LECTURE NOTES
ON
LOW POWER - VLSI
(2018 – 2019)
IV B. Tech II Semester (R15)
Mrs. N PRANAVI, Assistant Professor

CHADALAWADA RAMANAMMA ENGINEERING COLLEGE
(AUTONOMOUS)
Chadalawada Nagar, Renigunta Road, Tirupati – 517 506
Department of Electronics and Communication Engineering
SYLLABUS

UNIT I
Introduction, Historical background, why low power, sources of power dissipations, low-power design methodologies.
MOS Transistors: introduction, the structure of MOS Transistor, the Fluid model, Modes of operation of MOS Transistor, Electrical characteristics of MOS Transistors, MOS Transistors as a switch.

UNIT II
MOS Inverters: introduction, inverter and its characteristics, configurations, inverter ratio in different situations, switching characteristics, delay parameters, driving parameters, driving large capacitive loads.
MOS Combinational Circuits: introduction, Pass-Transistor logic, Gate logic, MOS Dynamic Circuits.

UNIT III
Sources of Power Dissipation: introduction, short-circuit power dissipation, switching power dissipation, glitching power dissipation, leakage power dissipation.
Supply voltage scaling for low power: introduction, device features size scaling, architecture-level approaches, voltage scaling, multilevel voltage scaling, challenges, dynamic voltage and frequency scaling, adaptive voltage scaling.

UNIT IV
Minimizing Switched Capacitance: introduction, system-level approaches, transmeta’s Crusoe processor, bus encoding, clock gating, gated-clock FSMs, FSM state encoding, FSM Partitioning, operand isolation, precomputation, logic styles for low power.

UNIT V
Chapter 1
Introduction

Introduction
Design for low power has become nowadays one of the major concerns for complex, very-large-scale-integration (VLSI) circuits. Deep submicron technology, from 130 nm onwards, poses a new set of design problems related to the power consumption of the chip. Tens of millions of gates are nowadays being implemented on a relatively small die, leading to a power density and total power dissipation that are at the limits of what packaging, cooling, and other infrastructure can support. As technology has shrunk to 90 nm and below, the leakage current has increased dramatically, and in some 65-nm designs, leakage power is nearly as large as dynamic power. So it is becoming impossible to increase the clock speed of high-performance chips as technology shrinks and the chip density increases, because the peak power consumption of these chips is already at the limit and cannot be increased further. Also, the power density leads to reliability problems because the mean time to failure decreases with temperature. Besides, the timing degrades and the leakage currents increase with temperature. For battery-powered devices also, this high on-chip power density has become a significant problem, and techniques are being used in these devices from software to architecture to impl
Historical Background [1]

The invention of transistor by William Shockley and his colleagues at Bell Laboratories, Murray Hills, NJ, ushered in the “solid state” era of electronic circuits and systems. Within few years after the invention, transistors were commercially available and almost all electronic systems started carrying the symbol “solid state,” signifying the conquest of the transistor over its rival—the vacuum tube. Smaller size, lower power consumption, and higher reliability were some of the reasons that made it a winner over the vacuum tube. About a decade later, Shockley and his colleagues, John Bardeen and Walter Brattain, of Bell Laboratories were rewarded with a Nobel Prize for their revolutionary invention.

The tremendous success of the transistor led to vigorous research activity in the field of microelectronics. Later, Shockley founded a semiconductor industry. Some of his colleagues joined him or founded semiconductor industries of their own. Gordon Moore, a member of Shockley’s team, founded Fairchild and later Intel. Research engineers of Fairchild developed the first planner transistor in the late 1950s, which was the key to the development of integrated circuits (ICs) in 1959. Planner technology allowed realization of a complete electronic circuit having a number of devices and interconnecting them on a single silicon wafer. Within few years of the development of ICs, Gordon Moore, director, Research and Development Laboratories, Fairchild Semiconductor, wrote an article entitled “Cramming More
Components onto Integrated Circuits” in the April 19, 1995 issue of the *Electronics Magazine*. He was asked to predict what would happen over the next 10 years in the semiconductor component industry. Based on the very few empirical data, he predicted that by 1975, it would be possible to cram as many as 65,000 components onto a single silicon chip of about one fourth of a square inch. The curve, which Moore used to make his prediction, is shown in Fig. 1.1. The starting point was the year 1959—the year of production of the first planner transistor. The other three points are based on the ICs made by Fairchild in the early 1960s, including an IC with 32 components in production in 1964. The last one was an IC to be produced in 1965 with 64 components. By extrapolating the plot to the year 1975, which he used for the purpose of prediction, he observed that “by 1975, the number of components per IC for minimum cost will be 65,000.” He also concluded that component density in an IC would double every year.

Later in the year 1975, Moore revisited the topic at the IEEE International Electron Devices Meeting and observed that his 10-year-old forecast of 65,000 components was on the mark. However, he revised his prediction rate from 1 year to 18 months, that is, the component density would double every 18 months. This became known as Moore’s law. Again after 30 years, Moore compared the actual performance of two kinds of devices—random-access memories (RAM) and microprocessors. Amazingly, it was observed that both kinds traced the slope fairly closely to the revised 1975 projection.

Moore’s law acted as a driving force for the spectacular development of IC technology leading to different types of products. Based on the scale of integration, the IC technology can be divided into five different categories, as summarized in Table 1.1. The first half of the 1960s was the era of small-scale integration (SSI), with about ten planner transistors on a chip. The SSI technology led to the fab-


Table 1.1 Evolution of IC technology

<table>
<thead>
<tr>
<th>Year</th>
<th>Technology</th>
<th>Number of components</th>
<th>Typical products</th>
</tr>
</thead>
<tbody>
<tr>
<td>1947</td>
<td>Invention of transistor</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>1950–1960</td>
<td>Discrete components</td>
<td>1</td>
<td>Junction diodes and transistors</td>
</tr>
<tr>
<td>1961–1965</td>
<td>Small-scale integration</td>
<td>10–100</td>
<td>Planner devices, logic gates, flip-flops</td>
</tr>
<tr>
<td>1966–1970</td>
<td>Medium-scale integration</td>
<td>100–1000</td>
<td>Counters, MUXs, decoders, adders</td>
</tr>
<tr>
<td>1971–1979</td>
<td>Large-scale integration</td>
<td>1000–20,000</td>
<td>8-bit μP, RAM, ROM</td>
</tr>
<tr>
<td>1980–1984</td>
<td>Very-large-scale integration</td>
<td>20,000–50,000</td>
<td>DSPs, RISC processors, 16-bit, 32-bit μP</td>
</tr>
<tr>
<td>1985–</td>
<td>Ultra-large-scale integration</td>
<td>&gt; 50,000</td>
<td>64-bit μP, dual-core μP</td>
</tr>
</tbody>
</table>


Evolution of IC technology:

- **1947**: Invention of transistor
- **1950–1960**: Discrete components
- **1961–1965**: Small-scale integration
- **1966–1970**: Medium-scale integration
- **1971–1979**: Large-scale integration
- **1980–1984**: Very-large-scale integration
- **1985–**: Ultra-large-scale integration

Evolution of IC technology has been characterized by a series of innovations and advancements in technology. The table above provides a summary of the key milestones and transitions in IC technology from the invention of the transistor in 1947 to the present day. Each era has been marked by a significant increase in the complexity and functionality of chips, leading to the development of more powerful and efficient computing devices.

In 1971, Intel marketed an IC with the capability of a general-purpose building block of digital systems. It contained all the functionality of the central processing unit (CPU) of a computer. The chip was code named as 4004. It was a 4-bit CPU. Later on, this device was given the name “microprocessor.” Thus, the microprocessor—“the CPU on a chip”—was born. The huge success of this chip led to the development of 8008 and 8085, the most popular 8-bit microprocessors, by Intel. Other semiconductor device manufacturers such as Motorola and Zilog joined the race of producing more and more powerful microprocessors. In the past three decades, the evolution tree of microprocessors has grown into a large tree with three main branches as shown in Fig. 1.2.

The main branch in the middle represents the general-purpose microprocessors, which are used to build computers of different kinds such as laptops, desktops, workstations, servers, etc. The fruits of this branch have produced more and more powerful CPUs with processing capability of increased number of bits starting from 4-bit processors to the present-day 64-bit processors. Moreover, the clock rates increased from few megahertz to thousands of megahertz, and many advanced architectural features such as pipelining, superscalar, on-chip cache memory, dual core, etc. Computers built using the present-day microprocessors have the capability of mainframe computers of the 1980s and 1990s. Figure 1.3 shows the series of
microprocessors produced by Intel in the past three-and-a-half decades conforming to Moore’s law very closely. It may be noted that the first microprocessor had only 2200 transistors and the latest microprocessors are having more than a billion transistors.

The left branch represents a new breed of processors, known as microcontrollers. A microcontroller can be considered a “computer on a chip.” Apart from the CPU, other subsystems such as ROM, RAM, input/output (I/O) ports, timer, and serial port are housed on a single chip in a microcontroller. The CPUs of the microcontroller are usually not as powerful as general-purpose microprocessors. Microcontrollers are typically used to realize embedded systems such as toys, home appliances, intelligent test and measurement equipment, etc.

The branch on the right side represents special-purpose processors such as DSP microprocessors (TMS 320), network processors (Intel PXA 210/215), etc. These special-purpose processors are designed to enhance performance of special applications such as signal processing, router and packet-level processing in communication equipment, etc.

With the increase in the number of transistors, the power dissipation also kept on increasing as shown in Fig. 1.4. This forced the chip designers to consider low power as one of the design parameters apart from performance and area. In the following section, we shall focus on the importance of low power in IC design.
Fig. 1.3 Moore’s law and the Intel microprocessors. (Source: Intel)

Fig. 1.4 Power dissipation of Intel processors. (Source: Intel)
Why Low Power?

Until recently, performance of a processor has been synonymous with circuit speed or processing power, e.g., million instructions per second (MIPS) or million floating point operations per second (MFLOPS). Power consumption was of secondary concern in designing ICs. However, in nanometer technology, power has become the most important issue because of:

- Increasing transistor count
- Higher speed of operation
- Greater device leakage currents

Increased process parameter variability due to aggressive device size scaling has created problems in yield, reliability, and testing. As a consequence, there is a change in the trend of specifying the performance of a processor. Power consumption is now considered one of the most important design parameters. Among various reasons for this change in trend, some important reasons are considered below.

In order to continuously improve the performance of the circuits and to integrate more and more functionality in the chip, the device feature size has to continuously shrink. Figure 1.4 shows the power dissipation of Intel processors. As a consequence, the magnitude of power per unit area known as power density is increasing as shown in Fig. 1.5. To remove the heat generated by the device, it is necessary to provide suitable packaging and cooling mechanism. There is an escalation in the cost of packaging and cooling as the power dissipation increases. To make a chip commercially viable, it is necessary to reduce the cost of packaging and cooling, which in turn demands lower power consumption.
Increased customer demand has resulted in proliferation of hand-held, battery-operated devices such as cell phone, personal digital assistant (PDA), palmtop, laptop, etc. The growth rate of the portable equipment is very high. Moreover, users of cell phones strive for increased functionality (as provided by smartphones) along with long battery life. As these devices are battery operated, battery life is of primary concern. Unfortunately, the battery technology has not kept up with the energy requirement of the portable equipment. Commercial success of these products depends on size, weight, cost, computing power, and above all on battery life. Lower power consumption is essential to make these products commercially viable.

It has been observed that reliability is closely related to the power consumption of a device. As power dissipation increases, the failure rate of the device increases because temperature-related failures start occurring with the increase in temperature as shown in Fig. 1.6. It has been found that every 10°C rise in temperature...
roughly doubles the failure rate. So, lower power dissipation of a device is essential for reliable operation. According to an estimate of the US Environmental Protection Agency (EPA), 80% of the power consumption by office equipment is due to computing equipment and a large part from unused equipment. Power is dissipated mostly in the form of heat. The cooling techniques, such as air conditioner, transfer the heat to the environment. To reduce adverse effect on environment, efforts such as EPA’s Energy Star program leading to power management standard for desktops and laptops has emerged.

1.4 Sources of Power Dissipations [3]

Although power and energy are used interchangeably in many situations, these two have different meanings and it is essential to understand the difference between the two, especially in the case of battery-operated devices. Figure 1.7 illustrates the difference between the two. Power is the instantaneous power in the device, while energy is the integration of power with time. For example, in Fig. 1.7, we can see that approach 1 takes less time but consumes more power than approach 2. But the energy consumed by the two, that is, the area under the curve for both the approaches is the same, and the battery life is primarily determined by this energy consumed.
Power dissipation is measured commonly in terms of two types of metrics:

1. *Peak power*: Peak power consumed by a particular device is the highest amount of power it can consume at any time. The high value of peak power is generally related to failures like melting of some interconnections and power-line glitches.

2. *Average power*: Average power consumed by a device is the mean of the amount of power it consumes over a time period. High values of average power lead to problems in packaging and cooling of VLSI chips.

In order to develop techniques for minimizing power dissipation, it is essential to identify various sources of power dissipation and different parameters involved in each of them. The total power for a VLSI circuit consists of dynamic power and static power. Dynamic power is the power consumed when the device is active, that is, when signals are changing values. Static power is the power consumed when the device is powered up but no signals are changing value. In CMOS devices, the static power consumption is due to leakage mechanism. Various components of power dissipation in CMOS devices can therefore be categorized as shown in Fig. 1.8.

### 1.4.1 Dynamic Power

Dynamic power is the power consumed when the device is active, that is, when the signals of the design are changing values. It is generally categorized into three types: switching power, short-circuit power, and glitching power, each of which will be discussed in details below.
Switching Power

The first and primary source of dynamic power consumption is the switching power, the power required to charge and discharge the output capacitance on a gate. Figure 1.9 illustrates switching power for charging a capacitor.

The energy per transition is given by

\[
\text{Energy/transition} = \frac{1}{2} \times C_L \times V_{dd}^2,
\]

where \(C_L\) is the load capacitance and \(V_{dd}\) is the supply voltage. Switching power is therefore expressed as:

\[
P_{\text{switch}} = \text{Energy/transition} \times f = C_L \times V_{dd}^2 \times P_{\text{trans}} \times f_{\text{clock}},
\]

where \(f\) is the frequency of transitions, \(P_{\text{trans}}\) is the probability of an output transition, and \(f_{\text{clock}}\) is the frequency of the system clock. Now if we take:

\[
C_{\text{switch}} = P_{\text{trans}} \times C_L,
\]

then, we can also describe the dynamic power with the more familiar expression:

\[
P_{\text{switch}} = C_{\text{eff}} \times V_{dd}^2 \times f_{\text{clock}}.
\]

Switching power is not a function of transistor size, but it is dependent on switching activity and load capacitance. Thus, it is data dependent.

In addition to the switching power dissipation for charging and discharging the load capacitance, switching power dissipation also occurs for charging and discharging of the internal node capacitance. Thus, total switching power dissipation is given by

\[
P_{\text{total switch}} = P_{\text{trans}} \times C_L \times V_{dd}^2 \times f_{\text{clock}} + \sum \alpha_i \times C_i \times V_{dd} \times (V_{dd} - V_{th}) \times f_{\text{clock}},
\]
where $\alpha_i$ and $C_i$ are the transition probability and capacitance, respectively, for an internal node $i$.

**Short-Circuit Power**

In addition to the switching power, short-circuit power also contributes to the dynamic power. Figure 1.10 illustrates short-circuit currents. Short-circuit currents occur when both the negative metal–oxide–semiconductor (NMOS) and positive metal–oxide–semiconductor (PMOS) transistors are on. Let $V_{\text{th}}$ be the threshold voltage of the NMOS transistor and $V_{\text{tp}}$ is the threshold voltage of the PMOS transistor. Then, in the period when the voltage value is between $V_{\text{tn}}$ and $V_{\text{dd}} - V_{\text{tp}}$, while the input is switching either from 1 to 0 or vice versa, both the PMOS and the NMOS transistors remain ON, and the short-circuit current follows from $V_{\text{dd}}$ to ground (GND).

The expression for short-circuit power is given by

$$P_{\text{short-circuit}} = t_{\text{sc}} \times V_{\text{dd}} \times I_{\text{peak}} \times f_{\text{clock}} = \frac{\mu \cdot \varepsilon_{\text{ox}} \cdot W}{12LD} \times (V_{\text{dd}} - V_{\text{th}})^3 \times t_{\text{sc}} \times f_{\text{clock}},$$

where $t_{\text{sc}}$ is the rise/fall time duration of the short-circuit current, $I_{\text{peak}}$ is the total internal switching current (short-circuit current plus the current to charge the internal capacitance), $\mu$ is the mobility of the charge carrier, $\varepsilon_{\text{ox}}$ is the permittivity of the silicon dioxide, $W$ is the width, $L$ is the length, and $D$ is the thickness of the silicon dioxide.

From the above equation it is evident that the short-circuit power dissipation depends on the supply voltage, rise/fall time of the input and the clock frequency apart from the physical parameters. So the short-circuit power can be kept low if the ramp (rise/fall) time of the input signal is short for each transition. Then the overall dynamic power is determined by the switching power.
1.4.1.3 Glitching Power Dissipation

The third type of dynamic power dissipation is the glitching power which arises due to finite delay of the gates. Since the dynamic power is directly proportional to the number of output transitions of a logic gate, glitching can be a significant source of signal activity and deserves mention here. Glitches often occur when paths with unequal propagation delays converge at the same point in the circuit. Glitches occur because the input signals to a particular logic block arrive at different times, causing a number of intermediate transitions to occur before the output of the logic block stabilizes. These additional transitions result in power dissipation, which is categorized as the glitching power.

1.4.2 Static Power

Static power dissipation takes place as long as the device is powered on, even when there are no signal changes. Normally in CMOS circuits, in the steady state, there is no direct path from $V_{dd}$ to GND and so there should be no static power dissipation, but there are various leakage current mechanisms which are responsible for static power dissipation. Since the MOS transistors are not perfect switches, there will be leakage currents and substrate injection currents, which will give rise to static power dissipation in CMOS. Since the substrate current reaches its maximum for gate voltages near $0.4V_{dd}$ and gate voltages are only transiently in this range when the devices switch, the actual power contribution of substrate currents is negligible as compared to other sources of power dissipation. Leakage currents are also normally negligible, in the order of nano-amps, compared to dynamic power dissipation. But with deep submicron technologies, the leakage currents are increasing drastically to the extent that in 90-nm technology and thereby leakage power also has become comparable to dynamic power dissipation.

Figure 1.11 shows several leakage mechanisms that are responsible for static power dissipation. Here, $I_1$ is the reverse-bias p–n junction diode leakage current,
I2 is the reverse-biased p–n junction current due to tunneling of electrons from the valence band of the p region to the conduction band of the n region, I3 is the subthreshold leakage current between the source and the drain when the gate voltage is less than the threshold voltage \( V_{th} \), I4 is the oxide tunneling current due to reduction in the oxide thickness, I5 is the gate current due to hot carrier injection of electrons (I4 and I5 are commonly known as IGATE leakage current), I6 is the gate-induced drain leakage current due to high field effect in the drain junction, and I7 is the channel punch through current due to close proximity of the drain and the source in short-channel devices.

These are generally categorized into four major types: subthreshold leakage, gate leakage, gate-induced drain leakage, and junction leakage as shown in Fig. 1.12. Apart from these four primary leakages, there are few other leakage currents which also contribute to static power dissipation, namely, reverse-bias p–n junction diode leakage current, hot carrier injection gate current, and channel punch through current.

**Low-Power DesignMethodologies**

Low-power design methodology needs to be applied throughout the design process starting from system level to physical or device level to get effective reduction of power dissipation in digital circuits based on MOS technology \([2–4]\). Various approaches can be used at different level of design hierarchy. Before venturing to do this, it is essential to understand the basics of MOS circuits and the way these are fabricated. So, we have started with fabrication technology in Chap. 2. The subsequent three chapters introduce MOS transistor, followed by MOS inverters, and then complex MOS combinational circuits. Chapter 6 introduces various sources of power dissipation in details. As the most dominant component has quadratic dependence and other components have linear dependence on the supply voltage, reducing the supply voltage is the most effective means to reduce dynamic power.
consumption. Unfortunately, this reduction in power dissipation comes at the expense of performance. It is essential to devise suitable mechanism to contain this loss in performance due to supply voltage scaling for the realization of low-power high-performance circuits. The loss in performance can be compensated by using suitable techniques at the different levels of design hierarchy; that is physical level, logic level, architectural level, and system level. Techniques like device feature size scaling, parallelism and pipelining, architectural-level transformations, dynamic voltage, and frequency scaling.

Apart from scaling the supply voltage to reduce dynamic power, another alternative approach is to minimize the switched capacitance comprising the intrinsic capacitances and switching activity. Choosing which functions to implement in hardware and which in software is a major engineering challenge that involves issues such as cost complexity, performance, and power consumption. From the behavioral description, it is necessary to perform hardware/software partitioning in a judicious manner such that the area, cost, performance, and power requirements are satisfied. Transmeta’s Crusoe processor is an interesting example that demonstrated that processors of high performance with remarkably low power consumption can be implemented as hardware–software hybrids. The approach is fundamentally software based, which replaces complex hardware with software, thereby achieving large power savings.

In CMOS digital circuits, the switching activity can be reduced by algorithmic optimization, by architectural optimization, by use of suitable logic-style or by logic-level optimization. The intrinsic capacitances of system-level busses are usually several orders of magnitude larger than that for the internal nodes of a circuit. As a consequence, a considerable amount of power is dissipated for transmission of data over I/O pins. It is possible to save a significant amount of power reducing the number of transactions, i.e., the switching activity, at the processors I/O interface. One possible approach for reducing the switching activity is to use suitable encoding of the data before sending over the I/O interface. The concept is also applicable in the context of multi-core system-on-a-chip (SOC) design. In many situations the switching activity can be reduced by using the sign-magnitude representation in place of the conventional two’s complement representation. Switching activity can be reduced by judicious use of clock gating, leading to considerable reduction in dynamic power dissipation. Instead of using static CMOS logic style, one can use other logic styles such as pass-transistor and dynamic CMOS logic styles or a suitable combination of pass-transistor and static CMOS logic styles to minimize energy drawn from the supply.

Although the reduction in supply voltage and gate capacitances with device size scaling has led to the reduction in dynamic power dissipation, the leakage power dissipation has increased at an alarming rate because of the reduction of threshold voltage to maintain performance. As the technology is scaling down from sub-micron to nanometer, the leakage power is becoming a dominant component of total power dissipation. This has led to vigorous research for the reduction of leakage power dissipation. Leakage reduction methodologies can be broadly classified into two categories, depending on whether it reduces standby leakage or runtime...
leakage. There are various standby leakage reduction techniques such as input vector control (IVC), body bias control (BBC), multi-threshold CMOS (MTCMOS), etc. and runtime leakage reduction techniques such as static dual threshold voltage CMOS (DTCMOS) technique, adaptive body biasing, dynamic voltage scaling, etc.

Aggressive device size scaling used to achieve high performance leads to increased variability due to short-channel and other effects. Performance parameters such as power and delay are significantly affected due to the variations in process parameters and environmental/operational ($V_{dd}$ temperature, input values, conditions. For designs, due to variability, the design methodology in the future nanometer VLSI circuit designs will essentially require a paradigm shift from deterministic to probabilistic and statistical design approach. The impact of process variations has been investigated and several techniques have been proposed to optimize the performance and power in the presence of process variations.

**MOS Fabrication Technology**

**Introduction**

Metal–oxide–semiconductor (MOS) fabrication is the process used to create the integrated circuits (ICs) that are presently used to realize electronic circuits. It involves multiple steps of photolithographic and chemical processing steps during which electronic circuits are gradually created on a wafer made of pure semiconducting material. Silicon is almost always used, but various compound semiconductors such as gallium–arsenide are used for specialized applications. There are a large number and variety of basic fabrication steps used in the production of modern MOS ICs. The same process could be used for the fabrication of n-type MOS (nMOS), p-type MOS (pMOS), or complementary MOS (CMOS) devices. The gate material could be either metal or poly-silicon. The most commonly used substrate is either bulk silicon or silicon on insulator (SOI). In order to avoid the presence of parasitic transistors, variations are brought in the techniques that are used to isolate the devices in the wafer. This chapter introduces various technologies that are used to fabricate MOS devices. Section 2.2 provides various processes used in the fabrication of MOS devices. Section 2.3 introduces fabrication of nMOS devices. Steps for
the fabrication of CMOS devices are presented in Sect. 2.4 Latch-up problem and various techniques to prevent it are highlighted in Sect. 2.5. Short-channel effects (SCEs) have been considered in Sect. 2.6 and emerging technologies for low power have been considered in Sect. 2.7.

**Basic Fabrication Processes [1, 2]**

Present day very-large-scale integration (VLSI) technology is based on silicon, which has bulk electrical resistance between that of a conductor and an insulator. That is why it is known as a semiconductor material. Its conductivity can be changed by several orders of magnitude by adding impurity atoms into the silicon crystal lattice. These impurity materials supply either free electrons or holes. The donor elements provide electrons and acceptor elements provide holes. Silicon having a majority of donors is known as n-type. On the other hand, silicon having a majority of acceptors is known as p-type. When n-type and p-type materials are put together, a junction is formed where the silicon changes from one type to the other type. Various semiconductor devices such as diode and transistors are constructed by arranging these junctions in certain physical structures and combining them with other types of physical structures, as we shall discuss in the subsequent sections.

**Wafer Fabrication**

The MOS fabrication process starts with a thin wafer of silicon. The raw material used for obtaining silicon wafer is sand or silicon dioxide. Sand is a cheap material and it is available in abundance on earth. However, it has to be purified to a high level by reacting with carbon and then crystallized by an epitaxial growth process. The purified silicon is held in molten state at about 1500 °C, and a seed crystal is slowly withdrawn after bringing in contact with the molten silicon. The atoms of the molten silicon attached to the seed cool down and take the crystalline structure of the seed. While forming this crystalline structure, the silicon is lightly doped by inserting controlled quantities of a suitable doping material into the crucible. The set up is for wafer fabrication to produce nMOS devices is shown in Fig. 2.1a. Here, boron may be used to produce p-type impurity concentration of $10^{15}$ to $10^{16}$ per cm$^3$. It gives resistivity in the range of 25–2 Ω cm. After the withdrawal of the seed, an “ingot” of several centimeters length and about 8–10 cm diameter as shown in Fig. 2.1b is obtained. The ingot is cut into slices of 0.3–0.4 mm thickness to obtain wafer for IC fabrication.

**Oxidation**

Silicon dioxide layers are used as an insulating separator between different conducting layers. It also acts as mask or protective layer against diffusion and high-energy
ion implantation. The process of growing oxide layers is known as oxidation because it is performed by a chemical reaction between oxygen (dry oxidation), or oxygen and water vapor (wet oxidation) and the silicon slice surface in a high-temperature furnace at about 1000 °C as shown in Fig. 2.2. To grow an oxide layer of thickness $t_{ox}$, the amount of silicon consumed is approximately $0.5t_{ox}$. Dry oxidation performed in O$_2$ with a few percent of hydrochloric acid added to produce thin, but robust oxide layers is used to form the gate structure. These layers are known as gate oxide layers. The wet oxidation produces a thicker and slightly porous layer. This layer is known as field oxide layer. The oxide thickness is limited by the diffusion rate of the oxidizing agent through the already grown layer and is about 1 µm at one atmospheric pressure, but can be doubled by using higher pressure, say approximately 20 atm. Another advantage of a high-pressure system is the possibility to grow thicker oxides in less time at high temperature.

**Mask Generation**

To create patterned layers of different materials on the wafer, masks are used at different stages. Masks are made of either inexpensive green glass or costly low-expansion glass plates with opaque and transparent regions created using photographic emulsion, which is cheap but easily damaged. Other alternative materials used for creating masks are iron oxide or chromium, both of which are more durable and give better line resolution, but are more expensive.
A mask can be generated either optically or with the help of an electron beam. In the optical process, a reticle, which is a photographic plate of exactly ten times the actual size of the mask, is produced as master copy of the mask. Transparent and opaque regions are created with the help of a pattern generator by projecting an image of the master onto the reticle. Special masking features such as parity masks and fiducials are used on the reticle to identify, align, and orient the mask. Master plates are generated from reticles in a step-and-repeat process by projecting an image of the reticle ten times reduced onto the photosensitized plate to create an array of geometrical shapes in one over the entire plate. Fiducials are used to control the separation between exposures and align the reticle images relative to one another. This process has the disadvantage that if there is a defect on the reticle, it is reproduced on all the chips. The step-and-repeat process not only is slow but also suffers from alignment problems and defect propagation due to dust specks. The electron beam mask generation technique overcomes these problems.

In the electron beam masking process, the masking plate is generated in one step. It is based on the raster scan approach where all the geometrical data are converted into a bit map of 1’s and 0’s. While scanning the masking plate in a raster scan manner, squares containing 1’s are exposed and those containing 0’s are not. Exposures are made by blanking and un-blanking the beam controlled by the bit map. Using this technique, several different chip types can be imprinted on the same set of masks. The main disadvantage of this approach is that it is a sequential technique. A better alternative is to use the soft X-ray photolithographic technique in which the entire chip can be eradicated simultaneously. This technique also gives higher resolution.

These master plates are usually not used for mask fabrication. Working plates made from the masters by contact printing are used for fabrication. To reduce turnaround time, specially made master plates can be used for wafer fabrication.

**Photolithography**

The photolithographic technique is used to create patterned layers of different materials on the wafer with the help of mask plates. It involves several steps. The first step is to put a coating of photosensitive emulsion called photo-resist on the wafer surface. After applying the emulsion on the surface, the wafer is spun at high speed (3000 rpm) to get a very thin (0.5–1 µm) and uniform layer of the photo-resist. Then the masking plate is placed in contact with the wafer in a precise position and exposed to the UV light. The mask plate, with its transparent and opaque regions, defines different areas. With negative photo-resist, the areas of the wafer exposed to UV light are polymerized (or hardened), while with positive photo-resist, the exposed areas are softened and removed.

The removal of the unwanted photo-resist regions is done by a process known as development. Unexposed (negative) or exposed (positive) portions of the photo-resist are chemically dissolved at the time of development. A low-temperature baking process hardens the subsequently remaining portion.
To create the desired pattern, actual removal of the material is done by the *etching* process. The wafer is immersed in a suitable etching solution, which eats out the exposed material leaving the material beneath the protective photo-resist intact. The etching solution depends on the material to be etched out. Hydrofluoric acid (HF) is used for SiO$_2$ and poly-silicon, whereas phosphoric acid is used for nitride and metal.

Another alternative to this wet chemical etching process is the plasma etching or ion etching. In this dry process, a stream of ions or electrons is used to blast the material away. Ions created by glow discharge at low pressure are directed to the target. Ions can typically penetrate about 800 Å of oxide or photo-resist layers, and thick layers of these materials are used as a mask of some area, whereas the exposed material is being sputtered away. This plasma technique can produce vertical etching with little undercutting. As a consequence, it is commonly used for producing fine lines and small geometries associated with high-density VLSI circuits.

Finally, the photo-resist material is removed by a chemical reaction of this material with fuming nitric acid or exposure to atomic oxygen which oxides away the photo-resist. Patterned layers of different materials in engraved form are left at the end of this process.

**Diffusion**

After masking some parts of the silicon surface, selective *diffusion* can be done in the exposed regions. There are two basic steps: pre-deposition and drive-in. In the pre-deposition step, the wafer is heated in a furnace at 1000 °C, and dopant atoms such as phosphorous or boron mixed with an inert gas, say nitrogen, are introduced into it. Diffusion of these atoms takes place onto the surface of the silicon, forming a saturated solution of the dopant atoms and solid. The impurity concentration goes up with a temperature up to 1300 °C and then drops. The depth of penetration depends on the duration for which the process is carried out. In the drive-in step, the wafer is heated in an inert atmosphere for few hours to distribute the atoms more uniformly and to a higher depth.

Another alternative method for diffusion is *ion* implantation. Dopant gas is first ionized with the help of an ionizer and ionized atoms are accelerated between two electrodes with a voltage difference of 150 kV. The accelerated gas is passed through a strong magnetic field, which separates the stream of dopant ions on the basis of molecular weights, as it happens in mass spectroscopy. The stream of these dopant ions is deflected by the magnetic field to hit the wafer. The ions strike the silicon surface at high velocity and penetrate the silicon layer to a certain depth as determined by the concentration of ions and accelerating field. This process is also followed by drive-in step to achieve uniform distribution of the ions and increase the depth of penetration.

Different materials, such as thick oxide, photo-resist, or metal can serve as mask for the ion implantation process. But implantation can be achieved through thin oxide layers. This is frequently used to control the threshold voltage of MOS
transistor. This control was not possible using other techniques, and ion implantation is now widely used not only for controlling the threshold voltage but also for all doping stages in MOS fabrication.

**Deposition**

In the MOS fabrication process, conducting layers such as poly-silicon and aluminium, and insulation and protection layers such as SiO$_2$ and Si$_3$N$_4$ are deposited onto the wafer surface by using the chemical vapor deposition (CVD) technique in a high-temperature chamber:

\[
\begin{align*}
\text{Poly: } & \text{SiH}_4 \xrightarrow{1000^\circ C} \text{Si} + 2\text{H}_2 \\
\text{SiO}_2: & \text{SiH}_4 + \text{O}_2 \xrightarrow{400–450^\circ C} \text{SiO}_2 + 2\text{H}_2 \\
\text{Si}_3\text{N}_4: & 3\text{SiCl}_2\text{H}_2 + 4\text{NH}_3 \xrightarrow{600–750^\circ C} \text{Si}_3\text{N}_4 + 6\text{HCl} + 6\text{H}_2
\end{align*}
\]

Poly-silicon is deposited simply by heating silane at about 1000°C, which releases hydrogen gas from silane and deposits silicon. To deposit silicon dioxide, a mixture of nitrogen, silane, and oxygen is introduced at 400–450°C. Silane reacts with oxygen to produce silicon dioxide, which is deposited on the wafer. To deposit silicon nitride, silane and ammonia are heated at about 700 °C to produce nitride and hydrogen. Aluminium is deposited by vaporizing aluminium from a heated filament in high vacuum.

