.

Chapter 3 details our power and delay model that incorporates pipelining, logic delay, voltage scaling and gate sizing. The logic delay is determined by factors such as the logic style, wire lengths, process technology, and process variation which affects the worse case delay.

From analysis with this model, an excellent ASIC using the low power techniques that we recommend below may close the power gap to a factor of 2.6 at a tight performance constraint for a typical ASIC [16].

### 2.5.2 Microarchitecture

Algorithmic and architectural choices can reduce the power by an order of magnitude [77]. We assume that ASIC and custom designers make similar algorithmic and architectural choices to find a low power implementation that is appropriate for the required performance and target application. Pipelining and parallelism are the two major microarchitectural techniques that can be used to maintain throughput (see Figure 2.3), when other power reduction techniques increase critical path delay. With similar microarchitectures, how do ASIC and custom pipelining and parallelism compare?



*Figure 2.3* This diagram shows pipelined (b) and parallel implementations (c) of the unpipelined direct form finite input response (FIR) filter in (a) [19][79]. The FIR filter calculates  $y_n = h_0x_n + h_1x_{n-1} + \dots + h_7x_{n-7}$ . The critical paths are shown in grey. The minimum clock period decreases as the registers break the critical path up into separate pipeline stages. Computation in each pipeline stage proceeds concurrently. The parallel implementation doubles the throughput, but the area is more than doubled. The multiplexer to select the odd or even result from the two parallel datapaths at each clock cycle is denoted by MUX.

On their own, pipelining and parallelism do not reduce power. Pipelining reduces the critical path delay by inserting registers between combinational logic. Glitches may not propagate through pipeline registers, but the switching activity of the combinational logic is otherwise unchanged. Additional pipeline registers add to the leakage power and especially to the dynamic power, because the clock signal going to the registers has high activity. Pipelining may reduce the instructions per cycle (IPC) due to branch misprediction and other hazards; in turn this reduces the energy efficiency. Parallelism trades off area for increased throughput, with overheads for multiplexing and additional wiring [6]. Both techniques enable the same performance to be met at lower supply voltage with smaller gate sizes, which can provide a net reduction in power.

Bhavnagarwala et al. [6] predict a 2 to 4 $\times$  reduction in power with voltage scaling by using 2 to 4 parallel datapaths. Generally, ASICs can make as full use of parallelism as custom designs, but careful layout is required to minimize additional wiring overheads.

Delay overheads for pipelining include: register delay; register setup time; clock skew; clock jitter; and any imbalance in pipeline stage delays that cannot be compensated for by slack passing or useful clock skew. For a given performance constraint, the pipelining delay overheads reduce the slack available to perform downsizing and voltage scaling.

In the IDCT, the cost of pipelining was about a 20% increase in total power, but pipelining reduced the critical path length by a factor of 4. For the same performance without pipelining,  $V_{dd}$  would have to be increased from 1.32V to 2.2V. Thus pipelining helped reduce power by 50% [102].

### 2.5.2.1 What's the problem?

The timing overhead per pipeline stage for a custom design is about 3 FO4 delays, but it may be 20 FO4 delays for an ASIC, substantially reducing the timing slack available for power reduction. For a typical ASIC, the budget for the register delay, register setup time, clock skew and clock jitter is about 10 FO4 delays. Unbalanced critical path delays in different pipeline stages can contribute an additional 10 FO4 delays in ASICs. If the delay constraint is tight, a little extra timing slack can provide substantial power savings from downsizing gates – for example, a 3% increase in delay gave a 20% reduction in energy for a 64-bit adder [104].

For pipeline registers, most ASICs use slow edge-triggered D-type flip-flops that present a hard timing boundary between pipeline stages, preventing slack passing. The clock skew between clock signal arrivals at different points on the chip must be accounted for. Faster pulse-triggered flip-flops were used in the custom StrongARM [62]. Some pulse-triggered flip-flops have greater clock skew tolerance [80]. Custom designs may use

level-sensitive latches to allow slack passing, and latches are also less sensitive to clock skew [19].

The custom XScale used clock-pulsed transparent latches [22]. A D-type flip-flop is composed of a master-slave latch pair. Thus a clock-pulsed latch has about half the delay of a D-type flip-flop and has a smaller clock load, which reduced the clock power by 33%. Clock-pulsed latches have increased hold time and thus more problems with races. The pulse width had to be carefully controlled and buffers were inserted to prevent races. The clock duty cycle also needs to be carefully balanced.

To estimate the impact of worse ASIC pipelining delay overhead, we developed a pipeline performance and power model, with power reduction from gate downsizing and voltage scaling versus timing slack (see Chapter 3). At a tight performance constraint for the ASIC design, we estimate that ASIC power consumption can be  $5.1\times$  that of custom, despite using a similar number of pipeline stages. While there is no timing slack available to the ASIC design, the lower custom pipeline delay overhead allows significant power reduction by gate downsizing and voltage scaling.

### **2.5.2.2 What can we do about it?**

Latches are well-supported by synthesis tools [83], but are rarely used other than in custom designs. Scripts can be used to convert timing critical portions of an ASIC to use latches instead of flip-flops [17]. High-speed flip-flops are now available in some standard cell libraries and can be used in an automated design methodology to replace slower D-type flip-flops on critical paths [33]. Useful clock skew tailors the arrival time of the clock signal to different registers by adjusting buffers in the clock tree and can be used in ASIC designs for pipeline balancing [26]. With these methods, the pipeline delay overhead in ASICs can be reduced to as low as 5 FO4 delays [18]. This enables more slack to be used for downsizing, voltage scaling, or increasing the clock frequency. From our pipeline model, ASICs can close the gap for the microarchitecture and timing overhead factor to within  $1.9\times$  of custom.

### **2.5.3 Clock gating**

In typical operation, pipeline stages and functional units are not always in use. For example, during a sequence of integer operations, the floating point unit may be idle. Providing the logical inputs to the idle unit are held constant, there are only two sources of power dissipation in the idle unit: static leakage; and switching activity at registers and any other clocked elements due to the clock signal – for example, precharge of domino logic.

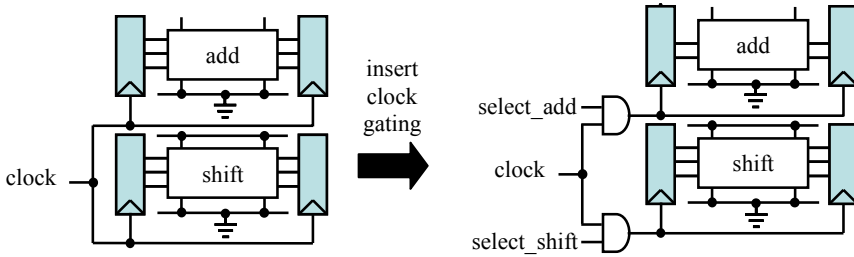


Figure 2.4 This is a simple illustration of clock gating. The clock signal to the registers is gated with a control signal that selects which functional unit is in use. A transparent low latch is usually inserted to de-glitch the enable signal [51].

Architectural or gate-level signals can turn off the clock to portions of the clock tree that go to idle units. This can be done with a clock gating control signal and clock signal at an AND gate, as illustrated in Figure 2.4. As the clock tree and registers can contribute 20% to 40% of the total power, this gives substantial dynamic power savings if units are often idle. The power overheads for logic to generate clock gating signals and the clock gating logic need to be compared versus the potential power savings. Usually clock gating signals can be generated within only one clock cycle, and there is only a small delay increase in the arrival of the gated clock signal at the register.

The StrongARM's total power when active would be about  $1.5\times$  worse without clock gating [62]. The StrongARM uses a 12 bit by 32 bit multiply-accumulate (MAC) unit. For some applications, one multiply operand will be 24-bit or less, or 12-bit or less, thus the number of cycles the  $12\times 32$  MAC is required is less than the three cycles for a full  $32\times 32$  multiply. This saves power by avoiding unnecessary computation. Typical code traces had shift operations of zero, so power could be saved by disabling the shifter in this case [62].

The custom DCT core uses clock gating techniques extensively. In typical operation, consecutive images are highly correlated. Calculations using the significant bits of pixels in common between consecutive images can be avoided. This reduced the number of additions required by 40%, and gave on average 22% power savings for typical images [101]. After the discrete cosine transform on a typical image, there are many coefficients of value zero. This was exploited in the custom IDCT to separately clock gate pipeline stages processing coefficients of zero [102].

We estimate that clock gating techniques can increase energy efficiency when the chip is active by up to  $1.6\times$ . Note that the power savings from clock gating vary substantially with the application.

### 2.5.3.1 What's the problem?

Clock gating requires knowledge of typical circuit operation over a variety of benchmarks. If a unit is seldom idle, clock gating would increase power consumption. Until recently, clock gating was not fully supported by commercial tools. Retiming to reposition the registers [75] can be essential to better balance the pipeline stages, but EDA tools did not support retiming of registers with clock gating.

Care must be taken with gated clock signals to ensure timing correct operation of the registers. Glitches in the enable signal must not propagate to the clock gate while the clock gate is high. This results in a long hold time for the enable signal, which may be avoided by inserting a transparent low latch to de-glitch the enable signal [51]. The transparent low latch prevents the signal that goes to the clock gate from changing while the clock is high. The setup time for the enable signal is longer to account for the clock gate and de-glitching latch. The clock signal arrives later to the register due to the delay of the clock gate, which increases the hold time for that register. The clock tree delay of the clock signal to the clock gate can be reduced to compensate for this, but that may require manual clock tree design.

### 2.5.3.2 What can we do about it?

An ASIC designer can make full use of clock gating techniques by carefully coding the RTL for the desired applications, or using automated clock-gating. The techniques used in custom DCT and IDCT designs were used in the synthesizable DCT/IDCT [28]. In the synthesizable DCT/IDCT, clock gating and data driven switching activity reduction increased the energy efficiency by 1.4× for DCT and 1.6× for IDCT [28].

In the last few years, commercial synthesis tools have become available to automate gate-level clock-gating, generating clock gating signals and inserting logic to gate the clock. There is now support for retiming of flip-flops with gated clock signals. Power Compiler was able to reduce the power of the synthesizable ARM9S core by 41% at 5% worse clock frequency [30] – primarily by gate downsizing, pin reordering, and clock gating. Useful clock skew tools can compensate for the additional delay on the gated clock signal [26].

There are tools for analyzing the clock-tree power consumption and activity of functional units during benchmark tests. These tools help designers identify signals to cut off the clock signal to logic when it is not in use, and to group logic that is clock gated to move the clock gating closer to the root of the clock tree to save more power.

As ASICs can make effective use of clock gating, there should be no power gap due to clock gating in comparison with custom.

