Low-Power Digital CMOS Design: A Survey

Krister Landernäs *

June 14, 2005

*Department of Computer Science and Electronics, Mälardalen University

Abstract

The aim of this document is to provide the reader with an overview of low-power digital design trends. Several research publications and books have been studied by the author and the knowledge gained has been compiled into this report. Low-power techniques adopted at different levels of the design process are presented and industrial application examples are given.

Contents

1 Motivation

In the last decade, low-power has become an important goal when designing digital systems. There are several reasons for this. The main motivation is perhaps the increasing market for portable electronic devices. Lap-top computers, cellular phones and MP3 players are today common consumer products. Maintaining reasonable battery life-time with an increasing number of functions integrated in these products requires a low-power design approach. The fact that battery life-times are only predicted to increase with 30-40% over the next years only further emphasises the need for low-power designs [1].

Decreasing the power consumption is also important in applications which are not battery-powered, e.g. stationary computers. Today, the power consumption for high-performance microprocessors can exceed 100 Watt. Without cooling the circuit would literally start to burn. Much time and effort is spent on packaging and cooling in order to remove superfluous heat. This increases developing time and the total manufacturing cost for the application.

2 Introduction

In this report, we study the power consumption of digital CMOS (complementary metal oxide semiconductor) circuits. Today, the majority of digital ICs are manufactured using CMOS, although there are exceptions (e.g. BiCMOS). This was not the case in the 70's, when most digital circuits were realized using only NMOS technology. The change into using complementary logic (both PMOS and NMOS) was caused by the fact that the static power consumption for NMOS circuits became too high. The solution was to use CMOS circuits which have (ideally) no direct path between power supply and ground in steady state and, hence, no static power consumption. This is illustrated in Fig. 1, where a typical CMOS circuit is shown. Note that the PMOS and NMOS net cannot conduct simultaneously.

The power consumption for a CMOS circuit can be expressed as,

$$P = P_{\text{dyn}} + P_{\text{stat}} + P_{\text{short}} + P_{\text{leak}}, \tag{1}$$

where P_{dyn} is the dynamic power consumption caused by charging and discharging capacitors driven by the circuit. Ideally the PMOS and NMOS net (see Fig. 1) cannot conduct at the same time. Unfortunately, a transistor is not an ideal switch and as a result a current will flow between the rails in steady state. This current causes a static power consumption P_{stat}. During input transitions the PMOS and NMOS net can, for a short time period, conduct simultaneously resulting in a short-circuit dissipation P_{short}. P_{leak} is caused by substrate and sub-threshold currents. Note, some publications does not distinguish between P_{stat} and P_{leak}.

Historically, for technologies above $0.8\mu m$, P_{dyn} dominated the total power consumption (over 90%). This is no longer the case as P_{leak} becomes more significant in deep-submicron technologies. The dynamic power consumption is

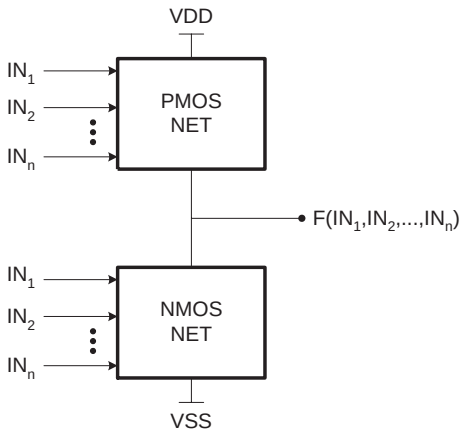

Figure 1: CMOS circuit.

still, however, important to minimize since it can lead to considerable power savings. Let us consider a typical CMOS inverter shown in Fig. 2. A model for

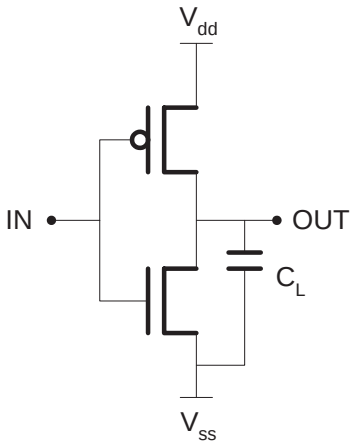

Figure 2: CMOS inverter.