**2.3 nMOS Fabrication Steps [2, 3]**

Using the basic processes mentioned in the previous section, typical processing steps of the poly-silicon gate self-aligning nMOS technology are given below. It can be better understood by considering the fabrication of a single enhancement-type transistor. Figure 2.3 shows the step-by-step production of the transistor.

**Step 1** The first step is to grow a thick silicon dioxide (SiO$_2$) layer, typically of 1 µm thickness all over the wafer surface using the wet oxidation technique. This oxide layer will act as a barrier to dopants during subsequent processing and provide an insulting layer on which other patterned layers can be formed.

**Step 2** In the SiO$_2$ layer formed in the previous step, some regions are defined where transistors are to be formed. This is done by the photolithographic process discussed in the previous section with the help of a mask (MASK 1). At the end of this step, the wafer surface is exposed in those areas where diffusion regions along with a channel are to be formed to create a transistor.
Step 3 A thin layer of SiO$_2$, typically of 0.1 μm thickness, is grown all over the entire wafer surface and on top of this poly-silicon layer is deposited. The poly-silicon layer, of 1.5 μm thickness, which consists of heavily doped poly-silicon is deposited using the CVD technique. In this step, precise control of thickness, impurity concentration, and resistivity is necessary.

Step 4 Again by using another mask (MASK 2) and photographic process, the poly-silicon is patterned. By this process, poly-gate structures and interconnections by poly layers are formed.

Step 5 Then the thin oxide layer is removed to expose areas where n-diffusions are to take place to obtain source and drain. With the poly-silicon and underlying thin oxide layer as the protective mask, the diffusion process is performed. It may be noted that the process is self-aligning, i.e., source and drain are aligned automatically with respect to the gate structure.
**Step 6** A thick oxide layer is grown all over again and holes are made at selected areas of the poly-silicon gate, drain, and source regions by using a mask (MASK 3) and the photolithographic process.

**Step 7** A metal (aluminium) layer of 1 μm thickness is deposited on the entire surface by the CVD process. The metal layer is then patterned with the help of a mask (MASK 4) and the photolithographic process. Necessary interconnections are provided with the help of this metal layer.

**Step 8** The entire wafer is again covered with a thick oxide layer—this is known as over-glassing. This oxide layer acts as a protective layer to protect different parts from the environment. Using a mask (MASK 5), holes are made on this layer to provide access to bonding pads for taking external connections and for testing the chip.

The above processing steps allow only the formation of nMOS enhancement-type transistors on a chip. However, if depletion-type transistors are also to be formed, one additional step is necessary for the formation of n-diffusions in the channel regions where depletion transistors are to be formed. It involves one additional step in between step 2 and step 3 and will require one additional mask to define channel regions following a diffusion process using the ion implantation technique.

## CMOS Fabrication Steps

There are several approaches for CMOS fabrication, namely, p-well, n-well, twin-tub, triple-well, and SOI. The n-well approach is compatible with the nMOS process and can be easily retrofitted to it. However, the most popular approach is the p-well approach, which is similar to the n-well approach. The twin-tub and silicon on sapphire are more complex and costly approaches. These are used to produce superior quality devices to overcome the latch-up problem, which is predominant in CMOS devices.

### 2.4.1 The n-Well Process

The most popular approach for the fabrication of n-well CMOS starts with a lightly doped p-type substrate and creates the n-type well for the fabrication of pMOS transistors. Major steps for n-well CMOS process are illustrated as follows:

**Step 1** The basic idea behind the n-well process is the formation of an n-well or tub in the p-type substrate and fabrication of p-transistors within this well. The formation of an n-well by ion implantation is followed by a drive-in step (1.8 × 10^2 p cm^−2, 80 kV with 1150 °C for 15 h of drive-in). This step requires a mask (MASK 1), which defines the deep n-well diffusions. The n-transistor is formed outside the well. The basic steps are mentioned below:
• Start with a blank wafer, commonly known as a substrate, which is lightly doped.

```
  p substrate
```

• Cover the wafer with a protective layer of SiO$_2$ (oxide) using the oxidation process at 900–1200°C with H$_2$O (wet oxidation) or O$_2$ (dry oxidation) in the oxidation furnace.

```
  SiO$_2$
  p substrate
```

• Spin on photoresist, which is a light-sensitive organic polymer. It softens where exposed to light.

```
  Photoresist
  SiO$_2$
  p substrate
```

• Expose photoresist through the n-well mask and strip off the exposed photoresist using organic solvents. The n-well mask used to define the n-well in this step is shown below.

```
  Photoresist
  SiO$_2$
  p substrate
```

• Etch oxide with HF, which only attacks oxide where the resist has been exposed.

```
  Photoresist
  SiO$_2$
  p substrate
```
• Remove the photoresist, which exposes the wafer.

![Diagram showing SiO₂ and p substrate]

• Implant or diffuse $n$ dopants into the exposed wafer using diffusion or ion implantation. The ion implantation process allows shallower wells suitable for the fabrication of devices of smaller dimensions. The diffusion process occurs in all directions and dipper the diffusion more it spreads laterally. This affects how closely two separate structures can be fabricated.

![Diagram showing SiO₂ and n well]

• Strip off SiO₂ leaving behind the p-substrate along with the n-well.

![Diagram showing p substrate and n well]

**Step 2** The formation of thin oxide regions for the formation of p- and n–transistors requires MASK 2, which is also known as active mask because it defines the thin oxide regions where gates are formed.

![Diagram showing polysilicon, thin gate oxide, p substrate, and n well]
**Step 3** The formation of patterned poly-silicon (nitride on the thin oxide) regions is done using MASK 3. Patterned poly-silicon is used for interconnecting different terminals.

**Step 4** The formation of n-diffusion is done with the help of the n+ mask, which is essentially MASK 4.

**Step 5** The formation of p-diffusion is done using the p+ mask, which is usually a negative form of the n+ mask. Similar sets of steps form p+ diffusion regions for the pMOS source and drain and substrate contact.

**Step 6** Thick SiO₂ is grown all over and then contact cut definition using another mask.
Step 7 The whole chip then has metal deposited over its surface to a thickness of 1 μm. The metal layer is then patterned by the photolithographic process to form interconnection patterns using MASK 7.

Step 8 Over-glassing is done by an overall passivation layer and a mask is required to define the openings for access to bonding pads (MASK 8).

Two transistors, one pMOS and another nMOS, which can be used to realize a CMOS inverter are formed using the n-well process shown in Fig. 2.4.

2.4.2 The p-Well Process

Typical p-well fabrication steps are similar to an n-well process, except that a p-well is implanted to form n-transistors rather than an n-well. p-Well processes are preferred in circumstances where the characteristics of the n- and p-transistors are required to be more balanced than that achievable in an n-well process. Because the transistor that resides in the native substrate has been found to have better characteristics, the p-well process has better p-devices than an n-well process.
2.4.3 Twin-Tub Process

In the twin-tub process, the starting material is either an n+ or p+ substrate with a lightly doped epitaxial layer, which is used for protection against latch-up. The process is similar to the n-well process, involving the following steps:

- Tub formation
- Thin oxide construction
- Source and drain implantations
- Contact cut definition
- Metallization

This process allows n-transistors and p-transistors to be separately optimized to provide balanced performance of both types of transistors. The threshold voltage, body effect, and the gain associated with n- and p-devices have to be independently optimized. Figure 2.5 visualizes a CMOS inverter fabricated using the twin-tub process.

2.5 Latch-Up Problem and Its Prevention

The latch-up [4, 5] is an inherent problem in both n-well- and p-well-based CMOS circuits. The phenomenon is caused by the parasitic bipolar transistors formed in the bulk of silicon as shown in Fig. 2.6a for the n-well process.Latch-up can be defined as the formation of a low-impedance path between the power supply and ground rails through the parasitic n–p–n and p–n–p bipolar transistors. Figure 2.6a shows a cross section of a CMOS inverter. Two parasitic bipolar transistors, Q₁ and Q₃ are shown in the figure. The p–n–p transistor has its emitter formed by the p+ source/drain implant used in the pMOS transistors. It may be noted that either the drain or the source may act as the emitter, although the source is the terminal that maintains the latch-up condition. The base is formed by the n-well, and the collector is formed by the p-substrate. The emitter of the n–p–n transistor is the n+ source/
High purity silicon grown with accurately determined dopant concentrations.

**Fig. 2.5** CMOS transistor realized using twin-tub process

排水装置

**Fig. 2.6** Latch-up problem of a CMOS transistor

**a**

The base is formed by the p-substrate and the collector is the n-well. The parasitic resistors R_{well} and R_{s} are formed because of the resistivity of the semiconductor material in the n-well and p-substrate, respectively.

As shown in Fig. 2.6b, the bipolar junction transistors (BJTs) are cross-coupled to form the structure of a silicon-controlled rectifier (SCR) providing a short-circuit path between the power rail and the ground. Leakage current through the parasitic resistors can cause one transistor to turn on, which in turn turns on the other transistor due to positive feedback, leading to heavy current flow and device failure. The mechanism of latch-up may be understood by referring to Fig. 2.6b.

In normal

**b**

**c**
operation, currents passing through the intrinsic resistors are diode-leakage currents, which are very small and the voltages developed across the resistors cannot turn on either of the BJTs. However, because of some external disturbance, current may increase through one of the two BJTs leading to a voltage drop across $R_s$ ($R_{well}$) which turns on the transistors. This leads to high collector current and causes higher voltage drop across $R_{well}$ (or $R_s$) and the resulting feedback leads to a self-sustaining low-resistance current path between $V_{dd}$ and ground (GND).

The latch-up process is triggered by transient currents or voltages generated internally during power-up, or externally due to voltages and currents beyond the normal operating ranges. Two distinct situations responsible for triggering are referred to as vertical triggering and lateral triggering. Vertical triggering takes place due to current flow in the vertical p–n–p transistor $Q_1$. The current is multiplied by the common-base current gain, which leads to a voltage drop across the emitter–base junction of the n–p–n transistor, due to resistance $R_s$. In a similar way, lateral triggering takes place when a current flows in the lateral n–p–n transistor leading to voltage drop across $R_{well}$. In either of the situations, the resulting feedback loop causes the current transients multiplied by $\beta_1 \times \beta_2$. It may be noted that when the condition $\beta_1 \times \beta_2 \geq 1$ is satisfied, both transistors continue to conduct a high current even after the initial disturbance no longer exists. At the onset of latch-up, the voltage drop across the BJT pair is given by

$$V_H = V_{BE1\text{sat}} + V_{CE2\text{sat}} = V_{BE2\text{sat}} + V_{CE1\text{sat}}$$

where $V_H$ is called the holding voltage. The latch-up condition is sustained as long as the current is greater than the holding current $I_H$; the holding current value depends on the total parasitic resistance $R_s$ in the current path. There are several approaches to reduce the tendency of latch-up. The slope of the $I$–$V$ curve depends on the total parasitic resistance $R_s$ in the current path. The possibility of internal latch-up can be reduced to a great extent by using the following rules:

- Every well must have an appropriate substrate contact.
- Every substrate contact should be directly connected to a supply pad by metal.
- Substrate contacts should be placed as close as possible to the source connection of transistors to the supply rails. This helps to reduce the value of both $R_s$ and $R_{well}$.
- Alternatively, place a substrate contact for every 5–10 transistors.
- nMOS devices should be placed close to $V_{ss}$ and pMOS devices close to $V_{dd}$.

In addition to the above, guard rings and trenches, as discussed below, are used to overcome latch-up.

**Use of Guard Rings**

The gain of the parasitic transistors can be reduced by using guard rings and making additional contacts to the ring as shown in Fig. 2.7. This reduces parasitic resis-
tance values and the contacts drain excess well or substrate leakage currents away from the active device such that trigger current which initiates latch-up is not attained. The guard bands act as dummy collectors and these reduce the gain of the parasitic transistors by collecting minority carriers and preventing them from being injected into the base. This, however, increases the space between the n-channel and p-channel devices and leads to reduction in gate density.

**Use of Trenches**

Another approach to overcome the latch-up problem is to use trenches between the individual transistor devices of the CMOS structure, and highly doped field regions are formed in the bottom of the trenches. Each n- and p-well includes a retrograde impurity concentration profile and extends beneath adjacent trenches as shown in Fig. 2.8.

**Short-Channel Effects [6]**

The channel length $L$ is usually reduced to increase both the speed of operation and the number of components per chip. However, when the channel length is the same order of magnitude as the depletion-layer widths ($xd_D$, $xd_S$) of the source and drain
junction, a metal–oxide–semiconductor field-effect transistor (MOSFET) behaves differently from other MOSFETs. This is known as short-channel effect (SCE). The SCEs are attributed to two physical phenomena:

- The limitation imposed on electron drift characteristics in the channel
- The modification of the threshold voltage due to the shortening of channel length

Some of the important SCEs are mentioned below.

**Channel Length Modulation Effect**

As the channel length is reduced, the threshold voltage of MOSFET decreases as shown in Fig. 2.9. This reduction of channel length is known as $V_{th}$ roll-off. The graph in Fig. 2.9b shows the reduction of threshold voltage with reduction in channel length. This effect is caused by the proximity of the source and drain regions leading to a 2D field pattern rather than a 1D field pattern in short-channel devices as shown in Fig. 2.9a. The bulk charge that needs to be inverted by the application of gate voltage is proportional to the area under the channel region. So, the gate voltage has to invert less bulk charge to turn the transistor on, leading to more band bending in the Si–SiO$_2$ interface in short-channel devices compared to long-channel devices. As a consequence, the threshold voltage is lower for a short-channel device for the same drain-to-source voltage. Moreover, the effect of the source–drain depletion region is more severe for high drain bias voltage. This results in further decrease in threshold voltage and larger subthreshold leakage current.

**Drain-Induced Barrier Lowering**

For long-channel devices, the source and drain regions are separated far apart, and the depletion regions around the drain and source have little effect on the potential distribution in the channel region. So, the threshold voltage is independent of the
channel length and drain bias for such devices. However, for short-channel devices, the source and drain depletion width in the vertical direction and the source drain potential have a strong effect on a significant portion of the device leading to a variation of the subthreshold leakage current with the drain bias. This is known as the drain-induced barrier-lowering (DIBL) effect. Because of the DIBL effect, the barrier height of a short-channel device reduces with an increase in the subthreshold current due to lower threshold voltage. Therefore, DIBL occurs when the depletion regions of the drain and the source interact with each other near the channel surface to lower the source potential barrier. The DIBL effect is visualized in Fig. 2.10.

Channel Punch Through

Due to the proximity of the drain and the source in short-channel devices, the depletion regions at the source–substrate and drain–substrate junctions extend into the channel. If the doping is kept constant while the channel length is reduced, the separation between the depletion region boundaries decreases. Increased reverse bias across the junction further decreases the separation. When the depletion regions merge, majority carriers in the source enter into the substrate and get collected by the drain. This situation is known as punch-through condition as shown in Fig. 2.11. The net effect of punch through is an increase in the subthreshold leakage current.
Emerging Technologies for Low Power

Over the past two decades, industries have closely followed Moore’s law by fabricating transistors with gate dielectric scaling using silicon dioxide (SiO$_2$). But, as transistor size shrinks, leakage current increases drastically. Managing that leakage is crucial for reliable high-speed operation. As a consequence, this is becoming an increasingly important factor in chip design. High-K (Hi-K) materials are proposed to reduce the gate leakage current, a metal gate is used to suppress the poly-silicon gate depletion, and SOI technologies with single or multiple gate transistors offer opportunities for further scaling down of the transistor dimensions. Many other alternatives such as dual-gated SOI and substrate biasing have recently been proposed to address the conflicting requirement of high performance during active mode of operation and low leakage during sleep mode of operation.

**Hi-K Gate Dielectric**

A significant breakthrough has been made by industries in solving the chip power problem, identifying a new “Hi-K” material called hafnium (Hf) to replace the transistor’s silicon dioxide gate dielectric, and new metals like nickel (Ni) silicide to replace the poly-silicon gate electrode of n-type and p-type MOS transistors. The scaling of CMOS transistors has led to the silicon dioxide layer to be used as a gate dielectric and, being very thin (1.4 nm), its leakage current is too large. It is necessary to replace the SiO$_2$ dielectric with a physically thicker layer of oxides of higher dielectric constant (K) or “Hi-K” gate oxides such as hafnium oxide (HfO$_2$) and hafnium silicate (HfSiO). Thus, for the sub-100-nm MOS structure, it reduces leakage current significantly more than the SiO$_2$ dielectric under the same electrical equivalent thickness. It has been established that the oxides must be implemented in conjunction with metal gate electrodes, the development of which is further behind. The metal gate electrode is a gate electrode with a metal or a compound with metallic conductivity. The current standard gate electrode is doped polycrystalline silicon (poly-Si), which is slightly depleted at its surface due to its semiconducting nature and decreases the current drivability of MOS transistors. But, the metal
Conventional structure. a Lightly doped drain–structure gate perfectly eliminates such depletion and, therefore, it is considered to be the indispensable component for advanced VLSI circuits. These new materials, along with the right process step, reduces gate leakage more than 100X while delivering record transistor performance. In early 2007, Intel announced the deployment of hafnium-based Hi-K dielectrics in conjunction with a metallic gate for components built on 45 nm technologies. At the same time, IBM announced plans to transition to Hi-K materials, also hafnium based, for some products in 2008. Although the International Technology Roadmap for Semiconductors (ITRS) predicted the implementation of Hi-K materials for gate dielectrics along with metal gate electrodes to be commonplace in the industry by 2010 but it is still far from reality.

**Lightly Doped Drain–Source**

In the lightly doped drain–source (LDD) structure, narrow, self-aligned n-regions are introduced between the channel and the n+ source–drain diffusions of an MOSFET. This helps to spread the high field at the drain pinch-off region and thus to reduce the maximum field intensity. N-channel devices are fabricated with LDD extensions in a CMOS process, without the requirement of an extra mask level. A smaller peak electric field near the drain is realized in this structure because of the reduced N gradient. This results in lowering hot-carrier effects (or fewer hot electrons into oxide) and increase in series resistance.

A pattern of lightly doped regions in the substrate is formed under the structures by multiple ion implantations. After the ion implantations, the lightly doped regions are annealed at a temperature and time to obtain a critical and desired dopant diffusion. A dielectric spacer structure is formed upon the sidewalls of each of the structures and over the adjacent portions of the substrate. A pattern of heavily doped n+ regions is formed in the substrate adjacent to the dielectric spacer structure on the sidewalls of the structures and over the adjacent portions of the substrate which form LDD structures of an MOSFET device to form the said integrated circuit device as shown in Fig. 2.12. The n+ regions provide smaller ohmic contacts required to avoid punch through. In the p-channel regions, the n-type LDD extensions are counterdoped by the regular p+ source/drain implant. This results in significant improvements in breakdown voltages, hot-electron effects, and short-channel threshold effects. A pattern of gate electrode structures is formed upon a semiconductor substrate whose structures each include a gate oxide and a poly-silicon layer as shown in Fig. 2.12.
Figure 2.13 shows a device with various channel-doping implants (source/drain extension, SDE; Gaussian halo; and vertical retrograde well) which have been developed to mitigate the SCEs and to improve the leakage characteristics.

**Silicon on Insulator**

Rather than using silicon as the substrate, technologies such as SOI have been developed that use an insulating substrate to improve process characteristics such as latch-up and speed. Figure 2.14 shows a CMOS inverter fabricated using the SOI approach. The steps used in a typical SOI CMOS process are as follows:

- A thin film (7–8 µm) of very lightly doped n-type Si is epitaxially grown over an insulator. Sapphire or SiO$_2$ is a commonly used insulator.
- An anisotropic etch is used to etch away the Si except where a diffusion area will be needed.
- Implantation of the p-island where an n-transistor is formed.
- Implantation of the n-island where a p-transistor is formed.
- Growing of a thin gate oxide (100–250 Å).
- Depositing of phosphorus-doped poly-silicon film over the oxide.
- Patterning of the poly-silicon gate.
- Forming of the n-doped source and drain of the n-channel devices in the p-islands.
• Forming of the p-doped source and drain of the p-channel devices in the n-islands.
• Depositing of a layer of insulator material such as phosphorus glass or SiO₂ over the entire structure.
• Etching of the insulator at contact cut locations. The metallization layer is formed next.
• Depositing of the passivation layer and etching of the bonding pad location.

**Advantages of SOI**

• Due to the absence of wells, transistor structures denser than bulk silicon are feasible.
• Lower substrate capacitance.
• No field-inversion problems (the existence of a parasitic transistor between two normal transistors).
• No latch-up is possible because of the isolation of transistors by insulating substrate.

**FinFET**

The finFET [7] is a transistor realization, first developed by Chenming Hu and his colleagues at the University of California at Berkeley, which attempts to overcome the worst types of SCE encountered by deep submicron transistors, such as DIBL. These effects make it difficult for the voltage on the gate electrode to deplete the channel underneath and stop the flow of carriers through the channel; in other words, to turn the transistor off. By raising the channel above the surface of the wafer instead of creating the channel just below the surface, it is possible to wrap the gate around up to three of its sides, providing much greater electrostatic control over the carriers within it. This led to the development of FinFET structure as shown in Fig. 2.15. In current usage, the term FinFET has a less precise definition. Among microprocessor manufacturers, AMD, IBM, and Motorola describe their double-gate development efforts as FinFET development, whereas Intel avoids using the
term to describe their closely related tri-gate architecture. In the technical literature, FinFET is used somewhat generically to describe any fin-based, multi-gate transistor architecture regardless of the number of gates. In a FinFET, gates turn on and off much faster than with planar transistors, since the channel is surrounded on three sides by the gate. As a result, leakage current is substantially reduced. $V_{dd}$ and dynamic power are significantly lower as well.
MOS Transistors

Abstract The fundamentals of metal–oxide–semiconductor (MOS) transistors are introduced in this chapter. Basic structure of an MOS transistor is introduced along with the concept of enhancement- and depletion-mode MOS transistors. The behavior of MOS transistors is explained with the help of the fluid model, which helps to visualize the operation of MOS transistors without going into the details of device physics. Then, the three modes of operation of an MOS transistor, namely accumulation, depletion, and inversion, are introduced. The electrical characteristics of MOS transistors are explained in detail by deriving the expression of drain current. The threshold voltage and transconductance of MOS transistors are defined, and their dependence on various parameters is highlighted. The body effect and channel-length modulation effect are explained. Use of MOS transistors to realize transmission gate and to use it as a switch is discussed in detail.

Keywords Fluid model · Threshold voltage · Transconductance · Cutoff region · Nonsaturated region · Saturated region · Figure of merit · Channel-length modulation effect · Body effect · MOS switch · Transmission gate

Introduction

The base semiconductor material used for the fabrication of metal–oxide–semiconductor (MOS) integrated circuits is silicon. Metal, oxide, and semiconductor form the basic structure of MOS transistors. MOS transistors are realized on a single crystal of silicon by creating three types of conducting materials separated by intervening layers of an insulating material to form a sandwich-like structure. The three conducting materials are: metal, poly-silicon, and diffusion. Aluminum as metal and polycrystalline silicon or poly-silicon are used for interconnecting different elements of a circuit. The insulating layer is made up of silicon dioxide (SiO₂). Patterned layers of the conducting materials are created by a series of photolithographic techniques and chemical processes involving oxidation of silicon, diffusion of impurities into the silicon and deposition, and etching of aluminum on the silicon to provide interconnection. In Sect. 3.2, we discuss the structure of various types of MOS transistors obtained after fabrication. In Sect. 3.3, characteristics of an MOS transistor will be studied with the help of the Fluid Model, which helps to understand the operation of an MOS transistor without going into detailed physics.
of the device. Electrical characteristics of MOS transistors are studied in detail in Sect. 3.5. Use of MOS transistors as a switch is explored in Sect. 3.6.

### The Structure of MOS Transistors

The structure of an MOS transistor is shown in Fig. 3.1. On a lightly doped substrate of silicon, two islands of diffusion regions of opposite polarity of that of the substrate are created. These two regions are called source and drain, which are connected via metal (or poly-silicon) to the other parts of the circuit. Between these two regions, a thin insulating layer of silicon dioxide is formed, and on top of this a conducting material made of poly-silicon or metal called gate is deposited. There are two possible alternatives. The substrate can be lightly doped by either a p-type or an n-type material, leading to two different types of transistors. When the substrate is lightly doped by a p-type material, the two diffusion regions are strongly doped by an n-type material. In this case, the transistor thus formed is called an nMOS transistor. On the other hand, when the substrate is lightly doped by an n-type material, and the diffusion regions are strongly doped by a p-type material, a pMOS transistor is created.

The region between the two diffusion islands under the oxide layer is called the channel region. The operation of an MOS transistor is based on the controlled flow of current between the source and drain through the channel region. In order to make a useful device, there must be suitable means to establish some channel current to flow and control it. There are two possible ways to achieve this, which have resulted in enhancement- and depletion-mode transistors. After fabrication, the structure of an enhancement-mode nMOS transistor looks like Fig. 3.2a. In this case, there is no conducting path in the channel region for the situation $V_{gs} = 0 \text{ V}$, that is when no voltage is applied to the gate with respect to the source. If the gate is connected to a suitable positive voltage with respect to the source, then the electric field established between the gate and the substrate gives rise to a charge inversion region in the substrate under the gate insulation, and a conducting path is formed between the source and drain. Current can flow between the source and drain through this conducting path.

By implanting suitable impurities in the channel region during fabrication, prior to depositing the insulation and the gate, the conducting path may also be established in the channel region even under the condition $V_{gs} = 0 \text{ V}$. This situation is shown in
Fig. 3.2 a nMOS enhancement-mode transistor. b nMOS depletion-mode transistor

Fig. 3.3 a nMOS enhancement. b nMOS depletion. c pMOS enhancement. d pMOS depletion-mode transistors

Fig. 3.2b. Here, the source and drain are normally connected by a conducting path, which can be removed by applying a suitable negative voltage to the gate. This is known as the depletion mode of operation.

For example, consider the case when the substrate is lightly doped in p-type and the channel region implanted with n-type of impurity. This leads to the formation of an nMOS depletion-mode transistor. In both the cases, the current flow between the source and drain can be controlled by varying the gate voltage, and only one type of charge carrier, that is, electron or hole takes part in the flow of current. That is the reason why MOS devices are called unipolar devices, in contrast to bipolar junction transistors (BJTs), where both types of charge carriers take part in the flow of current. Therefore, by using the MOS technology, four basic types of transistors can be fabricated—nMOS enhancement type, nMOS depletion type, pMOS enhancement type, and pMOS depletion type. Each type has its own pros and cons. It is also possible to realize circuits by combining both nMOS and pMOS transistors, known as Complementary MOS (CMOS) technology. Commonly used symbols of the four types of transistors are given in Fig. 3.3.

3.3 The Fluid Model

The operation of an MOS transistor can be analyzed by using a suitable analytical technique, which will give mathematical expressions for different device characteristics. This, however, requires an in-depth knowledge of the physics of the device. Sometimes, it is possible to develop an intuitive understanding about the operation of a system by visualizing the physical behavior with the help of a simple but very
Fig. 3.4  a An MOS capacitor. b The fluid model

effective model. The *Fluid model* [1] is one such tool, which can be used to visualize the behavior of charge-controlled devices such as MOS transistors, charge-coupled devices (CCDs), and bucket-brigade devices (BBDs). Using this model, even a novice can understand the operation of these devices.

The model is based on two simple ideas: (a) Electrical charge is considered as fluid, which can move from one place to another depending on the difference in their level, of one from the other, just like a fluid and (b) electrical potentials can be mapped into the geometry of a container, in which the fluid can move around. Based on this idea, first, we shall consider the operation of a simple MOS capacitor followed by the operation of an MOS transistor.

### 3.3.1 The MOS Capacitor

From the knowledge of basic physics, we know that a simple parallel-plate capacitor can be formed with the help of two identical metal plates separated by an insulator. An MOS capacitor is realized by sandwiching a thin oxide layer between a metal or poly-silicon plate on a silicon substrate of suitable type as shown in Fig 3.4a. As we know, in case of parallel-plate capacitor, if a positive voltage is applied to one of the plates, it induces a negative charge on the lower plate. Here, if a positive voltage is applied to the metal or poly-silicon plate, it will repel the majority carriers of the p-type substrate creating a depletion region. Gradually, minority carriers (electrons) are generated by some physical process, such as heat or incident light, or it can be injected into this region. These minority carriers will be accumulated underneath the MOS electrode, just like a parallel-plate capacitor. Based on the fluid model, the MOS electrode generates a pocket in the form of a surface
potential in the silicon substrate, which can be visualized as a container. The shape of the container is defined by the potential along the silicon surface. The higher the potential, the deeper is the container, and more charge can be stored in it. However, the minority carriers present in that region create an inversion layer. This changes the surface potential; increase in the quantity of charge decreases the positive surface potential under the MOS electrode. In the presence of inversion charge, the surface potential is shown in Fig. 3.4b by the solid line. The area between the solid line and the dashed line shows not only the presence of charge but also the amount of charge. The capacity of the bucket is finite and depends on the applied electrode voltage. Here, it is shown that the charge is sitting at the bottom of the container just as a fluid would stay in a bucket. In practice, however, the minority carriers in the inversion layer actually reside directly at the silicon surface. The surface of the fluid must be level in the equilibrium condition. If it were not, electrons would move under the influence of potential difference until a constant surface potential is established. From this simple model, we may conclude that the amount of charge accumulated in an MOS capacitor is proportional to the voltage applied between the plates and the area between the plates.

### 3.3.2 The MOS Transistor

By adding diffusion regions on either side of an MOS capacitor, an MOS transistor is realized. One of the diffusion regions will form the source and the other one will form the drain. The capacitor electrode acts as the gate. The cross-sectional view of an MOS transistor is shown in Fig. 3.5a. We can use the fluid model to explain the behavior of MOS transistors.

To start with, we may assume that the same voltage is applied to both the source and drain terminals \( V_{db} = V_{sb} \) with respect to the substrate. This defines the potential of these two regions. In the potential plot, the diffusion regions (where there is plentiful of charge carriers) can be represented by very deep wells, which are filled with charge carriers up to the levels of the potentials of the source and drain regions. The potential underneath the MOS gate electrode determines whether these
two wells are connected or separated. The potential in the channel region can be controlled with the help of the gate voltage. The potential at the channel region is shown by the dotted lines of Fig. 3.5b. The dotted line 1 corresponding to $V_{gb} = 0$ is above the drain and source potentials. As the gate voltage is gradually increased, more and more holes are repelled from the channel region, and the potential at the channel region moves downward as shown by the dotted lines 2, 3, etc. In this situation, the source and drain wells are effectively isolated from each other, and no charge can move from one well to the other. A point is reached when the potential level at the gate region is the same as that of the source and diffusion regions. At this point, the channel region is completely devoid of holes. The gate voltage at which this happens is called the threshold voltage ($V_t$) of the MOS transistor. If the gate voltage is increased further, there is an accumulation of electrons beneath the SiO$_2$ layer in the channel region, forming an inversion layer. As the gate voltage is increased further, the potential at the gate region moves below the source and drain potentials as shown by the dotted lines 3 and 4 in Fig. 3.5b. As a consequence, the barrier between the two regions disappears and the charge from the source and drain regions spills underneath the gate electrode leading to a uniform surface potential in the entire region. By varying the gate voltage, the thickness of the inversion layer can be controlled, which in turn will control the conductivity of the channel as visualized in Fig. 3.5b. Under the control of the gate voltage, the region under it acts as a movable barrier that controls the flow of charge between the source and drain areas.

When the source and drain are biased to different potentials ($V_{db} > V_{sb}$), there will be a difference in the potential levels. Let us consider two different situations. In the first case, the drain voltage is greater than the source voltage by some fixed value, and the gate voltage $V_{gb}$ is gradually increased from 0 V. Figure 3.6 shows different situations. Initially, for $V_{gb} = 0$ V, the potential level in the channel region is above the potential level of either of the source and drain regions, and the source and drain are isolated. Now, if the gated voltage is gradually increased, first, the gate region potential reaches the potential of the source region. Charge starts moving from the source to the drain as the gate voltage is slightly increased. The rate of flow of
Fig. 3.7 a Variation of drain current with gate voltage. b Voltage–current characteristics

charge moving from the source to the drain region, represented by the slope of the interface potential in the channel region, keeps on increasing until the gate region potential level becomes the same as that of the drain potential level. In this situation, the device is said to be operating in an active, linear, or unsaturated region. If the gate voltage is increased further, the width of the channel between the source and drain keeps on increasing, leading to a gradual increase in the drain current.

Let us consider another case when the gate voltage is held at a fixed value for a heavily turned-on channel. To start with, the drain voltage is the same as that of the source voltage, and it is gradually increased. Figure 3.6a shows the case when the source and drain voltages are equal. Although the path exists for the flow of charges, there will be no flow because of the equilibrium condition due to the same level. In Fig. 3.6b, a small voltage difference is maintained by externally applied voltage level. There will be continuous flow of charge resulting in drain current. With the increase in voltage difference between the source and drain, the difference in the fluid level increases, and the layer becomes more and more thin, signifying faster movement of charges. With the increasing drain potential, the amount of charge flowing from the source to drain per unit time increases. In this situation, the device is said to be operating in an active, linear, or unsaturated region. However, there is a limit to it. It attains a maximum value, when the drain potential \( V_{db} = (V_{gb} - V_t) \). Further increase in drain voltage does not lead to any change in the rate of charge flow. The device is said to be in the saturation region. In this condition, the drain current becomes independent of the drain voltage, and it is fully determined by the gate potential.

The strength of the fluid model is demonstrated above by the visualization of the operation of an MOS transistor. It can be applied to more complex situations where it is difficult to derive closed form of equations. In such situations, the fluid model will be of real help in understanding the operation of such circuits.