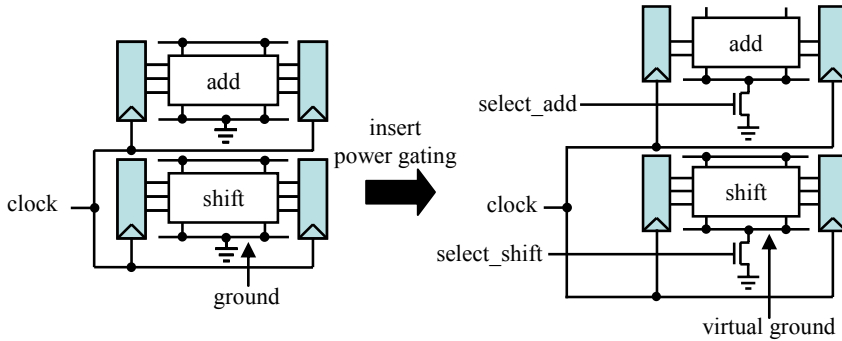


Figure 2.5 This is a simple illustration of power gating. The sleep transistors are turned on by a control signal that selects which functional unit is in use, reducing leakage from supply to ground. Registers may not be disconnected in this manner without losing state information.

## 2.5.4 Power gating and other techniques to reduce leakage in standby

After clock gating idle units, only static leakage remains. The leakage can be substantially reduced by several methods: reducing the supply voltage (see Section 2.5.9); disconnecting the power rails with sleep transistors [64], known as *power gating*; increasing  $V_{th}$  via substrate biasing to reduce subthreshold leakage; and assigning logic gate inputs to a lower leakage state [56]. All these methods take a significant amount of power and thus are only worthwhile when a unit will be idle for tens to thousands of clock cycles or more [27] – for example, when most of a mobile phone’s circuitry is idling while awaiting an incoming call. This requires architectural or software level signals to transition between normal operation and sleep mode.

Reducing the supply voltage reduces the subthreshold leakage current as there is less drain induced barrier lowering (DIBL), and also reduces the gate-oxide tunneling leakage [57]. For example, leakage decreases by 3× when  $V_{dd}$  is reduced from 1.2V to 0.6V with our 0.13μm libraries (see Section 4.5.1). Dynamic voltage scaling is discussed further in Section 2.5.9.

Subthreshold leakage and gate leakage vary substantially depending on which transistors in a gate are off, which is determined by the inputs. Leakage in combinational logic can be reduced by a factor of 2 to 4× by assigning primary inputs to a lower leakage state [56][58]. Additional circuitry is required in the registers to store the correct state, while outputting the low leakage state; or state information may be copied from the registers and restored on resumption from standby. There is also dynamic power consumption in the combinational logic in the cycle that inputs are assigned to a low leakage state. Thus, units must be idle for on the order of ten cycles to justify going to a low leakage state.



Normally, the p-well of the NMOS transistors is connected to ground and the n-well of the PMOS transistors is connected to the supply. The subthreshold leakage can be reduced by increasing the threshold voltages by reverse biasing the substrate, connecting the n-well to more than 0V and connecting the p-well to less than V<sub>dd</sub>. This requires charge pump circuitry to change the voltage, additional power rails, and a twin well or triple well process [24]. The advantage of reverse body bias is that the state is retained. Reverse body bias is less effective for reducing leakage in shorter channel transistors, for example providing a 4× reduction in leakage in 0.18μm technology and 2.5× reduction in 0.13μm [49], making it a less useful technique in deeper submicron technologies.

An alternate method of reverse body bias is to connect both the NMOS transistor source and well to a virtual ground  $V_{ss}$  (see Figure 2.5) which is raised to reduce leakage in standby. This avoids the need for charge pump circuitry and twin well or triple well process [24]. To avoid losing state information, the reduction in  $V_{dd} - V_{ss}$  must be limited by circuitry to regulate the voltage [23]. Reducing  $V_{dd} - V_{ss}$  also helps reduce the leakage. This reverse body bias and voltage collapse approach gave a 28× reduction in leakage in the 0.18μm XScale with minimal area penalty [24]. Returning from “drowsy” mode took 20us, corresponding to 18,000 cycles at 800MHz, as the phase-locked loop (PLL) was also turned off to limit power consumption. In comparison, using sleep transistors in the XScale would have reduced leakage by only about 5×, if power gating was not applied to latches and other memory elements that need to retain state, as they comprise about a sixth of the total transistor width [24].

Power gating with sleep transistors to disconnect the supply and/or ground rail (Figure 2.5) can provide more than an order of magnitude leakage reduction in circuitry that uses leaky low V<sub>th</sub> transistors on critical paths and high V<sub>th</sub> sleep transistors [64]. This is often referred to as MTCMOS, multi-threshold voltage CMOS. The “virtual” supply and “virtual” ground rails, which are connected to the actual power rails via sleep transistors, may be shared between logic gates to reduce the area overhead for the sleep transistors. Disconnecting the power rails results in loss of state, unless registers contain a latch connected to the actual supply and ground rails [64]. Registers also have connections to the virtual supply and virtual ground rails to limit leakage.

Leakage was reduced by 37× in a 0.13μm 32-bit arithmetic logic unit (ALU) using PMOS sleep transistors at the expense of a 6% area overhead and 2.3% speed decrease [89]. The leakage was reduced 64× by using reverse body bias in conjunction with PMOS sleep transistors in sleep mode, and forward body bias in active mode reduced the speed penalty to 1.8%. The total area overhead for the sleep transistors and the body bias circuitry was 8%. Using sleep transistors saved power if the ALU was idle for at least

a hundred clock cycles. Two clock cycles were required to turn the transistors back on from sleep mode, and four cycles were required to change from reverse body bias. With only forward body bias,  $V_{dd}$  could be reduced from 1.32V to 1.28V with no speed penalty, and leakage was reduced by  $1.9\times$  at zero bias in standby [89].

#### **2.5.4.1 What's the problem?**

Reducing leakage via state assignment, substrate biasing, reducing supply voltage, and sleep transistors requires architectural or software level signals to specify when units will be idle for many cycles. These techniques cannot be automated at the gate level and require architectural level support for signals to enter and exit standby over multiple cycles.

For state assignment, registers that retain data instead output a 0 or 1 in sleep mode. For registers that don't retain data in standby, extra circuitry is required if the reset output differs from the low leakage state output. These registers are larger and consume more power than a standard register.

Substrate biasing and reducing the supply voltage require a variable supply voltage from a voltage regulator. The cell libraries need to have delay, dynamic power, leakage power and noise immunity characterized at the different supply and substrate voltages. If functional units enter standby at different times, additional power rails may be required and wells biased at different potentials must be spatially isolated. These techniques are often used in low power custom designs, but are complicated to implement in ASICs.

There is a voltage drop across sleep transistors when they are on, degrading the voltage swing for logic. Wider sleep transistors degrade the voltage swing less, but have higher capacitance. Power up of sleep transistors takes substantial energy due to their large capacitance [64]. Standard cells must be characterized for the degraded supply voltage. Layout tools must cluster the gates that connect to the same virtual power rail that is disconnected by a given sleep signal, as having individual sleep transistors in each gate is too area expensive. As the registers that retain state connect to the virtual power rails and directly to the power rails, the standard cell rows on which registers are placed must be taller to accommodate the extra rails. The virtual and actual supply and ground voltages differ in standby. Thus, the substrates of the transistors connected to virtual power rails and those connected directly to the power rails are at different voltages and must be isolated spatially, increasing the area overhead. The floating output of a power-gated cell can cause large currents if it connects directly to a cell which is not power gated, so additional circuitry is required to drive the output of the power-gated cell [92].

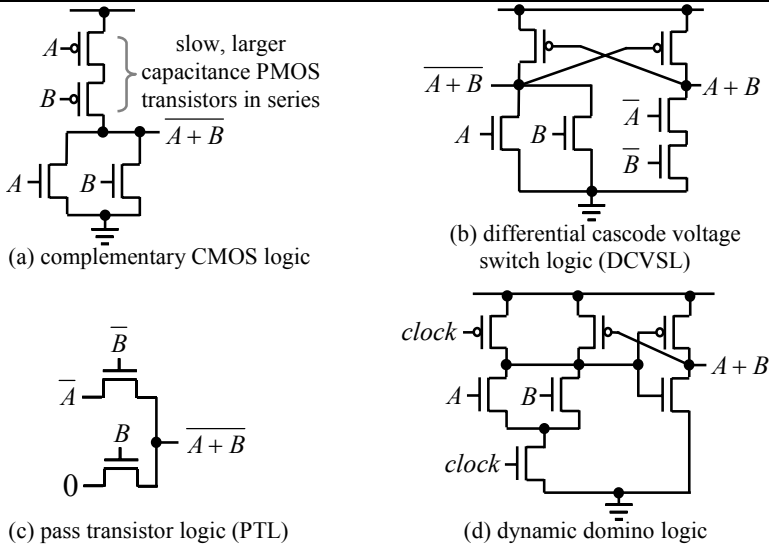
#### 2.5.4.2 What can we do about it?

ASICs seldom use standby power reduction techniques other than full power down, but there is now better tool support for power gating. An EDA flow with power gating can provide two orders of magnitude reduction in leakage if state is not retained, at the cost of 10% to 20% area overhead and 6% higher delay (see Chapter 10). The ARM1176JZ-S synthesizable core supports dynamic voltage scaling, allowing the supply voltage to be scaled from 1.21V to 0.69V in the 0.13 $\mu$ m process, but this requires additional hardware support [35].

To date state assignment and reverse substrate biasing have not been implemented in an EDA methodology. As state assignment cannot be effectively used with combinational logic that is power gated and provides far less leakage reduction than using sleep transistors, it is unlikely to be useful except for circuits that have only short idle periods, on the order of tens of clock cycles. Substrate biasing nicely complements power gating with forward body bias reducing the delay penalty for voltage drop across the sleep transistors, and with reverse body bias reducing the leakage in registers that are on to retain state information. As reverse substrate bias is less effective at shorter channel lengths, ASICs may have from 4 $\times$  higher standby leakage than custom designs that use reverse body bias in 0.18 $\mu$ m to 2 $\times$  worse than custom in deeper submicron technologies.

### 2.5.5 Logic style

ASICs almost exclusively use complementary static CMOS logic for combinational logic, because it is more robust to noise and V<sub>dd</sub> variation than other logic styles. Pass transistor logic (PTL), dynamic domino logic and differential cascode voltage switch logic (DCVSL) are faster than complementary static CMOS logic. These logic styles are illustrated in Figure 2.6. Complementary CMOS logic suffers because PMOS transistors are roughly 2 $\times$  slower than NMOS transistors of the same width, which is particularly a problem for NOR gates. With the two PMOS transistors in series in Figure 2.6(a), the PMOS transistors must be sized about 4 $\times$  larger for equal rise and fall delays, substantially increasing the load on the fanins. The high speed logic styles can be used to reduce the critical path delay, increasing performance. Alternatively, the additional timing slack can be used to achieve lower power at high performance targets. Complementary CMOS logic is lower power than other logic styles when high performance is not required. Hence, low power custom designs primarily use complementary CMOS, with faster logic only on critical paths. ASIC designs are mapped to slower, purely complementary CMOS logic standard cell libraries.



*Figure 2.6* This figure shows NOR2 logic gate implementations in different logic styles. The domino logic output is inverted, so that after precharging the inputs to domino logic gates are low to avoid them being discharged until an input transition to high occurs [71].