the dynamic power consumption in a CMOS inverter (approximately true for general CMOS gates) is given by,

$$P_{\mathrm{dyn}} = \alpha f_{\mathrm{clk}} C_{\mathrm{L}} V_{\mathrm{dd}}^2, \tag{2}$$

where α is the activity factor, f_{clk} is the clock frequency and C_{L} is the capacitive load being switched. The activity factor is the average number of transitions on the output of the gate during one clock cycle. Typically, $\alpha \leq 1$, but due to

glitches some systems may experience an activity factor larger than one. From (2), we have three parameters that we can target when trying to minimize the dynamic power consumption. The switching activity, i.e. αf_{clk}, the capacitive load and the voltage supply level. Their influence on the total power consumption and how we can affect these parameters will become clear in the following sections.

Power savings can be made at all abstraction levels during the design process. In this report, the levels are divided according to Fig. 3, where typical power saving examples are also given. The following sections will discuss each

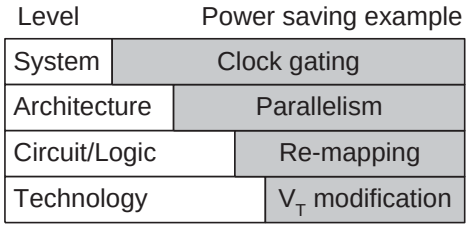

Figure 3: Levels of abstraction.

abstraction level and give examples on how power savings can be performed.

3 Technology Optimization

The quadratic relationship between P_{dyn} and V_{dd} (see (2)) has been used extensively over years to decrease the power consumption. The power dissipation can be lowered considerably by decreasing the supply voltage. This is known as *power-driven voltage scaling*. Reducing V_{dd} will, however, also increase the delay (T_{d}) of the circuit since,

$$T_{\mathrm{d}} = \frac{2CV_{\mathrm{dd}}}{\beta \left(V_{\mathrm{dd}} - V_{\mathrm{T}}\right)^{m}}, \tag{3}$$

where $1 \leq m \leq 2$ for short-channel devices [2]. It is important to note that the delay of a CMOS circuit is not determined by V_{dd} alone, but rather the difference between V_{dd} and the threshold voltage V_{T}. Both these parameters must be considered when optimizing the power-delay product of a CMOS circuit. Optimizing V_{dd} and V_{T} for low-power will result in very low values for both parameters. A threshold voltage of a few hundred millivolts and $V_{\mathrm{dd}} = 2V_{\mathrm{T}}$ is obtained. This is known as *ultra-low-power* CMOS (ULP). There are two main reasons why ULP is not so commonly used. Threshold control becomes hard and sub-threshold currents can become large (an exponential relationship between sub-threshold currents and V_{T} exists). Sub-threshold currents are also very temperature sensitive making temperature control difficult in ULP.

Voltage scaling is commonly used in industrial applications to lower the power consumption. The voltage level is, however, not scaled to the extreme for

reasons stated above. In Example 1 an application where voltage scaling was used is shown.

Example 1:
The StrongARM processor was designed in a three-metal layer 0.35μm process. It was later developed for low-power, where the voltage supply was scaled from 3.45 V to 1.5 V with $V_{\mathrm{T}} = 0.35V$. A power reduction of 5.3 times was obtained [3].

In high-performance systems, the power consumption generated by leakage currents is becoming an increasing problem. Fast transistors with low threshold voltages generate high static currents. One solution to the problem is to use a dual threshold logic. High-performance parts of a system can then be implemented using low-threshold devices while subsystems that are not timing critical can use a higher threshold level. A more dynamic solution is to increase the threshold level when the system, or parts of it, is in standby. Changing the threshold level is commonly realized by applying substrate biasing, see Example 2.

Example 2:
The MPEG4 Video Codec prototype adopted the variable threshold voltage scheme to reduce power dissipation. Substrate biasing is exploited to dynamically adjust the threshold. V_{T} is set to 0.2 V in active mode and 0.55 V in standby mode [3].

4 Logic Optimization

A number of logic optimization techniques for low-power have been developed over the years. These methods usually target either the switching activity or the capacitance of the logic. Today, due to design complexity, most logic optimization is done by synthesis tools. Fortunately many of the low-power methods discussed in this section can be performed by the tools.