To summarize this section, we can say that an MOS transistor acts as a voltage-controlled device. The device first conducts when the effective gate voltage \((V_{gb} - V_t)\) is more than the source voltage. The conduction characteristic is represented in Fig. 3.7a. On the other hand, as the drain voltage is increased with respect to the source, the current increases until \( V_{db} = (V_{gb} - V_t) \). For drain voltage \( V_{db} > (V_{gb} - V_t) \), the channel becomes pinched off, and there is no further increase in current. A plot of the drain current with respect to the drain voltage for different gate voltages is shown in Fig. 3.7b.
After having some insight about the operation of an MOS transistor, let us now have a look at the charge distribution under the gate region under different operating conditions of the transistor. When the gate voltage is very small and much less than the threshold voltage, Fig. 3.8a shows the distribution of the mobile holes in a p-type substrate. In this condition, the device is said to be in the **accumulation mode**. As the gate voltage is increased, the holes are repelled from the SiO$_2$–substrate interface and a depletion region is created under the gate when the gate voltage is equal to the threshold voltage. In this condition, the device is said to be in **depletion mode** as shown in Fig. 3.8b. As the gate voltage is increased further above the threshold voltage, electrons are attracted to the region under the gate creating a conducting layer in the p substrate as shown in Fig. 3.8c. The transistor is now said to be in **inversion mode**.

### Electrical Characteristics of MOS Transistors

The fluid model, presented in the previous section, gives us some basic understanding of the operation of an MOS transistor [3, 4]. We have seen that the whole concept of the MOS transistor is based on the use of the gate voltage to induce charge (inversion layer) in the channel region between the source and the drain. Application of the source-to-drain voltage $V_{ds}$ causes this charge to flow through the channel from the source to drain resulting in source-to-drain current $I_{ds}$. The $I_{ds}$ depends on two variable parameters—the gate-to-source voltage $V_{gs}$ and the drain-to-source voltage $V_{ds}$. The operation of an MOS transistor can be divided into the following three regions:
Fig. 3.9 Structural view of an MOS transistor

(a) Cutoff region: This is essentially the accumulation mode, when there is no effective flow of current between the source and drain.

(b) Nonsaturated region: This is the active, linear, or weak inversion mode, when the drain current is dependent on both the gate and the drain voltages.

(c) Saturated region: This is the strong inversion mode, when the drain current is independent of the drain-to-source voltage but depends on the gate voltage.

In this section, we consider an nMOS enhancement-type transistor and establish its electrical characteristics. The structural view of the MOS transistor, as shown in Fig. 3.9, shows the three important parameters of MOS transistors, the channel length $L$, the channel width $W$, and the dielectric thickness $D$. The expression for the drain current is given by

$$I_{ds} = \frac{\text{charge induced in the channel (}\mathcal{Q}_c\text{)} }{\text{electron transit time (}\tau_n\text{)}}, \quad (3.1)$$

Let us separately find out the expressions for $Q_c$ and $\tau_n$.

With a voltage $V$ applied across the plates, the charge is given by $Q = CV$, where $C$ is the capacitance. The basic formula for parallel-plate capacitor is $C = \frac{\varepsilon A}{D}$, where $\varepsilon$ is the permittivity of the insulator in units of F/cm. The value of $\varepsilon$ depends on the material used to separate the plates. In this case, it is silicon dioxide ($\text{SiO}_2$). For $\text{SiO}_2$, $\varepsilon_{\text{ox}} = 3.9\varepsilon_0$, where $\varepsilon_0$ is the permittivity of the free space. For the MOS transistor, the gate capacitance

$$C_G = \frac{\varepsilon_{\text{ox}} WL}{D}, \quad (3.2)$$

Now, for the MOS transistor,

$$Q_c = C_G \cdot V_{\text{eff}},$$

where $C_G$ is the gate capacitance and $V_{\text{eff}}$ is the effective gate voltage.

Now, the transit time, $\tau_n = \frac{\text{length of the channel (}}{\text{velocity of electron (}}\tau_n\text{)}$. \quad (3.3)
The velocity, \( \tau_n = \mu_n \cdot E_{ds} \), where \( \mu_n \) is the mobility of electron and \( E_{ds} \) is the drain to the source electric field due to the voltage \( V_{ds} \) applied between the drain and source.

Now, \( E_{ds} = V_{ds}/L \).

So,

\[
\tau_n = \frac{\mu_n V_{ds}}{L} \quad \text{and} \quad \tau_n = \frac{L^2}{\mu_n V_{ds}}. \tag{3.4}
\]

Typical value of \( \mu_n = 650 \text{cm} / \text{V} \) (at room temperature).

**The nonsaturated region:** As the channel formation starts when the gate voltage is above the threshold voltage and there is a voltage difference of \( V_{ds} \) across the channel, the effective gate voltage is

\[
V_{eff} = (V_{gs} - V_i - V_{ds}/2). \tag{3.5}
\]

Substituting this, we get

\[
Q = \frac{W L \epsilon_{ox}}{c} \left[ (V - V_{gs} - V_i) - \frac{V_{ds}}{2} \right]. \tag{3.6}
\]

Now, the current flowing through the channel is given by

\[
I = \frac{W L \mu_n \epsilon_{ox}}{c} \left[ (V - V_{gs} - V_i) - \frac{V_{ds}}{2} \right]. \tag{3.7}
\]

Substituting the value of \( \tau_n \), we get

\[
I = \frac{W L \mu_n \epsilon_{ox}}{c} \left[ (V - V_{gs} - V_i) - \frac{V_{ds}}{2} \right]. \tag{3.8}
\]

Assuming \( V_{ds} \leq V - V_{gs} - V_i \) in the nonsaturated region and \( K = \frac{\mu_n \epsilon_{ox}}{D} \), we get

\[
I = \frac{K W}{L} \left[ (V - V_{gs} - V_i) - \frac{V_{ds}}{2} \right]. \tag{3.9}
\]

Now, the gate-channel capacitance based on parallel-plate capacitor model is

\[
C_g = \frac{\epsilon_{ox} \epsilon_0 W L}{D} \quad \text{and} \quad K = \frac{C_g \mu_n}{W L}. \tag{3.10}
\]

So, in terms of the gate-channel capacitance the expression for drain-to-source current can be written as

\[
I = \frac{C_g \mu_n}{L^2} \left[ (V - V_{gs} - V_i) - \frac{V_{ds}}{2} \right]. \tag{3.11}
\]
The Saturated Region As we have seen in the previous section, the drain current ($I_{ds}$) increases as drain voltage increases until the IR drop in the channel equals the effective gate voltage at the drain. This happens when $V_{ds} = V_{gs} - V_t$. At this point, the transistor comes out of the active region and $I_{ds}$ remains fairly constant as $V_{ds}$ increases further. This is known as saturation condition. Assuming $V_{ds} = V_{gs} - V_t$ for this region, the saturation current is given by

$$I_{ds} = K \frac{W}{L} (V_{gs} - V_t)^2$$

or

$$I = \frac{C_g \mu_n}{2L^2} (V_{gs} - V_t)^2 = \frac{C_{ox} W \mu_n}{2L} (V_{gs} - V_t)^2 = \frac{\mu_n C_{ox} W}{2} \frac{(V_{gs} - V_t)^2}{L} \frac{(V_{gs} - V_t)^2}{2} \frac{(V_{gs} - V_t)^2}{L} \frac{(V_{gs} - V_t)^2}{2} \frac{(V_{gs} - V_t)^2}{2}. \quad (3.11)$$

It may be noted that in case of the enhancement-mode transistor, the drain-to-source current flows only when the magnitude exceeds the threshold voltage $V_t$. The $I_{ds} - V_{ds}$ characteristic for an enhancement-type nMOS transistor is shown in Fig. 3.10.

$$I_{ds} = 0 \text{ for } V_{gs} < V_t,$$

$$I_{(\text{lin})} = \frac{\mu_n C_{ox} W}{2L} (2(V_{gs} - V_t) - V_{ds} - V_{ds} - V_t)^2 \text{ for } V \geq V_{gs} \text{ and } V < V_{gs} - V_t,$$

$$I_{(\text{sat})} = \frac{\mu_n C_{ox} W}{2L} (V_{gs} - V_t)^2 \text{ for } V \geq V_{gs} \text{ and } V_{ds} \geq V_{gs} - V_t.$$

Electrical characteristics of the nMOS enhancement-type transistor have been discussed above. In the depletion-type nMOS transistor, a channel is created by implanting suitable impurities in the region between the source and drain during fabrication prior to depositing the gate insulation layer and the poly-silicon layer. As a result, channel exists even when the gate voltage is 0 V. Here, the channel current can also be controlled by the gate voltage. A positive gate voltage increases the channel width resulting in an increase of drain current. A negative gate voltage
Fig. 3.11 Voltage–current characteristics of nMOS depletion-type transistor

![Voltage–current characteristics of nMOS depletion-type transistor](image)

decreases the channel width leading to a reduced drain current. A suitable negative gate voltage fully depletes the channel isolating the source and drain regions. The characteristic curve, as shown in Fig. 3.11, is similar except the threshold voltage, which is a negative voltage in case of a depletion-mode nMOS transistor. In a similar manner, the expression for drain current can be derived and voltage–current characteristics can be drawn for pMOS enhancement-mode and pMOS depletion-mode transistors.

### 3.5.1 Threshold Voltage

One of the parameters that characterize the switching behavior of an MOS transistor is its threshold voltage \( V_{tn} \). As we know, this can be defined as the gate voltage at which an MOS transistor begins to conduct. Typical value for threshold voltage for an nMOS enhancement-type transistor is 0.2 \( V_{dd} \) i.e., for a supply voltage of 5 V, \( V_{tn} = 1.0 \) V. As we have seen, the drain current depends on both the gate voltage and the drain voltage with respect to the source. For a fixed drain-to-source voltage, the variation of conduction of the channel region (represented by the drain current) for different gate voltages is shown in Fig. 3.11 for four different cases: nMOS depletion, nMOS enhancement, pMOS enhancement, and pMOS depletion transistors, as shown in Fig. 3.12a–d, respectively.

The threshold voltage is a function of a number of parameters, including gate conductor material, gate insulation material, thickness of the gate insulator, doping level in the channel regions, impurities in the silicon–insulator interface and voltage between the source and substrate \( V_{sb} \).

Moreover, the absolute value of the threshold voltage decreases with an increase in temperature at the rate of \(-2\text{mV/°C}\) and \(-4\text{mV/°C}\) for low and high substrate doping levels, respectively.
The threshold voltage may be expressed as
\[
V_t = V_{t_0} + \gamma(\sqrt{|-2\varphi_b + V_{sb}|} - \sqrt{|2\varphi_b|})
\] (3.12)

where the parameter \(\gamma\) is the substrate bias coefficient, \(\varphi_b\) is substrate Fermi potential and \(V_{sb}\) is the substrate-bias coefficient.

The expression holds good for both n-channel and p-channel devices.

- The substrate Fermi potential \(\varphi_b\) is negative in nMOS and positive in pMOS.
- The substrate bias coefficient \(\gamma\) is positive in nMOS and negative in pMOS.
- The substrate bias voltage \(V_{sb}\) is positive in nMOS and negative in pMOS.
$V_{t0}$ is the threshold voltage for $V_{sb} = 0$.

$$\varphi_b = \frac{KT}{q} \ln \left( \frac{n_i}{N_A} \right) = 0.026 \ln \left( \frac{1.45 \times 10^{10}}{10^6} \right) = -0.35 \text{V}$$

$$C_{ox} = \frac{\varepsilon_{ox}}{t_{ox}} = \frac{3.97 \times 8.85 \times 10^{-14}}{500 \times 10^{-8}} = 7.03 \times 10^{-8} \text{F/cm}^2$$

$$\gamma = \sqrt{\frac{2q\varepsilon_{si}N_i}{C_{ox}}} = \sqrt{\frac{2 \times 1.6 \times 10^{-19} \times 10^{16} \times 11.7 \times 8.85 \times 10^{-14}}{7.03 \times 10^{-8}}} = 0.82$$

$$V_i = V_{t0} + \lambda \sqrt{2 \varphi_b} + V_{sb} - \sqrt{2 \varphi_b} = 0.4 + 0.82 \sqrt{0.7 + V_{sb} - 0.7},$$

where $q$ is the charge of electron, $\varepsilon_{ox}$ is the dielectric constant of the silicon substrate, $N_A$ is the doping concentration densities of the substrate ($10^{16}$ cm$^{-3}$), and $C_{ox}$ is the oxide capacitance, $N_i$ is the carrier concentration of the intrinsic silicon ($1.45 \times 10^{10}$ cm$^{-3}$).

### 3.5.2 Transistor Transconductance $g_m$

Transconductance is represented by the change in drain current for a change in gate voltage for a constant value of drain voltage. This parameter is somewhat similar to $\beta$, the current gain of BJTs.

$$g_m = \frac{\delta I_{ds}}{\delta V_{gs}} \bigg|_{V_{ds}=\text{constant}}$$

(3.14)

This can be derived from

$$I_{ds} = \frac{Q_c}{t_{sd}} \quad \text{or} \quad \delta I_{ds} = \frac{\delta Q_c}{t_{sd}}.$$  

(3.15)

$$t_{sd} = \frac{L^2}{\mu_n V_{ds}}.$$  

(3.16)

Thus,

$$\delta I_{ds} = \frac{\delta Q_c V}{L^2} \mu_n.$$  

(3.17)

But,

$$\delta Q_c = C_g \delta V_{gs}.$$  

Thus,

$$\delta I_{ds} = \frac{\mu_n C_i V}{L^2} \delta V_{gs}.$$  

(3.18)
\[ I_d = \frac{C_g \mu_n V_{ds}}{\delta V_{gs} L^2} \]

in saturation \( V = (V - V_t) \),

and substituting \( C_g = \frac{\varepsilon_{\text{ins}} \varepsilon_0 W L}{D} \).

We get

\[ g_m = \mu_n \varepsilon_{\text{ins}} \varepsilon_0 W L \frac{V - V_t}{g_s} \].

(3.20)

**Figure of Merit**

The figure of merit \( W_0 \) gives us an idea about the frequency response of the device

\[ W = \frac{g_m}{C_g} = \frac{\mu_n (V - V_t) L^2}{g_s} \frac{1}{I_{sd}}. \]

(3.21)

A fast circuit requires \( g_m \) as high as possible and a small value of \( C_g \). From Eq. 3.23, it can be concluded that higher gate voltage and higher electron mobility provide better frequency response.

**Body Effect**

All MOS transistors are usually fabricated on a common substrate and substrate (body) voltage of all devices is normally constant. However, as we shall see in subsequent chapters, when circuits are realized using a number of MOS devices, several devices are connected in series. This results in different source potentials for different devices. It may be noted from Eq. 3.13 that the threshold voltage \( V_t \) is not constant with respect to the voltage difference between the substrate and the source of the MOS transistor. This is known as the substrate-bias effect or body effect. Increasing the \( V_{sb} \) causes the channel to be depleted of charge carriers, and this leads to an increase in the threshold voltage.

Using Eq. 3.13, we compute and plot the threshold voltage \( V_t \) as a function of the source-to-substrate voltage \( V_{sb} \). The voltage \( V_{sb} \) will be assumed to vary between 0 and 5 V. The graph obtained is shown in Fig. 3.13.
The variation of the threshold voltage due to the body effect is unavoidable in many situations, and the circuit designer should take appropriate measures to overcome the ill effects of this threshold voltage variation.

**Channel-Length Modulation**

Simplified equations derived in Sect. 3.3 to represent the behavior of an MOS transistor is based on the assumption that channel length remains constant as the drain voltage is increased appreciably beyond the onset of saturation. As a consequence, the drain current remains constant in the saturation region. In practice, however, the channel length shortens as the drain voltage is increased. For long channel lengths, say more than 5 μm, this variation of length is relatively very small compared to the total length and is of little consequence. However, as the device sizes are scaled down, the variation of length becomes more and more predominant and should be taken into consideration.

To have better insight of this phenomenon, let us examine the mechanisms of the formation of channel and current flow in an MOS transistor in different operating conditions. Figure 3.14a shows the situation of an MOS transistor operating in the active or nonsaturation region \( (0 < V_{ds} < V_{gs} - V_{th}) \). In this mode, the inversion layer (i.e., channel) formed under the influence of gate voltage provides a current

**Fig. 3.13** Variation of the threshold voltage as a function of the source-to-substrate voltage

**Fig. 3.14** a Nonsaturated region. b Onset of saturation. c Deep in saturation
path between the source and drain. As the drain voltage is increased from zero, the current flow increases linearly with the drain voltage, and the channel depth at the drain end also gradually decreases. Eventually at drain voltage $V_{ds} = V_{gs} - V_t$, the inversion charge and the channel depth reduces to zero as shown in Fig. 3.14b. This is known as the pinch-off point. As the drain voltage is increased further, a depletion region is formed adjacent to the drain, and the depletion region gradually grows with the increase in drain voltage. This leads to gradual shifting of the pinch-off point towards the source, thereby reducing channel length as shown in Fig. 3.14c. This effective channel length $L_{eff}$ can be represented by

$$L_{eff} = L - \Delta L. \quad (3.22)$$

Substituting Eq. 3.14 in Eq. 3.11, we get

$$I_{ds(sat)} = \frac{\mu_n C_{ox}}{2} \left( \frac{W}{L} \right) \left( V_{gs} - V_t - V \right)^2. \quad (3.23)$$

This expression can be rewritten in terms of $\lambda$, known as channel-length modulation coefficient. It can be shown that $\Delta L \propto \sqrt{V_{ds} - V_{ds(sat)}}$

$$1 - \frac{\Delta L}{L} \approx 1 - \frac{\Delta L}{L}$$

Assuming $\lambda V_{ds} \ll 1$,

$$I_{ds(sat)} = \frac{\mu_n C_{ox}}{2} \left( \frac{W}{L} \right) \left( V_{gs} - V \right)^2 (1 + \lambda V_{ds}) \quad (3.23)$$

The channel-length modulation coefficient $\lambda$ has the value in the range of 0.02–0.005 per volt. Taking into consideration the channel-length modulation effect, the voltage–current characteristic is shown in Fig. 3.15.
MOS Transistors as a Switch

We have seen that in the linear region (when the drain-to-source voltage is small) an MOS transistor acts as a variable resistance, which can be controlled by the gate voltage. An nMOS transistor can be switched from very high resistance when the gate voltage is less than the threshold voltage $V_{th}$, to low resistance when $V_{gs}$ exceeds the threshold voltage $V_{th}$. This has opened up the possibility of using an MOS transistor as a switch, just like a relay. For example, an nMOS transistor when used as a switch is OFF when $V_{gs} = 0 \text{V}$ and ON when $V_{gs} = V_{dd}$. However, its behavior as a switch is not ideal. When $V_{gs} = V_{dd}$, the switch turns on but the on resistance is not zero. As a result, there is some voltage drop across the switch, which can be neglected when it is in series with a large resistance. Moreover, if $V_{dd}$ is applied to the input terminal, at the other end we shall get $(V_{dd} - V_{th})$. This is because when output voltage is more than $(V_{dd} - V_{th})$, the channel turns off, and it no longer functions as a closed switch as shown in Fig. 3.15a. However, a low-level signal can be passed without any degradation. The transistor used in the above manner is known as pass transistor. It may be noted that the roles of drain and source are interchangeable, and the device truly acts as a bilateral switch.

Similarly, a pMOS transistor can also be used as a switch. In this case, the minimum voltage that it can pass is $V_{tp}$, since below this value gate-to-source voltage will be higher than $-V_{tp}$ and the transistor turns off. This is shown in Fig. 3.16b. Therefore, a p-channel transistor passes a weak low-level signal but a strong high-level signal as shown below. Later, we shall discuss the use of pass transistors in realizing Boolean functions and discuss its advantages and disadvantages.

To overcome the limitation of either of the transistors, one pMOS and one nMOS transistor can be connected in parallel with complementary inputs at their gates. In this case, we can get both low and high levels of good quality of the output. The low level passes through the nMOS switch, and the high level passes through the pMOS switch without any degradation as shown in Fig. 3.16c. A more detailed discussion on transmission gates is given in the following subsection.

### 3.6.1 Transmission Gate

The transmission gate is one of the basic building blocks of MOS circuits. It finds use in realizing multiplexors, logic circuits, latch elements, and analog switches.
The characteristics of a transmission gate, which is realized by using one nMOS and one pMOS pass transistors connected in parallel, can be constructed by combining the characteristics of both the devices. It may be noted that the operation of a transmission gate requires a dual-rail (both true and its complement) control signal. Both the devices are off when “0” and “1” logic levels are applied to the gates of the nMOS and pMOS transistors, respectively. In this situation, no signal passes through the gate. Therefore, the output is in the high-impedance state, and the intrinsic load capacitance associated to the output node retains the high or low voltage levels, whatever it was having at the time of turning off the transistors. Both the devices are on when a “1” and a “0” prior to the logic levels are applied to the gates of the nMOS and pMOS transistors, respectively. Both the devices take part in passing the input signal to the output. However, as discussed below, their contributions are different in different situations.

To understand the operation of a transmission gate, let us consider two situations. In the first case, the transmission gate is connected to a relatively large capacitive load, and the output changes the state from low to high or high to low as shown in Fig. 3.17.

**Case I: Large Capacitive Load** First, consider the case when the input has changed quickly to $V_{dd}$ from 0 V and the output of the switch changes slowly from 0 V ($V_{ss}$) to $V_{dd}$ to charge a load capacitance $C_L$. This can be modeled by using $V_{dd}$ as an input and a ramp voltage generated at the output as the capacitor charges from $V_{ss}$ to $V_{dd}$. Based on the output voltage, the operations of the two transistors can be divided into the following three regions:

Region I: As the voltage difference between the input and output is large, both nMOS and pMOS transistors are in saturation. Here, $V_{out} < V_{tp}$.

Region II: nMOS is in saturation and pMOS in linear for $V_{tp} < V_{out} < V_{dd} - V_{tn}$.

Region III: nMOS is in cutoff and pMOS in linear for $V_{out} > V_{dd} - V_{tn}$.

Region I: Here,

$$V_{dsn} = V_{dd} - V_{out},$$
$$V_{gns} = V_{dd} - V_{out},$$
$$V_{dsp} = V_{out} - V_{dd},$$
$$V_{gps} = -V_{dd}$$

The current contributing to charge the load capacitor by the two transistors is

$$I = K \frac{W_n}{L_n} (V_{dd} - V_{out} - V_{tn})^2,$$ (3.24)

$$I = K \frac{W_p}{2L_n} \left( V_{dd} - V_{tp} \right)^2,$$ (3.25)

for the nMOS and pMOS transistors, respectively.
Now, the equivalent resistances for the two transistors are

$$ R_{cnp} = \frac{V_{dd} - V_{out}}{I_{dsn}} = \frac{2L_n}{K W_n} \cdot \frac{(V_{dd} - V_{out})}{(V_{dd} - V_{tn})^2} \quad (3.26) $$

and

$$ R_{cp} = \frac{V_{dd} - V_{out}}{I_{sdp}} = \frac{2L_p}{K P W_p} \cdot \frac{(V_{dd} - V_{out})}{(V_{dd} - |V_{tp}|)^2} \quad (3.27) $$
Region II: In this region, the nMOS transistor remains in saturation region, whereas the pMOS transistor operates in the linear region. Therefore, in this case

\[ I_{\text{dp}} = \frac{K \, W}{L_p} \left( V_{dd} - V_{tp} \right) \left( V - V_{dd} \right) \left( V_{out} - V_{out} - V_{dd} \right) \]  

(3.28)

\[ R_{\text{eqp}} = \frac{2L_p}{K_p \, W} \left( V_{dd} - V_{tp} \right) \frac{1}{2 \left( V_{dd} - V_{out} \right) - \left( V_{dd} - V_{out} \right)} \]  

(3.29)

Region III: In this region, the nMOS transistor turns off and pMOS transistor continues to operate in the linear region.

These individual nMOS and pMOS currents and the combined current are shown in Fig. 3.17c. It may be noted that the current decreases linearly as voltage builds up across the capacitor \( C_L \). The equivalent resistances and their combined values are shown in Fig. 3.17d.

Similarly, when the input voltage changes quickly from \( V_{dd} \) to 0 V and the load capacitance discharges through the switch, it can be visualized by Fig. 3.17e–h.

Region I: Both nMOS and pMOS are in saturation for \( V_{out} < \frac{1}{2} \left( V_{dd} - V_{tn} \right) \).

Region II: nMOS is in the linear region, and pMOS is in saturation for \( \left( V_{dd} - V_{tp} \right) \) \( V_{out} < V_{tn} \).

Region III: nMOS is in the linear region, and pMOS is cutoff for \( V_{out} < \left( V_{dd} - V_{tn} \right) \).

As shown in Fig. 3.17f, the current decreases linearly as voltage across the capacitor decreases from \( V_{dd} \) to 0 V. Note that the role of the two transistors reverses in the two cases.

**Case II: Small Capacitive Load** Another situation is the operation of the transmission gate when the output is lightly loaded (smaller load capacitance). In this case, the output closely follows the input. This is represented in Fig. 3.18a.
In this case, the transistors operate in three regions depending on the input voltage as follows:

Region I: nMOS is in the linear region, pMOS is cutoff for $V_{in} < V_{tp}$.

Region II: nMOS is in the linear region, pMOS linear for $V_{tp} < V_{in} < (V_{dd} - |V_{tn}|)$.

Region III: nMOS is cutoff, pMOS is in the linear region for $V_{in} > (V_{dd} - |V_{tn}|)$.

As the voltage difference between the transistors is always small, the transistors either operate in the nonsaturated region or are off as shown above. The individual currents along with the total current are shown in Fig. 3.18b. The variation of the on resistance and combined resistance is also shown in Fig. 3.18c.
Introduction

In Chap. 3, we have seen that a metal–oxide–semiconductor (MOS) transistor can be considered as a voltage-controlled resistor. This basic property can be used to realize digital circuits using MOS transistors. In this chapter, we discuss the realization of various types of MOS inverters. The inverter forms the basic building block of gate-based digital circuits. An inverter can be realized with the source of an n-type metal–oxide–semiconductor (nMOS) enhancement transistor connected to the ground, and the drain connected to the positive supply rail $V_{dd}$ through a pull-up device. The generalized block diagram is shown in Fig. 4.1. The input voltage is applied to the gate of the nMOS transistor with respect to ground and output is taken from the drain. When the MOS transistor is ON, it pulls down the output voltage to the low level, and that is why it is called a pull-down device, and the other device, which is connected to $V_{dd}$, is called the pull-up device.
Fig. 4.1 General structure of an nMOS inverter. nMOS n-type metal–oxide–semiconductor

Fig. 4.2 Truth table and logic symbol of the inverter

<table>
<thead>
<tr>
<th>$V_{\text{in}}$</th>
<th>$V_{\text{out}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The pull-up device can be realized in several ways. The characteristics of the inverter strongly depend on the pull-up device used to realize the inverter. Theoretically, a passive resistor of suitable value can be used. Although the use of a possible resistor may be possible in realizing an inverter using discrete components, this is not feasible in very-large-scale integration (VLSI) implementation. Instead, an active pull-up device realized using a depletion-mode nMOS transistor or an enhancement-mode nMOS transistor or a p-type metal–oxide–semiconductor (pMOS) transistor could be used. Basic characteristics of MOS inverters are highlighted in Sect. 4.2. The advantages and disadvantages of different inverter configurations are explored in Sect. 4.3 Section 4.3 explores the inverter ratio in different situations. The switching characteristics on MOS inverters are considered in Sect. 4.5 Various delay parameters have been estimated in Sect. 4.6 Section 4.7 presents the different circuit configurations to drive a large capacitive load.

**Inverter and Its Characteristics**

Before we discuss about the practical inverters realized with MOS transistors, we consider the characteristics of an ideal inverter [1, 2]. The truth table and logic symbol of an inverter are shown in Fig. 4.2. The input to the inverter is $V_{\text{in}}$ and output is $V_{\text{out}}$.

Figure 4.3 shows how the output of an ideal inverter changes as the input of the inverter is varied from 0 V (logic level 0) to $V_{\text{dd}}$ (logic level 1). Initially, output is $V_{\text{dd}}$ when the output is 0 V, and as the input crosses $V_{\text{dd}}/2$, the output switches to 0 V, and it remains at this level till the maximum input voltage $V_{\text{dd}}$. This diagram is known as the input–output or transfer characteristic of the inverter. The input voltage, $V_{\text{dd}}/2$, at which the output changes from high ‘1’ to low ‘0’, is known as inverter threshold voltage. For practical inverters realized with MOS devices, the voltage transfer characteristics will be far from this ideal voltage transfer characteristic represented by Fig. 4.3. A more realistic voltage transfer characteristic is shown in Fig. 4.4a. As shown in Fig. 4.4a, because of some voltage drop across the pull-up device, the out-
put high voltage level is less than $V_{dd}$ for the low input voltage level. This voltage is represented by $V_{OH}$, which is the maximum output voltage level for output level ‘1’.

As the input voltage increases and crosses the threshold voltage of the pull-down transistor, it starts conducting, which leads to a decrease in the output voltage level.

However, instead of an abrupt change in the voltage level from logic level ‘1’ to logic level ‘0’, the voltage decreases rather slowly. The unity gain point at which $\frac{dV_o}{dV_{in}} = -1$ is defined as the input high voltage $V_{IH}$, which is the maximum input voltage which can be treated as logic level ‘0’.

As the input voltage is increased further, the output crosses a point where $V_{in} = V_{out}$. The voltage at which this occurs is referred to as the inverter threshold voltage $V_T$.

Fig. 4.4 a Various voltage levels on the transfer characteristics; b low- and high-level noise margins
It may be noted that the inverter threshold voltage may not be equal to \( V_{dd}/2 \) for practical inverters. Before the output attains the output low voltage \( V_{OL} \), which is the minimum output voltage for output logic level ‘0’, the transfer-characteristic curve crosses another important point \( V_{IH} \), the minimum input voltage that can be accepted as logic ‘1’. This point is also obtained at another unity gain point at which \( \frac{dV_o}{dV_i} = -1 \) as shown in Fig. 4.4a.

An important parameter called the noise margin is associated with the input–output voltage characteristics of a gate. It is defined as the allowable noise voltage on the input of a gate so that the output is not affected. The deviations in logic levels from the ideal values, which are restored as the signal propagates to the output, can be obtained from the DC characteristic curves. The logic levels at the input and output are given by

- logic 0 input: \( 0 \leq V_{in} \leq V_{IL} \),
- logic 1 input: \( V_{IH} \leq V_{in} \leq V_{dd} \),
- logic 0 output: \( 0 \leq V_{0} \leq V_{OL} \),
- logic 1 output: \( V_{OH} \leq V_{0} \leq V_{dd} \).

The low-level noise margin is defined as the difference in magnitude between the minimum low output voltage of the driving gate and the maximum input low voltage accepted by the driven gate.

\[
NM_L = |V_{IL} - V_{OL}| \tag{4.1}
\]

The high-level noise margin is defined as the difference in magnitude between the minimum high output voltage of the driving gate and the minimum voltage acceptable as high level by the driven gate:

\[
NM_H = |V_{OH} - V_{IH}| \tag{4.2}
\]

To find out the noise margin, we can use the transfer characteristics as shown in Fig. 4.4a. The noise margins are shown in Fig. 4.4b.

When any of the noise margins is low, the gate is susceptible to a switching noise at the input.

**MOS Inverter Configurations**

The various MOS inverter configurations [3] realized using different types of pull-up devices are discussed in this section. In Sect. 4.3.1, the use of a passive resistor as the pull-up device is discussed and disadvantages are highlighted. The use of a depletion-mode nMOS transistor as the pull-up device is discussed in Sect. 4.3.2. Section 4.3.3 discusses the use of an enhancement mode of nMOS transistor, whereas Sect. 4.3.4 discusses the use of a pMOS transistor as a pull-up
device in configuration. The pMOS device can also be used to realize the CMOS inverter, where the two transistors are used in complementary mode, as discussed in Sect. 4.3.5. Various inverters introduced in this section are compared in Sect. 4.3.6.

**Passive Resistive as Pull-up Device**

A passive resistor $R_L$ can be used as the pull-up device as shown in Fig. 4.5a. The value of the resistor should be chosen such that the circuit functionally behaves like an inverter. When the input voltage $V_{in}$ is less than $V_{tn}$, the transistor is OFF and the output capacitor charges to $V_{dd}$. Therefore, we get $V_{dd}$ as the output for any input voltage less than $V_{tn}$. When $V_{in}$ is greater than $V_{tn}$, the MOS transistor acts as a resistor $R_c$, where $R_c$ is the channel resistance with $V_{gs} > V_{tn}$. The output capacitor discharges through this resistor and output voltage is given by

$$V_{OL} = V_{dd} \frac{R_c}{R_c + R_L} \quad (4.3)$$

Normally, this output is used to drive other gates. Functionally, this voltage can be accepted as low level provided it is less than $V_t$. So,

$$V_{OL} = V_{dd} \frac{R_c}{R_c + R_L} < V_{tn}$$

Assuming the typical value of threshold voltage $V_{tn} = 0.2V_{dd}$, we get

$$V_{OL} = V_{dd} \frac{R_c}{R_c + R_L} \leq 0.2V_{dd} \quad \text{or} \quad R_L > 4R_c \quad (4.4)$$

This imposes a restriction on the minimum value of load resistance for a successful operation of the circuit as an inverter. The input–output characteristic of the inverter...
Fig. 4.6 Realization of a resistive load

is shown in Fig. 4.5b. The circuit operates along the load line as shown in Fig. 4.5b. For $V_{in} = 0$ V, the output voltage $V_{out} = V_{dd}$ (point A), and for $V_{in} = V_{dd}$, the output voltage $V_{out} = V_{OL}$, as shown by point B. The transfer characteristic is shown in Fig. 4.5c, which shows that the output is $V_{dd}$ for $V_{in} = 0$ V, but for $V_{in} = V_{dd}$ the output is not $0$ V.