The StrongARM used primarily complementary CMOS, with static DCVSL to implement wide NOR gates [62]. In the custom IDCT multiplier, the carry and sum of the full adder cells are both on the critical path [102]. A complementary CMOS gate generated the carry out, and a static DCVSL gate generated the sum. This full adder was 37% faster than a purely complementary CMOS mirror adder.

The StrongARM and XScale used some dynamic logic. Dynamic DCVSL (dual rail domino logic) has twice the activity of single rail domino logic. The Samsung Halla used dynamic DCVSL and is higher power than the complementary CMOS ARM1020E at 400MHz. However, the Halla runs at up to 1.2GHz, while the ARM1020E is limited to 400MHz [60]. Zlatanovici [104] compared 0.13 $\mu$ m single rail domino and complementary static CMOS 64-bit adders. Domino could achieve as low as 6.8 FO4 delays at 34pJ/cycle. The fastest static CMOS version was 12.5 FO4 delays, but only 18pJ/cycle.

PTL is a high speed and low energy logic style [7]. In a 0.6 $\mu$ m study, a complementary CMOS carry-lookahead 32-bit adder was 20% slower than complementary PTL, but the complementary CMOS adder was 71% lower power [103]. At maximum frequency in 0.25 $\mu$ m, a complementary CMOS 3-input XOR ring oscillator had 1.9 $\times$  delay and 1.3 $\times$  power compared to versions in PTL and DCVSL [52]. The XScale ALU bypass adder was implemented in PTL. At 1.1V, this was 14% slower than single rail domino, but it has no precharge and lower switching activity [22].

High speed logic styles can increase the speed of combinational logic by  $1.5\times$  [18]. We discuss the potential power savings with reduced combinational logic delay calculated from the pipeline model in Section 3.5. We optimistically assumed no extra power consumption for using a high speed logic style on critical paths. At a tight performance constraint, pipelines with only complementary static CMOS combinational logic had up to  $2.0\times$  higher energy per operation.

### 2.5.5.1 What's the problem?

PTL, DCVSL, and dynamic logic libraries are used as in-house aids to custom designers. Standard cell libraries with these logic styles are not available to ASIC designers. All of these logic styles are less robust than complementary CMOS logic, and have higher leakage power.

Differential cascode voltage switch logic is faster than complementary CMOS logic, but is higher energy [7][20]. DCVSL requires both input signal polarities and has higher switching activity than complementary CMOS logic. Static DCVSL has cross-coupled outputs, resulting in longer periods of time with a conducting path from supply to ground and larger short circuit current. The DCVSL inputs and their negations must arrive at the same time to limit the duration of the short circuit current, requiring tight control of the layout to ensure similar signal delays.

Dynamic logic is precharged on every clock cycle, increasing the clock load, activity, and dynamic power. The precharged node may only be discharged once, so glitches are not allowed. Shielding may be required to prevent electromagnetic noise due to capacitive cross-coupling discharging the precharged node. To avoid leakage through the NMOS transistors discharging the node, a weak PMOS transistor is required as a “keeper” [99]. There can be charge sharing between dynamic nodes or on PTL paths.

Pass transistor logic suffers a voltage drop of  $V_{th}$  across the NMOS pass transistor when the input voltage is high [71]. Consequently, the reduced voltage output from PTL may need to be restored to full voltage to improve the noise margin and to avoid large leakage currents in fanouts. The voltage drop can be avoided by using a complementary PMOS transistor in parallel with the NMOS transistor, but this increases the loading on the inputs, reducing the benefit of PTL. Buffering is needed if the fanins and fanouts are not near the PTL gates, and an inverter may be needed to generate a negated input.

Using these logic styles requires careful cell design and layout. A typical EDA flow gives poor control over the final layout, thus use of these logic styles would result in far more yield problems and chip failures.

### 2.5.5.2 What can we do about it?

The foundry requirement of high yield means that the only standard cell libraries available to ASIC designers will continue to be robust complementary static CMOS logic. Thus an EDA design flow cannot reduce the power gap for logic style.

An alternative is for designers to adopt a semi-custom design flow: high speed custom cells and manual layout can be used for timing critical logic; or custom macros can be used.

## 2.5.6 Logic design

Logic design refers to the topology and the logic structure used to implement datapath elements such as adders and multipliers. Arithmetic structures have different power and delay trade-offs for different logic styles, technologies, and input probabilities.

### 2.5.6.1 What's the problem?

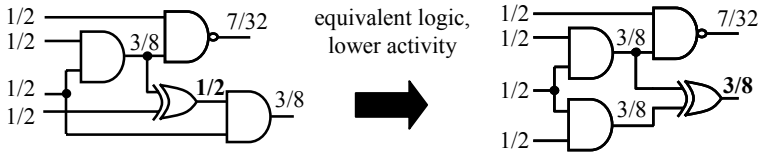
Custom designers tend to pay more attention to delay critical datapaths. Specifying logic design requires carefully structured RTL and tight synthesis constraints. For example, we found that flat synthesis optimized out logic that reduced switching activity in multiplier partial products [47], so the scripts were written to maintain the multiplier hierarchy during synthesis. The reduced switching activity reduced the power-delay product by  $1.1\times$  for the 64-bit multiplier.

Careful analysis is needed to compare alternate algorithmic implementations for different speed constraints. For example, high-level logic transition analysis showed that a 32-bit carry lookahead adder had about 40% lower power-delay product than carry bypass or carry select adders [13]. There was also a 15% energy difference between 32-bit multipliers. Zlatanovici compared 64-bit domino adders in 0.13 $\mu\text{m}$ , and found that the radix-4 adders achieved smaller delay and about 25% lower energy than radix-2 [104].

We estimate that incomplete evaluation of logic design alternatives may result in  $1.2\times$  higher power for a typical ASIC.

### 2.5.6.2 What can we do about it?

Synthesis tools can compile to arithmetic modules. The resulting energy and delay is on par with tightly structured RTL. In general, ASIC designers should be able to fully exploit logic design.



*Figure 2.7* This figure illustrates how refactoring logic can reduce the switching activity while giving the same functional result. Switching activities are annotated on the diagram, as propagated from independent inputs that have equal probability of being zero or one.

## 2.5.7 Technology mapping

In technology mapping a logical netlist is mapped to a standard cell library in a given technology. Different combinations of cells can implement a gate with different activity, capacitance, power and delay. For example to implement an XOR2, an AO22 with inverters may be smaller and lower power, but slower. (An AO22 logic gate computes  $ab + cd$ , so XOR2 may be implemented by  $a\bar{b} + \bar{a}b$ .) Refactoring can reduce switching activity (see Figure 2.7). Common sub-expression elimination reduces the number of operations. Balancing path delays and reducing the logic depth decreases glitch activity. High activity nets can be assigned to gate pins with lower input capacitance. [63][77]

### 2.5.7.1 What's the problem?

While there are commercial tools for power minimization, power minimization subject to delay constraints is still not supported in the initial phase of technology mapping. Minimizing the total cell area minimizes circuit capacitance, but it can increase activity. For a 0.13 $\mu$ m 32-bit multiplier after post-synthesis power minimization, the power was 32% higher when using minimum area technology mapping. This was due to more (small) cells being used, increasing activity. We had to use technology mapping combining delay and area minimization targets for different parts of the multiplier. Technology mapping for low power may improve results; without this and other low power technology mapping techniques, ASICs may have 1.4 $\times$  higher power than custom.

### 2.5.7.2 What can we do about it?

Power minimization tools do limited remapping and pin reassignment, along with clock gating and gate sizing [84]. These optimizations are applied after technology mapping for minimum delay, or minimum area with delay constraints. EDA tools should support technology mapping for minimum power with delay constraints. This requires switching activity analysis, but it is not otherwise substantially more difficult than targeting minimum area.

For a given delay constraint, technology mapping can reduce the power by 10% to 20%, for a 10% to 20% increase in area [63][77]. Low power encoding for state assignment can also give 10% to 20% power reduction [90]. Logic transformations based on logic controllability and observability, common sub-expression elimination, and technology decomposition can give additional power savings of 10% to 20% [68]. Pin assignment can provide up to 10% dynamic power savings by connecting higher activity inputs to gate input pins with lower capacitance [74].

ASICs should not lag custom power consumption due to technology mapping, if better EDA tool support is provided.

## 2.5.8 Gate sizing and wire sizing

Wires and transistors should be sized to ensure correct circuit operation, meet timing constraints, and minimize power consumption. ASICs must choose cell sizes from the range of drive strengths provided in the library. ASIC wire widths are usually fixed. Downsizing transistors gives a linear reduction in their capacitance and thus dynamic power, and also gives a linear reduction in leakage. Reducing the wire width gives a linear reduction in wire capacitance but a linear increase in wire resistance, increasing signal delay on the wire.

### 2.5.8.1 What's the problem?

There is a trade-off between power and delay with gate sizing. To reduce delay, gates on critical paths are upsized, increasing their capacitance. In turn, their fanin gates must be upsized to drive the larger capacitance. This results in oversized gates and buffer insertion on the critical paths. Delay reductions come at the price of increasingly more power and worse energy efficiency.

To balance rise and fall delays, an inverter has PMOS to NMOS width ratio of about 2:1 as a PMOS transistor has about half the drain current of a NMOS transistor of the same width. Accounting for the number of transistors in series, other logic gates also have 2:1 P/N ratio to balance rise and fall delays for an inverter of equivalent drive strength, as illustrated in Figure 2.8. However, to minimize the average of the rise delay and fall delay, the P/N ratio for an inverter should be about 1.5:1 [37]. Reducing the P/N ratio provides a small reduction in delay and a substantial reduction in power consumption, by reducing the capacitance of the larger PMOS transistors. The optimal P/N ratio to minimize the delay is larger for larger loads [73]. In addition, sometimes the rise and fall drive strengths needed are different – for example, the rising output transition from a cell may be on a critical path, but the falling transition may not be critical.



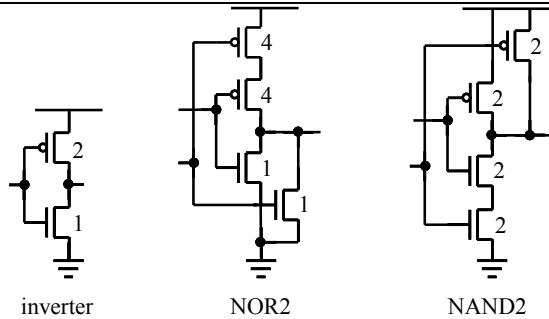


Figure 2.8 This figure shows the relative NMOS and PMOS transistor widths for equal rise and fall delays in different logic gates of equivalent drive strength.

The ratio of pullup to pulldown drive strength determines at what input voltage a gate switches from low to high or high to low [99]. Equal rise and fall delays maximize the noise margin for a high or low input. Thus skewing the P/N ratio reduces the noise margin. Ideally, standard cell libraries should provide a range of drive strength skews and lower power cells with reduced P/N ratio, but often only cells with equal rise and fall drive strength are available to ensure high yield.