Traditional cell libraries were developed for high-speed. This implies large transistors with high power consumption. Libraries with low-power capabilities are, however, becoming more common. Usually a low-power model of each cell is included in the library. This cell is optimized with smaller transistor sizes and shorter interconnect. In a synchronous design the number of flip-flops is usually large. It is therefore important that the library contains registers optimized for low-power. Naturally, the synthesis tool must include power optimization routines in order to take advantage of a low-power library.

The synthesis tool will try to minimize power consumption after all timing requirements are met. This is done by replacing cells in the netlist to decrease the power consumption. Any replacement that lead to better power characteristics without violating timing will be accepted. This procedure is known as *re-mapping*. Re-mapping may cause a cell to be resized or a group of cells to be

replaced with a more complex cell. This is an intricate optimization problem involving both switching power as well as internal power on a number of cells at the same time.

In addition to re-mapping, *pin swapping* is also used to minimize power dissipation. Pin swapping changes the input order to connect signals with high activity to low-capacitance inputs. Both re-mapping and pin swapping are time consuming operations which are applied to local parts of a netlist. As a result the power savings are quite small with 10-20% savings usually obtained.

5 Architecture Optimization

In arithmetic units, e.g., adders and multipliers glitches can consume a large amount of power. Glitches are caused by mismatch in signal arrival times and will increase the activity factor α (given in (2)). A simple example is shown in Fig. 4, where a mismatch in input signal arrival time causes a glitch on the output of the XNOR gate. These unwanted signal transitions can either

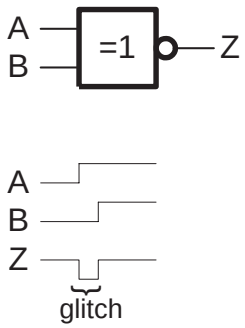

Figure 4: Glitch generation.

be generated by or propagated through the logic and can increase the power dissipation considerably. In Table 1, the increase in power consumption caused by glitches are given for some common arithmetic units [4].

Arithmetic unit	Power increase
8-bit Ripple carry adder	30%
8x8 Array multiplier	150%
16x16 Array multiplier	326%

Table 1: Increase in power consumption due to glitches.

The amount of glitches can be reduced by equalizing path delays. This can be achieved by inserting delay elements or registers (pipelining). Different logic

units can be more or less prone to generate glitches. A Wallace-tree multiplier has more balanced paths than an Array multiplier and will, therefore, experience fewer glitches (see Example 3).

Example 3:
The MAC unit in the StrongARM processor is based on a Wallace-tree multiplier and a Carry-look-ahead adder. Power savings of 23% were made compared to using an Array multiplier [3].

In the former section, we discussed voltage scaling and it's impact on power consumption. Voltage scaling can only be used when a system experiences positive timing slack (timing constraints are met). Parallelism can be exploited for a system to obtain positive slack and, thereby, enable voltage scaling. In Fig. 5, a simple arithmetic unit is shown. We can retain the same operating

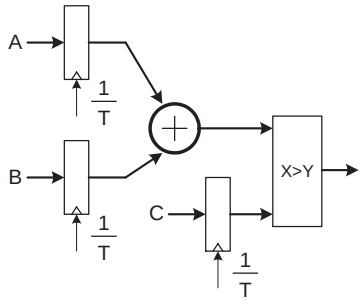

Figure 5: Arithmetic unit.

speed by using two parallel structures working at half the frequency, see Fig. 6. Since the timing constraints for the adders and comparators have decreased the voltage supply level can be lowered accordingly. Measurements on the systems shown in Fig. 5 and Fig. 6 using a $2\mu m$ process shows that power savings of about 60% can be obtained [2]. The trade-off in this case is increased area and routing versus lower power consumption.

6 System Optimization

Performing optimizations on the system level can have a significant impact on the power consumption. A common approach is to partition the system and study the workload for different subsystems. Power savings can be made by disabling the power supply and/or clock signal on parts of the system that are idle for a period of time. This is known as *dynamic power management*.