This implementation of this inverter has a number of disadvantages:

- As the charging of the output capacitor takes place through the load resistor $R_L$ and discharge through $R_c$ and their values must be different as per Eq. 4.4, there is asymmetry in the ON-to-OFF and OFF-to-ON switching times.
- To have higher speeds of operation, the value of both $R_c$ and $R_L$ should be reduced. However, this increases the power dissipation of the circuit. Moreover, as we shall see later, to achieve a smaller value of $R_c$, the area of the MOS inverter needs to be increased.
- The resistive load can be fabricated by two approaches—using a diffused resistor approach or using an undoped poly-silicon approach. In the first case, an n-type or a p-type isolated diffusion region can be fabricated to realize a resistor between the power supply line and the drain of the nMOS transistor. To realize a resistor of the order of few $K\ \Omega$, as required for proper operation of the circuit, the length to width must be large. To realize this large length-to-width ratio in a small area, a serpentine form is used as shown in Fig. 4.6. However, this requires a very large chip area. To overcome the limitation of this approach, the second approach based on undoped poly-silicon can be used. Instead of using doped poly-silicon, which is commonly used to realize the gate and interconnect regions having lower resistivity, undoped poly-silicon is used here to get higher resistivity. Although this approach leads to a very compact resistor compared to the previous approach, the resistance value cannot be accurately controlled leading to large process parameter variations. In view of the above discussion, it is evident that this inverter configuration is not suitable for VLSI realization. Better alternatives for the realization of the pull-up resistor are explored in the following subsections.

**nMOS Depletion-Mode Transistor as Pull up**

To overcome the limitations mentioned above, MOS transistors can be used as pull-up devices instead of using a passive resistor. There are three possible alternatives for pull-up devices—an nMOS enhancement-mode transistor, a depletion-mode
Fig. 4.7  a nMOS inverter with depletion-mode transistor as pull-up device; b voltage current characteristic; c transfer characteristic. nMOS n-type metal–oxide–semiconductor

nMOS transistor, or a pMOS transistor. Any one of the transistors can be used as a pull-up device. First, we consider the use of an nMOS depletion-mode transistor as an active pull-up (pu) device as shown in Fig. 4.7a. As the output of an inverter is commonly connected to the gate of one or more MOS transistors in the next stage, there is no fan-out current, and the currents flowing through both the transistors must be equal. The input voltage is applied to the gate of the pull-down (pd) transistor, and the output is taken out from the drain of the pd device.

1. **Pull-down device off and pull-up device in linear region:** This corresponds to point ‘A’ on the curve with the input voltage $V_{\text{in}} < V_{\text{tn}}$. $V_{\text{out}} = V_{\text{dd}}$ and $I_{\text{ds}} = 0$.

In this situation, there is no current flow from the power supply and no current flows through either of the transistors.

2. **Pull-down device in saturation and pull-up device in linear region:** This corresponds to point B. Here,

$$I = \frac{K_n W_{\text{pd}}}{2L_{\text{pd}}} (V - V_{\text{in}})^2$$

and

$$I = \frac{K_n W_{\text{pu}}}{L_{\text{pu}}} \left[ (V - V_{\text{out}} - V_{\text{out}}^2) \right]$$

where $V_{\text{tpd}}$ and $V_{\text{tpu}}$ are the threshold voltages of the enhancement- and depletion-mode MOS transistors, respectively.

3. **Pull-down and pull-up device, both in saturation:** This is represented by point C on the curve. In this situation,

$$I = \frac{K_n W_{\text{pd}}}{2L_{\text{pd}}} (V - V_{\text{in}})^2$$

and

$$I = \frac{K_n W_{\text{pu}}}{2L_{\text{pu}}} V^2.$$
4. **Pull-down device in linear region and pull-up device in saturation:** This situation occurs when input voltage is equal to $V_{dd}$. Here,

$$
I_{pd} = \beta_{pd} \left( V_{pd}^2 - V_{OL} \right),
$$

$$
I_{pu} = \frac{\beta_{pu}}{2} V_{pu}^2,
$$

(4.8)

where $\beta_{pd} = K_{W_{pd}}$ and $\beta_{pu} = K_{W_{pu}}$.

Equating the two currents and ignoring $V_{OL}/2$ term, we get

$$
\beta_{pd} \left( V_{dd} - V_{in} \right) = \frac{\beta_{pu}}{2} \left( V_{pu} \right)^2,
$$

(4.9)

$$
V_{OL} = \frac{1}{2K} \frac{\left( V_{pu} \right)^2}{\left( V_{dd} - V_{OL} \right)} = \frac{\beta_{pu}}{2\beta_{pd}} \frac{\left( V_{pu} \right)^2}{\left( V_{dd} - V_{OL} \right)} = \frac{1}{2K} \frac{\left( V_{pu} \right)^2}{\left( V_{dd} - V_{OL} \right)},
$$

(4.10)

where

$$
K = \frac{\beta_{pd}}{\beta_{pu}} \frac{(W/L)_{pd}}{(W/L)_{pu}}
$$

(4.11)

The quantity $K$ is called the *ratio of the inverter*. For successful inverter operation, the low output voltage, $V_{OL}$, should be smaller than the threshold voltage of the pull-down transistor of the next stage. From the above discussion, we can make the following conclusion:

- The output is not ratioless, which leads to asymmetry in switching characteristics.
- There is static power dissipation when the output logic level is low.
- It produces strong high output level, but weak low output level.

### 4.3.3 nMOS Enhancement-Mode Transistor as Pull up

Alternatively, an enhancement-mode nMOS transistor with gate normally connected to its drain ($V_{dd}$) can be used as an active pull-up resistor as shown in Fig. 4.8a. Let us consider the output voltage for two situations—when $V_{in} = 0$ and $V_{in} = V_{dd}$. In the first case, the desired output is $V_{dd}$. But as the output, $V_{out}$ approaches the voltage ($V_{dd} - V_{in}$), the pull-up transistor turns off. Therefore, the output voltage cannot
reach $V_{dd}$. The maximum output voltage that can be attained is $(V_{dd} - V_{tn})$, where $V_{tn}$ is the threshold voltage of the enhancement-mode pull-up transistor. The output voltage for $V_{in} = V_{dd}$ is not 0 V, because in this case both the transistors are conducting and act as a voltage divider. The transfer characteristic is shown in Fig. 4.8b. From the above discussion, we can make the following conclusion:

- The output is not ratioless, which leads to asymmetry in switching characteristics.
- There is static power dissipation when the output level is low.
- It produces weak low and high output levels.

As a consequence, nMOS enhancement-type transistor is not suitable as a pull-up device for realizing an MOS inverter.

**The pMOS Transistor as Pull Up**

We can realize another type of inverter with a pMOS transistor as a pull-up device with its gate permanently connected to the ground as shown in Fig. 4.9a. As it is functionally similar to a depletion-type nMOS load, it is called a ‘pseudo-nMOS’ inverter. Unlike the CMOS inverter, discussed in Sect. 4.2.4, the pull-up transistor always remains ON, and there is DC current flow when the pull-down device is ON. The low-level output is also not zero and is dependent on the $\beta_n / \beta_p$ ratio like the depletion-type nMOS load. The voltage-transfer characteristic is shown in Fig. 4.9b.
pMOS Transistor as a Pull Up in Complementary Mode

In this case, a pMOS enhancement type transistor is used as a pull-up device. However, here the gates of both the pull-up and pull-down transistors are tied together and used as input as shown in Fig. 4.10a. Output is taken from the drain of the pull-down device as usual. In this case, when the input voltage \( V_{\text{in}} = 0 \) V, the gate input of the pull-up transistor is below \( V_{dd} \) of its source voltage, i.e., \( V_{gs} = -V_{dd} \), which makes the pull-up transistor ON, and the pull-down transistor OFF. So, there is no DC current flow between \( V_{dd} \) to ground. When the input voltage \( V_{\text{in}} = V_{dd} \), the gate input of the pull-up transistor is zero with respect to its source, which makes it OFF. The pull-down transistor, however, is ON because the \( V_{gs} = V_{dd} \). In this situation also, there is no DC current flow between \( V_{dd} \) and ground. However, as the gate voltage is gradually increased from ‘0’ to ‘1’, the pull-up transistor switches from ON to OFF and the pull-down transistor switches from OFF to ON. Around the midpoint, both transistors are ON and DC current flows between \( V_{dd} \) and ground. Detailed analysis can be made by dividing the entire region of operation into five basic regions as follows:

Region 1: \( 0 \leq V_{\text{in}} < V_{tn} \) The pull-down transistor is off and the pull-up transistor is in the linear (subthreshold) region as shown by a representative point ‘A’ on the superimposed nMOS and pMOS transistor’s drain-to-source voltage–current characteristic curves. In this region, there is no DC current flow and output voltage remains close to \( V_{dd} \). This corresponds to point ‘A’ in Fig. 4.10b.

Region 2: \( V_{tn} < V_{\text{in}} < V_{\text{inv}} \) Here, the pull-down transistor moves into a saturation region and the pull-up transistor remains in the linear region as represented by point B, when the input is \( V_{il} \). The pull-down transistor acts as a current source and pull-up transistor acts as a resistor. The current flow through the transistor increases as \( V_{\text{in}} \) increases from \( V_{tn} \) to \( V_{\text{inv}} \) and attains a maximum value when \( V_{\text{in}} = V_{\text{inv}} \).

Since the same current flows through both the devices, \( I_{\text{isp}} = -I_{\text{dsn}} \).

For the pMOS devices, the drain current is given by

\[
I_{\text{dsp}} = -\beta_p \left[ (V_{\text{in}} - V_{\text{dd}})(V_{\text{dd}} - V_{\text{tp}}) - \frac{1}{2} (V_{\text{dd}} - V_{\text{oph}})^{2} \right]
\]

(4.12)
where

\[ \beta_p = K_p \frac{W}{L_p} \]

\[ V_{gs} = V_{dd} - V_{in} \]

\[ V_{dp} = V_{in} - V_{out} \]

The saturation current of the nMOS transistor is given by

\[ I_{dsn} = \beta_n \frac{(V_{in} - V_{tn})^2}{2} \]

where \( \beta_n = K_n \frac{W_n}{L_n} \) and \( V_{gsn} = V_{in} \).

Equating these two, we get

\[ V = (V_{in} - V) + \sqrt{\frac{(V_{in} - V)^2 - 2(V_{out} - V_{dd})}{2}} \frac{V_{dd} - V_{tn}}{\beta_p} \cdot \frac{(V_{in} - V_{tn})^2}{2} \]

(4.13)

A plot of this is shown in Region II of Fig. 4.10c.

The \( V_{in} \) corresponds to the point \( \frac{dV_{out}}{dV_{in}} = -1 \), and, at this point, the nMOS transistor operates in the saturation region, whereas the pMOS transistor operates in the linear region. Equating \( I_{dsn} = -I_{dsp} \), we get

\[ \frac{\beta_p (V_{in} - V)^2}{2} = \frac{2}{\beta_p} \left[ 2(V_{out} - V_{in}) \right] \left[ (V_{in} - V_{tn})^2 \right] \]

Substituting

\[ V_{gsn} = V_{in}, \quad V_{gsp} = -(V_{dd} - V_{in}) \quad \text{and} \quad V_{dsp} = -(V_{dd} - V_{out}) \]

we get

\[ \frac{\beta_n}{2} (V_{in} - V)^2 = \frac{\beta_p}{2} \left[ 2(V_{out} - V_{in}) \right] \left[ (V_{out} - V_{in}) - (V_{out} - V_{in}) \right] \]

(4.14)

Differentiating both sides with respect to \( V_{in} \), we get

\[ \frac{\beta_n}{4} (V_{in} - V)^2 = \frac{\beta_p}{4} \left[ 2(V_{out} - V_{in}) \right] \left[ \frac{dV_{out}}{dV_{in}} - 2(V_{out} - V_{in}) \right] \left[ \frac{dV_{out}}{dV_{in}} \right] \]

(4.15)

Substituting, \( V_{in} = V_{in} \) and \( \frac{dV_{out}}{dV_{in}} = -1 \), we get

\[ \beta_n (V_{tn} - V_{in}) = \beta_p (2V_{out} - V_{in} + V_{in} - V_{pp}) \]

or

\[ V_{in} = \frac{2V_{out} + V_{in} - V_{pp} + (\beta_n / \beta_p) \cdot V_{tn}}{(1 + (\beta_n / \beta_p))} \]

(4.15)
For $\beta_n / \beta_p = 1$, and $V_{out} \equiv V_{dd}$, the value of $V_{IL}$ can be found out to be

$$V_{IL} = \frac{1}{8} (3V + 2V_{dd})$$  \hspace{1cm} (4.16)

**Region 3:** $V_{in} = V_{inv}$. At this point, both the transistors are in the saturation condition as represented by the point C on the superimposed characteristic curves. In this region, both the transistors can be modeled as current sources.

Assuming

$$V_{gs}^{pd} = V_{in} \quad \text{and} \quad V_{gs}^{pu} = V_{in} - V_{dd} = V_{inv} - V_{dd},$$

we may equate the saturation currents of the pull-up and pull-down transistors to get

$$I_{dsn} = \frac{1}{2} K_n \frac{W_n}{L_n} (V - V_{inv})^2$$

$$I_{dsp} = -\frac{1}{2} K_p \frac{W_p}{L_p} (V_{inv} - V_{dd} - V_{tp})^2$$

Equating we get,

$$\frac{\beta_n}{2} (V - V_{inv})^2 = - \frac{\beta_p}{2} (V_{inv} - V_{dd} - V_{tp})^2$$

$$or \quad \frac{V_{inv} - V_{dd} - V_{tp}}{V_{inv} - V_{in}} = \sqrt{\frac{\beta_n}{\beta_p}}$$

$$or \quad V_{inv} = \left(1 + \sqrt{\frac{\beta_n}{\beta_p}}\right) V_{in} + V + \sqrt{\frac{\beta_n}{\beta_p}}$$

$$or \quad V_{inv} = \frac{V_{dd} + V_{tp} + \sqrt{\frac{\beta_n}{\beta_p}}}{1 + \sqrt{\frac{\beta_n}{\beta_p}}}$$  \hspace{1cm} (4.17)

For

$$\beta_n = \beta_p \quad \text{and} \quad V_{in} = -V_{tp}, \quad \text{we get} \quad V_{inv} = V_{dd} / 2.$$  

In a CMOS process,
\[
\frac{K_n}{K_p} = \frac{\mu_n}{\mu_p} \approx 2.5
\]

To make \( \beta_p = \beta_p \), one may choose
\[
\left[ \frac{W}{L} \right]_p = 2.5 \left[ \frac{W}{L} \right]_n,
\]
which can be realized with pMOS channel 2.5 times wider than the nMOS channel of the same length. The inverter acts as a symmetrical gate when this is realized.

Since both the transistors are in saturation, they act as current sources and at this point the current attains a maximum value as shown in Fig. 4.10c. This leads to power dissipation, known as short-circuit power dissipation. Later, we will derive a detailed expression for short-circuit power dissipation.

**Region 4:** \( V_{\text{inv}} < V_n \leq V_{dd} \)

As the input voltage has been increased above \( V_{\text{inv}} \), the nMOS transistor moves from the saturation region to the linear region, whereas the pMOS transistor remains in saturation. With the increase in input voltage beyond \( V_{\text{inv}} \), the output voltage and also the drain current continue to drop. A representative point in this region is point D. In this region, nMOS transistor acts as a resistor and pMOS transistor acts as a current source.

The drain current for the two transistors are given by
\[
I_{dsn} = \beta_n \left( V_{\text{in}} - V_{tn} \right) V_O - \frac{\alpha}{2} \quad \text{and} \quad I_{dsp} = -\beta_p \left( V_{\text{in}} - V_{dd} - V_{tp} \right)^2
\]

As
\[
I_{dsn} = -I_{dsp},
\]
we get
\[
\beta_n \left( V_{\text{in}} - V_{tn} \right) V_O - \frac{\alpha}{2} = \beta_p \left( V_{\text{in}} - V_{dd} - V_{tp} \right)^2
\]
or
\[
\frac{V_{\text{in}}}{2} - (V_{\text{in}} - V_{\text{in}}) V_O + \frac{\beta_p}{\beta_n} (V - V_{dd} - V_{tp})^2 = 0.
\]
Solving for $V_O$, we get

$$V = \frac{(V - V_{\text{in}}) - \beta_n \sqrt{(V_{\text{in}} - V_{\text{IH}})^2 - \beta_p (V_{\text{in}} - V_{\text{dd} - \text{tp}})^2}}{\beta_n}$$

Similarly, we can find out the value of $V_{\text{IH}}$ when the nMOS transistor operates in the linear region and pMOS transistor operates in the saturation region. Equating $I_{\text{dsn}} = I_{\text{dsp}}$, at this point, we get

$$\frac{\beta_n}{2} \left[ 2(V_{\text{gsn}} - V_{\text{in}}) \cdot V_{\text{dsn}} - V_{\text{out}} \right] = \frac{\beta_p}{2} (V_{\text{gsp}} - V_{\text{in}})^2.$$

Substituting $V_{\text{gsp}} = -(V_{\text{dd} - V_{\text{in}}})$ and $V_{\text{dsp}} = -(V_{\text{dd} - V_{\text{out}}})$, we get

$$\frac{\beta_n}{2} \left[ 2(V_{\text{in}} - V_{\text{out}}) \cdot V_{\text{dsn}} - V_{\text{out}}^2 \right] = \frac{\beta_p}{2} (V_{\text{in}} - V_{\text{dd} - \text{tp}}^2).$$

Differentiating both sides with respect to $V_{\text{in}}$, we get

$$\beta_n \left[ (V - V_{\text{in}}) \frac{dV_{\text{out}}}{dV_{\text{in}}} + V_{\text{out}} \right] - \beta_p \left( V_{\text{out}} \frac{dV_{\text{in}}}{dV_{\text{out}}} \right) = \beta_p \left( V_{\text{in}} - V_{\text{dd} - \text{tp}} \right).$$

Now, substituting $V_{\text{in}} = V_{\text{IH}}$ and $dV_{\text{out}}/dV_{\text{in}} = -1$, we get

$$\beta_n (-V_{\text{IH}} + V_{\text{in}} + 2V_{\text{out}}) = \beta_p (V_{\text{IH}} - V_{\text{dd} - \text{tp}})$$

or

$$V_{\text{IH}} = \frac{V + V_{\text{dd} - \text{tp}} + \beta_n (2V_{\text{out}} - V_{\text{in}})}{1 + \frac{\beta_n}{\beta_p}}.$$

This equation can be solved with the Kirchhoff’s current law (KCL) equation to obtain numerical values of $V_{\text{IH}}$ and $V$. For $\beta / \beta_{\text{p}} = 1$, the value of $V = \frac{1}{8} (5V_{\text{dd} - \text{tn}} - 2V_{\text{IH}})$.

For a symmetric inverter, $V_{\text{IH}} + V_{\text{IL}} = V_{\text{dd}}$.

Noise Margin:
which are equal to each other.

Region 5: \( V_{dd} - V_{tp} \leq V_{in} \leq V_{dd} \) In this region, the pull-up pMOS transistor remains OFF and the pull-down nMOS transistor goes to deep saturation. However, the current flow through the circuit is zero as the p transistor is OFF and the output voltage \( V_O = 0 \).

Based on the above discussions, key features of the CMOS inverter are highlighted below:

It may be noted that unlike the use of nMOS enhancement- or depletion-mode transistor as a pull-up device, in this case, there is no current flow either for ‘0’ or ‘1’ inputs. So, there is no static power dissipation. Current flows only during the transition period. So, the static power dissipation is very small. Moreover, for low and high inputs, the role of the pMOS and nMOS transistors are complementary; when one is OFF, the other one is ON. That is why this configuration is known as the complementary MOS or CMOS inverter. Another advantage is that full high and low levels are generated at the output. Moreover, the output voltage is independent of the relative dimensions of the pMOS and nMOS transistors. In other words, the CMOS circuits are ratioless.

\( \beta_n / \beta_p \) Ratio: As we have mentioned earlier, the low- and high-level outputs of a CMOS inverter are not dependent on the inverter ratio. However, the transfer characteristic is a function of the \( \beta_n / \beta_p \) ratio. The transfer characteristics for three different ratio values are plotted in Fig. 4.11. Here, we note that the voltage at which the gate switches from high to low level \( (V_{inv}) \) is dependent on the \( \beta_n / \beta_p \) ratio. \( V_{inv} \) increases as \( \beta_n / \beta_p \) decreases. For a given process technology, the \( \beta_n / \beta_p \) can be changed by changing the channel dimensions, i.e., the channel length and width. Keeping \( L \) the same, if we increase \( W_n / W_p \) ratio, the transition moves towards the left and as \( W_n / W_p \) is decreased, the transition moves towards the right as shown in Fig. 4.11. As the carrier mobility depends on temperature, it is expected that the transfer characteristics will be affected with the temperature. However, both \( \beta_n \) and \( \beta_p \) are affected in the same manner \( (\beta \propto T^{-1.5}) \) and the ratio \( \beta_n / \beta_p \) remains more or less the same. On the other hand, both the threshold voltages \( V_{in} \) and \( V_{tp} \)
Table 4.1 Comparison of the inverters

<table>
<thead>
<tr>
<th>Inverters</th>
<th>( V_{LO} )</th>
<th>( V_{HI} )</th>
<th>Noise margin</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resistor</td>
<td>Weak</td>
<td>Strong</td>
<td>Poor for low</td>
<td>High</td>
</tr>
<tr>
<td>nMOS depletion</td>
<td>Weak</td>
<td>Strong</td>
<td>Poor for low</td>
<td>High</td>
</tr>
<tr>
<td>nMOS enhancement</td>
<td>Weak</td>
<td>Weak</td>
<td>Poor for both low</td>
<td>High</td>
</tr>
<tr>
<td>Pseudo-nMOS</td>
<td>Weak</td>
<td>Strong</td>
<td>Poor for low</td>
<td>High</td>
</tr>
<tr>
<td>CMOS</td>
<td>Strong</td>
<td>Strong</td>
<td>Good</td>
<td>Low</td>
</tr>
</tbody>
</table>

\( nMOS \) n-type metal–oxide–semiconductor, \( CMOS \) complementary metal–oxide–semiconductor

decrease with increase in temperature leading to some shrinkage of the region I and expansion of region V.

4.3.6 Comparison of the Inverters

Table 4.1 summarizes the performance of the five different types of inverters discussed in this section. As given in column 2, low output level is weak, i.e., the output does not go down to 0 V, for all the inverters except CMOS. High-output level is weak only for the inverter with the enhancement-mode nMOS transistor as pull-up devices. For all other inverters, full high-level (\( V_{dd} \)) is attained as shown in column 3. As shown in column 4, CMOS gives a good noise margin both for high- and low-logic levels. The inverters with nMOS enhancement-mode transistor as pull up have poor noise margin both for high- and low-logic levels. For all other types of inverters, noise margin is poor for low levels. So far, as power consumption is concerned, all inverters except CMOS draw DC current from the supply when the input is high. As a consequence, only CMOS consumes lower power compared to other types of inverters. From this table, we can conclude that CMOS is superior in all respects compared to other types of logic circuits because of these advantages. As a consequence, CMOS has emerged as the dominant technology for the present-day VLSI circuits. In case of nMOS depletion mode of transistor as a pull-up device, low-level noise margin is poor, and in case of nMOS enhancement-mode transistor, the noise margin is poor for both low and high levels. It may be noted that CMOS circuits provide better noise margin compared to other two cases as given in Table 4.1.

4.4 Inverter Ratio in Different Situations

In a practical circuit, an inverter will drive other circuits, and, in turn, will be driven by other circuits. Different inverter ratios will be necessary for correct and satisfactory operation of the inverters. In this section, we consider two situations—an inverter driven by another inverter and an inverter driven through one or more pass transistors—and find out the inverter ratio for these two cases.
**An nMOS Inverter Driven by Another Inverter**

Let us consider an nMOS inverter with depletion-type transistor as an active load is driving a similar type of inverter as shown in Fig. 4.12. In order to cascade two or more inverters without any degradation of voltage levels, we have to meet the condition $V_{in} = V_{out} = V_{inv}$; and for equal margins, let us set $V_{inv} = 0.5V_{dd}$. This condition is satisfied when both the transistors are in saturation, and the drain current is given by

$$I_{ds} = \frac{K}{L} (V_{gs} - V)^2$$

(4.23)

For the depletion-mode transistor,

$$I_{ds} = K \frac{W_p}{L_{pu}} \left(\frac{-V_{tp}}{2}\right)^2$$

(4.24)

where $V_{tp}$ is the threshold voltage of the depletion-mode transistor and $V_{gs} = 0$.

And for the enhancement-mode transistor,

$$I_{ds} = K \frac{W_{pd}^{inv}}{2L_{pd}} (V_{n} - V)^2$$

(4.25)

Equating these currents, we get

$$\frac{W_{pd}^{inv}}{L_{pd}} (V_{n} - V)^2 = \frac{W_{pu}^{inv}}{L_{pu}} (-V)^2$$

(4.26)

Assuming $Z_{pd} = L_{pd}/W_{pd}$ and $Z_{pu} = L_{pu}/W_{pu}$, where $Z$ is known as the **aspect ratio** of the MOS devices, we have
Fig. 4.13 An inverter driven through one or more pass transistors

\[ \frac{1}{Z_{pd}} (V_{inv} - V_{tn})^2 = \frac{1}{Z_{pu}} (-V_{tdp})^2 \]

\[ \frac{Z_{pu}}{Z_{pa}} (V_{inv} - V_{tn}) = -V_{tdp} \]

or \[ V_{inv} = V_{tn} - \frac{V_{tdp}}{\sqrt{Z_{pu}/Z_{pd}}} \]

Substituting typical values \( V_{m} = 0.2V_{dd} \), \( V_{tp} = -0.6V_{dd} \) and \( V_{inv} = 0.5V_{dd} \), we get

\[ \frac{Z_{pu}}{Z_{pd}} = 4 \]

This ratio \( Z_{pu}/Z_{pd} \) is known as the inverter ratio \( (R_{inv}) \) of the inverter, and it is 4:1 for an inverter directly driven by another inverter.

**An nMOS Inverter Driven Through Pass Transistors**

Here, an nMOS inverter is driven through one or more pass transistors as shown in Fig. 4.13. As we have seen in Sect. 4.1, a pass transistor passes a weak high level. If \( V_{dd} \) is applied to the input of a pass transistor, at the output, we get \( (V_{dd} - V_{tp}) \), where \( V_{tp} \) is the threshold voltage of the pass transistor. Therefore, instead of \( V_{dd} \), a degraded high level \( (V_{dd} - V_{tp}) \) is applied to the second inverter. We have to ensure that the same voltage levels are produced at the outputs of the two inverters in spite of different input voltage levels.

First, let us consider inverter 1 with input voltage \( V_{dd} \). In this situation, the pull-down transistor is in active mode and behaves like a resistor, whereas the pull-up transistor is in saturation, and it represents a current source.

For the pull-down transistor

\[ I_{ds} = K \frac{W_{pd1}}{L_{pd1}} [(V_{dd} - V_{m}) V_{ds1} - \frac{kT}{2}] \]
Therefore, \[ R_1 = \frac{V_{sd1}}{I_{K W}} = \frac{1}{K W} \left( \frac{V}{V_{dd} - V_{tn}} \right) \]

\[ \text{(4.30)} \]

Ignoring \( V_{sd1} \) factor, \[ R_1 = \frac{Z_{pd1}}{V_{dd} - V_{tn}} \]

\[ \text{(4.31)} \]

Now, for the depletion-mode pull-up transistor, \( V_{gs} = 0 \),

\[ I_1 = I_{ds} = K \frac{W}{L_{pu1}} \left( -V_{tp} \right)^2 \]

\[ \text{(4.32)} \]

The product \[ I_1 R = \frac{Z}{V_{out1}} \left( V - V_{dd} - V_{tn} \right) \]

\[ \text{(4.33)} \]

In a similar manner, we can consider the case of inverter 2, where the input voltage is \( V_{dd} - V_{tp} \). As in case of inverter 1, we get

\[ R_2 \approx \frac{Z_{pd2}}{K} \left( V_{dd} - V_{tp} - V_{tn} \right) \]

\[ \text{and } I_2 = K \frac{Z_{pu2}}{Z_{pu2}^2} \left( -V_{td} \right)^2 \]

\[ \text{(4.34)} \]

Therefore, \[ V_{out2} = I_2 R = \frac{Z_{pu2}}{Z_{pu2}^2} \left( V_{dd} - V_{tp} - V_{tn} \right) \]

\[ \left( -V_{td} \right)^2 \]

\[ \text{(4.35)} \]

If we impose the condition that both the inverters will have the same output,

\[ V_{out1} = V_{out2} \text{ then, } I_1 R = I_2 R \]

\[ \text{or } \frac{Z_{pu2}}{Z_{pu2}^2} = \frac{Z_{pu1}}{Z_{pu1}^2} \]

\[ \frac{Z_{pd1}}{Z_{pd1}^2} (V_{dd} - V_{tn}) \]

\[ \text{(4.36)} \]

Substituting typical values \( V_{tn} = 0.2 V_{dd} \) and \( V_{tp} = 0.3 V_{dd} \),

we get

\[ \frac{Z_{pu2}}{Z_{pu2}^2} = 4.0 \frac{Z_{pu1}}{Z_{pu1}^2} \approx \frac{Z_{pu1}}{Z_{pu1}^2} \]

\[ \frac{Z_{pd2}}{Z_{pd2}^2} \approx 2, \frac{Z_{pu1}}{Z_{pu1}^2} = 8 \]

\[ \text{(4.37)} \]
It may be noted that, if there are more than one-pass transistors before the second inverter, the degradation in the high voltage level is the same. Therefore, we may conclude that, if an inverter is driven through one or more pass transistors, it should have inverter ratio $Z_{pu}/Z_{pd} \geq 8/1$.

**Switching Characteristics**

So far, we have discussed the DC characteristics of an inverter, which gives us the behaviour of the inverter in static conditions when inputs do not change. To understand the switching characteristics, it is necessary to understand the parameters that affect the speed of the inverters. In this section, we consider the dynamic behaviour of a CMOS inverter. A CMOS inverter driving a succeeding stage is shown in Fig. 4.14a. As shown in the figure, various parasitic capacitances exist (Fig. 4.15).

The capacitances $C_{gd}$ and $C_{gs}$ are mainly due to gate overlap with the diffusion regions, whereas $C_{db}$ and $C_{gb}$ are voltage-dependent junction capacitances. The capacitance $C_{out}$ is the lumped value of the distributed capacitances due to interconnection and $C_{gn}$ and $C_{gp}$ are due to the thin oxide capacitances over the gate area of the nMOS and pMOS transistors, respectively. For the sake of simplicity, all the capacitances are combined into an equivalent lumped capacitance $C_{L}$, which is connected as the load capacitance of the inverter as shown in Fig. 4.14b. Here,
$$C_L = C_{dgn} + C_{dgp} + C_{dhn} + C_{dhp} + C_{int} + C_{gn} + C_{gp}.$$  

We assume that an ideal step waveform, with zero rise and fall time is applied to the input as shown in Fig. 4.14b. The delay $t_d$ is the time difference between the midpoint of the input swing and the midpoint of the swing of the output signal. The load capacitance shown at the output of the inverter represents the total of the input capacitance of driven gates, the parasitic capacitance at the output of the gate itself, and the wiring capacitance. In the following section, we discuss the estimation of the load capacitances.

### 4.5.1 Delay-Time Estimation

The circuit for high-to-low propagation delay time $t_{pHL}$ estimation can be modeled as shown in Fig. 4.16c. It is assumed that the pull-down nMOS transistor remains in saturation region and the pMOS transistor remains off during the entire discharge period. The saturation current for the nMOS transistor is given by

$$I_{dsn} = \frac{\beta_n}{2} (V_{gs} - V)^2$$  

(4.38)
The capacitor $C_L$ discharges from voltage $V_{dd}$ to $V_{dd}/2$ during time $t_0$ to $t_1$.

$$V_{dd} = \frac{V_{dd}}{2} = V_{dd} - \beta_n \cdot \left( \frac{V - V}{m} \right)^2 \times (t - t_0).$$  \hspace{1cm} (4.39)

Substituting $V_{gs} = V_{dd}$ and $t_{ph} = t_1 - t_0$,
we get the fall delay

$$t_{phi} = \frac{V_{dd} \times C_L}{\beta_n \frac{V_{dd} - V}{m}}.$$  \hspace{1cm} (4.40)

When the input goes from high ($V_{dd}$) to low, initially the output is at a low level. The pull-up pMOS transistor operates in the saturation region. In a similar manner, based on the model of Fig. 4.16d, the rise time delay is given by

$$t_{plh} = \frac{C_L}{V_{dd} \cdot \left( \frac{1}{1 - \frac{V_{tp}}{V_{dd}}} \right)}.$$  \hspace{1cm} (4.41)

For equally sized MOS transistors, the rise delay is greater than the fall delay because of lower mobility of p-type carriers. So, the pMOS transistor should be sized by increasing the width $W_p$ in order to get equal size and fall delays.

**Delay Time** By taking average of the rise and full delay time, we get the delay time

$$t = \frac{1}{2} \left( t_{phi} + t_{phh} \right)$$

$$= \frac{C_L}{2V_{dd} \beta_n \left( \frac{1}{1 - \frac{V_{tn}}{V_{dd}}} \right)^2} \left[ \frac{1}{\beta_n} \left( \frac{V_{mn}}{V_{dd}} \right)^2 + \frac{1}{\beta_p} \left( \frac{V_{tp}}{V_{dd}} \right)^2 \right].$$  \hspace{1cm} (4.42)

Assuming $V_{in} = -V_{tp} = V_t$, the delay is given by

$$t = \left[ \frac{L_n}{K \frac{W}{n}} + \frac{L_p}{K \frac{W}{p}} \right] \frac{C_L}{V_{dd} \beta_n \left( \frac{1}{1 - \frac{V_t}{V_{dd}}} \right)^2}.$$  \hspace{1cm} (4.43)
This expression gives us a simple analytical expression for the delay time. It is observed that the delay is linearly proportional to the total load capacitance $C_L$. The delay also increases as the supply voltage is scaled down, and it increases drastically as it approaches the threshold voltage. To overcome this problem, the threshold voltage is also scaled down along with the supply voltage. This is known as the constant field scaling. Another alternative is to use constant voltage scaling, in which the supply voltage is kept unchanged because it may not be always possible to reduce the supply voltage to maintain electrical compatibility with other subsystems used to realize a complete system. The designer can control the following parameters to optimize the speed of CMOS gates.