A design-specific standard cell library developed for the iCORE [73] gave a 20% speed increase by using reduced P/N width ratio, and by using larger transistor widths to increase drive strength instead of buffering. The larger transistor widths required increased cell height, but the net impact on layout area was minimal as they were only used in the most critical paths. However, the design time for this library was about two worker years.

Custom libraries may be finer grained, which avoids oversizing gates, and have skewed drive strengths. Cells in datapath libraries are denser and have smaller input capacitance [18]. Specific cell instances can be optimized. Cells that connect to nearby cells don't need guard-banding. This avoids the need for buffering to handle driving or being driven by long wires.

Wire widths can also be optimized in custom designs. Gong et al. [34] optimized global clock nets on a 1.2 $\mu$ m chip. By simultaneously optimizing buffer and wire sizes, they reduced the clock net power by about 63%. This amounts to a 10% to 20% saving in total power.

The basic approach to gate sizing in commercial EDA software has changed little in the past 20 years. These gate sizers like TILOS [29] proceed in a greedy manner, picking the gate with the best power or area versus delay tradeoff to change, and iterating. There are known circuit examples where these approaches perform suboptimally, but it has not been clear how much of a problem this is for typical circuits for real world applications. We found power savings of up to 32.3% versus gate sizing in Design Compiler, which is commonly used in EDA flows for circuit synthesis, and 16.3%

savings on average across the ISCAS'85 benchmarks and three typical datapath circuits (see Section 6.5.3). Gate sizing is an NP-complete problem, but circuit sizes are large and optimization software must have runtimes of  $O(n^2)$  or less to be of practical use [81], where  $n$  is the number of gates in the circuit. The TILOS-like greedy approaches are relatively fast, being  $O(n^2)$ , and other approaches that perform better with similar static timing analysis (STA) accuracy have had worse computational complexity.

Some commercial power minimization software has only recently provided the option of minimizing the total power. Previously, the user had to prioritize minimizing either the dynamic power or the leakage power, which can be suboptimal.

We estimate that these limitations in gate sizing and wire sizing for typical ASICs may lead to a power gap of  $1.6\times$  versus custom.

### 2.5.8.2 What can we do about it?

Gate downsizing to reduce power consumption is well supported by power minimization tools. Some commercial tools support clock tree wire sizing, but there are no commercial tools available for sizing other wires. Automated cell creation, characterization and in-place optimization tools are available. Standard cell libraries with finer grained drive strengths and lower power consumption are available, though users may be charged a premium.

We synthesized the base configuration of a Tensilica Xtensa processor in  $0.13\mu\text{m}$ . The power/MHz was 42% lower and the area was 20% less at 100MHz than at the maximum clock frequency of 389MHz, due to using smaller gates and less buffers. If delay constraints are not too tight, tools can reduce power by gate downsizing without impacting delay. At 325MHz, Power Compiler was able to reduce the power consumption by 26% and reduce the area by 12% for no delay penalty.

Libraries with fine granularity help to reduce the power by avoiding use of oversized gates. In a  $0.13\mu\text{m}$  case study of digital signal processor (DSP) functional macros, using a fine grained library reduced power consumption by 13% (see Chapter 13).

After place and route when wire lengths and capacitive loads are accurately known, in place optimization can remove guard banding where it is unnecessary. ASIC designers have tended to distrust this approach, as the optimized cells without guard banding cannot be safely used at earlier stages in the EDA flow. Skewing the pullup to pulldown drive strength to optimize the different timing arcs through a gate can also improve energy efficiency. A prototype tool flow for in place cell optimization increased circuit speed by 13.5% and reduced power consumption by 18%, giving a  $1.4\times$  increase in energy efficiency for the  $0.35\mu\text{m}$  12,000 gate bus controller [25]. 300 optimized cells were generated in addition to the original standard cell library that had 178 cells.

Our linear programming gate sizing approach discussed in 0 takes a global view of the circuit rather than performing greedy “peephole” optimization. We achieved up to 32.3% power savings and on average 16.3% power savings versus gate sizing in Design Compiler for the combinational netlists. Our optimization approach has between  $O(n)$  and  $O(n^2)$  runtime growth, making it scalable to large circuit sizes.

ASICs may have  $1.1\times$  worse power than custom due to gate and wire sizing, as wire sizing tools are not available other than for the clock tree, and some design-specific cell optimizations are not possible without custom cell design, beyond what is possible with automated cell creation.

### 2.5.9 Voltage scaling

Reducing the supply voltage  $V_{dd}$  quadratically reduces switching power. Short circuit power also decreases with  $V_{dd}$ . Reducing  $V_{dd}$  also reduces leakage. For example, with our 0.13 $\mu$ m library leakage decreases by a factor of three as  $V_{dd}$  is decreased from 1.2V to 0.6V. As  $V_{dd}$  decreases, a gate’s delay increases. To reduce delay, threshold voltage  $V_{th}$  must also be scaled down. As  $V_{th}$  decreases, leakage increases exponentially. Thus there is a tradeoff between performance, dynamic power and leakage power.

Ideally, we want to operate at as low  $V_{dd}$  as possible, with  $V_{th}$  high enough to ensure little leakage. For example, dynamic scaling of the supply voltage from 3.8V to 1.2V gives a  $10\times$  increase in energy efficiency at the price of decreasing performance by a factor of 14 for Burd’s 0.6 $\mu$ m ARM implementation [11]. Reducing the power consumption in this manner requires timing slack.

Power consumption may be reduced by using multiple supply voltages and multiple threshold voltages. High  $V_{dd}$  and low  $V_{th}$  can be used on critical paths to reduce their delay, while lower  $V_{dd}$  and higher  $V_{th}$  can be used elsewhere to reduce dynamic and leakage power.

#### 2.5.9.1 What’s the problem?

Custom designs can achieve at least twice the speed of ASICs with high performance design techniques [18]. At the same performance target as an ASIC, a custom design can reach lower  $V_{dd}$  using the additional timing slack. Compare  $V_{dd}$  of Burd, StrongARM and XScale to other ARMs in Table 2.1. With lower  $V_{dd}$  they save between 40% and 80% dynamic power versus other ARMs in the same technology. This is the primary reason for their higher energy efficiency. To use lower  $V_{dd}$ , ASICs must either settle for lower performance or use high speed techniques, such as deeper pipelining, to maintain performance.

Dynamically adjusting the supply voltage for the desired performance requires a variable voltage regulator and takes time, during which correct

signals must be maintained to avoid transitioning into illegal states from which behavior is unknown. To change from 1.2V to 3.8V in Burd's ARM [11] required energy equal to that consumed in 712 cycles of peak operation, and there was a delay of 70us. Increasing or decreasing the supply voltage by 800mV took 50us in the XScale [22].

Several barriers remain to ASICs using low Vdd. Using lower Vdd requires lower Vth to avoid large increases in gate delay. Vth is determined by the process technology. A foundry typically provides two or three libraries with different Vth: high Vth for low power; and low Vth for high speed at the expense of significant leakage power. Most ASIC designers cannot ask a foundry to fine tune Vth for their particular design, even if an intermediate Vth might be preferable to reduce leakage. Vdd can be optimized for ASICs, but typical ASIC libraries are characterized at only two nominal supply voltages – say 1.2V and 0.9V in 0.13um. To use Vdd of 0.6V, the library must be re-characterized. There is also less noise immunity at lower Vdd.

Use of multiple supply voltages either requires that the wells of PMOS transistors in low Vdd gates are reverse biased by connecting them to high Vdd, or spatial isolation between the wells connected to low Vdd and high Vdd. Layout tools must support these spacing constraints. Low voltage swing signals must be restored to full voltage swing with a voltage level converter to avoid large leakage currents when a high Vdd gate is driven by a low Vdd input. Most level converter designs require access to both high Vdd and low Vdd, which complicates layout and may require that they straddle two standard cell rows, additionally the PMOS wells connected to different Vdd must be spatially isolated. Voltage level converters are not available in ASIC libraries. Synthesis and optimization tools must insert level converters where needed, and prevent low Vdd gates driving high Vdd gates in other cases.

If voltage level converters are combined with the flip-flops, the power and delay overheads for voltage level restoration are less. Due to the additional power and delay overheads for asynchronous level converters (those not combined with flip-flops), there have been reservations about whether they provide any practical benefits over only using level converter flip-flops [93]. There has also been concern about their noise immunity [46].

Multi-Vdd circuitry has more issues with capacitive cross-coupling noise due to high voltage swing aggressors on low voltage swing lines. Thus it may be best to isolate high Vdd and low Vdd circuitry into separate voltage islands, rather than using multi-Vdd at a gate level. Multi-Vdd at the gate-level can also require additional voltage rails. Gate level multi-Vdd requires tool support to cluster cells of the same Vdd to achieve reasonable layout density. An additional voltage regulator is needed to generate the lower Vdd.

Using multiple threshold voltages is expensive. Each additional PMOS and NMOS threshold voltage requires another mask to implant a different

dopant density, substantially increasing processing costs. A set of masks costs on the order of a million dollars today and an additional  $V_{th}$  level increases the fabrication cost by 3% [69]. Each additional mask increases the difficulty of tightly controlling yield, motivating some manufacturers to limit designs to a single NMOS and single PMOS threshold voltage.

To take full advantage of multiple threshold voltages within gates, standard cells with multi- $V_{th}$  and skewed transistor widths must be provided. High  $V_{th}$  can be used to reduce leakage while low  $V_{th}$  can be used to reduce dynamic power. For example, using low  $V_{th}$  PMOS transistors and high  $V_{th}$  NMOS transistors in a complementary CMOS NOR gate, as leakage is less through the PMOS transistors that are in series. In gates that have an uneven probability of being high or low, there is more advantage to using high  $V_{th}$  to reduce leakage for the pullup or pulldown network that is more often off. Similarly, for wider transistors with high  $V_{th}$  may be preferable for gates that have low switching activity, while narrower transistors with low  $V_{th}$  is better when there is higher switching activity.

#### **2.5.9.2 What can we do about it?**

There are tools to automate characterizing a library at different  $V_{dd}$  operating points. Characterization can take several days or more for a large library. Standard cell library vendors can help by providing more  $V_{dd}$  characterization points. Commercial tools do not adequately support multi- $V_{dd}$  assignment or layout, but separate voltage islands are possible.

There are voltage level converter designs that only need to connect to high  $V_{dd}$  (see Figure 13.8). Some asynchronous level converters designs have been shown to be robust and have good noise immunity in comparison to typical logic gates at low  $V_{dd}$  [53]. It would help if voltage level converters were added to standard cell libraries.

Foundries often support high and low  $V_{th}$  cells being used on the same chip. Power minimization tools can reduce power by using low  $V_{th}$  cells on the critical path, with high  $V_{th}$  cells elsewhere to reduce leakage. Combining dual  $V_{th}$  with sizing reduced leakage by 3 to 6× for a 5% increase in delay on average versus using only low  $V_{th}$  [78]. From a design standpoint, an advantage of multiple threshold voltages is that changing the threshold voltage allows the delay and power of a logic gate to be changed without changing the cell footprint, and thus not perturbing the layout. As discussed in Chapter 7, multi-threshold voltage optimization is straightforward, providing those cells are provided in the library. Optimization runtime increases at worst linearly with the number of cells in the library.