The clock network (flip-flops, buffers) of a subsystem will consume power even if the system performs no computation. Clock gating is used to prevent this by turning off the clock signal to inactive blocks. A clock gating example

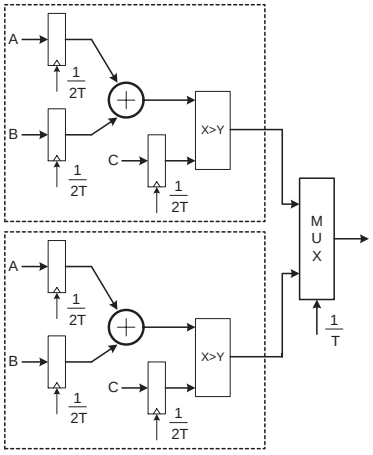

Figure 6: Parallel arithmetic unit.

is shown in Fig. 7, where the input and state value are used in a combinational block to disable the clock signal. The latch is needed to remove glitches. There

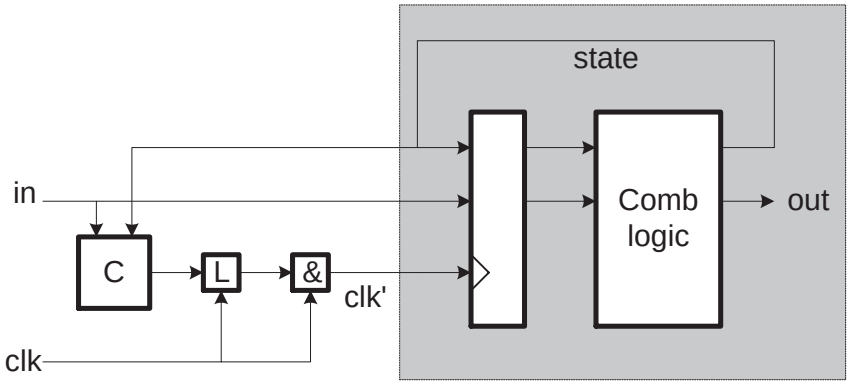

Figure 7: Clock gating example.

are some drawbacks with clock gating concerning system performance. A latency is introduced in the system when enabling the clock signal. The total capacitive load on the clock network will also increase. Note that clock gating does not affect the power consumption caused by sub-threshold currents (i.e. P_{leak}). To eliminate P_{leak} the power supply must be disabled.

The penalty for disabling the power supply or clock signal is a latency when returning from the idle state. Dynamic power management is, therefore, a trade-off between power and performance. This is illustrated in Example 4.

Example 4:
The latency for the CPU of the TMS320C5x DSP is $50\mu s$ when returning from idle mode. This is the time it takes for the on-chip PLL to lock to the external clock signal [3].

Voltage scaling (discussed in Section 3) can be applied to parts of a system to obtain a lower power consumption. The challenge is to efficiently generate multiple voltage levels and distribution grids. Power efficient level converters are essential when designing an application with multiple voltage levels. In digital IC design different voltage levels are often used for I/O and core logic. The UMC $0.18\mu m$ standard cell process uses 3.3 V and 1.8 V for I/O and core logic, respectively [5].

In Fig. 8, a multiple voltage level example is given. The clock tree is divided into two parts, where the intermediate stage uses a lower voltage level. This reduces voltage swing and, hence, the dynamic power consumption for these buffers [6]. The dynamic power consumption for this low-power stage can be written as,

$$P_{\mathrm{dyn}} = \alpha f_{\mathrm{clk}} C_{\mathrm{L}} V_{\mathrm{dd}} V_{\mathrm{s}}, \tag{4}$$

where V_{s} is the reduced voltage swing. Experiments show that power savings of about 45% can be made for a 0.25μm process [6].

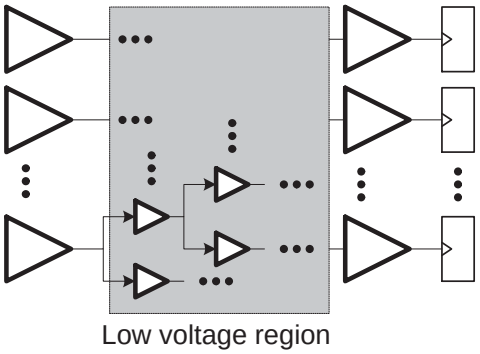

Figure 8: Example of multiple voltage levels in a clock tree.