- The width of the MOS transistors can be increased to reduce the delay. This is known as gate sizing, which will be discussed later in more detail.
- The load capacitance can be reduced to reduce delay. This is achieved by using transistors of smaller and smaller dimensions as provided by future-generation devices.
- Delay can also be reduced by increasing the supply voltage $V_{dd}$ along and/or reducing the threshold voltage $V_t$ of the transistors.

### 4.5.2 Ring Oscillator

To characterize a particular technology generation, it is necessary to measure the gate delay as a measure of the performance [1]. It is difficult to measure delays of the order of few nanoseconds with the help of an oscilloscope. An indirect approach is to use a ring oscillator to measure the delay. A ring oscillator is realized by a cascade connection of odd number of a large number of inverters, where the output of the last stage is connected to the input of the first stage, as shown in Fig. 4.17. The circuit oscillates because the phase of the signal fed to the input stage from the output stage leads to positive feedback. The frequency of oscillation for this closed-loop cascade connection is determined by the number of stages and the delay of each stage. The output waveform of a three-stage ring oscillator is shown in Fig. 4.18. The time period can be expressed as the sum of the six delay times

$$T = t_{ph11} + t_{ph2} + t_{ph13} + t_{ph1} + t_{ph2} + t_{ph3}$$

$$= (t_{ph11} + t_{ph1}) + (t_{ph2} + t_{ph2}) + t_{ph3} + t_{ph3}$$

$$= 2t_d + 2t_d + 2t_d$$

$$= 6t_d$$

$$= 2.3t_d.$$
For an $n$-stage (where $n$ is an odd number) inverter, the time period $T = 2n \tau_d$. Therefore, the frequency of oscillation $f = 1 / 2nt_d$ or $t_d = 1 / 2nf$. It may be noted that the delay time can be obtained by measuring the frequency of the ring oscillator. For better accuracy of the measurement of frequency, a suitable value of $n$ (say 151) can be chosen. This can be used for the characterization of a fabrication process or a particular design. The ring oscillator can also be used for on-chip clock generation. However, it does not provide a stable or accurate clock frequency due to dependence on temperature and other parameters. To generate stable and accurate clock frequency, an off-chip crystal is used to realize a crystal oscillator.

## 4.6 Delay Parameters

In order to understand the delay characteristics of MOS transistors, we have to consider various parameters such as resistance and capacitances of the transistors along with wiring and parasitic capacitances. For ease of understanding and simplified treatment, we will introduce simplified circuit parameters as follows:

---

**Fig. 4.18** Output waveform of a three-stage ring oscillator
**Resistance Estimation**

Let us consider a rectangular slab of conducting material of resistivity $\rho$, of width $W$, of thickness $t$ and length $L$ as shown in Fig. 4.19. The resistance between A and B, $R_{AB}$ between the two opposite sides is given by

$$ R_{AB} = \frac{\rho L}{W} = \frac{\rho L}{t} \Omega, \quad (4.44) $$

where $A$ is the cross section area. Consider the case in which $L = W$, then

$$ R_{AB} = \frac{\rho}{t} = R \Omega \quad (4.45) $$

where $R_s$ is defined as the resistance per square or the sheet resistance.

Thus, it may be noted that $R_s$ is independent of the area of the square. The actual value of $R_s$ will be different for different types of layers, their thickness, and resistivity. For poly-silicon and metal, it is very easy to envisage the thickness, and their resistivities are also known. But, for diffusion layer, it is difficult to envisage the depth of diffusion and impurity concentration level. Although the voltage–current characteristics of an MOS transistor is nonlinear in nature, we may, approximate its behaviour in terms of a ‘channel’ resistance to estimate delay performance. We know that in the linear region

$$ I_{ds} = \beta \left[ \left( V_{gs} - V_t \right) V_{ds} \frac{V_{ds}^2}{2} \right] \quad (4.46) $$

Assuming, $V_{ds} \ll \left( V_{gs} - V_t \right)$, we may ignore the quadratic term to get

$$ I_{ds} = \beta \left( V_{gs} - V_t \right) V_{ds} \quad \text{or} \quad R_c = \frac{1}{I_{ds}} \left( \frac{1}{\beta \left( V_{gs} - V_t \right)} \right) \frac{L}{W} \mu C_g \left( V - V_g \right) W $$

$$ = K \left( \frac{L}{W} \right) \text{ where } K = \frac{1}{\mu C_g \left( V_{gs} - V_t \right)} \quad (4.47) $$
Table 4.2 Sheet resistances of different conductors

<table>
<thead>
<tr>
<th>Layer</th>
<th>Min.</th>
<th>Typical</th>
<th>Max.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metal</td>
<td>0.03</td>
<td>0.07</td>
<td>0.1</td>
</tr>
<tr>
<td>Diffusion (n, p')</td>
<td>10</td>
<td>25</td>
<td>100</td>
</tr>
<tr>
<td>Silicide</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Poly-silicon</td>
<td>15</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>n-channel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p-channel</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sheet resistance in ohm/sq.

$K$ may take a value between 1000 to 3000 $\Omega$/sq. Since the mobility and the threshold voltage are functions of temperature, the channel resistance will vary with temperature. Typical sheet resistances for different conductors are given in Table 4.2.

Sheet resistance concept can be conveniently applied to MOS transistors and inverters. A diffusion and a poly-silicon layer are laid orthogonal to each other with overlapping areas to represent a transistor. The thinox mask layout is the hatched overlapped region. The poly-silicon and the thinox layer prevent diffusion to occur below these layers. Diffusion takes place in the areas where these two layers do not overlap. Assuming a channel length of $2\lambda$ and channel width of $2\lambda$, the resistance of this channel is given by

$$R = 1 \text{ square} \times R_s \text{ per square where } R_s = 10^4 \Omega.$$  

In this case, the length to width ratio is 1:1.

For a transistor with $L = 8\lambda$ and $W = 2\lambda$,

$$Z = \frac{L}{W} = \frac{4}{1}.$$  

Thus, the channel resistance $= 4 \times R_s = 4 \times 10^4 \Omega$. It can be regarded as four-unit squares in series. It is also possible to consider an inverter with a given inverter ratio of $z_{pu}:z_{pd}$. The most common ratio of 4:1 can be realized in one of the two ways as shown in Fig. 4.20.

**Area Capacitance of Different Layers**

From the structure of MOS transistors, it is apparent that each transistor can be represented as a parallel-plate capacitor. The area capacitance can be calculated based on the dielectric thickness (silicon dioxide) between the conducting layers.

Here, area capacitance

$$C = \frac{\varepsilon_0 \varepsilon_{\text{ox}} A}{D} \text{ Farads},$$  \hspace{1cm} (4.48)
Table 4.3 Capacitance of different materials

<table>
<thead>
<tr>
<th>Capacitance</th>
<th>Value in pF/μm²</th>
<th>Relative value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gate to channel</td>
<td>4 x 10⁻⁴</td>
<td>1</td>
</tr>
<tr>
<td>Diffusion</td>
<td>1 x 10⁻⁴</td>
<td>0.25</td>
</tr>
<tr>
<td>Poly-silicon</td>
<td>4 x 10⁻⁴</td>
<td>0.1</td>
</tr>
<tr>
<td>Metal 1</td>
<td>0.3 x 10⁻⁴</td>
<td>0.075</td>
</tr>
<tr>
<td>Metal 2</td>
<td>0.2 x 10⁻⁴</td>
<td>0.50</td>
</tr>
<tr>
<td>Metal 2 to metal</td>
<td>0.4 x 10⁻⁴</td>
<td>0.15</td>
</tr>
<tr>
<td>Metal 2 to poly</td>
<td>0.3 x 10⁻⁴</td>
<td>0.075</td>
</tr>
</tbody>
</table>

Fig. 4.20 Two different inverter configurations with inverter ratio 4:1

where $D$ is the thickness of the silicon dioxide, $A$ is the Area of place, $\varepsilon_{\text{ins}}$ is the relative permittivity of $\text{SO}_2 = 3.9$, and $\varepsilon_0 = 8.85 \times 10^{-14}$ F/cm, permittivity of free space.

For 5-μm MOS technology, typical area capacitance is given in Table 4.3:

**Standard Unit of Capacitance $C_g$**

The standard unit of capacitance $C_g$ is defined as the gate-to-channel capacitance of a minimum-size MOS transistor. It gives a value approximate to the technology and can be conveniently used in calculations without associating with the absolute value.

Considering 5-μm technology, where gate area = 5 μm x 5 μm = 25 μm², area capacitance = $4 \times 10^{-4}$ pF/cm².

Therefore, standard value of $C_g = 25 \times 4 \times 10^{-4}$ pF = 0.01 pF.

**Example 2.2**

Let us consider a rectangular sheet of length $L$ and width $W$. Assuming $L = 20$ and $W = 3$, the relative area and capacitances can be calculated as follows:

Relative area =$\frac{20 \times 3}{2 \times 3} = 10$

a. Consider the area in metal capacitance to substrate = $15 \times 0.075 \ C_g = 1.125 \ C_g$
b. Consider the same are in poly
   capacitance to substrate = 15 × 0.20 = 1.5 \( C_g \)

c. Considering it in diffusion
   capacitance to substrate = 15 × 0.25 = 3.70 \( C_g \)

A structure occupying different layers can be calculated in the same manner.

**The Delay Unit**

Let us consider the situation of one standard gate capacitance being charged through one square of channel resistance.

Time constant = \( 1 R_s \times 1 \ C_g \ s \)

For 5-\( \mu \)m technology

\( \tau = 10^4 \Omega \times 0.01 \text{ pF} = 0.1 \text{ ns} \)

when circuit wiring and parasitic capacitances are taken into account, this figure increases by a factor of 2–3.

So, in practice \( \tau \approx 0.2–0.3 \text{ ns} \).

The value of \( \tau \) obtained from transit time calculations is

\[
I_{sd} = \frac{L^2}{\mu \cdot V_n \cdot ds}
\]

Substituting the values corresponding to 5 \( \mu \)m technology

\[
\tau_{sd} = \frac{25^2 \mu m^2 \text{Vs}}{650 \text{cm}^2 \text{3V} \times 10^9 \text{ns}10^4} = 0.13 \text{ns}.
\]

(Assuming \( C_g \) is charged from 0 V to 63 % of \( V_{dd} \))

It may be noted that this is very close to the theoretical estimate, and it is used as the fundamental time unit and all timings can be evaluated in terms of \( \tau \).

**Driving Large Capacitive Loads**

There are situations when a large load capacitance such as, long buffers, off-chip capacitive load or I/O buffer are to be driven by a gate [1]. In such cases, the delay can be very high if driven by a standard gate. A super buffer or a BiCMOS inverter is used or cascade of such gates of increasing can be used to achieve smaller delays as discussed in the following subsections.
4.7.1 Super Buffers

We have seen that one important drawback of the basic nMOS inverters (because of ratioed logic) in driving capacitive load is asymmetric drive capability of pull-up and pull-down devices. This is because of longer channel length (four times) of the pull-up device. Moreover, when the pull-down transistor is ON, the pull-up transistor also remains ON. As a consequence, a complete pull-down current is not used to discharge the load capacitance, but part of it is neutralized by the current passing through the pull-up transistors. This asymmetry can be overcome in especially designed circuits known as super buffers. It is possible to realize both inverting and noninverting super buffers as shown in Fig. 4.21. In standard nMOS inverter, the gate of the depletion-mode pull-up device is tied to the source. Instead, the gate of the pull-up device of the super buffer is driven by another inverter with about twice the gate drive. Thus, the pull-up device is capable of sourcing about four times the current of the standard nMOS inverter. This is the key idea behind both the inverting and noninverting types of super buffers. This not only overcomes the asymmetry but also enhances the drive capability.

A schematic diagram of nMOS super buffers, both inverting and noninverting types, are shown in Fig. 4.21. As shown, the output stage is a push–pull stage. The gate of the pull-up device is driven by a signal of opposite level of the pull-down device, generated using a standard inverter. For the inverting-type super buffer, when the input voltage is low, the gates of both the pull-down transistors are low, the gates of both the pull-up devices are high, and the output is high. When the input voltage is high, the gates of both pull-down devices switch to high, the gates of both pull-up devices switch to low, and the output switches from high to low. The small-channel resistance of the pull-down device discharges the output load capacitor quickly. When the input voltage goes from high to low, the gate of the pull-up device quickly switches to high level. This drives the pull-down stage very quickly, and the output switches quickly from low to high. It can be shown that the high-to-low and low-to-high switching times are comparable.
In order to compare the switching speed of a standard inverter and a super buffer, let us compare the current sourcing capability in both the situations.

For a standard inverter, the drain current of the pull-up device in saturation ($0 < V_0 < 2$ V) and linear region are given as follows:

$$I_{ds}^{\text{sat}} = \frac{\beta_{pu}}{2} (V_{gs} - V_{tdep})^2, \quad \text{where} \quad V_{tdep} = -3V$$

$$\quad = \frac{\beta_{pu}}{2} (0 + 3)^2 = 4.5 \beta_{pu}. \quad (4.49)$$

$$I_{ds}^{\text{lin}} = \beta_{pu} \left[ (V_{gs} - V_{tdep}) V_{ds} - \frac{V_{ds}^2}{2} \right]$$

$$\quad = \frac{\beta_{pu}}{2} \left[ 2(0 + 3)2.5 - (2.5)^2 \right] = 4.38 \beta_{pu}. \quad (4.50)$$

The average of the saturation current for $V_{ds} = 5$ V and linear current for $V_{ds} = 2.5$ V is approximately $4.4 \beta_{pu}$.

For the super buffer, the pull-up device is always in linear region because $V_{gs} = V_{ds}$. The average pull-up current can be assumed to be an average of the linear currents at $V_{ds} = 5$ V and $V_{ds} = 2.5$ V.

$$I_{ds}^{(5V)} = \frac{\beta_{pu}}{2} \left[ 2(5 + 3)5 - 5^2 \right]$$

$$\quad = 27.5 \beta_{pu},$$

$$I_{ds}^{(2.5V)} = \frac{\beta_{pu}}{2} \left[ 2(2.5 + 3)2.8 - (2.5)^2 \right]$$

$$\quad = 10.62 \beta_{pu}.$$

The average source current

$$\quad = \frac{(27.5 + 10.62)\beta_{pu}}{2} = 19.06 \beta_{pu}$$

Therefore, the current drive is $= 19.06 / 4.44 = 4.3$ times that of standard inverter for the super buffer using totem pole output configuration. This high drive also alleviates the asymmetry in driving capacitive loads and makes the nMOS inverter behave like a ratioless circuit.
**BiCMOS Inverters**

Higher current drive capability of bipolar NPN transistors is used in realizing BiCMOS inverters. Out of the several possible configurations, the conventional BiCMOS inverter, shown in Fig. 4.22, is considered here. The operation of the inverter is quite straightforward. The circuit requires four MOS transistors and two bipolar NPN transistors $Q_1$ and $Q_2$. When the input $V_{in}$ is low, the pMOS transistor $P_1$ is ON, which drives the base of the bipolar transistor $Q_1$ to make it ON. The nMOS transistor $N_1$ and $N_2$ are OFF, but transistor $N_3$ is ON, which shunts the base of $Q_2$ to turn it OFF. The current through $Q_1$ charges capacitor $C_L$ and at the output $V_{out}$, we get a voltage ($V_{dd} - V_{be}$), where $V_{be}$ is the base–emitter voltage drop of the transistor $Q_1$. If the input is high ($V_{dd}$), transistors $P_1$ and $N_3$ are OFF and $N_1$ and $N_2$ are ON. The drain current of $N_2$ drives the base of the transistor $Q_2$, which turns ON and transistor $N_1$ shunts the base of $Q_1$ to turn it OFF. The capacitor $C_L$ discharges through $Q_2$. The conventional BiCMOS gate gives high-current-drive capability, zero static power dissipation and high input impedance. The DC characteristic of the conventional BiCMOS inverter is shown in Fig. 4.22a. For a zero input voltage, the pMOS transistor operates in the linear region. This drives the NPN transistor $Q_1$, and the output we get is $V_{dd} - V_{be}$, where $V_{be}$ is the base–emitter voltage drop of $Q_1$. As input voltage increases, the subthreshold leakage current of the transistor $N_3$ increases leading to a drop in the output voltage. For $V_{in} = V_{inv} (V_{dd}/2)$, both $P_1$ and $N_2$ transistors operate in the saturation region driving both the NPN transistors ($Q_1$ and $Q_2$) ON. In this region, the gain of the inverter is very high, which leads to a sharp fall in the output voltage, as shown in Fig. 4.22b. As the input voltage is further increased, the output voltage drops to zero. The output voltage characteristics of CMOS and BiCMOS are compared in Fig. 4.22b. It may be noted that the BiCMOS inverter does not provide strong high or strong low outputs. High output is $V_{dd} - V_{CE1}$, where $V_{CE1}$ is the saturation voltage across $Q1$ and the low-level output is $V_{CE2}$, which is the saturation voltage across $Q2$.  

---

**Fig. 4.22** a A conventional BiCMOS inverter; b output characteristics of static CMOS and BiCMOS. CMOS complementary metal–oxide–superconductor.
The delays of CMOS and BiCMOS inverters are compared in Fig. 4.23 for different fan-outs. Here, it is assumed that for CMOS, $W_p = 15 \, \mu m$ and $W_n = 7 \, \mu m$ and for the BiCMOS, $W_p = W_n = 10 \, \mu m$ and $W_{Q1} = W_{Q2} = 2 \, \mu m$. It may be noted that for fan-out of 1 or 2, CMOS provides smaller delay compared to BiCMOS due to longer delay through its two stages. However, as fan-out increases further, BiCMOS performs better. So, BiCMOS inverters should be used only for larger fan-out.

**Buffer Sizing**

It may be observed that an MOS transistor of unit length $(2\lambda)$ has gate capacitance proportional to its width $(W)$, which may be multiple of $\lambda$. With the increase of the width, the current driving capability is increased. But this, in turn, also increases the gate capacitance. As a consequence, the delay in driving a load capacitance $C_L$ by a transistor of gate capacitance $C_g$ is given by the relationship $(C_L / C_g)\tau$, where $\tau$ is the unit delay, or delay in driving an inverter by another of the same size.

Let us now consider a situation in which a large capacitive load, such as an output pad, is to be driven by an MOS gate. The typical value of such load capacitance is about $100 \, \text{pF}$, which is several orders of magnitude higher than $C_g$. If such a load is driven by an MOS gate of minimum dimension $(2\lambda \times 2\lambda)$, then the delay will be $10^3\tau$. To reduce this delay, if the driver transistor is made wider, say $10^3 \times 2\lambda$, the delay of this stage becomes $\tau$, but the delay in driving this driver stage is $1000\tau$, so, the total delay is $1001\tau$, which is more than the previous case. It has been observed that the overall delay can be minimized by using a cascaded stage of inverters of increasing size as shown in Fig. 4.24. For example, considering each succeeding stage is ten times bigger than that of the preceding stage, each stage gives a delay of $10\tau$. This results in a total delay of $31\tau$ instead of $1001\tau$. Now, the question arises about the relative dimension of two consecutive stages. If relative dimension is large, fewer stages are needed, and if relative dimension is small, a large number of driver stages is needed. Let us assume that there are $n$ stages and each buffer is scaled up by a factor of $f$, which gives stage delay of $f\tau$. For $n$ stages, the delay is $nfr$. Let $y$ be the ratio of the load capacitance to one gate capacitance. Then, for $n$ buffer stages in cascade
Using a single driver with $W$ to $L$ ratio of 1000:1; using drivers of increasing size with stage ratio of 10. $W$ width; $L$ length

$$y = \frac{c_L}{c_g} = \frac{c_{g1}}{c_{g2}} \ldots \frac{c_{gl}}{c_{gn}} = f^n$$

or $\ln y = n \ln f$

or $n = \frac{\ln y}{\ln f}$

As each stage gives a delay of $f\tau$, the total delay of $n$ stages is

$$nf \tau = f \tau \left[ \frac{\ln y}{\ln f} \right] = f \cdot \tau \cdot \ln y.$$  \hspace{1cm} (4.52)

The delay is proportional to $\ln(y)$, which is dependent on the capacitive load $C_L$ and for a given load $\ln(y)$ is constant. To find out the value of $f$ at which the delay is minimum, we can take the derivative of $f/\ln(f)$ with respect to $f$ and equate it to zero.

$$\frac{d}{df} \left[ \ln\left(\frac{f}{\ln f}\right) \right] = \ln f - f \cdot \ln\left(\frac{1}{f}\right) = 0.$$ \hspace{1cm} (4.53)

or $\ln(f) = 1$ or $f = e$, where $e$ is the base of natural logarithm.

The minimum total delay is

$$t_{\min} = nf \tau = e \tau \ln \left[ \frac{c_L}{c_g} \right]$$

\hspace{1cm} (4.54)

Therefore, the total delay is minimized when each stage is larger than the previous one by a factor of $e$. For 4:1 nMOS inverters, delay per pair is $5\tau$ and overall delay is

$$\frac{n}{2} e5\tau = \frac{5}{2} ne\tau = 2.5en\tau (n \text{ even}).$$
Fig. 4.25 Variation of delay with stage ratio
Table 4.4 Variation of delay with buffer sizing

<table>
<thead>
<tr>
<th>f</th>
<th>f/e ln(f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.062</td>
</tr>
<tr>
<td>e</td>
<td>1.000</td>
</tr>
<tr>
<td>3</td>
<td>1.005</td>
</tr>
<tr>
<td>4</td>
<td>1.062</td>
</tr>
<tr>
<td>5</td>
<td>1.143</td>
</tr>
<tr>
<td>6</td>
<td>1.232</td>
</tr>
</tbody>
</table>

For CMOS, the overall delay is $= 3.5enr (n \text{ even})$

It may be noted that the minimum total delay is a slowly varying function of $f$ as shown in Fig. 4.25. Therefore, in practice, it may be worthwhile to scale the buffers by 4 or 5 to 1. This allows saving of the real estate at the cost small speed penalty, say about 10% that the minimum possible delay (Table 4.4).

**MOS Combinational Circuits**

**Introduction**

There are two basic approaches of realizing digital circuits by metal–oxide–semiconductor (MOS) technology: switch logic and gate logic. A switch logic is based on the use of “pass transistors” or transmission gates, just like relay contacts, to steer logic signals through the device. On the other hand, gate logic is based on the realization of digital circuits using inverters and other conventional gates, as it is typically done in transistor–transistor logic (TTL) circuits. Moreover, depending on how circuits function, they can also be categorized into two types: static and dynamic gates. In case of static gates, no clock is necessary for their operation and the output remains steady for as long as the supply voltage is maintained. Dynamic circuits are realized by making use of the information storage capability of the intrinsic capacitors present in the MOS circuits.
A basic operation of pass-transistor logic circuits is considered in Sect. 5.2. Realization of pass-transistor logic circuits, overcoming the problems highlighted in Sect. 5.2.1, has been presented in Sect. 5.2.2. The advantages and disadvantages of pass-transistor logic circuits are considered in Sect. 5.2. Pass-transistor logic families are introduced in Sect. 5.2.3. Logic circuits based on gate logic are considered in Sect. 5.3. The operation of n-type MOS (nMOS), NAND, and NOR gates has been presented in Sect. 5.3.2. Realization of complementary metal–oxide–semiconductor (CMOS) NAND and NOR gates has been discussed in Sect. 5.3.3. The switching characteristic of CMOS gates is considered in Sect. 5.3.4. Section 5.4 introduces MOS dynamic logic circuits. Section 5.5 presents the realization of some example circuits.

Pass-Transistor Logic

As pass transistors functionally behave like switches, logic functions can be realized using pass transistors in a manner similar to relay contact networks. Relay contact networks have been presented in the classical text of Caldwell (1958). However, a closer look will reveal that there are some basic differences between relay circuits and pass-transistor networks. Some of the important differences are discussed in this section [1]. In our discussion, we assume that the pass transistors are nMOS devices.

As we have seen in Sect. 3.6, a high-level signal gets degraded as it is steered through an nMOS pass transistor. For example, if both drain and gate are at some high voltage level, the source will rise to the lower of the two potentials: $V_{dd}$ and $(V_{gs} - V_t)$. If both drain and gate are at $V_{dd}$, the source voltage cannot exceed $(V_{dd} - V_t)$. We have already seen that when a high-level signal is steered through a pass transistor and applied to an inverter gate, the inverter has to be designed with a high inverter ratio (8:1) such that the gate drive (3.5 V assuming the threshold voltage equal to 1.5 V) is sufficient enough to drive the inverter output to an acceptable low level. This effect becomes more prominent when a pass-transistor output is allowed to drive the gate of another pass-transistor stage as shown in Fig. 5.1. As the gate voltage is 3.5 V, the high-level output from the second pass transistor can never exceed 2.0 V, even when the drain voltage is 5 V as shown in Fig. 5.1. In such a situation, the gate potential will not be sufficient enough to drive the output low of an inverter with an aspect ratio of 8:1. Therefore, while synthesizing nMOS pass-transistor logic, one must not drive the gate of a pass transistor by the output of another pass transistor. This problem does not exist either in relay logic or in case of transmission gate.
In the synthesis of relay logic, the network is realized such that high-level signal reaches the output for which the function is “1.” Absence of the high voltage level is considered to be “0.” In pass-transistor logic, however, this does not hold good. For example, to realize the function \( f = a + b'c \) the relay logic network is shown in Fig. 5.2a. The pass-transistor realization based on the same approach is shown in Fig. 5.2b. It can be easily proved that the circuit shown in Fig. 5.2b does not realize the function \( f = a + b'c \). Let us consider an input combination (say 100) for which the function is “1,” the output will rise to high level by charging the output load capacitance. Now, if we apply an input combination 010, for which the function is “0,” then the output remains at “1” logic level because the output capacitance does not get a discharge path when input combination of 010 is applied. A correct nMOS pass-transistor realization of \( f = a + b'c \) is shown in Fig. 5.2c. The difference between the circuit of Fig. 5.2c and that of Fig. 5.2b is that it provides a discharge path of the output capacitor when the input combination corresponds to a low-load output. Therefore, in synthesizing pass-transistor logic, it is essential to provide both charging and discharging path for the load capacitance. In other words, charging path has to be provided for all input combinations requiring a high output level and discharge path has to be provided for all input combinations requiring a low output level.

Finally, care should be taken such that advertently or inadvertently an output point is not driven by signals of opposite polarity. In such a situation, each transistor acts as a resistor and a voltage about half of the high level is generated at the output. Presence of this type of path is known as sneak path. In such a situation, an undefined voltage level is generated at a particular node. This should be avoided.

### 5.2.1 Realizing Pass-Transistor Logic

Pass transistors can be used to realize multiplexers of different sizes. For example, 2-to-1 and 4-to-1 multiplexer realization using pass transistor is shown in Fig. 5.3a, b, respectively. Now, any two-variable function can be expanded around the variables using Shannon’s expansion thereon. The expanded function can be represented by the following expression:

\[
f(a, b) = g_0a'b' + g_1a'b + g_2ab' + g_3ab.
\] (5.1)
Each \( g_i \) can assume a binary value, either 0 or 1, depending on the function. By applying the variables to control the gate potential and applying the logic value \( g_i \) to an input \( a_i \) the function can be realized. In this way, any function of two variables can be realized.

The realization of the function \( f = a'b + ab' \) is shown in Fig. 5.4a. When a function is realized in the above manner, none of the problems mentioned earlier arises. This approach may be termed as universal logic module (ULM)-based approach because of the ability of multiplexers to realize any function up to a certain number of input variables.

However, the circuit realized in the above manner may not be optimal in terms of the number of pass transistors. That means, there may exist another realization with lesser number of pass transistors. For example, optimal realization of \( f = a'b + ab' \) is shown in Fig. 5.4b. Optimal network may be obtained by expanding a function iteratively using Shannon’s expansion theorem. The variable around which expansion is done has to be suitably chosen and each reduced function is further expanded until it is a constant or a single variable.

**Example 5.1** Consider the function \( f = a + b'c \). Realize it using pass-transistor logic.

By expanding around the variable “\( a \)”, we get \( f = a.1 + a'.b'c \).

Now, by expanding around the variable “\( b \)”, we get \( f = a \cdot 1 + a'(b \cdot 0 + b' \cdot c) \).
No further expansion is necessary because there does not exist any reduced function, which is a function of more than one variable. Realization of the function \( f = a + b'c \) based on this approach is shown in Fig. 5.2c. Circuit realization based on the above approach is not only area efficient but also satisfies all the requirements of pass-transistor logic realization mentioned earlier.

**Advantages and Disadvantages**

Realization of logic functions using pass transistors has several advantages and disadvantages. The advantages are mentioned below:

a. Ratioless: We have seen that for a reliable operation of an inverter (or gate logic), the width/length (W/L) ratio of the pull-up device is four times (or more) that of the pull-down device. As a result, the geometrical dimension of the transistors is not minimum (i.e., \( 2\lambda \times 2\lambda \)). The pass transistors, on the other hand, can be of minimum dimension. This makes pass-transistor circuit realization very area efficient.

b. Powerless: In a pass-transistor circuit, there is no direct current (DC) path from supply to ground (GND). So, it does not require any standby power, and power dissipation is very small. Moreover, each additional input requires only a minimum geometry transistor and adds no power dissipation to the circuit.

c. Lower area: Any one type of pass-transistor networks, nMOS or p-type MOS (pMOS), is sufficient for the logic function realization. This results in a smaller number of transistors and smaller input loads. This, in turn, leads to smaller chip area, lower delay, and smaller power consumption.

However, pass-transistor logic suffers from the following disadvantages:

1. When a signal is steered through several stages of pass transistors, the delay can be considerable, as explained below.

When the number of stages is large, a series of pass transistors can be modeled as a network of resistance capacitance (RC) elements shown in Fig. 5.5. The ON resistance of each pass transistor is \( R_{\text{pass}} \) and capacitance \( C_L \). The value of \( R_{\text{pass}} \) and \( C_L \) depends on the type of switch used. The time constant \( R_{\text{pass}} C_L \) approximately gives the time constant corresponding to the time for \( C_L \) to charge to 63% of its final value. To calculate the delay of a cascaded stage of \( n \) transistors, we can simplify the equivalent circuit by assuming that all resistances and capacitances are equal and can be lumped together into one resistor of value \( n \times R_{\text{pass}} \) and one capacitor of value \( n \times C_L \). The equivalent time constant is \( n^2 R_{\text{pass}} C_L \). Thus, the delay increases as the square of number of pass transistors in series. However, this simplified analysis leads to overestimation of the delay. A more accurate estimate of the delay can be obtained by approximating the effective time constant to be

\[
\tau_{\text{ef}} = \sum_k R_{k,0} C_{k,1},
\]
Fig. 5.5 a Pass-transistor network. b RC model for the pass-transistor network. RC resistance capacitance

\[ where \ R_k \text{ stands for the resistance for the path common to } k \text{ and input.} \]

For \( k = 4 \),

\[
\tau_{\text{eff}} = R_1C_1 + (R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3 + (R_1 + R_2 + R_3 + R_n)C_L. \quad (5.3)
\]

Time constant \( \tau = CR_{\text{eq}} \frac{n(n + 1)}{2} \).

Assuming all resistances are equal to \( R_{\text{pass}} \) and all capacitances are equal to \( C \), the delay of the \( n \)-stage pass-transistor network can be estimated by using the Elmore approximation:

\[
t_p = 0.69 \frac{n(n + 1)}{2} \frac{R_{\text{pass}}}{C}. \quad (5.4)
\]

For \( n = 4 \), the delay is

\[ t_p = 6.9R_{\text{pass}}C, \]

which are ten times that of single-stage delays.

The above delay estimate implies that the propagation delay is proportional to \( n^2 \) and increases rapidly with the number of cascaded stages. This also sets the limit on the number of pass transistors that can be used in a cascade. To overcome this problem, the buffers should be inserted after every three of four pass-transistor stages.

For a large value of \( n \), let buffers are introduced after each \( k \) stages, as shown in Fig. 5.6 for \( k = 3 \). Assuming a propagation delay of each buffer is \( t_{\text{buf}} \), the overall propagation delay can be computed as follows:

\[
t = 0.69 \left[ \frac{n}{k} CR \frac{k(k + 1)}{2} \right]_p + \left( \binom{n}{k} - 1 \right) t_{\text{buf}} = 0.69 \left[ \frac{C}{L_{\text{pass}}} \frac{n(n + 1)}{2} \right]_p + \left( \binom{n}{k} - 1 \right) t_{\text{buf}}.
\]
In contrast to quadratic dependence on $n$ in a circuit without buffer, this expression shows a linear dependence on the number $n$. An optimal value of $k$ can be obtained from

$$\frac{\delta f}{\delta k} = 0 \Rightarrow k_{opt} = 1.7 \sqrt{\frac{I_{buf}}{R_{pass}C_L}},$$

for $R_{pass} = 10k\Omega$, $C = 10fF$, $t_{buf} = 0.5n$ sec, we get $k = 3.8$.

Therefore, a cascade of more than four pass-transistor stages is not recommended to restrict the delay within a reasonable limit.

2. As we have mentioned earlier, there is a voltage drop ($V_{out} = V_{dd} - V_{tn}$) as we steer the signal through nMOS transistors. This reduced level leads to high static currents at the subsequent output inverters and logic gates. In order to avoid this, it is necessary to use additional hardware known as the swing restoration logic at the gate output.

3. Pass-transistor structure requires complementary control signals. Dual-rail logic is usually necessary to provide all signals in the complementary form. As a consequence, two MOS networks are again required in addition to the swing restoration and output buffering circuitry.

The required double inter-cell wiring increases wiring complexity and capacitance by a considerable amount.

### 5.2.3 Pass-Transistor Logic Families

Several pass-transistor logic styles have seen proposed in recent years to overcome the limitations mentioned above. The most common ones are: the conventional nMOS pass-transistor logic or complementary pass-transistor logic (CPL), the dual pass-transistor logic (DPL), and the swing-restored pass-transistor logic (SRPL) [2]. In this section, they are introduced and compared.