Geometric programming optimization results on small benchmark circuits suggest that multi- $V_{dd}$  and multi- $V_{th}$  may only offer 20% power savings versus optimal choice of single  $V_{dd}$ , single NMOS  $V_{th}$ , and single PMOS  $V_{th}$  [16]. As ASIC designers are limited to  $V_{th}$  values specified by the

foundry, there may be more scope for power savings in ASICs when  $V_{th}$  is suboptimal. After scaling  $V_{dd}$  from 1.2V to 0.8V by using a low  $V_{th}$  of 0.08V, we found power savings of up to 26% by using a second higher  $V_{th}$  to reduce leakage with our linear programming optimization approach in Chapter 7, and average power savings were 16%. We found that power savings with gate-level multi- $V_{dd}$  were generally less than 10%. Using multi- $V_{dd}$  is more appropriate at a module level, making a good choice of a single supply voltage for the module based on the delay of critical paths.

With 9% timing slack versus the maximum clock frequency, Stok et al. in Chapter 13 reduced power consumption by 31% by scaling from  $V_{dd}$  of 1.2V to  $V_{dd}$  of 1.0V. Usami et al. [94] implemented automated tools to assign dual  $V_{dd}$  and place dual  $V_{dd}$  cells, with substrate biasing for the transistors to operate at low  $V_{th}$  in active mode to increase performance and high  $V_{th}$  in standby mode to reduce leakage. They achieved total power reduction of 58% with only a 5% increase in area. The ARM1176JZ-S [35] synthesizable core supports dynamic voltage scaling, but this requires additional software and hardware support. This demonstrates that ASICs can use such methods with appropriately designed RTL, software, and EDA tool support, reducing the power gap due to voltage scaling alone to 1.0 $\times$ .

### 2.5.10 Floorplanning, cell placement and wire routing

The quality of floorplanning of logic blocks and global routing for wires, followed by cell placement and detailed wire routing, have a significant impact on wire lengths. A significant portion of the capacitance switched in a circuit is wiring capacitance. The power consumption due to interconnect is increasing from about 20% in 0.25 $\mu$ m to 40% in 0.09 $\mu$ m [82]. Wire lengths depend on cell placement and congestion. Larger cells and additional buffers are needed to drive long wires. We estimate that poor floorplanning, cell sizing and cell placement with inaccurate wire load models can result in 1.5 $\times$  worse power consumption in ASICs compared to custom.

#### 2.5.10.1 What's the problem?

Custom chips are partitioned into small, tightly placed blocks of logic. Custom datapaths are manually floorplanned and then bit slices of layout may be composed. Automatic place and route tools are not good at recognizing layout regularity in datapaths.

We used BACPAC [82] to examine the impact of partitioning. We compared partitioning designs into blocks of 50,000 or 200,000 gates in 0.13 $\mu$ m, 0.18 $\mu$ m, and 0.25 $\mu$ m. Across these technologies, using 200,000 gate blocks increased average wire length by about 42%. This corresponds to a 9% to 17% increase in total power. The delay is also about 20% worse with larger partitions [18]. The net increase in energy per operation is 1.3 to 1.4 $\times$ .

When sizing gates and inserting buffers, the first pass of synthesis uses wire load models to estimate wire loads. Wire load models have become increasingly inaccurate, with wires contributing a larger portion of load capacitance in the deep submicron. A conservative wire load model is required to meet delay constraints, but this results in most gates being over-sized [18], making the power higher.

Physical synthesis iteratively performs placement and cell sizing, to refine the wire length estimates. Cell positions are optimized then wire lengths are estimated with Steiner trees. Steiner tree wire length models used by physical synthesis are inaccurate if a wire route is indirect. There can be too many critical paths to give them all a direct route. Power minimization increases path delay, so more paths are critical, increasing congestion. This may degrade performance. For example for the base configuration of Tensilica's Xtensa processor for a tight performance target of 400MHz clock frequency in 0.13um, we found that the clock frequency was 20% worse after place and route when power minimization was used.

#### **2.5.10.2 What can we do about it?**

Physical synthesis, with iteratively refined wire length estimates and cell placement, produces substantially better results than a tool flow using only wire load models. In our experience, physical synthesis can increase speed by 15% to 25%. The cell density (area utilization) increases, reducing wire lengths, and then cells may be downsized, which reduces power by 10% to 20%.

Earlier power minimization tools often ended up increasing the worst critical path delay after layout if the delay constraint was tight. This is less of a problem in today's tools, where power minimization is integrated with physical synthesis. Tool flow integration has also improved, particularly as some of the major CAD software vendors now have complete design flows with tools that perform well throughout the design flow – rather than using for example Synopsys tools for synthesis and Cadence tools for place and route.

An ASIC designer can generate bit slices from carefully coded RTL with tight aspect ratio placement constraints. Bit slices of layout may then be composed. With bit slices, Chang showed a 70% wire length reduction versus automated place-and-route [15], which would give a 1.2 to 1.4× increase in energy efficiency. Stok et al. in Chapter 13 found that bit slicing and some logic optimization, such as constant propagation, improved clock frequency by 22% and reduced power consumption by 20% for seven DSP functional macros implemented in 0.13um, improving the energy efficiency by a factor of 1.5×. Compared to bit slicing using a library of datapath cells, manual placement and routing can still achieve smaller wire lengths [15], leaving a gap of about 1.1×.

### 2.5.11 Process technology

After the layout is verified, the chip is fabricated in the chosen process technology by a foundry. Within the same nominal technology generation, the active power, leakage power, and speed of a chip differ substantially depending on the process used to fabricate the circuit. Older technologies are slower and are cheaper per mask set. However, newer technologies have more dies per wafer and thus may be cheaper per die for larger production runs. Newly introduced technologies may have lower yield, though these problems are typically ironed out as the technology matures [61].

High performance chips on newer technologies have substantially higher subthreshold leakage power as threshold voltage is scaled down with supply voltage to reduce dynamic power. Gate tunneling leakage is also higher as transistor gate oxide thickness is reduced for the lower input voltage to the transistor gate to retain control of the transistor.

Gate leakage can be reduced if the gate oxide thickness  $t_{ox}$  is increased, which requires a high-k gate dielectric permittivity  $\epsilon_{ox}$  to maintain the drive current (see Equation (4.1)). For example, Intel will use hafnium oxide in their 45nm process [44], which has dielectric permittivity of about an order magnitude larger than silicon oxide that is used in most of today's processes, enabling Intel to reduce the gate leakage by more than  $10\times$ .

The power consumption and power per unit area can be less in deeper submicron technologies if performance is not increased [55]. For example in 65nm, Intel's low power P1265 process reduces leakage  $300\times$ , but has 55% lower saturation drain current and hence is about  $2.2\times$  slower [48], compared to their higher performance P1264 technology [91]. To reduce leakage they increased oxide thickness from 1.2nm to 1.7nm, increased gate length from 35nm to 55nm, and increased threshold voltage from about 0.4V to 0.5V (at drain-source voltage of 0.05V) [48]. Note that the higher threshold voltage results in a greater delay increase if supply voltage is reduced.

While Intel started selling processors produced in 65nm bulk CMOS technology at the start of 2006, AMD is still producing chips in 90nm silicon-on-insulator (SOI) technology [8][36]. AMD is on track to offer 65nm SOI chips in the last quarter of 2006 [67]. Intel is a technology generation ahead, and has the cost advantage of using cheaper bulk CMOS and more dies per wafer with its smaller technology. However, SOI has better performance per watt than bulk CMOS, so Intel has only a slight advantage in terms of performance and energy efficiency.

In the same nominal technology generation, there are substantial differences between processes. Different technology implementations differ by up to 25% in speed [18], 60% in dynamic power, and an order of magnitude in leakage. We compared several gates in Virtual Silicon's IBM 8SF and UMC L130HS 0.13 $\mu$ m libraries. 8SF has about 5% less delay and



only 5% of the leakage compared to L130HS, but it has  $1.6\times$  higher dynamic power [97]. Our study of two TSMC 0.13um libraries with the base configuration of Tensilica’s Xtensa processor showed that TSMC’s high  $V_{th}$ , low-k library was 20% lower power/MHz, with 66% less leakage power and 14% less dynamic power, than the low  $V_{th}$ , low-k library (see Table 2.5).

The power consumption, wire RC delays, and IR drop in the wires can be reduced by use of copper wires and low-k interlayer dielectric insulator. Copper interconnect has 40% lower resistivity than aluminum. Low-k dielectrics of 2.7 to 3.6 electrical permittivity (k) are used in different processes, compared to  $SiO_2$ ’s dielectric constant of 3.9. Using low-k interlayer dielectric insulator reduces interconnect capacitance by up to 25%, reducing dynamic power consumption by up to 12%. High-k transistor gate dielectrics increase the transistor drive strength and thus speed, and can also reduce the gate tunneling leakage by an order of magnitude [59].

Narendra et al. showed that silicon-on-insulator (SOI) was 14% to 28% faster than bulk CMOS for some 0.18um gates. The total power was 30% lower at the same delay, but the leakage was 1.2 to  $20\times$  larger [65]. A 0.5um DSP study showed that SOI was 35% lower power at the same delay as bulk CMOS [76]. Double-gated fully depleted SOI is less leaky than bulk CMOS.

The StrongARM caches were 90% of the chip area and were primarily responsible for leakage. A 12% increase in the NMOS channel length reduced worst case leakage  $20\times$ . Lengthening transistors in the cache and other devices reduced total leakage by  $5\times$  [62]. Transistor capacitance, and thus dynamic power, increases linearly with channel length. Channel length can be varied in ASICs to reduce leakage if such library cells are available.

As a process technology matures, incremental changes can improve yield, improve performance and reduce power consumption. In Intel’s 0.25um P856 process the dimensions were shrunk by 5% and, along with other modifications, this gave a speed improvement of 18% in the Pentium II [10]. The 0.18um process for the Intel XScale had a 5% shrink from P858, and other changes to target system-on-chip applications [22]. There was also a 5% linear shrink in Intel’s 0.13um P860 process and the effective gate length was reduced from 70nm to 60nm [87]. A 5% shrink reduces transistor capacitance and dynamic power by about 5%. These process improvements are typical of what is available to high volume custom designs.

We estimate that different choices within the same process technology generation may give up to  $1.6\times$  difference in power.

Table 2.5 Dynamic and leakage power for two different 0.13um TSMC libraries for Tensilica’s Xtensa processor for the base configuration with a clock frequency of 100MHz.

Library	low Vdd, low k-dielectric	low Vdd, low k-dielectric, high Vth
Dynamic power (uW)	6.48	5.66
Leakage power (mW)	0.67	0.25
Total power (mW)	7.15	5.90

### 2.5.11.1 What's the problem?

Standard cells are characterized in a specific process. The cells must be modified and libraries updated for ASIC customers to take advantage of process improvements. Without such updates, 20% speed increase and greater reductions in power may be unavailable to ASIC customers. Finding the lowest power for an ASIC requires synthesis with several different libraries to compare power at performance targets of interest. The lowest power library and process may be too expensive.