A static voltage scaling is not always possible due to performance constraints. Considerable power savings can still be obtained using *dynamic voltage scaling*. The goal with dynamic voltage scaling is to lower the voltage supply level and clock frequency when the system is not operating at maximum performance. There are several ways to implement this functionality, one is shown in Fig. 9. In Fig. 9, the ring oscillator will match the critical path frequency of the chip. The operating system (o.s.) sets the desired frequency and this value is compared with the actual value from the oscillator. The frequency error is used as a feedback signal to control the voltage supply level which in turn affects the frequency of the oscillator. It should be clear from Fig. 9 that implementing a

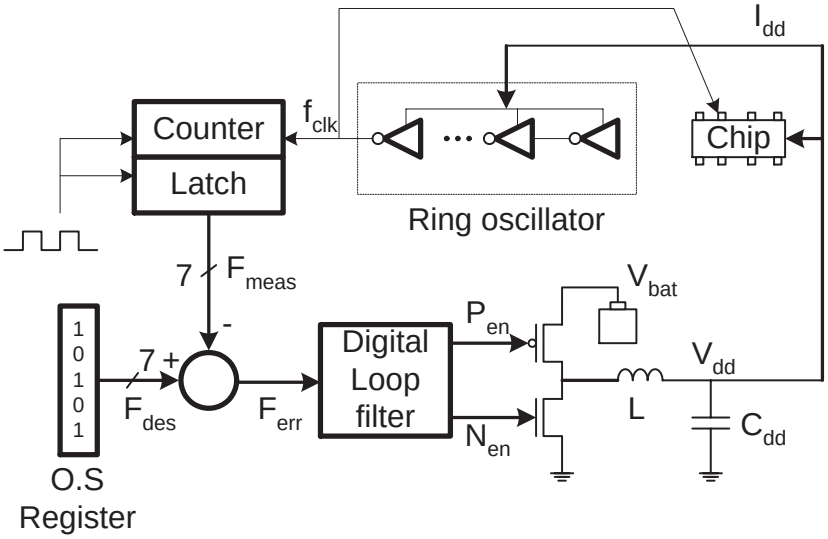

Figure 9: Principle of dynamic voltage scaling.

dynamic voltage supply is a rather complex task. Power savings can, however, be considerable. In Example 5 an application is given where dynamic voltage scaling is utilized.

Example 5:
Transmeta's Crusoe microprocessor with LongRun technology has 32 levels of clock frequency and supply voltage. The clock frequency ranges from 200MHz-700MHz and the voltage supply level from 1.1V-1.6V. It takes $20\mu s$ to change level [7].

7 Conclusions

Low-power digital CMOS design continues to be an important research topic. A shift in focus can be observed in recent years. Static approaches (such as static voltage scaling) are being replaced with more dynamic solutions. Technology scaling and higher performance demands are the main motivation behind this change. Simply lowering the voltage supply or threshold voltage for a whole system is becoming harder in high-performance applications. Instead a dynamic approach must be used where parts of the system are targeted for low-power as the workload permits it.

References

[1] J. M. Rabaey, A. Chandrakasan and B. Nikolic, *Digital Integrated Circuits,* Prentice Hall, 2003.

[2] A. Chandrakasan and R. W. Brodersen, "Minimizing power consumption on digital CMOS circuits," *IEEE Proceedings*, Vol-83, Issue. 4, pp. 498–523, April 1995.

[3] L. Benini, G. De Micheli and E. Macii, "Designing low-power circuits: practical recipes," *IEEE Circuits and Systems Magazine*, Vol-1, Issue. 1, pp. 6–25, 2001.

[4] H. Veendrick, *Deep-Submicron CMOS IC's,* Kluwer, 2000.

[5] High Performance 0.18μ Standard cell library data book, Rev.2.1, Virtual Silicon Technology, Inc., Sunnyvale, CA, January 2001.

[6] J. Pangjun and S. S. Sapatnekar, "Low-power clock distribution using multiple voltages and reduced swings," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol-10, No. 3, pp. 309–318, June 2002.

[7] Crusoe Processor model TM5700/TM5900 databook, Rev.1.0, Transmeta Corporation, 2004.