**Complementary Pass-Transistor Logic** A basic structure of a CPL gate is shown in Fig. 5.7a. It comprises two nMOS pass-transistor logic network (one for each rail), two small pull-up pMOS transistors for swing restoration, and two output inverters for the complementary output signals. CPL realization for the 2-to-1
Fig. 5.7 a Basic complementary pass-transistor logic (CPL) structure; and b 2-to-1 multiplexer realization using CPL logic

Fig. 5.8 Complementary pass-transistor logic (CPL) logic circuit for a 2-input AND/NAND, b 2-input OR/NOR, and c 2-input EX-OR

multiplexer is shown in Fig. 5.7b. This is the basic and minimal gate structure with ten transistors. All two-input functions can be implemented by this basic gate structure. Realizations of 2-input NAND, NOR, and EX-OR functions are shown in Fig. 5.8a–c, respectively.

**Swing-Restored Pass-Transistor Logic** The SRPL logic style is an extension of CPL to make it suitable for low-power low-voltage applications. The basic SRPL logic gate is shown in Fig. 5.9a, where the output inverters are cross-coupled with a latch structure, which performs both swing restoration and output buffering. The pull-up pMOS transistors are not required anymore and that the output nodes of the nMOS networks are the gate outputs. Figure 5.9b shows an example of AND/NAND gate using SRPL logic style. As the inverters have to drive the outputs and must also be overridden by the nMOS network, transistor sizing becomes very difficult and results in poor output-driving capacity, slow switching, and larger short-circuit currents. This problem becomes worse when many gates are cascaded.

**Double Pass-Transistor Logic** The DPL is a modified version of CPL, in which both nMOS and pMOS logic networks are used together to alleviate the problem of the CPL associated with reduced high logic level. As this provides full swing on
the output, no extra transistors for swing restoration is necessary. Two-input AND/NAND DPL realization is shown in Fig. 5.10. As shown in the figure, both nMOS and pMOS pass transistors are used. Although, owing to the addition of pMOS transistor, the output of the DPL is full rail-to-rail swing, it results in increased input capacitance compared to CPL. It may be noted that DPL can be regarded as dual-rail pass-gate logic. The DPL has a balanced input capacitance, and the delay is independent of the input delay contrary to the CPL and conventional CMOS pass-transistor logic, where the input capacitance for different signal inputs is the same. As two current paths always drive the output in DPL, the reduction in speed due to the additional transistor is compensated.

**Single-Rail Pass-Transistor Logic** A single-rail pass-transistor logic style has been adopted in the single-rail pass-transistor logic (LEAP; LEAn integration with pass transistors) [7] design which exploits the full functionality of the multiplexer structure scheme. Swing restoration is done by a feedback pull-up pMOS transistor. This is slower than the cross-coupled pMOS transistors of CPL working in differential mode. This swing restoration scheme works for $V_{dd} > V_{th} + |V_{tp}|$, because the threshold voltage drop through the nMOS network for a logic “1” prevents the nMOS of the inverter and with that the pull-up pMOS from turning ON. As a consequence, robustness at low voltages is guaranteed only if the threshold voltages are appropriately selected. Complementary inputs are generated locally by inverters.
Three basic cells, $Y_1$, $Y_2$, and $Y_3$, used in LEAP logic design is shown in Fig. 5.11. It may be noted that $Y_1$ cell corresponds to a 2-to-1 multiplexer circuit, whereas $Y_3$ corresponds to 4-to-1 multiplexer circuit realized using three 2-to-1 multiplexers. $Y_3$ is essentially a 3-to-1 multiplexer realized using two 2-to-1 multiplexers. Any complex logic circuit can be realized using these three basic cells.

Comparison of the pass-transistor logic styles based on some important circuit parameters, such as number of MOS transistors, the output driving capabilities, the presence of input/output decoupling, requirement for swing restoration circuits, the number of rails, and the robustness with respect to voltage scaling and transistor sizing, is given in Table 5.1.
5.3 Gate Logic

Table 5.1 Qualitative comparisons of the logic styles

<table>
<thead>
<tr>
<th>Logic style</th>
<th>#MOS networks</th>
<th>Output driving</th>
<th>I/O decoupling</th>
<th>Swing restoration</th>
<th># Rails</th>
<th>Robustness</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOS</td>
<td>$2n$</td>
<td>Medium/good</td>
<td>Yes</td>
<td>No</td>
<td>Single</td>
<td>High</td>
</tr>
<tr>
<td>CPL</td>
<td>$2n+6$</td>
<td>Good</td>
<td>Yes</td>
<td>Yes</td>
<td>Dual</td>
<td>Medium</td>
</tr>
<tr>
<td>SRPL</td>
<td>$2n+4$</td>
<td>Poor</td>
<td>No</td>
<td>Yes</td>
<td>Dual</td>
<td>Low</td>
</tr>
<tr>
<td>DPL</td>
<td>$4n$</td>
<td>Good</td>
<td>Yes</td>
<td>No</td>
<td>Dual</td>
<td>High</td>
</tr>
<tr>
<td>LEAP</td>
<td>$n+3$</td>
<td>Good</td>
<td>Yes</td>
<td>Yes</td>
<td>Single</td>
<td>Medium</td>
</tr>
<tr>
<td>DCVSPG</td>
<td>$2n+2$</td>
<td>Medium</td>
<td>Yes</td>
<td>No</td>
<td>Dual</td>
<td>Medium</td>
</tr>
</tbody>
</table>


Fig. 5.12 a Fan-in of gates; and b fan-out of gates

Gate Logic

In gate logic, conventional gates, such as inverters, NAND gates, NOR gates, etc., are used for the realization of logic functions. In Chap. 4, we have already discussed the possible realization of inverter circuits by using both nMOS and CMOS technology. Realization of NAND, NOR, and other types of gates are considered in this section.

A basic inverter circuit can be extended to incorporate multiple inputs leading to the realization of standard logic gates such as NAND and NOR gates [4]. These gates along with the inverters can be used to realize any complex Boolean function. In this section, we shall consider various issues involved in the realization of these multi-input gates. But, before that we introduce the concept of fan-in and fan-out, which are important parameters in the realization of complex functions using these gates.

**Fan-In and Fan-Out**

Fan-in is the number of signal inputs that the gate processes to generate some output. Figure 5.12a shows a 2-input NAND gate and a four-input NOR gate with fan-in of 2 and 4, respectively. On the other hand, the fan-out is the number of logic inputs driven by a gate. For example, in Fig. 5.12b a four-input NOR gate is shown with fan-out of 3.
**5 MOS Combinational Circuits**

**Fig. 5.13** a n-input nMOS NAND gate; b equivalent circuits; and c n-input nMOS NOR gate. nMOS n-type MOS

---

### nMOS NAND and NOR Gates

Realizations of nMOS NAND gate with two or more inputs are possible. Let us consider the generalized realization with \( n \) inputs as shown in Fig. 5.13 with a depletion-type nMOS transistor as a pull-up device and \( n \) enhancement-type nMOS transistors as pull-down devices. In this kind of realization, the length/width (L/W) ratio of the pull-up and pull-down transistors should be carefully chosen such that the desired logic levels are maintained. The critical factor here is the low-level output voltage, which should be sufficiently low such that it turns off the transistors of the following stages. To satisfy this, the output voltage should be less than the threshold voltage, i.e., \( V_{\text{out}} \leq V_t = 0.2 V_{dd} \). To get a low-level output, all the pull-down transistors must be ON to provide the GND path. Based on the equivalent circuit shown in Fig. 5.13b, we get the following relationship:

\[
\frac{V_{dd} \times nZ_{pd}}{nZ_{pd} + Z_{pu}} \leq 0.2 V_{dd} \tag{5.5}
\]

for the boundary condition

\[
\frac{nZ_{pd}}{nZ_{pd} + Z_{pu}} \leq 0.2
\]

or \( nZ_{pd} \leq 0.2 nZ_{pu} + 0.2 Z_{pu} \)

or \( \frac{Z_{pu}}{nZ_{pd}} \leq 4. \)

Therefore, the ratio of \( Z_{pu} \) to the sum of all the pull-down \( Z_{pd} \)’s must be at least 4:1.

It may be noted that, not only one pull-down transistor is required per input of the NAND gate stage but also the size of the pull-up transistor has to be adjusted.
5.3 Gate Logic

5.3.3 CMOS Realization

The structure of a CMOS network to realize any arbitrary Boolean function is shown in Fig. 5.14a. Here, instead of using a single pull-up transistor, a network of pMOS transistors is used as pull up. The pull-down circuit is a network of nMOS transistors, as it is done in the case of an nMOS realization.

As a consequence, nMOS NAND gates are only used when absolutely necessary and the number of inputs is restricted so that area and delay remain within some limit.

An n-input NOR gate is shown in Fig. 5.13c. Unlike the NAND gate, here the output is low when any one of the pull-down transistors is on, as it happens in the case of an inverter. As a consequence, the aspect ratio of the pull-up to any pull-down transistor will be the same as that of an inverter, irrespective of the number of inputs of the NOR gate.

The area occupied by the nMOS NOR gate is reasonable, because the pull-up transistor geometry is not affected by the number of inputs. The worst-case delay of an NOR gate is also comparable to the corresponding inverter. As a consequence, the use of NOR gate in circuit realization is preferred compared to that of NAND gate, when there is a choice.
5.3.3.1 CMOS NAND Gates

Let us consider an \( n \)-input NAND gate as shown in Fig. 5.14b. It has fan-in of \( n \) having \( n \) number of nMOS transistor in series in the pull-down path and \( n \) number of pMOS transistors in parallel in the pull-up network. We assume that all the transistors are of the same size having a width \( W = W_n = W_p \) and length \( L = L_n = L_p \). If all inputs are tied together, it behaves like an inverter. However, static and dynamic characteristics are different from that of the standard inverter. To determine the inversion point, pMOS transistors in parallel may be equated to a single transistor with the width \( n \) times that of a single transistor. And nMOS transistors in series may be considered to be equivalent to have a length equal to \( n \) times that of a single transistor as shown in Fig. 5.15a. This makes the transconductance ratio \( \frac{\beta_n}{n^2 \beta_p} \).

Substituting these values in Eq. 4.17, we get

\[
V_{\text{inv}} = \frac{V_{\text{in}} + V_{dd} + \frac{\beta_n}{n^2 \beta_p}}{1 + \sqrt[n]{\beta_p}} \tag{5.6}
\]

or

\[
V_{\text{inv}} = \frac{V_{\text{in}} + \frac{\beta_n}{n \beta_p}}{1 + \frac{1}{n \sqrt[n]{\beta_p}}} \tag{5.7}
\]

As the fan-in number \( n \) increases, the \( \frac{\beta_n}{\beta_p} \) ratio (transconductance ratio) decreases leading to increase in \( V_{\text{inv}} \). In other words, with the increase in fan-in, the switching point (inversion point) moves towards the right. It may be noted that in our analysis, we have ignored the body effect.
5.3.3.2 CMOS NOR Gates

The realization of the \( n \)-input CMOS NOR gate is shown in Fig. 5.16a. Its equivalent circuit is shown in Fig. 5.16b. In this case, the transconductance ratio is equal to \( n^2 \beta_n / \beta_p \).

Substituting this ratio in Eq. 5.6, we get

\[
V_{\text{inv}} = \frac{V_{\text{dd}} + \frac{1}{\beta_p} nV_{\text{th}} \sqrt{\beta_n / \beta_p}}{1 + n \sqrt{\gamma_n / \gamma_p}}. \tag{5.8}
\]

As the fan-in number \( n \) increases, the \( \beta_n / \beta_p \) ratio (transconductance ratio) increases leading to a decrease in \( V_{\text{inv}} \). In other words, with the increase in fan-in, the switching point (inversion point) moves towards the left. Here, also, we have ignored the body effect. Therefore, with the increase in fan-in the inversion point moves towards left for NOR gates, whereas it moves towards right in case of NAND gates.

5.3.4 Switching Characteristics

To study the switching characteristics, let us consider the series and parallel transistors separately. Figure 5.17a shows \( n \) pull-up pMOS transistors with their gates tied together along with a load capacitance \( C_L \). An equivalent circuit is shown in Fig. 5.17b. Intrinsic time constant of this network is given by

\[
t_{\text{dr}} = \frac{R_p}{n} \left( nC_{\text{outp}} + \frac{R_p}{n} \right),
\]

\[
C_L = \frac{R_p}{n} \left( nC_{\text{outp}} + C_L \right). \tag{5.9}
\]
Here, the load capacitance $C_L$ includes all capacitances on the output node except the output capacitances of the transistors in parallel. The worst-case rise time occurs when only one pMOS transistor is ON and remaining pMOS transistors are OFF. This rise time is close to the rise time of an inverter.

To find out the fall-time (high-to-low time), let us consider the $n$ numbers of series-connected-nMOS transistors as shown in Fig. 5.18a. The intrinsic switching time for this network based on Fig. 5.18b is given by

$$ t = nR \left( \frac{C_{\text{out}}}{n} + C \right) + 0.35R C \left( n - 1 \right)^2 $$

(5.10)

The first term represents the intrinsic switching time of the $n$-series-connected transistors and the second term represents the delay caused by $R_n$ charging $C_{\text{inn}}$. For an $n$-input NAND gate

$$ t = R_p \left( \frac{nC}{n} + C_{\text{out}} + C \right) $$

(5.11)

and

$$ t = nR \left( \frac{C_{\text{out}}}{n} + nC \right) + 0.35R C \left( n - 1 \right)^2. $$

(5.12)

Thus, the delay increases linearly with the increase of fan-in number $n$. 
CMOS NOR Gate

In a similar manner, the rise and fall times of an \( n \)-input NOR gate can be obtained as

\[
t = \frac{R_n}{n} (nC_{\text{outp}} + C_L)
\]

and

\[
t = nR \left( \frac{C_{\text{outp}}}{p} + nC_{\text{oupn}} + C_{\text{outn}} \right) + 0.35R \frac{C_{\text{pin}}}{p} (n-1)^2.
\]

It may be noted that in case of NAND gate, the discharge path is through the series–connected-nMOS transistors. As a consequence, the high-to-low delay increases with the number of fan-in. If the output load capacitance is considerably larger than other capacitances then the fall time reduces to \( t_{df} = nR_nC_L \) and \( t_{dr} = R_pC_L \). On the other hand, in case of NOR gate the charging of the load capacitance take place through the series connected pMOS transistors giving rise time \( t_{dr} = nR_pC_L \) and \( t_{df} = R_nC_L \). As \( R_p > R_n \), the rise-time delay for NOR gates is more than the fall-time delay of NAND gates with the same fan-in. This is the reason why NAND gates are generally a better choice than NOR gates in complementary CMOS logic. NOR gates may be used for limited fan-out.

CMOS Complex Logic Gates

By combining MOS transistors in series–parallel fashion, basic CMOS NAND/NOR gates can be extended to realize complementary CMOS complex logic gates. Basic concept of realizing a complex function \( f \) by a single gate is shown in Fig. 5.19a. Here, the nMOS network corresponds to the function \( f' \) and the pMOS network is complementary of the nMOS network. For example, the realization of the function \( f = A' + BC \) is shown in Fig. 5.19b.

Here, \( f = A' + BC \Rightarrow f' = (A' + BC)' = A(B' + C') \).

The pull-down nMOS network realizes \( A(B' + C') \), whereas pull-up pMOS network realizes \( A' + BC \). In this realization, both nMOS and pMOS network are, in general, series–parallel combination MOS transistors as shown in Fig. 5.19b. Full-complementary CMOS realization of half-adder function is shown in Fig. 5.19c.

Here, \( S = A \oplus B = A'B + AB' \Rightarrow S' = (A'B + AB')' = (A + B')(A' + B) \).

In realizing complex functions using full-complementary CMOS logic, it is necessary to limit the number of MOS transistors in both the pull-up and pull-down network, so that the delay remains within the acceptable limit. Typically, the limit is in the range of 4–6 MOS transistors.
5.4 MOS Dynamic Circuits

The MOS circuits that we have discussed so far are static in nature [2]. In static circuits, the output voltage levels remain unchanged as long as inputs are kept the same and the power supply is maintained. nMOS static circuits have two disadvantages: they draw static current as long as power remains ON, and they require larger chip area because of “ratioed” logic. These two factors contribute towards slow operation of nMOS circuits. Although there is no static power dissipation in a full-complementary CMOS circuit, the logic function is implemented twice, one in the pull-up p-network and the other in the pull-down n-network. Due to the extra area and extra number of transistors, the load capacitance on gates of a full-complementary CMOS is considerably higher. As a consequence, speeds of operation of the CMOS and nMOS circuits are comparable. The CMOS not only has twice the available current drive but also has twice the capacitance of nMOS. The trade-off in choosing one or the other is between the lower power of the CMOS and the lower area of nMOS (or pseudo nMOS).

In the static combinational circuits, capacitance is regarded as a parameter responsible for poor performance of the circuit, and therefore considered as an undesirable circuit element. But, a capacitance has the important property of holding charge. Information can be stored in the form of charge. This information storage capability can be utilized to realize digital circuits. In MOS circuits, the capacitances need not be externally connected. Excellent insulating properties of silicon dioxide provide very good quality gate-to-channel capacitances, which can be used for information storage. This is the basis of MOS dynamic circuits. In Sect. 5.4.1, we discuss how dynamic MOS circuits are realized utilizing these capacitances using single-phase and two-phase clocks.

We shall see that the advantage of low power of full-complementary CMOS circuits and smaller chip area of nMOS circuits are combined in dynamic circuits leading to circuits of smaller area and lower power dissipation. MOS dynamic circuits

Fig. 5.19 a Realization of a function $f$ by complementary MOS (CMOS) gate; b realization of $f = A' + BC$; and c realization of $S = A \oplus B$
are also faster in speed. However, these are not free from disadvantages. Like any other capacitors, charge stored on MOS capacitors also leak. To retain information, it is necessary to periodically restore information by a process known as refreshing. There are other problems like charge sharing and clock skew leading to hazards and races. Suitable measures should be taken to overcome these problems. These problems are considered in Sect. 5.4.3.

**Single-Phase Dynamic Circuits**

To realize dynamic circuits, it is essential to use a clock. Two types of clocks are commonly used; single-phase clock and nonoverlapping two-phase clock. The single-phase clock consists of a sequence of pulses having high (logic 1) and low (logic 0) levels with width $W$ and time period $T$ as shown in Fig. 5.20a. A single-phase clock has two states (low and high) and two edges per period. The schematic diagram of a single-phase dynamic nMOS inverter is shown in Fig. 5.20b. Operation of a single-phase inverter circuit is explained below.

When the clock is in the high state, both transistors $Q_2$ and $Q_3$ are ON. Depending on the input, $Q_1$ is either ON or OFF. If the input voltage is low, $Q_1$ is OFF and the output capacitor (gate capacitor of the next stage) charges to $V_{dd}$ through $Q_2$ and $Q_3$. When the input voltage is high, $Q_1$ is ON and the output is discharged through it to a low level. When the clock is in the low state, the transistors $Q_2$ and $Q_3$ are OFF, isolating the output capacitor. This voltage is maintained during the OFF period of the clock, provided the period is not too long. During this period, the power supply is also disconnected and no current flows through the circuit. As current flows only when the clock is high, the power consumption is small, and it depends on the duty cycle (ratio of high time to the time period $T$). It may be noted that the output of the circuit is also ratioed, because the low output voltage depends on the ratio of the ON resistance of $Q_1$ to that of $Q_2$. As we know that, this ratio is related to the physical dimensions of $Q_1$ to $Q_2$ ($L:W$ ratio) and is often referred to as the inverter ratio. The idea of the MOS inverter can be extended to realize dynamic MOS NAND, NOR, and other gates as shown in Fig. 5.21.

The circuits realized using the single-phase clocking scheme has the disadvantage that the output voltage level is dependent on the inverter ratio and the number of transistors in the current path to GND. In other words, single-phase dynamic
circuits are ratioed logic. Moreover, as we have mentioned above, the circuit dissipates power when the output is low and the clock is high.

Another problem arising out of single-phase clocked logic is known as clock skew problem. This is due to a delay in a clock signal during its journey through a number of circuit stages. This results in undesired signals like glitch, hazards, etc. Some of the problems can be overcome using two-phase clocking scheme as discussed in the following subsection.

Two-Phase Dynamic Circuits

A two-phase nonoverlapping clock is shown in Fig. 5.22a. As the two phases ($\phi_1$ and $\phi_2$) are never high simultaneously, the clock has three states and four edges and satisfies the property $\phi_1.\phi'_2 = 0$. There is a dead time, $\Delta$, between transitions of the clock signals as shown in Fig. 5.22a. The schematic diagram of a circuit that generates two-phase clock is shown in Fig. 5.22b. The circuit takes a single-phase clock as an input and generates two-phase clock as the output. The dead time, $\Delta$, is decided by the delay through the NAND gates and the two inverters. As the clock (clk) signal goes low, $\phi_1$ also goes low after four gate delays. $\phi_2$ can go high after seven gate delays. In a similar manner, as clk goes high, $\phi_2$ goes low after five gate delays and $\phi_1$ goes high after eight gate delays. So $\phi_2(\phi_1)$ goes high after three gate delays and $\phi_1(\phi_2)$ goes low. Here the dead time is three gate delays. A longer dead time can be obtained by inserting more number of inverters in the feedback path.

This two-phase clocking gives a great deal of freedom in designing circuits. An inverter based on two-phase clock generator is shown in Fig. 5.23. When the clock $\Phi_2$ is high, the intrinsic capacitor charges to $V_{dd}$ through $Q_1$. And clock $\Phi_1$, which comes after $\Phi_2$ performs the evaluation. If $V_{in}$ is high, $Q_3$ is turned ON and since $Q_3$ is ON, the capacitor discharges to the GND level and the output $V_0$ attains low logic level. If $V_{in}$ is low, the $Q_2$ is OFF and there is no path for the capacitor to discharge. Therefore, the output $V_0$ remains at high logic level. It may be noted that the pull-up and pull-down transistors are never simultaneously ON. The circuit has no DC current path regardless of the state of the clocks or the information stored on the parasitic capacitors. Moreover, the output is not ratioed, i.e., the low-level output is independent of the relative value of the aspect ratio of the transistors. That
is why the circuits based on two-phase clocking are often termed as *ratioless* and *powerless*. Moreover, in dynamic circuits, there can be at the most one transition per clock cycles, and therefore there cannot be multiple spurious transitions called *glitches*, which can take place in static CMOS circuits. Like inverter, NAND, NOR, and other complex functions can be realized by replacing $Q_2$ with a network of two or more nMOS transistors.

### 5.4.3 CMOS Dynamic Circuits

Dynamic CMOS circuits avoid area penalty of static CMOS and at the same time retains the low-power dissipation of static CMOS, and they can be considered as extension of pseudo-nMOS circuits. The function of these circuits can be better explained using the idea of *pre-charged* logic. Irrespective of
several alternatives available in dynamic circuits, in all cases the output node is pre-charged to a particular level, while the current path to the other level is turned OFF. The charging of inputs to the gate must take place during this phase. At the end of this pre-charge phase, the path between the output node and the pre-charge source is turned OFF by a clock, while the path to the other level is turned ON through a network of MOS transistors, which represents this function. During this phase, known as evaluation phase, the inputs remain stable and depending on the state of the inputs, one of the two possibilities occurs. The network of transistors either connects the output to the other level discharging the output or no path is established between the other level and output, thereby retaining the charge. This operation is explained with the help of a circuit realization for the function \( f = x_3(x_1 + x_2) \). The realization of this function using full-complementary CMOS approach is shown in Fig. 5.24a. There are two alternative approaches for realizing it using dynamic CMOS circuits. As shown in Fig. 5.23b, the circuit is realized using the nMOS block scheme, where there is only one pMOS transistor, which is used for pre-charging. During \( \varphi = 0 \), the output is pre-charged to a high level through the pMOS transistor. And during the evaluation phase (\( \varphi = 1 \)), the output is evaluated through the network of nMOS transistors. At the end of the evaluation phase, the output level depends on the input states of the nMOS transistors. In the second approach, the circuit uses the pMOS block scheme, where there is only one nMOS transistor used for pre-discharging the output as shown in Fig. 5.24c. Here, during pre-charge phase, the output is pre-discharged to a low level through the nMOS transistor and during evaluation phase the output is evaluated through the network of pMOS transistors. In the evaluation phase, the low-level output is either retained or charged to a high level through the pMOS transistor network.
The dynamic CMOS circuits have a number of advantages. The number of transistors required for a circuit with fan-in $N$ is $(N+2)$, in contrast to $2N$ in case of static CMOS circuit. Not only dynamic circuits require $(N+2)$ MOS transistors but also the load capacitance is substantially lower than that for static CMOS circuits. This is about 50% less than static CMOS and is closer to that of nMOS (or pseudo nMOS) circuits. But, here full pull-down (or pull-up) current is available for discharging (or charging) the output capacitance. Therefore, the speed of the operation is faster than that of the static CMOS circuits. Moreover, dynamic circuits consume static power closer to the static CMOS. Therefore, dynamic circuits provide superior (area-speed product) performance compared to its static counterpart. For example, a dynamic NOR gate is about five times faster than the static CMOS NOR gate. The speed advantage is due to smaller output capacitance and reduced overlap current. However, dynamic circuits are not free from problems, as discussed in the following subsections.

**Charge Leakage Problem**

However, the operation of a dynamic gate depends on the storage of information in the form of charge on the MOS capacitors. The source–drain diffusions form parasitic diodes with the substrate. If the source (or drain) is connected to a capacitor $C_L$ with some charge on it, the charge will be slowly dissipated through the reverse-biased parasitic diode. Figure 5.25 shows the model of an nMOS transistor. The leakage current is expressed by the diode equation:

$$i_D = I_s (e^{\frac{-qV}{kT}} - 1),$$

where $I_s$ is the reverse saturation current, $V$ is the diode voltage, $q$ is the electronic charge ($1.602 \times 10^{-19}$ C), $k$ is the Boltzmann constant ($1.38 \times 10^{-23}$ J/K), and $T$ is the temperature in Kelvin.

It may be noted that the leakage current is a function of the temperature. The current is in the range 0.1–0.5 nA per device at room temperature. The current doubles...
for every 10 °C increase in temperature. Moreover, there will be some subthreshold leakage current. Even when the transistor is OFF and the current can still flow from the drain to source. This component of current increases when the gate voltage is above zero and as it approaches the threshold voltage, this effect becomes more pronounced. A sufficiently high threshold voltage $V_t$ is recommended to alleviate this problem. The charge leakage results in gradual degradation of high-level voltage output with time, which prevents it to operate at a low clock rate. This makes it unattractive for many applications, such as toys, watches, etc., which are operated at lower frequency to minimize power dissipation from the battery. As a consequence, the voltage on the charge storage node slowly decreases. This needs to be compensated by refreshing the charge at regular intervals.

**Charge Sharing Problem**

Dynamic circuits suffer from charge sharing problem because of parasitic capacitances at different nodes of a circuit. For example, consider the circuit of Fig. 5.26a. The node A charges to $V_{dd}$ during the pre-charge phase with a charge of $C_L V_{dd}$ stored on the capacitor $C_1$. Assume that the parasitic capacitances of nodes B and C are $C_1$ and $C_2$, respectively. To start with these capacitors are assumed to have no charges. As there is no current path from node A to GND, it should stay at $V_{dd}$ during the evaluation phase. But because of the presence of $C_1$ and $C_2$, redistribution of charge will take place leading to the so-called charge-sharing problem. This charge-sharing mechanism can be modeled as shown in Fig. 5.26b, where $C_1$ and $C_2$ are connected to node A through two switches representing the MOS transistors. Before the switches are closed, the charge on $C_L$ is given by $Q_A = V_{dd} C_L$ and charges at node B and C are $Q_B = 0$ and $Q_C = 0$, respectively.

After the switches are closed, there will be a redistribution of charges based on the charge conservation principle, and the voltage $V_A$ at node A is given by

$$C_L V_{dd} = (C_L + C_1 + C_2) V_A$$  \hspace{1cm} (5.15)
5.4 MOS Dynamic Circuits

Fig. 5.27 A weak p-type MOS (pMOS) transistor to reduce the impact of charge leakage and charge sharing problem

Therefore, the new voltage at node A, at the end of the evaluation phase, will be

\[ V_A = \frac{C_L}{C_L + C_1 + C_2} V_{dd}. \]  

(5.16)

If \( C_1 = C_2 = 0.5 \, C_L \), then \( V_A = 0.5 \, V_{dd} \). This may not only lead to incorrect interpretation of the output but also results in the high-static-power dissipation of the succeeding stage.

To overcome the charge sharing and leakage problem, some techniques may be used. As shown in Fig. 5.27, a weak pMOS (low W/L) is added as a pull-up transistor. The transistor always remains ON and behaves like a pseudo-nMOS circuit during the evaluation phase. Although there is static power dissipation due to this during evaluation phase, it helps to maintain the voltage by replenishing the charge loss due to the leakage current or charge sharing.

5.4.4.3 Clock Skew Problem

It is not uncommon to use several stages of dynamic circuits to realize a Boolean function. Although same clock is applied to all these stages, it suffers delay due to resistance and parasitic capacitances associated with the wire that carry the clock pulse and this delay is approximately proportional to the square of the length of the wire. As a result, different amounts of delays are experienced at different points in the circuit and the signal-state changes that are supposed to occur simultaneously may never actually occur at the same time. This is known as clock skew problem and it results in hazard and race conditions. For example, let us consider the two stages of CMOS dynamic circuits. Timing relationship of the two clocks of two consecutive stages is shown in Fig. 5.28a. The two clocks are not in synchronization. When pre-charging is in progress for the first stage, evaluation phase has already started in the second stage. As the output is high at the time of the pre-charge phase, this will discharge the output of the second stage through the MOS transistor \( Q_1 \), irrespective of what will be the output of the first stage during the evaluation phase. The charge loss leads to reduced noise margins and even malfunctioning
of the circuit; because, if the output depends on the difference in delay time of the two clocks, then the charge on the load capacitance $C_L$ of the second stage may discharge to the extent that the voltage is less than the threshold voltage of the MOS transistor of the next stage. As a consequence, this problem may lead to incorrect output as a cascading problem arises because the output is pre-charged to a high level, which leads to inadvertent discharge of the next stage. This problem can be overcome if the output can be set to low during pre-charge. This basic concept is used in realizing domino CMOS.

There are several approaches to overcome this problem. One straightforward approach to deal with this problem is to introduce a delay circuit in the path of the clock that will compensate for the discharge delay between the first and the second stage of the dynamic circuit. It may be noted that the delay is a function of the complexity of the discharge path. This “self-timing” scheme is fairly simple and often used in CMOS programmable logic array (PLA) and memory design. However, there exist other approaches by which this problem can be overcome. Special type of circuits, namely domino and (no race) NORA CMOS circuits, as discussed below, can also be used.

### 5.4.5 Domino CMOS Circuits

A single-domino CMOS gate is shown in Fig. 5.29. It consists of two distinct components. The first component is a conventional dynamic pseudo-nMOS gate, which
works based on the pre-charge technique. The second component is a static inverting CMOS buffer. Only the output of the static buffer is fed as an input to the subsequent stage. During the pre-charge phase, the dynamic gate has a high output, so the buffer output is low. Therefore, during pre-charge, all circuit inputs which connect the output of one domino gate to the input of another are low, and the transistors which are driven by these outputs are OFF. Moreover, during the evaluation phase, a domino gate can make only one transition namely a low to high. This is due to the fact that the dynamic gate can only go from high to low during the evaluation phase. As a result, the buffer output could only go from low to high during evaluation phase, which makes the circuits glitch-free.

During the evaluation phase, output signal levels change from low to high from the input towards the output stages, when several domino logic gates are cascaded. This phenomenon resembles the chain action of a row of falling dominos, hence the name domino logic [5].

Domino CMOS circuits have the following advantages:

- Since no DC current path is established either during the pre-charge phase or during the evaluation phase, domino logic circuits have lower power consumption.
- As n-block is only used to realize the circuit, domino circuits occupy lesser chip area compared to static CMOS circuits.
- Due to lesser number of MOS transistors used in circuit realization, domino CMOS circuits have lesser parasitic capacitances and hence faster in speed compared to static CMOS.

Full pull-down current is also available to drive the output nodes. Moreover, the use of single clock edge to activate the circuit provides simple operation and maximum possible speed. One limitation of domino logic is that each gate requires an inverting buffer. However, this does not pose a serious problem, because buffers are needed anyway to achieve higher speed and higher fan-out. Another limitation of the domino logic is that all the gates are non-inverting in nature, which calls for new logic design paradigm which is different from the conventional approach.

5.4.6 NORA Logic

Another alternative is to use the duality property of the n-blocks and p-blocks used in the realization CMOS circuits. The pre-charge output of an n-block is “1,” whereas the pre-charge output of a p-block is 0. By alternatively using p-blocks and n-blocks, as shown in Fig. 5.30, the clock skew problem is overcome in NORA logic circuits.

NORA stands for NO RAce [6]. Here, both the n- and p-blocks are in pre-charge phase when $\text{clk} = 1$ or $\text{clk} = 0$. The outputs of n-mostly and p-mostly blocks are pre-charged and pre-discharged to 1 and 0, respectively. During this phase, the inputs are set up. The n-mostly and p-mostly blocks are in evaluation phase when
During this phase, inputs are held constant and outputs are evaluated as a function of the inputs.

As pre-charged condition output is 1 (0) for an n-block (p-block), it cannot lead to any discharge of the outputs of a p-block (n-block). As a consequence, the internal delays cannot result in a race condition, and we may conclude that the circuits designed based on the NORA logic are internal-delay race-free.

Compared to the domino technique, this approach gives higher flexibility. As both inverted and non-inverted signals are available from the NORA technique, alternating n-mostly and p-mostly blocks can be used to realize circuits of arbitrary logical depth. When connection between same types of blocks is necessary, domino-like inverters may be used between them. The NORA logic style has the disadvantage that p-block is slower than n-blocks, due to lower mobility of the current carrier of the pMOS circuits. Proper sizing, requiring extra area, may be used to equalize the delays. Compared to domino logic circuits, the NORA logic circuits are faster due to the elimination of the static inverter and the smaller load capacitance. The Digital Equipment Corporation (DEC) alpha processor, the first 250 MHz CMOS microprocessor, made extensive use of NORA logic circuits.