### 2.5.11.2 What can we do about it?

Generally, it requires little extra work to re-target an ASIC EDA flow to a different library. ASICs can be migrated quickly to different technology generations, and updated for process improvements. In contrast, the design time to migrate custom chips is large. Intel started selling 90nm Pentium 4 chips in February 2004 [36], but a 90nm version of the XScale was only reported in June 2005 [72] and is not currently in production to our knowledge. Meanwhile, ARM has synthesized the more recent Cortex-A8 core for 65nm [4]. ASICs should be able to take full advantage of process improvements, closing the gap for process technology to 1.0 $\times$ .

## 2.5.12 Process variation

Chips fabricated in the same process technology vary in power and speed due to process variation, as illustrated in Figure 2.9. Some of the chips fabricated may be too slow, while some are significantly faster. In previous technology generations, the faster chips could be sold at a premium. However, faster chips have more leakage power and greater variation in leakage power [9]. Thus the faster chips may consume too much power, particularly if run at a higher clock frequency where dynamic power is also higher as it increases linearly with clock frequency.

There are a number of sources of process variation, such as optical proximity effects, and wafer defects. The channel length  $L$ , transistor width, wire width and wire height have about 25% to 35% variation from nominal at three standard deviations ( $3\sigma$ ). Transistor threshold voltage  $V_{th}$  and oxide thickness have about 10% variation at  $3\sigma$  [66]. Decreased transistor oxide thickness substantially increases gate tunneling leakage, and a decrease in  $V_{th}$  or  $L$  can cause a large increase in subthreshold leakage current, though these transistors are faster. Dynamic power scales linearly with transistor and wire dimensions, as capacitances increase.

To ensure high yield accounting for process variation, libraries are usually characterized at two points. To meet the target speed, the process' worst case speed corner is used – typically 125°C, 90% of nominal  $V_{dd}$ ,

with slow transistors. To prevent excessive power, the active power may be characterized at a worst case power corner, e.g.  $-40^{\circ}\text{C}$ , 110% of nominal  $V_{dd}$ , and fast transistors. Leakage is worse at high temperature. Due to  $V_{dd}$  alone, the active power is 50% higher at the worst case power corner than at the worst case speed corner. These process corners are quite conservative and limit a design. The fastest chips fabricated in a typical process may be 60% faster than estimated from the worst case speed corner [18]. Similarly, examining the distribution of power of fabricated 0.3um MPEG4 codecs [85], the worst case power may be 50% to 75% higher than the lowest power chips produced.

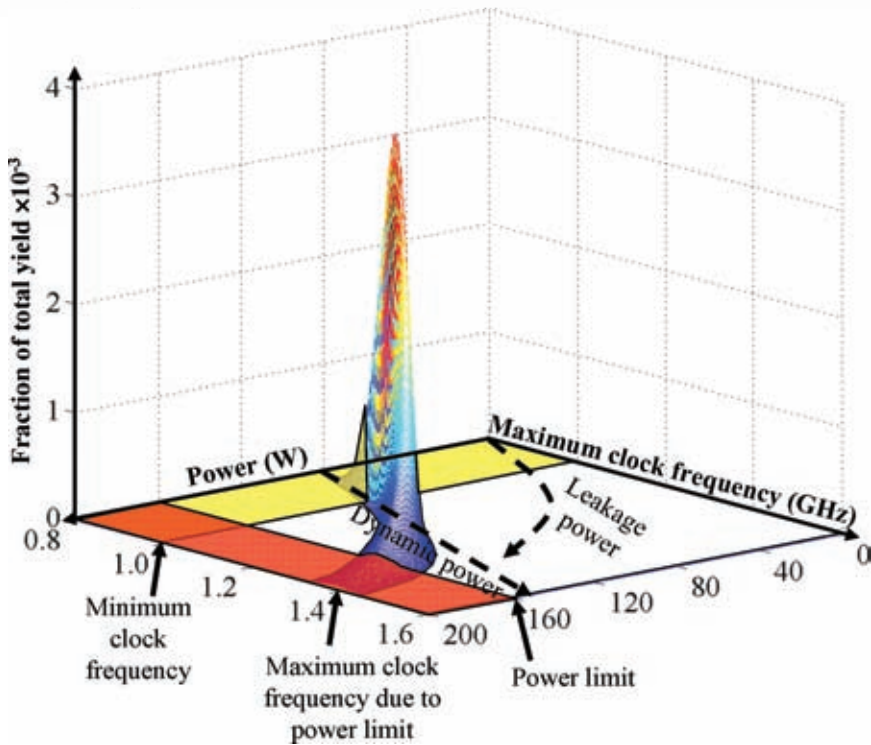


Figure 2.9 This graph illustrates yield versus maximum clock frequency  $f$  and total power  $P$  at that clock frequency. The minimum frequency is 1.0GHz and the maximum power is 160W. The maximum frequency of about 1.4GHz is determined from the power constraint. 2.3% of the chips are slower than 1.0GHz and 2.2% are faster than 1.4GHz. 10.7% of the chips have power consumption of more than 160W. Data was generated with a normal distribution of  $f = N(1.2, 0.1)$  and distribution for total power of  $P = 100f + e^{-10+10f/N(0,0.4)}$ , with dynamic power of  $100f$ . The leakage distribution is similar to the 0.18um technology in [9].

*Table 2.6* This table compares the rated power consumption of chips operating at the same clock frequency that are sold by Intel and AMD today [1][2][3][41][42][45][86]. Higher speed parts can operate at a lower supply voltage, reducing the power consumption. These lower power processors are sold at a premium.

Processor	Model	Codename	Technology (nm)	Frequency (GHz)	Voltage (V)	Power (W)	Power Increase
Athlon 64 X2	4800+	Windsor	90	2.40	1.25	65	
Athlon 64 X2	4800+	Windsor	90	2.40	1.35	89	×1.4
Athlon 64 X2	3800+	Windsor	90	2.00	1.08	35	
Athlon 64 X2	3800+	Windsor	90	2.00	1.25	65	×1.9
Athlon 64 X2	3800+	Windsor	90	2.00	1.35	89	×2.5
Athlon 64	3500+	Orleans	90	2.20	1.25	35	
Athlon 64	3500+	Orleans	90	2.20	1.40	62	×1.8
Turion 64	MT-40	Lancaster	90	2.20	1.20	25	
Turion 64	ML-40	Lancaster	90	2.20	1.35	35	×1.4
Core 2 Duo	T7600	Merom	65	2.33	1.30	35	
Xeon 5100	5148	Woodcrest	65	2.33	1.25	40	×1.1
Xeon 5100	5140	Woodcrest	65	2.33	1.40	65	×1.9
Core 2 Duo	T7200	Merom	65	2.00	1.30	35	
Xeon 5100	5130	Woodcrest	65	2.00	1.40	65	×1.9
Core Duo	L2500	Yonah	65	1.83	1.21	15	
Core Duo	T2400	Yonah	65	1.83	1.33	31	×2.1
Core Duo	L2400	Yonah	65	1.66	1.21	15	
Core Duo	T2300	Yonah	65	1.66	1.33	31	×2.1

Exploiting the variation in power consumption, Intel and AMD have been selling lower power chips at a premium. The power consumption of the cheaper, higher power parts is typically up to about  $2\times$  that of the low power chips, as shown in Table 2.6. Note that Intel's Merom (laptop), Conroe (desktop) and Woodcrest (server) chips are essentially the same [40], though voltages, caching strategies and so forth may be changed for lower power but lower performance for the laptop version.

Custom circuitry can be designed to ameliorate process variation in fabricated chips. In the Pentium 4, the clock is distributed across the chip to 47 domain buffers, which each have a 5 bit programmable register to remove skew from the clock signal in that domain to compensate for process variation [54]. A similar scheme was used to reduce clock skew in the 90nm XScale [21]. The body bias can be changed to adjust the transistor threshold voltage, and thus the delay and leakage power. Body bias can be applied at a circuit block level to reduce the standard deviation in clock frequency between dies from 4.1% to 0.21%, improving speed by 15% versus the slower chips without body bias, while limiting the range in leakage power to  $3\times$  for a 0.15 $\mu$ m test chip [88]. To do this, representative critical path delays and the leakage current must be measured while the bias is varied. Additional power rails are needed to route the alternate NMOS and PMOS

body bias voltages from the body bias generator circuitry, resulting in a 3% area overhead [88]. Forward body bias allowed  $V_{dd}$  to be reduced from 1.43V to 1.37V giving a 7% reduction in total power for a 0.13 $\mu$ m 5GHz 32-bit integer execution core [95].

#### **2.5.12.1 What's the problem?**

For ASIC parts that are sold for only a few dollars per chip, additional testing for power or speed binning is too expensive. Such ASICs are characterized under worst case process conditions to guarantee good yield. Thus ASIC power and speed are limited by the worst case parts. Without binning, there may be a power gap of  $\times 2$  versus custom chips that are binned. Custom chips that have the same market niche as ASICs have the same limitation on testing for binning, unless they are sold at a much higher price per chip.

The complicated circuitry and tight control of layout and routing required to compensate for process variation in a fabricated chip is not possible within an ASIC methodology.

#### **2.5.12.2 What can we do about it?**

To account for process variation, ASIC power may be characterized after fabrication. Parts may then be advertised with longer battery life. However, post-fabrication characterization of chip samples does not solve the problem if there is a maximum power constraint on a design. In this case, ASICs may be characterized at a less conservative power corner, which requires better characterization of yield for the standard cell library in that process. For typical applications, the power consumption is substantially less than peak power at the worst case power corner. Additional steps may be taken to limit peak power, such as monitoring chip temperature and powering down if it is excessive.

We estimate a power gap of up to  $1.3\times$  due to process variation for ASICs in comparison to custom designs that compensate for process variation, from analysis of a 15% increase in custom speed with the pipeline model in Chapter 3.

## **2.6 SUMMARY**

We compared synthesizable and custom ARM processors from 0.6 $\mu$ m to 0.13 $\mu$ m. We also examined discrete cosine transform cores, as an example of dedicated low power functional units. In these cases, there was a power gap of 3 to  $7\times$  between custom and ASIC designs.

We have given a top-down view of the factors contributing to the power gap between ASIC and custom designs. From our analysis, the most significant opportunity for power reduction in ASICs is using microarchitectural techniques to maintain performance while reducing power by voltage scaling.

Reducing the pipeline delay overhead and using pipelining to increase timing slack can enable substantial power savings by reducing the supply voltage and downsizing gates. Multiple threshold voltages may be used to limit leakage while enabling a lower  $V_{dd}$  to be used. Choosing a low power process technology and limiting the impact of process variation reduces power by a large factor.

In summary, at a tight performance constraint for a typical ASIC design, we believe that the power gap can be closed to within  $2.6\times$  by using these low power techniques with fine granularity standard cell libraries, careful RTL design and EDA tools targeting low power. The remaining gap is mostly from custom designs having lower pipelining overhead and using high speed logic on critical paths. Using a high speed logic style on critical paths can provide timing slack for significant power savings in custom designs. High speed logic styles are less robust and require careful layout, and thus are not amenable to use in an ASIC EDA methodology.

An example of combining low power and high performance design techniques on DSP functional macros is in Chapter 13. To improve performance and reduce power consumption, they used arithmetic optimizations, logic optimization, a finer grained library, voltage scaling from 1.2V to 1.0V, and bit-slicing. Performance improved from 94MHz to 177MHz and energy efficiency increased from 0.89MHz/mW to 2.78MHz/mW – a factor of  $3.1\times$ . This demonstrates the power savings that may be achieved by using low power techniques in ASICs.

The next chapter details our power and delay model that incorporates the major factors that contribute to the power gap between ASIC and custom. It includes pipelining, logic delay, voltage scaling and gate sizing. The logic delay is determined by factors such as the logic style, wire lengths after layout, process technology, and process variation which affects the worst case delay.

## 2.7 REFERENCES

- [1] AMD, AMD Processors for Desktops: AMD Athlon 64 Processor and AMD Sempron Processor, September 2006. <http://www.amdcompare.com/us-en/desktop/>
- [2] AMD, AMD Turion 64 Mobile Technology Model Number, Thermal Design Power, Frequency, and L2 Cache Comparison, September 2006. [http://www.amd.com/us-en/Processors/ProductInformation/0,,30\\_118\\_12651\\_12658,00.html](http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_12651_12658,00.html)
- [3] AMD, Processor Pricing, September 2006. [http://www.amd.com/us-en/Corporate/VirtualPressRoom/0,,51\\_104\\_609,00.html?redir=CPPR01](http://www.amd.com/us-en/Corporate/VirtualPressRoom/0,,51_104_609,00.html?redir=CPPR01)
- [4] ARM, ARM Cortex-A8, 2006. [http://www.arm.com/products/CPUs/ARM\\_Cortex-A8.html](http://www.arm.com/products/CPUs/ARM_Cortex-A8.html)
- [5] ARM, ARM Processor Cores. [http://www.armdevzone.com/open.nsf/htmlall/A944EB65693A4EB180256A440051457A/\\$File/ARM+cores+111-1.pdf](http://www.armdevzone.com/open.nsf/htmlall/A944EB65693A4EB180256A440051457A/$File/ARM+cores+111-1.pdf)
- [6] Bhavnagarwala, A., et al., “A Minimum Total Power Methodology for Projecting Limits on CMOS GSI,” *IEEE Transactions on VLSI Systems*, vol. 8, no. 3, June 2000, pp. 235-251.

- [7] Bisdounis, L., et al., "A comparative study of CMOS circuit design styles for low-power high-speed VLSI circuits," *International Journal of Electronics*, vol. 84, no. 6, 1998, pp. 599-613.
- [8] Bohr, M., "Staying Ahead of the Power Curve: Q&A with Intel Senior Fellow Mark T. Bohr," *Technology@Intel Magazine*, August 2006.
- [9] Borkar, S. et al., "Parameter variation and impact on Circuits and Microarchitecture," in *Proceedings of the Design Automation Conference*, 2003, pp. 338-342.
- [10] Brand, A., et al., "Intel's 0.25 Micron, 2.0 Volts Logic Process Technology," *Intel Technology Journal*, Q3 1998, 8 pp. <http://developer.intel.com/technology/itj/q31998/pdf/p856.pdf>
- [11] Burd, T., et al., "A Dynamic Voltage Scaled Microprocessor System," *IEEE Journal of Solid State Circuits*, vol. 35, no. 11, 2000, pp. 1571-1580.
- [12] Burd, T., *Energy-Efficient Processor System Design*, Ph.D. dissertation, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 2001, 301 pp.
- [13] Callaway, T., and Swartzlander, E., "Optimizing Arithmetic Elements for Signal Processing," *VLSI Signal Processing*, 1992, pp. 91-100.
- [14] Chandrakasan, A., and Brodersen, R., "Minimizing Power Consumption in Digital CMOS Circuits," in *Proceedings of the IEEE*, vol. 83, no. 4, April 1995, pp. 498-523.
- [15] Chang, A., "VLSI Datapath Choices: Cell-Based Versus Full-Custom," S.M. Thesis, Massachusetts Institute of Technology, February 1998, 146 pp. [http://cva.stanford.edu/publications/1998/achang\\_sm\\_thesis.pdf](http://cva.stanford.edu/publications/1998/achang_sm_thesis.pdf)
- [16] Chinnery, D., *Low Power Design Automation*, Ph.D. dissertation, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2006.
- [17] Chinnery, D., et al., "Automatic Replacement of Flip-Flops by Latches in ASICs," chapter 7 in *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*, Kluwer Academic Publishers, 2002, pp. 187-208.
- [18] Chinnery, D., and Keutzer, K., *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*, Kluwer Academic Publishers, 2002, 432 pp.
- [19] Chinnery, D., Nikolić, B., and Keutzer, K., "Achieving 550 MHz in an ASIC Methodology," in *Proceedings of the Design Automation Conference*, 2001, pp. 420-425.
- [20] Chu, K., and Pulfrey, D., "A Comparison of CMOS Circuit Techniques: Differential Cascode Voltage Switch Logic Versus Conventional Logic," *Journal of Solid-State Circuits*, vol. sc-22, no. 4, August 1987, pp. 528-532.
- [21] Clark, L., "The XScale Experience: Combining High Performance with Low Power from 0.18um through 90nm Technologies," presented at the Electrical Engineering and Computer Science Department of the University of Michigan, September 30, 2005. [http://www.eecs.umich.edu/vlsi\\_seminar/f05/Slides/VLSI\\_LClark.pdf](http://www.eecs.umich.edu/vlsi_seminar/f05/Slides/VLSI_LClark.pdf)
- [22] Clark, L., et al., "An Embedded 32-b Microprocessor Core for Low-Power and High-Performance Applications," *Journal of Solid-State Circuits*, vol. 36, no. 11, November 2001, pp. 1599-1608.
- [23] Clark, L., et al., "Standby Power Management for a 0.18um Microprocessor," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2002, pp. 7-12.
- [24] Clark, L., Morrow, M., and Brown, W., "Reverse-Body Bias and Supply Collapse for Low Effective Standby Power," *IEEE Transactions on VLSI Systems*, vol. 12, no. 9, 2004, pp. 947-956.
- [25] Cote, M., and Hurat, P. "Faster and Lower Power Cell-Based Designs with Transistor-Level Cell Sizing," chapter 9 in *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*, Kluwer Academic Publishers, 2002, pp. 225-240.

- [26] Dai, W., and Staepelaere, D., "Useful-Skew Clock Synthesis Boosts ASIC Performance," chapter 8 in *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*, Kluwer Academic Publishers, 2002, pp. 209-223.
- [27] Duarte, D., et al., "Evaluating run-time techniques for Leakage Power Reduction," in *Proceedings of the Asia and South Pacific Design Automation Conference*, 2002, pp. 31-38.
- [28] Fanucci, L., and Saponara, S., "Data driven VLSI computation for low power DCT-based video coding," *International Conference on Electronics, Circuits and Systems*, vol.2, 2002, pp. 541-544.
- [29] Fishburn, J., and Dunlop, A., "TILOS: A Posynomial Programming Approach to Transistor Sizing," in *Proceedings of the International Conference on Computer-Aided Design*, 1985, pp. 326-328.
- [30] Flynn, D., and Keating, M., "Creating Synthesizable ARM Processors with Near Custom Performance," chapter 17 in *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*, Kluwer Academic Publishers, 2002, pp. 383-407.
- [31] Furber, S., *ARM System-on-Chip Architecture*. 2nd Ed. Addison-Wesley, 2000.
- [32] Ganswijk, J., Chip Directory: ARM Processor family. <http://www.xs4all.nl/~ganswijk/chipdir/fam/arm/>
- [33] Garg, M., "High Performance Pipelining Method for Static Circuits using Heterogeneous Pipelining Elements," in *Proceedings of the European Solid-State Circuits Conference*, 2003, pp. 185-188.
- [34] Gong, J., et al., "Simultaneous buffer and wire sizing for performance and power optimization," *International Symposium on Low Power Electronics and Design*, 1996, pp. 271-276.
- [35] Greenhalgh, P., "Power Management Techniques for Soft IP," *Synopsys Users Group European Conference*, May 6, 2004, 12 pp.
- [36] Hare, C. 786 Processors Chart. <http://users.erols.com/chare/786.htm>
- [37] Harris, D., "High Speed CMOS VLSI Design – Lecture 2: Logical Effort & Sizing," November 4, 1997.
- [38] Harris, D., et al. "The Fanout-of-4 Inverter Delay Metric," unpublished manuscript, 1997, 2 pp. <http://odin.ac.hmc.edu/~harris/research/FO4.pdf>
- [39] Ho, R., Mai, K.W., and Horowitz, M., "The Future of Wires," in *Proceedings of the IEEE*, vol. 89, no. 4, April 2001, pp. 490-504.
- [40] Horan, B., "Intel Architecture Update," presented at the IBM EMEA HPC Conference, May 17, 2006. [www-5.ibm.com/fr/partenaires/forum/hpc/intel.pdf](http://www-5.ibm.com/fr/partenaires/forum/hpc/intel.pdf)
- [41] Intel, Dual-Core Intel Xeon Processor 5100 Series, Features, September 2006. <http://www.intel.com/cd/channel/reseller/asmona/eng/products/server/processors/5100/feature/index.htm>
- [42] Intel, Intel Core Duo Processor Specifications, September 2006. <http://www.intel.com/products/processor/coreduo/specs.htm>
- [43] Intel, Intel XScale Microarchitecture: Benchmarks. <http://developer.intel.com/design/intelxscale/benchmarks.htm>
- [44] Intel, Meet the World's First 45nm Processor, 2007. [http://www.intel.com/technology/silicon/45nm\\_technology.htm?iid=search](http://www.intel.com/technology/silicon/45nm_technology.htm?iid=search)
- [45] Intel, Processor Number Feature Table, September 2006. [http://www.intel.com/products/processor\\_number/proc\\_info\\_table.pdf](http://www.intel.com/products/processor_number/proc_info_table.pdf)
- [46] Ishihara, F., Sheikh, F., and Nikolić, B., "Level Conversion for Dual-Supply Systems," *IEEE Transactions on VLSI Systems*, vol. 12, no. 2, 2004, pp. 185-195.
- [47] Ito, M., Chinnery, D., and Keutzer, K., "Low Power Multiplication Algorithm for Switching Activity Reduction through Operand Decomposition," in *Proceedings of the International Conference on Computer Design*, 2003, pp. 21-26.