NORA technique can be also used to realize pipelined circuit in a convenient manner. A single stage of a pipeline block is shown in Fig. 5.30. It consists of one n-mostly, one p-mostly, and a clocked CMOS (C²MOS) block, which is used as a latch stage for storing information. The pipeline circuits can be realized by applying $\phi$ and $\phi'$ to consecutive pipeline blocks. For $\phi = 0$ and $\phi' = 1$, the $\phi$-sections are pre-charged; while, the $\phi$-sections outputs are held constant by the C²MOS latch stages. Then for phase $\phi = 1$ and $\phi' = 0$, the $\phi$-sections are in evaluation phase and $\phi$-sections are in pre-charge phase. Now, the $\phi$-section outputs, evaluated in the previous phase, are held constant and the $\phi$-sections can use this information for computation. In this way, information flows through the pipeline of alternating $\phi$ and $\phi'$ sections.

5.5 Some Examples

In this section, realization of several example functions such as full adder, parity generator and priority encoder and using different logic styles are considered and the number of transistors required for realization by different logic styles is compared.
5.5 Some Examples

Fig. 5.31 Block diagram of the full adder

<table>
<thead>
<tr>
<th>A</th>
<th>FULL</th>
<th>C</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>ADDER</td>
<td>CIN</td>
<td>Co</td>
</tr>
</tbody>
</table>

The block diagram of a full adder is shown in Fig. 5.31. The sum of product for the sum and carry functions are given below:

\[
S = ABC_{in} + \overline{C_0} (A + B + C_{in}) \\
C_0 = AB + C_{in} (A + B).
\]

Realizations of full-adder function by using static CMOS, dynamic CMOS, and pass-transistor circuit are shown in Figs. 5.32, 5.33, and 5.34, respectively.

de. Full adder

The block diagram of a 4-bit parity generator is shown in Fig. 5.35. The truth table is given in Table 5.2. Realizations of the 4-bit parity generator using static CMOS, domino CMOS, and pass-transistor circuit are shown in Figs. 5.36, 5.37, and 5.38, respectively.

e. Parity generator

The block diagram of a 4-bit parity generator is shown in Fig. 5.35. The truth table is given in Table 5.2. Realizations of the 4-bit parity generator using static CMOS, domino CMOS, and pass-transistor circuit are shown in Figs. 5.36, 5.37, and 5.38, respectively.

Fig. 5.32 Static complementary MOS (CMOS) realization of full adder
Fig. 5.33 NORA complementary MOS (CMOS) realization of full adder

Fig. 5.34 Pass-transistor realization of the full adder

Fig. 5.35 Block diagram of 4-bit parity generator

f. Priority encoder
The block diagram of the 8-bit priority encoder is shown in Fig. 5.39. The truth table is given in Table 5.3. The sum-of-product form of the three functions to be realized are given below:

\[
y_0 = x_4x_6(x_1\overline{x_2} + x_3) + x_5\overline{x_6} + x_7 \\
y_1 = x_4x_5(x_2 + x_3) + x_6 + x_7 \\
y_2 = x_4 + x_5 + x_6 + x_7.
\]
Some Examples

Table 5.2 Parity generator

<table>
<thead>
<tr>
<th>$x$</th>
<th>$x^2$</th>
<th>$x$</th>
<th>$x_0$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig. 5.36 Static complementary MOS (CMOS) realization of parity generator
Fig. 5.37  Domino complementary MOS (CMOS) realization of 4-bit parity generator

Fig. 5.38  Pass-transistor realization 4-bit parity generator

Fig. 5.39  Block diagram of 8-input priority encoder

Table 5.3  Truth Table of the priority encoder

<table>
<thead>
<tr>
<th>$x_7$</th>
<th>$x_6$</th>
<th>$x_5$</th>
<th>$x_4$</th>
<th>$x_3$</th>
<th>$x_2$</th>
<th>$x_1$</th>
<th>$x_0$</th>
<th>$y_2$</th>
<th>$y_1$</th>
<th>$y_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
The realizations of the function using static CMOS, domino CMOS, and pass-transistor circuit are shown in Figs. 5.40, 5.41, and 5.42, respectively.

Comparison Table 5.4 compares the area requirements in terms of the number of transistors for the three examples using three logic styles. We find that both dynamic CMOS and pass-transistor realizations require lesser number of transistors and hence the smaller area. Delay of different implementations, given in nanosecond, has been estimated using Cadence SPICE simulation tool. Delay of different implementations is presented in Table 5.5. We find that the domino CMOS has the minimum delay.
UNIT-3
Sources of Power Dissipation

Introduction

In order to develop techniques for minimizing power dissipation, it is essential to identify various sources of power dissipation and different parameters involved in each of them. Power dissipation may be specified in two ways. One is maximum power dissipation, which is represented by “peak instantaneous power dissipation.” Peak instantaneous power dissipation occurs when a circuit draws maximum power, which leads to a supply voltage spike due to resistances on the power line. Glitches may be generated due to this heavy flow of current and the circuit may malfunction, if proper care is not taken to suppress power-line glitches. The second one is the “average power dissipation,” which is important in the context of battery-operated
portable devices. The average power dissipation will decide the battery lifetime. Here, we will be concerned mainly with the average power dissipation, although the techniques used for reducing the average power dissipation will also lead to the reduction of peak power dissipation and improve reliability by reducing the possibility of power-related failures.

Power dissipation can be divided into two broad categories—static and dynamic. Static power dissipation takes place continuously even if the inputs and outputs do not change. For some logic families, such as nMOS and pseudo-nMOS, both pull-up and pull-down devices are simultaneously ON for low output level causing direct current (DC) flow. This leads to static power dissipation. However, in our low-power applications, we will be mainly using complementary metal–oxide–semiconductor (CMOS) circuits, where this type of static power dissipation does not occur. On the other hand, dynamic power dissipation takes place due to a change in input and output voltage levels.

At this point, it is important to distinguish between the two terms—power and energy. Although we often use both the terms interchangeably, these two do not have the same meaning. Power dissipation is essentially the rate at which energy is drawn from the power supply, which is proportional to the average power dissipation. Power dissipation is important from the viewpoint of cooling and packaging of the integrated circuit (IC) chips. On the other hand, energy consumed is important for battery-driven portable systems. As power dissipation is proportional to the clock frequency, to reduce power dissipation by half we may reduce the clock frequency by half. Although this will reduce the power dissipation and keep the chip cooler, the time required to perform a computation will double that of the previous case. In this case, the energy is drawn from the battery at half the rate of the previous case. But, the total energy drawn from the battery for performing a particular computation remains the same. Figure 6.1 illustrates the difference between power and energy. The height of the graphs represents the power and the area under the curve represents the energy. Here, the same energy is drawn for different average power dissipation. For battery-operated portable systems, a reduction in energy consumption is more important than a reduction in power consumption. Instead of using power as a performance metric, the power-delay product, which is dimensionally the same as energy, is the natural metric for low-power systems, where battery life is the index of energy efficiency. We usually mean low energy when we say low power.

**Fig. 6.1** Power versus energy
In CMOS circuits, power dissipation can be divided into two broad categories: dynamic and static. Dynamic power dissipation in CMOS circuits occur when the circuits are in working condition or active mode, that is, there are changes in input and output conditions with time. In this section, we introduce the following three basic mechanisms involved in dynamic power dissipation:

- **Short-circuit power**: Short-circuit power dissipation occurs when both the nMOS and pMOS networks are ON. This can arise due to slow rise and fall times of the inputs as discussed in Sect. 6.2.
- **Switching power dissipation**: As the input and output values keep on changing, capacitive loads at different circuit points are charged and discharged, leading to power dissipation. This is known as switching power dissipation. Until recently, this was the most dominant source of power dissipation. The switching power dissipation is discussed in Sect. 6.3.
- **Glitching power dissipation**: Due to a finite delay of the logic gates, there are spurious transitions at different nodes in the circuit. Apart from the abnormal behavior of the circuits, these transitions also result in power dissipation known as glitching power dissipation. This is discussed in Sect. 6.4.

Static power dissipation occurs due to various leakage mechanisms. The following seven leakage current components are discussed in Sect. 6.5:

- Reverse-bias p–n junction diode leakage current
- Reverse-biased p–n junction current due to the tunneling of electrons from the valence bond of the p region to the conduction bond of the n region, known as band-to-band-tunneling current
- Subthreshold leakage current between source and drain when the gate voltage is less than the threshold voltage \( V_t \)
- Oxide-tunneling current due to a reduction in the oxide thickness
- Gate current due to a hot-carrier injection of electrons
- Gate-induced drain-leakage (GIDL) current due to high field effect in the drain junction
- Channel punch-through current due to close proximity of the drain and the source in short-channel devices

### 6.2 Short-Circuit Power Dissipation

When there are finite rise and fall times at the input of CMOS logic gates, both pMOS and nMOS transistors are simultaneously ON for a certain duration, shorting the power supply line to ground. This leads to current flow from supply to ground. Short-circuit power dissipation takes place for input voltage in the range \( V_{in} < V_{in} < V_{dd} - | V_{tp} | \), when both pMOS and nMOS transistors turn ON creating a conducting path between \( V_{dd} \) and ground (GND). It is analyzed in the case of a CMOS inverter as shown in Fig. 6.2. To estimate the average short-circuit current,
we have used a simple model shown in Fig. 6.3. It is assumed that \( \tau \) is both rise and fall times of the input \( (\tau_r = \tau_f = \tau) \) and the inverter is symmetric, i.e., \( \beta_n = \beta_p = \beta \) and \( V_{in} = -V_{ip} = V_t \). The mean short-circuit current of the inverter having no load attached to

\[
I_{\text{mean}} = 2 \times \frac{1}{T} \left[ \int_{t_1}^{2t_1} i(t) dt + \int_{t_2}^{2t_2} i(t) dt \right].
\]  

(6.1)

Because of the symmetry, we may write

\[
I_{\text{mean}} = \frac{4}{T} \left[ \int_{t_1}^{t_2} i(t) dt \right].
\]  

(6.2)

As the nMOS transistor is operating in the saturation region, \( i(t) = \frac{\beta}{2} (V(t) - V)^2 \), substituting this, we get

\[
I_{\text{mean}} = \frac{4}{T} \left[ \int_{t_1}^{t_2} \frac{\beta}{2} (V_{in}(t) - V_t)^2 dt \right].
\]  

(6.3)

Here, \( V_{in}(t) = \frac{V_{dd}}{\tau} t \), \( t_1 = \frac{\tau}{V_{dd}} V_t \), and \( t_2 = \frac{\tau}{2} \).

Substituting these in Eq. 6.2, we get

\[
I_{\text{mean}} = \frac{2\beta}{T} \int_{V_{dd}/V_t}^{V_t} (V_{in}(t) - V_t)^2 dt.
\]  

(6.4)

\[
= \frac{2\beta}{2} \int_{V_{dd}/V_t}^{V_t} \left( \frac{V}{V_{dd}/t-V} \right)^2 dt.
\]  

(6.5)

Substituting \( \frac{V_{dd}}{\tau} - V = X \) and \( \frac{V_{dd}}{\tau} dt = dX \), we get
The short-circuit power is given by

\[
P_{\text{sc}} = V_{\text{dd}} I_{\text{mean}} = \frac{\beta}{12} (V_{\text{dd}} - 2V_f)^3 \tau_f = \frac{\beta}{12} V_{\text{dd}}^3 \left(1 - 2 \frac{\tau_f}{V_{\text{dd}}} \right)^3.\] 

As the clock frequency decides how many times the output changes per second, the short-circuit power is proportional to the frequency. The short-circuit current is also proportional to the rise and fall times. Short-circuit currents for different input slopes are shown in Fig. 6.4. The power supply scaling affects the short-circuit power considerably because of cubic dependence on the supply voltage.

So far, we have assumed that there is no load (capacitive) at the output. In practice, there will be some load capacitance at the output and the output rise and fall times will be dependent on that. If the load capacitance is very large, the input changes from high to low or low to high before the output changes significantly. In
such a situation, the short-circuit current will be very small. It is maximum when there is no load capacitance. The variation of short-circuit current for different output capacitances is shown in Fig. 6.5. From this analysis, it is evident that the short-circuit power dissipation can be minimized by making the output rise/fall times smaller. The short-circuit power dissipation is also reduced by increasing the load capacitance. However, this makes the circuit slower. One good compromise is to have equal input and output slopes.

Because of the cubic dependence of the short-circuit power on supply voltage, the supply voltage may be scaled to reduce short-circuit power dissipation. It may also be noted that if the supply voltage is scaled down such that \( V_{dd} \leq (V_{tn} + |V_{tp}|) \), the short-circuit current can be virtually eliminated, because the nMOS and pMOS transistors will not conduct simultaneously for any input values. The variation of voltage transfer characteristics (VTC) for different supply voltages are shown in Fig. 6.6. It is evident from the diagram that the circuit can function as an inverter even when the supply voltage is equal to or less than the sum of the two threshold voltages \((V_{tn} + |V_{tp}|)\). In such a situation, however, the delay time of the circuit increases drastically. Figure 6.7 shows the VTC when the supply voltage is less than the sum of the two threshold voltages. Here, the VTC contains a region when none of the two transistors conduct. However, the previous voltage level is maintained by the stored charge in the output capacitor and the VTC exhibits a hysteresis behavior.

We may conclude this subsection by stating that the short-circuit power dissipation depends on the input rise/fall time, the clock frequency, the load capacitance, gate sizes, and above all the supply voltage.
6.3 Switching Power Dissipation

There exists capacitive load at the output of each gate. The exact value of capacitance depends on the fan-out of the gate, output capacitance, and wiring capacitances and all these parameters depend on the technology generation in use. As the output changes from a low to high level and high to low level, the load capacitor charges and discharges causing power dissipation. This component of power dissipation is known as switching power dissipation.

Switching power dissipation can be estimated based on the model shown in Fig. 6.8. Figure 6.8a shows a typical CMOS gate driving a total output load capacitance $C_L$. For some input combinations, the pMOS network is ON and nMOS network is OFF as modeled in Fig. 6.8b. In this state, the capacitor is charged to $V_{dd}$ by drawing power from the supply. For some other input combinations, the nMOS network is ON and pMOS network is OFF, which is modeled in Fig. 6.8c. In this state, the capacitor discharges through the nMOS network. For simplicity, let us assume that the CMOS gate is an inverter.
During the transition of the output from 0 to $V_{dd}$, the energy drawn from the power supply is given by

$$E_{0\rightarrow1} = \int_0^{V_{dd}} p(t) \, dt = \int V_{dd} \cdot i(t) \, dt,$$

(6.8)

where $i(t)$ is an instantaneous current drawn from the supply voltage $V_{dd}$ and it can be expressed as

$$i(t) = C_{\text{L}} \frac{dV_0}{dt},$$

(6.9)

where $V_0$ is the output voltage. Substituting this in Eq. 6.8, we get

$$E = V \int_0^{V_{dd}} C \, dV = CV^2.$$

(6.10)

Regardless of the waveform and time taken to charge the capacitor, $C V^2$ is the energy drawn from the power supply for 0 to $V_{dd}$ transition at the load capacitance. Part of the energy drawn from the supply is stored in the load capacitance

$$E = V \int_0^{V_{dd}} i(t) \, dt = CV \int_0^{V_{dd}} V \, dV = \frac{1}{2} C V^2.$$

(6.11)

This implies that half of the energy is stored in the capacitor, and the remaining half $(1/2)C V^2$ is dissipated in the pMOS transistor network. During the $V$ to 0 transition at the output, no energy is drawn from the power supply and the charge stored in the capacitor is discharged in the nMOS transistor network.

If a square wave of repetition frequency $f (1/T)$ is applied at the input, average power dissipated per unit time is given by

$$P = \frac{1}{T} V \int_0^{V_{dd}} C \, dV = CV^2 f.$$

(6.12)
The switching power is proportional to the switching frequency and independent of device parameters. As the switching power is proportional to the square of the supply voltage, there is a strong dependence of switching power on the supply voltage. Switching power reduces by 56%, if the supply voltage is reduced from 5 to 3.3 V, and if the supply voltage is lowered to 1 V, the switching power is reduced by 96% compared to that of 5 V. This is the reason why voltage scaling is considered to be the most dominant approach to reduce switching power.

**Dynamic Power for a Complex Gate**

For an inverter having a load capacitance $C_L$, the dynamic power expression is $C_L V^2 f$. Here, it is assumed that the output switches from rail to rail and input switching occurs for every clock. This simple assumption does not hold good for complex gates because of several reasons. First, apart from the output load capacitance, there exist capacitances at other nodes of the gate. As these internal nodes also charge and discharge, dynamic power dissipation will take place on the internal nodes. This leads to two components of dynamic power dissipation-load power and internal node power. Second, at different nodes of a gate, the voltage swing may not be from rail to rail. This reduced voltage swing has to be taken into consideration for estimating the dynamic power. Finally, to take into account the condition when the capacitive node of a gate might not switch when the clock is switching, a concept known as switching activity is introduced. Switching activity determines how often switching occurs on a capacitive node. These three issues are considered in the following subsections.

**Reduced Voltage Swing**

There are situations where a rail-to-rail swing does not take place on a capacitive node. This situation arises in pass transistor logic and when the pull-up device is an enhancement-type nMOS transistor in nMOS logic gates as shown in Fig. 6.9. In such cases, the output can only rise to $V_{dd} - V_t$. This situation also happens in interval nodes of CMOS gates. Instead of $C_L V^2$ for full-rail charging, the energy drawn from power supply for charging the capacitance to $(V_{dd} - V_t)$ is given by

$$E_{0\rightarrow1} = C_L V_{dd} (V_{dd} - V_t).$$

(6.13)
A three-input NAND gate is shown in Fig. 6.10. Apart from the output capacitance $C_L$, two capacitances $C_1$ and $C_2$ are shown in two interval nodes of the gate. For input combination 110, the output is “1” and transistors $Q_3$, $Q_4$, and $Q_5$ are ON. All the capacitors will draw energy from the supply. Capacitor $C_L$ will charge to $V_{dd}$ through $Q_3$, capacitor $C_1$ will charge to $(V_{dd} - V_i)$ through $Q_3$ and $Q_4$. Capacitor $C_2$ will also charge to $(V_{dd} - V_i)$ through $Q_3$, $Q_4$, and $Q_5$. For each 0-to-$V_{dd}$ transition at an internal node, the energy drawn is given by

$$E_{0\rightarrow1} = C_i V_i V_{dd},$$

(6.14)

where $C_i$ is the internal node capacitance and $V_i$ is internal voltage swing at node $i$.

### Switching Activity [2, 3]

For a complex logic gate, the switching activity depends on two factors—the topology of the gate and the statistical timing behavior of the circuit. To handle the transition rate variation statistically, let $n(N)$ be the number of 0-to-$V_{dd}$ output transitions in the time interval $[0,N]$. Total energy $E_N$ drawn from the power supply for this interval is given by

$$E_N = C V_{ldd}^2 \times n(N).$$

(6.15)

The average power dissipation during an extended interval is

$$P_{\text{avg}} = \lim_{N \to \infty} \frac{E_N}{N} \times f,$$

where $f$ is the clock frequency.

$$P_{\text{avg}} = \left( \lim_{N \to \infty} \frac{n(N)}{N} \right) C V_{ldd}^2 f.$$

(6.16)

The term $\lim_{N \to \infty} (n(N) / N)$ gives us the expected (average) value of the number of transitions per clock cycle, which is defined as the switching activity. Therefore,
\[ \alpha = \lim_{n \to \infty} \frac{n(N)}{N} . \quad (6.17) \]

Taking into account various aspects discussed so far, the dynamic power consumption is given by
\[ P = \alpha C V^2 f + \sum_{i=1}^{k} \alpha CVV f , \quad (6.18) \]

where \( \alpha_0 \) is the switching activity at the output node, \( \alpha_i \) is the switching activity on the \( i \)th internal node, and \( f \) is the clock frequency. Here, it is assumed that there are \( k \) internal nodes.

**Switching Activity of Static CMOS Gates**

The switching activity at the output of a static CMOS gate depends strongly on the function it realizes. It is assumed that the inputs are independent to each other and the probability of occurrence of “0” and “1” is same, let \( P_0 \) be the probability that the output will be “0” and \( P_1 \) is the probability that the output will be “1.” Then,
\[ P_{0 \to 1} = P_0 P_1 = P_0 (1 - P_0) . \quad (6.19) \]

For an \( n \)-input gate, the total number of input combinations is \( 2^n \). Out of \( 2^n \) combinations in the truth table, let \( n_0 \) be the total number of combinations for which the output is 0 and \( n_1 \) is the total number of combinations for which the output is “1,” then
\[ P_0 = \frac{n_0}{2^n} \quad \text{and} \quad P_1 = \frac{n_1}{2^n} . \quad (6.20) \]

We get
\[ P_{0 \to 1} = \frac{n_0}{2^n} \times \frac{n_1}{2^n} = \frac{n_0 n_1}{2^{2n}} . \quad (6.21) \]

For example, consider a two-input NAND gate with the truth table given in Table 6.1. In this case, \( P_0 = \frac{1}{4} \) and \( P_1 = \frac{3}{4} \).
So \( P_{0 \to 1} = \frac{3}{16} \).

For NAND/NOR gates, the switching activity decreases with the increase in the number of inputs. On the other hand, the switching activity at the output of an EX-OR gate in 1/4, which is independent on the number of inputs. Output activities for different gates are shown in Table 6.2. The variation of the switching activity at the output of NAND, NOR, and EX-OR gates with the increase in the number of inputs is shown in Fig. 6.11.
Table 6.1 Truth table of NAND gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>( F_{\text{NAND}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Inputs Not Equiprobable**

So far, we have assumed that the inputs are independent of each other and equiprobable. But, inputs might not be equiprobable. In such cases, the probability of transitions at the output depends on the probability of transitions at the primary inputs. Let us consider a two-input NAND gate with two inputs A and B having mutually independent signal probabilities of \( P_A \) and \( P_B \). The probability of logical “0” at the output is \( P_0 = P_A P_B \) and the probability of “1” at the output is \( P_1 = (1 - P_0) = (1 - P_A P_B) \). Therefore,

\[
P_{0 \rightarrow 1} = P_0 P_1 = (1 - P_A P_B)P_A P_B. \tag{6.22}
\]

In a similar manner, the switching activity of a two-input NOR gate (output is 1 when both the inputs are 0) is

\[
P_1 = (1 - P_A)(1 - P_B)
\]

\[
P_0 = [1 - (1 - P_A)(1 - P_B)]
\]

\[
P_{0 \rightarrow 1} = (1 - P_A)(1 - P_B)[1 - (1 - P_A)(1 - P_B)]. \tag{6.23}
\]

In this manner, the switching activity at the output for any complex gate can be obtained. The output switching activity of different gates is given in Table 6.3.

**Mutually Dependent Inputs**

When a multilevel network of complex gates are considered, the inputs to a gate at a particular stage might not be independent. This can happen because the signal from one input may propagate through multiple paths and again recombine at a particular stage. This problem of re-convergent fan-out is shown with the help of a simple ex-

Table 6.2 Switching activity of different gates

<table>
<thead>
<tr>
<th>GATE</th>
<th>INV</th>
<th>Two-input NAND/AND</th>
<th>Two-input NOR/OR</th>
<th>Two-input EX-OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{0 \rightarrow 1} )</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>16</td>
<td>16</td>
<td>4</td>
</tr>
</tbody>
</table>
Fig. 6.11 Variation of switching activity with increase in the number of inputs

Table 6.3 Switching activity of different gates for inputs not equiprobable

<table>
<thead>
<tr>
<th>Function</th>
<th>( P_{0 \rightarrow 1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAND/AND</td>
<td>( P_A P_B (1 - P_A P_B) )</td>
</tr>
<tr>
<td>NOR/OR</td>
<td>( (1 - P_A)(1 - P_B) \left[ 1 - (1 - P_A)(1 - P_B) \right] )</td>
</tr>
<tr>
<td>XOR</td>
<td>( \left[ 1 - (P_A + P_B - 2P_A P_B) \left[ P_A + P_B - 2P_A P_B \right] \right] )</td>
</tr>
</tbody>
</table>

Example given in Fig. 6.12. Two circuits are shown, one without re-convergent fan-out and the other with re-convergent fan-out. In the case of the circuit of Fig. 6.12a, signals at nodes E and F are independent and it gives a switching activity \( P_{0 \rightarrow 1} = \frac{63}{256} \) at node G.

This computation of transition probability is quite straightforward. Starting from the primary inputs, signal probabilities are calculated till the output node is reached. But, this approach will not work for circuits having re-convergent fan-out. Circuits having re-convergent fan-out result in inter-signal dependencies. For example, in circuit 6.12b, E and F signals had inter-signal dependencies leading to different switching activities at node G of the circuit and Fig. 6.9. In this case, the inputs E and F are mutually dependent, and the switching activity \( P_{0 \rightarrow 1} \) at node G is \( \frac{15}{64} \).

Example 6.1 Let us now consider three different implementations of a six-input OR function, which drives a capacitive load of 0.1 pF. The three implementations are:

Fig. 6.12 a Circuit without re-convergent fan-out. b Circuit with re-convergent fan-out
Fig. 6.13  Three different realizations for the six-input OR function

(i) A six-input NOR and an inverter
(ii) Two three-input NOR followed by a two-input NAND
(iii) Three two-input NOR followed by a three-input NAND

The three realizations are shown in Fig. 6.13.

Table 6.4 provides details of various gates used for the realization of the three implementations. The table provides the area in terms of unit area called cell grid, output and input capacitances, and delay in terms of output capacitance $C_0$ in pico-farad. For all transistors, $W_p = 2W_n = 10 \ \mu m$. In Table 6.5, the switching activity and different node points for the three realizations are given. The power cost, which is the sum of the product of switching activity and node capacitance at different points are given. As it is evident, implementation-1 requires minimum area and it is the most power efficient. However, the delay time of this implementation is longer than the other two implementations. In the other two cases, the delay is smaller but the power cost is higher.

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Area (cell unit)</th>
<th>Input capacitance (fF)</th>
<th>Output capacitance (fF)</th>
<th>Average delay (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INV</td>
<td>2</td>
<td>85</td>
<td>48</td>
<td>$0.22 + 1.00 \ C_0$</td>
</tr>
<tr>
<td>NAND2</td>
<td>3</td>
<td>105</td>
<td>48</td>
<td>$0.30 + 1.24 \ C_0$</td>
</tr>
<tr>
<td>NAND3</td>
<td>4</td>
<td>132</td>
<td>48</td>
<td>$0.37 + 1.50 \ C_0$</td>
</tr>
<tr>
<td>NAND6</td>
<td>7</td>
<td>200</td>
<td>48</td>
<td>$0.65 + 2.30 \ C_0$</td>
</tr>
<tr>
<td>NOR2</td>
<td>3</td>
<td>101</td>
<td>48</td>
<td>$0.27 + 1.50 \ C_0$</td>
</tr>
<tr>
<td>NOR3</td>
<td>4</td>
<td>117</td>
<td>48</td>
<td>$0.31 + 2.00 \ C_0$</td>
</tr>
</tbody>
</table>
Table 6.5 Transition activity at different points and relative performance of the three implementations

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th></th>
<th></th>
<th>II</th>
<th></th>
<th></th>
<th></th>
<th>III</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>O_1</td>
<td>Z</td>
<td>O_1</td>
<td>O_2</td>
<td>Z</td>
<td>O_1</td>
<td>O_2</td>
<td>O_3</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P_1</td>
<td>63/64</td>
<td>1/64</td>
<td>7/8</td>
<td>7/8</td>
<td>1/64</td>
<td>3/4</td>
<td>3/4</td>
<td>3/4</td>
<td>1/64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P_0</td>
<td>1/64</td>
<td>63/64</td>
<td>1/8</td>
<td>1/8</td>
<td>63/64</td>
<td>1/4</td>
<td>1/4</td>
<td>1/4</td>
<td>63/64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P_{0\rightarrow1}</td>
<td>63/4096</td>
<td>63/4096</td>
<td>7/64</td>
<td>7/64</td>
<td>63/4096</td>
<td>3/16</td>
<td>3/16</td>
<td>3/16</td>
<td>63/4096</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>9</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>1.1</td>
<td>0.85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.87</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P_{cost}</td>
<td>6.7</td>
<td>42.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>89.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6.6 Transition activity of dynamic gates

<table>
<thead>
<tr>
<th>Gate</th>
<th>INV</th>
<th>Two-input NOR</th>
<th>Two-input NAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_{0\rightarrow1}</td>
<td>1/2</td>
<td>3/4</td>
<td>1/4</td>
</tr>
</tbody>
</table>

**Transition Probability in Dynamic Gates**

The logic style has a strong influence on the node transition probability. For static CMOS, we calculated the transition probability as $P_0P_1 = P_0(1 - P_0)$. In the case of dynamic gates, the situation is different. As we have seen, in the case of domino or NORA logic styles, the output nodes are pre-charged to a high level in the pre-charge phase and then are discharged in the evaluation phase, depending on the outcome of evaluation. In other words, the output transition probability will depend on the signal probability $P_0$, i.e.,

$$P_{0\rightarrow1} = P_0,$$

(6.24)

where $P_0$ is the probability for the output is in the zero state. For $n$ independent inputs to a gate

$$P_{0\rightarrow1} = \frac{N_0}{2^n},$$

(6.25)

where $N_0$ is the number of zero entries in the truth table. Table 6.6 gives the output transition probabilities for NAND, NOR, and inverter. Compared to the static CMOS, transition probabilities are more, because the output is always pre-charged to 1, i.e., $P_1 = 1$.

It may be noted that the switching activity for NAND and NOR gates is more in the dynamic gates than their static gate counterparts.
**Power Dissipation due to Charge Sharing**

Moreover, in the case of dynamic gates, power dissipation takes place due to the phenomenon of charge sharing even when the output is not 0 at the time evaluation, i.e., the output load capacitance is not discharged, but part of the charge of the load capacitance might get redistributed leading to a reduction in the output voltage level. In the next pre-charge period, the output is again pre-charged back to $V_{dd}$. For example, consider the three-input NAND dynamic gate shown in Fig. 6.14.

As shown in Fig. 6.14, after charge sharing, the new voltage level is

\[
(C_1 + C_2 + C_L) V_{\text{new}} = V_{dd} C_L
\]

or

\[
V_{\text{new}} = \frac{C_L V_{dd}}{C_1 + C_2 + C_3}.
\]  

(6.26)

Therefore,

\[
\Delta V = V_{dd} - \frac{C_1}{C_1 + C_2 + C_3} \times V_{dd}
\]

\[
= \frac{C_1 + C_2}{C_1 + C_2 + C_L} V_{dd}.
\]  

(6.27)

The power transferred from the supply is

\[
P = C \frac{V_{dd} \Delta V}{C_1 + C_2 + C_L} V^2.
\]  

(6.28)

The power is additional to the normal switching activity power dissipation.
Glitching Power Dissipation

In the power calculations so far, we have assumed that the gates have zero delay. In practice, the gates will have finite delay and this delay will lead to spurious undesirable transitions at the output. These spurious signals are known as *glitches*. In the case of a static CMOS circuit, the output node or internal nodes can make undesirable transitions before attaining a stable value. Consider the circuit shown in Fig. 6.15. If the inputs ABC change value from 101 to 000, ideally for zero gate delay the output should remain at the 0 logic level. However, considering unit gate delay of the first gate stage, output O\textsubscript{1} is delayed compared to the C input. As a consequence, the output switches to 1 logic level for one gate delay duration. This transition increases the dynamic power dissipation and this component of dynamic power is known as *glitching power*. Glitching power may constitute a significant portion of dynamic power, if circuits are not properly designed.

Usually, cascaded circuits as shown in Fig. 6.16a exhibit high glitching power. The glitching power can be minimized by realizing a circuit by balancing delays, as shown in Fig. 6.16b. On highly loaded nodes, buffers can be inserted to balance delays and cascaded implementation can be avoided, if possible, to minimize glitching power.

---

**Fig. 6.15** Output waveform showing glitch at output O\textsubscript{2}

**Fig. 6.16** Realization of A, B, C, and D, a in cascaded form, b balanced realization
Leakage Power Dissipation

When the circuit is not in an active mode of operation, there is static power dissipation due to various leakage mechanisms. In deep-submicron devices, these leakage currents are becoming a significant contributor to power dissipation of CMOS circuits. Figure 6.17 illustrates the seven leakage mechanisms. Here, $I_1$ is the reverse-bias p–n junction diode leakage current; $I_2$ is the reverse-biased p–n junction current due to tunneling of electrons from the valence bond of the $p$ region to the conduction bond of the $n$ region; $I_3$ is the subthreshold leakage current between the source and the drain when the gate voltage is less than the threshold voltage $V_t$; $I_4$ is the oxide-tunneling current due to a reduction in the oxide thickness; $I_5$ is gate current due to hot-carrier injection of electrons; $I_6$ is the GIDL current due to a high field effect in the drain junction; and $I_7$ is the channel punch-through current due to the close proximity of the drain and the source in short-channel devices. These leakage components are discussed in the following subsections.

6.5.1 p–n Junction Reverse-Biased Current

Let us consider the physical structure of a CMOS inverter shown in Fig. 6.18. As shown in the figure, source–drain diffusions and n-well diffusions form parasitic diodes in the bulk of silicon substrate. As parasitic diodes are reverse-biased, their leakage currents contribute to static power dissipation. The current for one diode is given by

$$I_{rdlc} = A J_s \left[ e^{\frac{V_D}{nKT}} - 1 \right],$$

(6.29)

where $J_s$ is the reverse saturation current density (this increases with temperature), $I_s$ is the $AJ_s$, $V_d$ is the diode voltage, $n$ is the emission coefficient of the diode (sometimes equal to 1), $q$ is the charge of an electron ($1.602 \times 10^{-19}$), $K$ is the Boltzmann
constant \((1.38 \times 10^{-23} \text{ J/k})\), \(T\) is the temperature in K, \(V = \frac{KT}{q}\) is known as the thermal voltage.

At room temperature, \(V_T \approx 26 \text{mV}\).