- [48] Jan, C., et al., "A 65nm Ultra Low Power Logic Platform Technology using Uni-axial Strained Silicon Transistors," *Technical Digest of the International Electron Devices Meeting*, 2005, pp. 60-63.
- [49] Keshavarzi, A., et al., "Effectiveness of Reverse Body Bias for Leakage Control in Scaled Dual Vt CMOS ICs," *International Symposium on Low-Power Electronics Design*, 2001, pp. 207-212.
- [50] Kim, J., "GHz ARM Processor Design," tutorial at the International System-on-Chip Conference, October 23, 2002.
- [51] Kitahara, T., et al., "A clock-gating method for low-power LSI design," in *Proceedings of the Asia and South Pacific Design Automation Conference*, 1998, pp. 307-312.
- [52] Kosonocky, S., et al., "Low-Power Circuits and Technology for Wireless Data Systems," *IBM Journal of Research and Development*, vol. 47, no. 2/3, March/May 2003, pp. 283-298.
- [53] Kulkarni, S., and Sylvester, D., "Fast and Energy-Efficient Asynchronous Level Converters for Multi-VDD Design," *IEEE Transactions on VLSI Systems*, September 2004, pp. 926-936.
- [54] Kurd, N.A, et al., "A Multigigahertz Clocking Scheme for the Pentium® 4 Microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 11, November 2001, pp. 1647-1653.
- [55] Kuroda, T., "Low-power CMOS design in the era of ubiquitous computers," *OYO BUTURI*, vol. 73, no. 9, 2004, pp. 1184-1187.
- [56] Lee, D., et al., "Analysis and Minimization Techniques for Total Leakage Considering Gate Oxide Leakage," *proceedings of the Design Automation Conference*, 2003, pp. 175-180.
- [57] Lee, D., et al., "Simultaneous Subthreshold and Gate-Oxide Tunneling Leakage Current Analysis in Nanometer CMOS Design," in *Proceedings of the International Symposium on Quality Electronic Design*, 2003, pp. 287-292.
- [58] Lee, D., and Blaauw, D., "Static Leakage Reduction through Simultaneous Threshold Voltage and State Assignment," in *Proceedings of the Design Automation Conference*, 2003, pp. 191-194.
- [59] Lee, D., Blaauw, D., and Sylvester, D., "Gate Oxide Leakage Current Analysis and Reduction for VLSI Circuits," *IEEE Transactions on VLSI Systems*, vol. 12, no. 2, 2004, pp. 155-166.
- [60] Levy, M., "Samsung Twists ARM Past 1GHz," *Microprocessor Report*, October 16, 2002.
- [61] McDonald, C., "The Evolution of Intel's Copy Exactly! Technology Transfer Method," *Intel Technology Journal*, Q4 1998. <http://developer.intel.com/technology/itj/q41998/pdf/copyexactly.pdf>
- [62] Montanaro, J., et al., "A 160MHz, 32-b, 0.5W, CMOS RISC Microprocessor," *Journal of Solid-State Circuits*, vol. 31, no. 11, 1996, pp. 1703-1714.
- [63] Moyer, B., "Low-Power Design for Embedded Processors," in *Proceedings of the IEEE*, vol. 89, no. 11, November 2001.
- [64] Mutoh, S., et al., "1-V Power Supply High-Speed Digital Circuit Technology with Multithreshold-Voltage CMOS," *Journal of Solid-State Circuits*, vol. 30, no. 8, 1995, pp. 847-854.
- [65] Narendra, S., et al., "Comparative Performance, Leakage Power and Switching Power of Circuits in 150 nm PD-SOI and Bulk Technologies Including Impact of SOI History Effect," *Symposium on VLSI Circuits*, 2001, pp. 217-218.
- [66] Nassif, S., "Delay Variability: Sources, Impact and Trends," *International Solid-State Circuits Conference*, 2000.
- [67] Ostrander, D., "Logic Technology and Manufacturing," slides presented at AMD's Technology Analyst Day, 2006. [http://www.amd.com/us-en/assets/content\\_type/DownloadableAssets/DarylOstranderAMDAAnalystDay.pdf](http://www.amd.com/us-en/assets/content_type/DownloadableAssets/DarylOstranderAMDAAnalystDay.pdf)

- [68] Pradhan, D., et al., "Gate-Level Synthesis for Low-Power Using New Transformations," *International Symposium on Low Power Electronics and Design*, 1996, pp. 297-300.
- [69] Puri, R., et al., "Pushing ASIC Performance in a Power Envelope," in *Proceedings of the Design Automation Conference*, 2003, pp. 788-793.
- [70] Quinn, J., *Processor98: A Study of the MPU, CPU and DSP Markets*, Micrologic Research, 1998.
- [71] Rabaey, J.M., *Digital Integrated Circuits*. Prentice-Hall, 1996.
- [72] Ricci, F., "A 1.5 GHz 90 nm Embedded Microprocessor Core," *Digest of Technical Papers of the Symposium on VLSI Circuits*, 2005, pp. 12-15.
- [73] Richardson, N., et al., "The iCORE™ 520MHz Synthesizable CPU Core," Chapter 16 of *Closing the Gap Between ASIC and Custom*, 2002, pp. 361-381.
- [74] Shen, W., Lin, J., and Wang, F., "Transistor Reordering Rules for Power Reduction in CMOS Gates," in *Proceedings of the Asia South Pacific Design Automation Conference*, 1995, pp. 1-6.
- [75] Shenoy, N., "Retiming Theory and Practice," *Integration, The VLSI Journal*, vol. 22, no. 1-2, August 1997, pp. 1-21.
- [76] Simonen, P., et al., "Comparison of bulk and SOI CMOS Technologies in a DSP Processor Circuit Implementation," *International Conference on Microelectronics*, 2001.
- [77] Singh, D., et al., "Power Conscious CAD Tools and Methodologies: a Perspective," in *Proceedings of the IEEE*, vol. 83, no. 4, April 1995, pp. 570-594.
- [78] Sirichotiyakul, S., et al., "Stand-by Power Minimization through Simultaneous Threshold Voltage Selection and Circuit Sizing," in *Proceedings of the Design Automation Conference*, 1999, pp. 436-441.
- [79] Staszewski, R., Muhammad, K., and Balsara, P., "A 550-MSample/s 8-Tap FIR Digital Filter for Magnetic Recording Read Channels," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, 2000, pp. 1205-1210.
- [80] Stojanovic, V., and Oklobdzija, V., "Comparative Analysis of Master-Slave Latches and Flip-Flops for High-Performance and Low-Power Systems," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 4, April 1999, pp. 536-548.
- [81] Stok, L., et al., "Design Flows," chapter in the *CRC Handbook of EDA for IC Design*, CRC Press, 2006.
- [82] Sylvester, D. and Keutzer, K., "Getting to the Bottom of Deep Sub-micron," in *Proceedings of the International Conference on Computer Aided Design*, November 1998, pp. 203-211.
- [83] Synopsys, *Design Compiler User Guide*, version U-2003.06, June 2003, 427 pp.
- [84] Synopsys, *Power Compiler User Guide*, version 2003.06, 2003.
- [85] Takahashi, M., et al., "A 60-mW MPEG4 Video Codec Using Clustered Voltage Scaling with Variable Supply-Voltage Scheme," *Journal of Solid-State Circuits*, vol. 33, no. 11, 1998, pp. 1772-1780.
- [86] techPowerUp! CPU Database, August 2006. <http://www.techpowerup.com/cpubd/>
- [87] Thompson, S., et al., "An Enhanced 130 nm Generation Logic Technology Featuring 60 nm Transistors Optimized for High Performance and Low Power at 0.7 – 1.4 V," *Technical Digest of the International Electron Devices Meeting*, 2001, 4 pp.
- [88] Tschanz, J. et al., "Adaptive Body Bias for Reducing the Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," *IEEE International Solid-State Circuits Conference*, 2002.
- [89] Tschanz, J., et al., "Dynamic Sleep Transistor and Body Bias for Active Leakage Power Control of Microprocessors," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 11, 2003, pp. 1838-1845.
- [90] Tsui, C., et al., "Low Power State Assignment Targeting Two- And Multi-level Logic Implementations," in *Proceedings of the International Conference on Computer-Aided Design*, 1994, pp. 82-87.

- [91] Tyagi, S., et al., "An advanced low power, high performance, strained channel 65nm technology," *Technical Digest of the International Electron Devices Meeting*, 2005, pp. 245-247.
- [92] Usami, K., et al., "Automated Selective Multi-Threshold Design For Ultra-Low Standby Applications," in *Proceedings of the International Symposium on Low Power Design*, 2002, pp. 202-206.
- [93] Usami, K., and Horowitz, M., "Clustered voltage scaling technique for low power design," in *Proceedings of the International Symposium on Low Power Design*, 1995, pp. 3-8.
- [94] Usami, K., and Igarashi, M., "Low-Power Design Methodology and Applications Utilizing Dual Supply Voltages," in *Proceedings of the Asia and South Pacific Design Automation Conference*, 2000, pp. 123-128.
- [95] Vangal, S., et al., "5GHz 32b Integer-Execution Core in 130nm Dual-V<sub>T</sub> CMOS," *Digest of Technical Papers of the IEEE International Solid-State Circuits Conference*, 2002, pp. 334-335, 535.
- [96] Veendrick, H., "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," *Journal of Solid-State Circuits*, vol. SC-19, August 1984, pp. 468-473.
- [97] Virtual Silicon. <http://www.virtual-silicon.com/>
- [98] Weicker, R., "Dhrystone: A Synthetic Systems Programming Benchmark," *Communications of the ACM*, vol. 27, no. 10, 1984, pp. 1013-1030.
- [99] Weste, N., and Eshraghian, K., *Principles of CMOS VLSI Design*, Addison-Wesley, 1992.
- [100] Wolfe, A., "Intel Clears Up Post-Tejas Confusion," VARBusiness magazine, May 17, 2004. <http://www.varbusiness.com/sections/news/breakingnews.jhtml?articleId=18842588>
- [101] Xanthopoulos, T., and Chandrakasan, A., "A Low-Power DCT Core Using Adaptive Bitwidth and Arithmetic Activity Exploiting Signal Correlations and Quantization," *Journal of Solid State Circuits*, vol. 35, no. 5, May 2000, pp. 740-750.
- [102] Xanthopoulos, T., and Chandrakasan, A., "A Low-Power IDCT Macrocell for MPEG-2 MP@ML Exploiting Data Distribution Properties for Minimal Activity," *Journal of Solid State Circuits*, vol. 34, May 1999, pp. 693-703.
- [103] Zimmerman, R., and Fichtner, W., "Low-Power Logic Styles: CMOS Versus Pass-Transistor Logic," *Journal of Solid-State Circuits*, vol. 32, no. 7, July 1997, 1079-1090.
- [104] Zlatanovici, R., and Nikolić, B., "Power-Performance Optimal 64-bit Carry-Lookahead Adders," *European Solid-State Circuits Conference*, 2003, pp. 321-324.

<http://www.springer.com/978-0-387-25763-1>

Closing the Power Gap between ASIC & Custom  
Tools and Techniques for Low Power Design

Chinnery, D.; Keutzer, K.

2007, XII, 388 p. 138 illus., Hardcover

ISBN: 978-0-387-25763-1