The leakage current approaches \(I_s\), the reverse saturation current even for a small reverse-biased voltage across the diode. The reverse saturation current per unit area is defined as the current density \(J_s\), and the resulting current is approximately \(I = A \cdot J_s\), where \(A\) is area of drain diffusion. For a typical CMOS process, \(J_s \approx 1 - 5 \text{pA/\mu m}^2\) at room temperature, and the \(A\) is about 6 \(\mu \text{m}^2\) for a 1.0-\(\mu \text{m}\) minimum feature size. It leads to a leakage current of about 1 fA per device at room temperature. The reverse diode leakage current \(I_{rdc}\) increases significantly with temperature. The total static diode leakage current for \(n\) devices is given by
\[
I_{rdc} = \sum_{i=1}^{n} I_i
\]
and the total static power dissipation for \(n\) devices is equal to
\[
P = V_{dd} \cdot \sum_{i=1}^{n} I_i. \tag{6.30}
\]

Then, the total static power dissipation due to diode leakage current for one million transistors is given by
\[
P = V_{dd} \sum_{i=1}^{10^6} I_{di} \approx 0.01 \mu \text{W}. \tag{6.31}
\]

This is small enough to make any significant impact on the power dissipation when the circuit is in a normal mode (active) of operation. However, when a particular portion of the chip remains in the standby mode for a considerable period of time, this component of power dissipation cannot be ignored.
6.5.2 **Band-to-Band Tunneling Current**

When both n regions and p regions are heavily doped, a high electric field across a reverse biased p–n junction causes a significant current to flow through the junction due to tunneling of electrons from the valence band of the p region to the conduction band of n region. This is illustrated in Fig. 6.19. It is evident from this diagram that for the voltage drop across that junction should be more than the band gap. The tunneling current density is given by

\[ J_{b-b} = A \frac{E V_{app} \exp \left( -B \frac{E^{3/2}}{E} \right)}{E^{3/2}} \]

where

\[ A = \frac{\sqrt{2m^* q^3}}{4 \pi^3 h^2} \quad \text{and} \quad B = \frac{4\sqrt{m^*}}{3qh}, \]

where \( m^* \) is the effective mass of electron, \( E \) is the elective field at the junction, \( q \) is the electronic charge, and \( h \) is the reduced Planck’s constant (1/2π times).

Assuming a step junction, the electric field at the junction is given by

\[ E = \frac{2qN_a N_d (V_{app} + V_{bi})}{\varepsilon_s (N_a + N_d)}, \]

where \( N_a \) and \( N_d \) are the doping in the p and n side, respectively. \( \varepsilon_s \) is the permittivity of silicon and \( V_{bi} \) is the built-in voltage across the junction. High doping concentrations and abrupt doping profiles are responsible for a significant band-to-band tunneling BTBT current in scaled devices through the drain–well junction.

6.5.3 **Subthreshold Leakage Current**

The subthreshold leakage current in CMOS circuits is due to carrier diffusion between the source and the drain regions of the transistor in weak inversion, when
the gate voltage is below $V_t$. The behavior of an MOS transistor in the subthreshold operating region is similar to a bipolar device, and the subthreshold current exhibits an exponential dependence on the gate voltage. The amount of the subthreshold current may become significant when the gate-to-source voltage is smaller than, but very close to, the threshold voltage of the device.

The subthreshold current expression as given by the BSIM3 model is stated below:

$$I_{stlc} = Ae^{\frac{-q(V - V_{th})}{N^*KT}} \left(1 - e^{\frac{-qV_{ds}}{kT}}\right)^{2} \times \left(\frac{1}{e^{\frac{-q(V - V_{ds})}{mV}} - 1}\right)$$

$$m = 1 + \frac{c_{dm}}{c_{ox}} = 1 + \frac{\varepsilon_{si}}{\varepsilon_{ox}/t_{ox}} = 1 + \frac{3t_{ox}}{w_{dm}}$$

where $m$ is the subthreshold swing coefficient, $V_T = \frac{KT}{q}$ is the thermal voltage, $\mu_0$ is the zero bias mobility, $c_{ox}$ is the gate oxide capacitance per unit area, $w_{dm}$ is the maximum depletion layer width, and $t_{ox}$ is the gate oxide thickness.

The typical value of this current for a single transistor is 1–10 nA. In long-channel devices, the subthreshold current is independent of the drain voltage for $V_{ds}$ larger than few $V_t$. However, for short-channel devices, the dependence on the gate voltage is exponential as illustrated in Fig. 6.20. The inverse of the slope of the log10($I_{ds}$) versus $V_{gs}$ characteristic is called the subthreshold slope ($S_t$), which is given by the following relationship

$$S_t = \left[ \frac{c_{dm}}{C_{ox}} \right]^{-1} = 2.3 \frac{mkT}{q} = 2.3 \frac{kT}{q} \left[ \frac{C}{1 + \frac{c_{dm}}{C_{ox}}} \right]$$

where $C_{dm}$ is capacitance of the depletion layer and $C_{ox}$ is oxide capacitance. Here, $S_t$ is measured in mVs per decade of the drain current. Typical values $S_t$ for a bulk CMOS process can range from 70 to 120 mV/decade. As a low value of $S_t$ is desirable, the gate oxide thickness $t_{ox}$ is reduced or the doping concentration is lowered to achieve this.

Various mechanisms which affect the subthreshold leakage current are:

- Drain-induced barrier lowering (DIBL)
- Body effect
- Narrow-width effect
- Effect of channel length and $V_{th}$ roll-off
- Effect of temperature
Fig. 6.20 Log($I_D$) versus $V_G$ at two different drain voltages for $20 \times 0.4$-µm n-channel transistor in a 0.35-µm CMOS process

6.5.3.1 Drain-Induced Barrier Lowering

For long-channel devices, the sources and drain region are separated far apart and the depletion regions around the drain and source have little effect on the potential distribution in the channel region. So, the threshold voltage is independent of the channel length and drain bias for such devices. However, for short-channel devices, the source and drain depletion width in the vertical direction and the source drain potential have a strong effect on a significant portion of the device leading to variation of the subthreshold leakage current with the drain bias. This is known as the DIBL effect. Because of the DIBL effect, the barrier height of a short-channel device reduces with an increase in the subthreshold current due to a lower threshold voltage.

DIBL occurs when the depletion regions of the drain and the source interact with each other near the channel surface to lower the source potential barrier. Figure 6.19 shows the lateral energy band diagram at the surface versus distance from the source to the drain. It is evident from the figure that DIBL occurs for short-channel lengths and it is further enhanced at high drain voltages. Ideally, the DIBL effect does not change the value of $S_t$, but does lower $V_{th}$. Figure 6.23 shows the DIBL effect as it shows the $I_D - V_G$ curve for different gate voltages (Fig. 6.22).
6.5.3.2 Body Effect

As a negative voltage is applied to the substrate with respect to the source, the well-to-source junction, the device is reverse-biased and bulk depletion region is widened. This leads to an increase in the threshold voltage. This effect is known as the body effect. The threshold voltage equation given below gives the relationship of the threshold voltage with the body bias

\[
V_{th} = V_{fb} + 2\psi_B + \frac{\sqrt{2\phi_i q N_a (2\tau_B + V_{Th})}}{C_{ox}},
\]
Fig. 6.23 n-Channel drain current versus gate voltage illustrating various leakage components

where $V_{fb}$ is the flat-band voltage, $N_a$ is the doping density in the substrate, and

$$
\psi = \left( \frac{K T}{e} \right) \ln \left( \frac{N_a}{n_i} \right) + \phi
$$

is the difference between the Fermi potential and the intrinsic potential in the substrate.

The variation of the threshold voltage with respect to the substrate bias $d V_{th}/d V_{bs}$ is referred to as the substrate sensitivity:

$$
d V_{th} = \frac{\sqrt{\epsilon_s q N_a}}{C_{ox}} \frac{d V_{bs}}{2(2 \psi_B + V_{bs})}.
$$

Substrate sensitivity is higher for higher bulk doping concentration, and it decreases as the substrate reverse bias increases. At $V_{bs} = 0$, it is equal to $C_{dm}/C_{ox}$ or $m - 1$ where $m$ is also called the body effect coefficient.

Figure 6.24 shows the $I_d$ versus $V_G$ curve for different body bias from 0 to $-5$ V. The curves show that the $S_t$ does not change with body bias, but $I_{OFF}$ decreases due to an increase in the threshold voltage.

Taking into consideration all the three effects, e.g., weak inversion, DIBL, body effect and subthreshold leakage can be modeled as

$$
I_{subth} = A \times e^{\mu/mr (V_G - V_T - V_{th} - \psi_B)} \times (1 - e^{-V_{bs}/V_T}),
$$
where
\[
A = \mu_0 C_{ox} \frac{W}{L_{eff}} (V_T)^2 e^{1.8} e^{-\Delta V / \eta V_T}
\]

\(V_{th0}\) is the zero-bias threshold voltage and \(V_T = \frac{KT}{q}\) is the thermal voltage. \(\gamma'\) is the linearized body effect coefficient for small values of source to bulk voltage, \(\eta\) is the DIBL coefficient, \(C_{ox}\) is the gate oxide capacitance, \(\mu_0\) is the zero bias mobility, \(m\) is the subthreshold swing coefficient \(\left( m = 1 + \frac{3}{L_{ox}} \right)\), and \(\Delta V_{th}\) is the parameter to take into account the transistor-to-transistor leakage variations.

Narrow-Width Effect

The width of a gate, particularly when it becomes narrow, affects the threshold voltage as shown in Fig. 6.25. The reduction in threshold voltage also leads to an increase in the subthreshold leakage current (Fig. 6.24).

\(V_{th}\) Roll-Off

As the channel length is reduced, the threshold voltage of metal–oxide–semiconductor field-effect transistor (MOSFET) decreases. This reduction of channel length is known as \(V_{th}\) roll-off. Figure 6.26 shows the reduction of threshold voltage with a reduction in channel length. This effect is caused by the proximity of the source and drain regions leading to a 2D field pattern rather than a 1D field pattern in short-channel devices. The bulk change that needs to be inverted by the application of gate voltage is proportional to the area under the trapezoidal region, shown in Fig. 6.27, given by \(Q_B' \propto W_{dm} (L + L') / 2\). So, the gate voltage has to invert less bulk charge to turn the transistor on leading to more band bending in the Si–SiO\(_2\) interface in the short-channel device compared to long-channel devices. As a consequence, the threshold voltage is lower for a short-channel device. Moreover, the effect of the source–drain depletion region is more severe for high drain bias voltage. This results in a further decrease in threshold voltage and larger subthreshold current.

Temperature Effect

There is a strong dependence of threshold voltage on temperature. The equation
\[
S = 2.3 \frac{KL}{q} 1 + \frac{C_{dm}}{C_{ox}}
\]
indicates that the subthreshold slope changes linearly with
temperature. For a 0.3-μm technology, $S_t$ varies from 58.2 to 81.9 mV/decade. As the temperature varies from $-50$ to $+25^\circ$C in the 0.3-μm technology as shown in Fig. 6.28, the $I_{OFF}$ increases from 0.45 to 100 pA, an increase by a factor of 356, for a 20-μm-wide device (23 fA/μm to 8 pA/μm). The following two parameters are responsible for increases in subthreshold leakage current:

(i) $S_t$ increases linearly with temperature.

(ii) The threshold voltage decreases with temperature at the rate of 0.8 mV/°C.
Tunneling Through Gate Oxide

Device size scaling leads to a reduction in oxide thickness, which in turn results in an increase in the field across the oxide. The high electric field along with low oxide thickness leads to the tunneling of electrons from the substrate to the gate and vice versa, through the gate. The basic phenomenon of tunneling is explained with the help of a heavily doped n+ type poly-silicon gate and a p-type substrate.
Because of higher mobility, primarily electrons take part in the tunneling process. An energy band diagram in flat-band condition is shown in Fig. 6.29a, where $\Phi$ is the Si–SiO$_2$ interface barrier height for electrons. The energy band diagram changes to that of Fig. 6.29b, when a positive bias is applied to the gate. The electrons at the strongly inverted surface can tunnel through the SiO$_2$ because of the small width of the potential barrier arising out of small oxide thickness. Similarly, if a negative gate bias is applied, the electrons from the n+ poly-silicon can tunnel through the oxide layer as shown in Fig. 6.29c.

This results in gate oxide-tunneling current, which violates the classical infinite input impedance assumption of MOS transistors and thus affects the circuit performance significantly.

**Hot-Carrier Injection**

Electrons and holes near the Si–SiO$_2$ interface can gain sufficient energy due to the high electric field in short-channel MOSFETs and cross the interface potential barrier and enter into the oxide layer as shown in Fig. 6.30. This phenomenon is known
as the hot-carrier injection. Because of the lower effective mass and smaller barrier height (3.1 eV) for electrons compared to holes (4.5 eV), the possibility of an injection due to electrons is more than the holes.

**Gate-Induced Drain Leakage**

Due to a high electric field ($V_{ds}$) across the oxide, a deep depletion region under the drain overlap region is created, which generates electrons and holes by band-to-band tunneling. The resulting drain to body current is called GIDL current. When the negative gate bias is large ($V_{gate}$ at zero or negative voltages with respect to drain at $V_{dd}$), the n+ region under the gate can be depleted and even inverted as shown in Fig. 6.31. As a consequence, minority carriers are emitted in the drain region underneath the gate. As the substrate is at a lower potential, the minority carriers at the drain depletion region are swept away laterally to the substrate, creating a GIDL current. Thinner oxide thickness and higher $V_{dd}$ enhance the electric field and therefore enhance GIDL current.

At low drain doping, the electric field is not high enough to cause tunneling. At very high drain doping, the depletion width and hence the tunneling is limited. The GIDL is worse for moderated doping, when the depletion width and electric field are both considerable.

**Punch-Through**

Due to the proximity of the drain and the source in short-channel devices, the depletion regions at the source–substrate and drain–substrate junctions extend into the channel. If the doping is kept constant while the channel length is reduced, the separation between the depletion region boundaries decreases. Increased reverse bias (higher $V_{ds}$) across the junction further decreases the separation. When the depletion region merge, a majority of the carriers in the source enter into the substrate and get collected by the drain. This situation is known as punch-through condition. The net
Fig. 6.31 GIDL effect. *GDIL* gate-induced drain leakage

**Fig. 6.31 a**

- $V_d > 0$
- $V_d < 0$
- n+ ploy gate
- n+ drain
- Depletion edge

**Fig. 6.31 b**

- $V = V_{DD}$
- $V_d = V_{DD}$
- n+ ploy gate
- n+ drain
- Tunnel created minority carrier
- Depletion edge
- GIDL
- p-substrate

Effect of punch through is an increase in the subthreshold current. Moreover, punch-through degrades the subthreshold slope.

The punch-through voltage $V_{PT}$ estimates the value of $V_{ds}$ for which punch through occurs at $V_{gs} = 0$:

$$V_{PT} \propto N_B (L - W_j)^3,$$

where $N_B$ is the doping concentration at the bulk, $L$ is the channel length, and $W_j$ is the junction width.

One method for controlling the punch through is to have a halo-implant at the leading edges of the drain–source junctions.
In this chapter, various sources of power dissipation in digital CMOS circuits have been presented. The contribution of various sources of power dissipation in the total power for present-generation static CMOS circuits is shown in Fig. 6.32. It is evident from the figure that the switching power dissipation constitutes 80–90% of the total power dissipation. Next dominant source of power dissipation is due to threshold leakage current, which constitutes 10–30% of the total power dissipation. However, the size of the MOS transistors is shrinking, the power dissipation due to leakage current is increasing rapidly as shown in Fig. 6.33. It is anticipated that the dynamic power and subthreshold leakage power dissipation will be comparable in terms of percentage of the total power in the next generation circuits of submicron technology. Both short-circuit power dissipation and static power dissipation constitute about 5% of the total power dissipation.

However, the above situation holds good when the circuit is switching at the same rate of the operating frequency. This is not true when some subsystems remain in the standby mode. In such cases, the standby power dissipations due to diode leakage current and subthreshold leakage current takes place, of which the subthreshold leakage current is dominant.
From the equation for dynamic power, we find that, because of quadratic dependence of the dynamic power on the supply voltage, the supply voltage reduction is the dominant technique for realizing low-power circuits. The other parameters that affect power (or energy) are the switching activity \( \alpha \) and the capacitance \( C_L \). The product of the switching activity and capacitance, known as switched capacitance, is another parameter that can be minimized to reduce power dissipation. In the subsequent chapters, we discuss various low-power synthesis approaches at different levels of design hierarchy.

**Supply Voltage Scaling for Low Power**

![Graph](image)

Fig. 7.1 a Variation of normalized energy with respect to supply voltage; b variation of delay with respect to supply voltage

Although the dynamic power has three components, the switching power is the most dominant component. The switching power \( P_{\text{switching}} = \alpha C V^2 f \) caused by the charging and discharging of capacitances at different nodes of a circuit can be optimized by reducing each of the components such as the clock frequency \( f \), the total switched capacitance \( \sum \alpha_i C_i \), and the supply voltage \( V_{dd} \). Another dynamic power component, the glitching power is often neglected. But, it can account for up to 15% of the dynamic power. The third component, the short-circuit power, captures the power dissipation as a result of a short-circuit current, which flows between the supply voltage and ground (GND), when the CMOS logic gates switches from 0 to −1 or from 1 to 0. This can be minimized by minimizing the rise and fall times. The static power dissipation has also three dominant components. The most significant among them is the subthreshold leakage power due to the flow of current between the drain and source. The second important component is the gate leakage power due to the tunneling of electrons from the bulk silicon through the gate oxide potential barrier into the gate. In sub-50-nanometer devices, the source–substrate and drain–substrate reversed p–n junction band-to-band tunneling current, the third component, is also large. Because of the quadratic dependence of dynamic power on the supply voltage, supply voltage scaling was initially developed to reduce dynamic power. But, the supply voltage scaling also helps to reduce the static power because the subthreshold leakage power decreases due to the reduction of the drain-induced barrier lowering (DIBL), the gate-induced drain leakage (GIDL), and the gate tunneling current as well. It has been demonstrated that the supply voltage scaling leads to the reduction of the subthreshold leakage and gate leakage currents of the order of \( V^3 \) and \( V^4 \), respectively.
From the above discussion, it is quite evident that reducing the supply voltage, $V_{dd}$, is the most effective way to reduce both dynamic and static power dissipations. A plot of the normalized energy with supply voltage variation is shown in Fig. 7.1. The normalized energy, which is equivalent to the power-delay product (PDP), can be considered as the most appropriate performance metric for low-power applications. Unfortunately, this reduction in power dissipation comes at the expense of
performance. The delay of a circuit is related to the supply voltage by the following equation:

$$\text{Delay} \propto \left( \frac{V_{dd}}{V_t} \right)^2 = \frac{1}{V_{dd}} \left( 1 - \frac{V_t}{V_{dd}} \right)^2 \quad (7.2)$$

As it is evident from Eq. (7.2), there is a performance penalty for the reduction in the supply voltage. If the threshold voltage is not scaled along with the supply voltage to avoid an increase in leakage current, a plot of the variation of the normalized delay with the supply voltage variation is shown in Fig. 7.1b. The plot shows that the delay increases with the decrease in supply voltage in a nonlinear manner, and it increases sharply as the supply voltage approaches the threshold voltage.

It is essential to devise a suitable mechanism to contain this loss in performance due to the supply voltage scaling for the realization of low-power high-performance circuits. The loss in performance can be compensated by using suitable techniques at different levels of design hierarchy, that is, the physical level, logic level, architectural level, algorithmic level, and system level.

The voltage scaling approaches can be divided into the following four categories:

**Static Voltage Scaling (SVS)** In this case, fixed supply voltages are applied to one or more subsystems or blocks.

**Multilevel Voltage Scaling (MVS)** This is an extension of the SVS, where two or few fixed discrete voltages are applied to different blocks or subsystems.

**Dynamic Voltage and Frequency Scaling (DVFS)** This is an extension of the MVS, where a large number of discrete voltages are applied in response to the changing workload conditions of the subsystems.

**Adaptive Voltage Scaling (AVS)** This is an extension of the DVFS, where a close-loop control system continuously monitors the workload and adjusts the supply voltage.

In this chapter, we discuss the abovementioned voltage scaling techniques starting with SVS. In the first physical-level-based approach, the device feature size is scaled to overcome the loss in performance as discussed in Sect. 7.2. In Sect. 7.3, we focus on architecture-level approaches, such as parallelism and pipelining for SVS. SVS using high-level transformation has been discussed in Sect. 7.4. Dynamic voltage and frequency scheduling approach is discussed in Sect. 7.5. Section 7.5 introduces multilevel voltage scaling (MVS), and various challenges of MVS are highlighted in Sect. 7.6. Dynamic voltage and frequency scaling (DVFS) has been discussed in Sect. 7.7. Adaptive voltage scaling has been highlighted in Sect. 7.8. Then, subthreshold logic circuits has been introduced in Sect. 7.9.
Device Feature Size Scaling [1]

Continuous improvements in process technology and photolithographic techniques have made the fabrication of metal–oxide–semiconductor (MOS) transistors of smaller and smaller dimensions to provide a higher packaging density. As a reduction in feature size reduces the gate capacitance, this leads to an improvement in performance. This has opened up the possibility of scaling device feature sizes to compensate for the loss in performance due to voltage scaling. The reduction of the size, i.e., the dimensions of metal–oxide–semiconductor field-effect transistors (MOSFETs), is commonly referred to as scaling. To characterize the process of scaling, a parameter $S$, known as scaling factor, is commonly used. All horizontal and vertical dimensions are divided by this scaling factor, $S > 1$, to get the dimensions of the devices of the new generation technology. Obviously, the extent of scaling, in other words the value of $S$, is decided by the minimum feature size of the prevalent technology. It has been observed that over a period of every 2 to 3 years, a new generation technology is introduced by downsizing the device dimensions by a factor of $S$, lying in the range 1.2–1.5. Table 7.1 presents the recent history of device size scaling. If the trend continues, according to International Technology Roadmap for Semiconductors (ITRS), the feature size will be 8 nm by the year 2018. The trend in MOS device scaling is represented by the curves shown in Fig. 7.2. It may be noted that the slope of all the curves in this figure is equal to the scaling parameter $S$. Figure 7.2a and b shows the reduction in gate delay for the n-type MOS (nMOS) and p-type MOS (pMOS) transistor, respectively. Figure 7.2c shows how gate oxide thickness varies with the scaling of channel length, whereas Fig. 7.2d shows how the supply voltage is scaled with the scaling of channel length.

Figure 7.3 shows the basic geometry of an MOSFET and the various parameters scaled by a scaling factor $S$. It may be noted that all the three dimensions are proportionally reduced along with a corresponding increase in doping densities. There are two basic approaches of device size scaling—constant-field scaling and constant-voltage scaling. In constant-field scaling, which is also known as full scaling, the supply voltage is also scaled to maintain the electric fields same as the previous generation technology as shown in Fig. 7.2d. In this section, we examine, in detail, both the scaling strategies and their effect on the vital parameters of an MOSFET.

### 7.2.1 Constant-Field Scaling

In this approach, the magnitudes of all the internal electric fields within the device are preserved, while the dimensions are scaled down by a factor of $S$. This requires...
Fig. 7.2 Trends in metal–oxide–semiconductor (MOS) device scaling

Fig. 7.3 Scaling of a typical metal–oxide–semiconductor field-effect transistors (MOSFET) by a scaling factor $S$

that all potentials must be scaled down by the same factor. Accordingly, supply and threshold voltages are scaled down proportionately. This also dictates that the doping densities are to be increased by a factor of $S$ to preserve the field conditions. A list of scaling factors for all device dimensions, potentials, and doping densities are given in Table 7.2.
As a consequence of scaling, various electrical parameters are affected. For example, the gate oxide capacitance per unit area increases by a factor of $S$ as given by the following relationship:

$$C' = \frac{\varepsilon_{ox}}{t_{ox}} = S \cdot C_{ox}. \quad (7.3)$$

As both length and width parameters are scaled down by the same factor, the $W/L$ remains unchanged. So, the transconductance parameter $K_n$ is also scaled by a factor $S$. Both linear-mode and saturation-mode drain currents are reduced by a factor of $S$, as given below:

$$I'_{ds} (lin) = \frac{K_n'}{2} \left( 2(V_{gs} - V) \cdot V'_{ds} - V^2_{ds} \right)$$

$$= \frac{SK_n}{2} \cdot \frac{1}{S^2} \left( 2(V_{gs} - V) V_{ds} - V^2_{ds} \right)$$

$$= \frac{I_D (lin)}{S}. \quad (7.6)$$

As both supply voltage and the drain current are scaled down by a factor of $S$, the power dissipation is reduced by a factor of $S^2$. This significant reduction in power dissipation is the most attractive feature of the constant-field scaling approach:

$$P = I_{ds} \cdot V_{ds}$$

$$P' = I'_{ds} \cdot V'_{ds} = \frac{1}{S^2} \cdot \frac{P}{S^2} \quad (7.7)$$

Another important parameter is the power density per unit area, which remains unchanged because the area and power dissipation, both are reduced by a factor of $S^2$. As the gate oxide capacitance reduces by a factor of $S$, there will be a reduction in both the rise-time and fall-time of the device. This leads to the reduction of delay.
Table 7.3 Effects of constant-field scaling on the key device parameters

<table>
<thead>
<tr>
<th>Quality</th>
<th>Before scaling</th>
<th>After scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gate capacitance</td>
<td>$C_g$</td>
<td>$C'_g = C_g / S$</td>
</tr>
<tr>
<td>Drain current</td>
<td>$I_D$</td>
<td>$I'_D = I_D / S$</td>
</tr>
<tr>
<td>Power dissipation</td>
<td>$P$</td>
<td>$P' = P / S^2$</td>
</tr>
<tr>
<td>Power density</td>
<td>$P/\text{area}$</td>
<td>$P'/\text{area}' = (P/\text{area})$</td>
</tr>
<tr>
<td>Delay</td>
<td>$t_d$</td>
<td>$t'_d = t_d / S$</td>
</tr>
<tr>
<td>Energy</td>
<td>$E = Pt_d$</td>
<td>$E' = \frac{P}{S^2} \cdot \frac{t_d}{S} \cdot \frac{P \cdot t_d}{S^3} \cdot \frac{1}{E}$</td>
</tr>
</tbody>
</table>


time and the consequent improvement in performance. Effects of constant-field scaling on the key device parameters are shown in Table 7.3:

$$C'_g = W' \cdot L' \cdot C'_{\text{ox}} = W \cdot L \cdot \frac{C_g}{S} = \frac{C_{\text{ox}}}{S} \quad (7.8)$$

Important benefits of constant-field scaling are: (i) smaller device sizes leading to a reduced chip size, higher yield, and more number of integrated circuits (ICs) per wafer, (ii) higher speed of operation due to smaller delay, and (iii) reduced power consumption because of the smaller supply voltage and device currents.

**Constant-Voltage Scaling**

In constant-voltage scaling, all the device dimensions are scaled down by a factor of $S$ just like constant-voltage scaling. However, in many situations, scaling of supply voltage may not be feasible in practice. For example, if the supply voltage of a central processing unit (CPU) is scaled down to minimize power dissipation, it leads to electrical compatibility with peripheral devices, which usually operate at higher supply voltages. It may be necessary to use multiple supply voltages and complicated-level translators to resolve this problem. In such situations, constant-voltage scaling may be preferred. In a constant-voltage scaling approach, power supply voltage and the threshold voltage of the device remain unchanged. To preserve the charge–field relations, however, the doping densities have to be scaled by a factor of $S^2$. Key device dimensions, voltages, and doping densities for constant-voltage scaling are shown in Table 7.4.

Constant-voltage scaling results in an increase in drain current (both in linear mode and in saturation mode) by a factor of $S$. This, in turn, results in an increase in the power dissipation by a factor of $S$ and the power density by a factor of $S^3$, as shown in Table 7.5. As there is no decrease in delay, there is also no improvement in performance. This increase in power density by a factor of $S^3$ has possible adverse effects on reliability such as electromigration, hot-carrier degradation, oxide breakdown, and electrical overstress.
Table 7.4 Constant-voltage scaling of the device dimensions, voltages, and doping densities

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Before scaling</th>
<th>After scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel length</td>
<td>L</td>
<td>L' = L / S</td>
</tr>
<tr>
<td>Channel width</td>
<td>W</td>
<td>W' = W / S</td>
</tr>
<tr>
<td>Gate oxide thickness</td>
<td>t_{ox}</td>
<td>t'<em>{ox} = t</em>{ox} / S</td>
</tr>
<tr>
<td>Junction depth</td>
<td>x_j</td>
<td>x_j / S</td>
</tr>
<tr>
<td>Power supply voltage</td>
<td>V_{dd}</td>
<td>V'<em>{dd} = V</em>{dd}</td>
</tr>
<tr>
<td>Threshold voltage</td>
<td>V_{to}</td>
<td>V'<em>{to} = V</em>{to}</td>
</tr>
<tr>
<td>Doping densities</td>
<td>N_A</td>
<td>N'_A = N_A \cdot S \cdot 2</td>
</tr>
<tr>
<td></td>
<td>N_D</td>
<td>N'_D = N_D \cdot S \cdot 2</td>
</tr>
</tbody>
</table>

Table 7.5 Effects of constant-voltage scaling on the key device parameters

<table>
<thead>
<tr>
<th>Quality</th>
<th>Before scaling</th>
<th>After scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gate capacitance</td>
<td>C_g</td>
<td>C'_g = C_g / S</td>
</tr>
<tr>
<td>Drain current</td>
<td>I_D</td>
<td>I'_D = I_D \cdot S</td>
</tr>
<tr>
<td>Power dissipation</td>
<td>P</td>
<td>P' = P \cdot S</td>
</tr>
<tr>
<td>Power density</td>
<td>P/area</td>
<td>P' / area' = S^3 P/area</td>
</tr>
<tr>
<td>Delay ((t_d \propto C_g \cdot V_{dd} / I_{ds}))</td>
<td>t_d</td>
<td>t'_d = t_d / S^2</td>
</tr>
</tbody>
</table>

\[
I'(\text{lin}) = S \cdot I(\text{lin}) \\
I'_{\text{sat}} = S \cdot I_D(\text{sat}),
\]

\[
P' = I'_0 \cdot V_{ds'} = (S I_0) V_{ds} = S \cdot P.
\]

**Short-Channel Effects**

However, associated with the benefits mentioned above, scaling has some unwanted side effects, commonly referred to as **short-channel effects**, which has been discussed at length in Sect. 2.6. Short-channel effects arise when channel length is of the same order of magnitude as depletion region thickness of the source and drain junctions or when the length is approximately equal to the source and drain junction depths. As the channel length is reduced below 0.6 μm, the short-channel effect starts manifesting. This leads to an increase in subthreshold leakage current, reduction in threshold voltage with \( V_{gs} \), and a linear increase in the saturation current instead of square of the gate-to-source voltage.

Moreover, if channel length is scaled down without scaling the supply voltage (constant-voltage scaling), the electric field across a gate oxide device continues to increase, creating a hot carrier. Hot carriers can cause an avalanche breakdown of the gate oxide. It is necessary to restrict the maximum electric field across the gate oxide to 7 MV/cm, which translates into 0.7 V/10 Å of gate oxide thickness. For
gate oxide thickness of 70 Å, the applied gate voltage should be limited to 4.9 V for long-term reliable operation.

From the above discussion, it is evident that voltage scaling and scaling down of the device feature size are complementary to each other. However, if voltage scaling is not done along with the scaling of feature size because of some other design constraints, it will be necessary to use an appropriate measure to reduce the number of hot carriers. One technique is to use lightly doped drain structure shown in Fig. 7.4. The physical device structure is modified so that the carriers do not gain energy from the field to become hot carriers. Of course, the performance of the device is traded to obtain long-term reliability.

7.3 Architectural-Level Approaches

Architectural-level refers to register-transfer-level (RTL), where a circuit is represented in terms of building blocks such as adders, multipliers, read-only memories (ROMs), register files, etc. [2, 3]. High-level synthesis technique transforms a behavioral-level specification to an RTL-level realization. It is envisaged that low-power synthesis technique on the architectural level can have a greater impact than that of gate-level approaches. Possible architectural approaches are: parallelism, pipelining, and power management, as discussed in the following subsections.

7.3.1 Parallelism for Low Power

Parallel processing is traditionally used for the improvement of performance at the expense of a larger chip area and higher power dissipation. Basic idea is to use multiple copies of hardware resources, such as arithmetic logic units (ALUs) and processors, to operate in parallel to provide a higher performance. Instead of using parallel processing for improving performance, it can also be used to reduce power.

We know that supply voltage scaling is the most effective way to reduce power consumption. Unfortunately, the savings in power come at the expense of performances or, more precisely, maximum operating frequency. This follows from the equation:

$$f_{\text{max}} \propto \frac{(V - V_t)^2}{V \left(1 - \frac{V}{V_{dd}}\right)}$$

(7.11)
Fig. 7.5  a  A 16-bit adder; b parallel architecture of the 16-bit adder. *MUX* multiplexer

If the threshold voltage is scaled by the same factor as the supply voltage, the maximum frequency of operation is roughly linearly dependent on the power supply voltage. Reducing the supply voltage forces the circuit to operate at a lower frequency. In simple terms, if the supply voltage is reduced by half, the power is reduced by one fourth and performance is lowered by half. The loss in performance can be compensated by parallel processing. This involves splitting the computation into two independent tasks running in parallel. This has the potential to reduce the power by half without reduction in the performance. Here, the basic approach is to trade the area for power while maintaining the same throughput.

To illustrate this, let us consider the example of Fig. 7.5a, where two 16-bit registers supply two operands to a $16 \times 16$ adder. This is considered as the reference
Table 7.6 Impact of parallelism on area, power, and throughput

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Without $V_{dd}$ scaling</th>
<th>With $V_{dd}$ scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>2.2X</td>
<td>2.2X</td>
</tr>
<tr>
<td>Power</td>
<td>2.2X</td>
<td>0.227X</td>
</tr>
<tr>
<td>Throughput</td>
<td>2X</td>
<td>1X</td>
</tr>
</tbody>
</table>

architecture and all the parameters, such as power supply voltage, frequency of operation, power dissipation, etc., of this architecture are referred by ref notation. If 10 ns is the delay of the critical path of the adder, then the maximum clock frequency that can be applied to the registers is $f_{ref} = 100$MHz. In this situation, the estimated dynamic power of the circuit is:

$$P = C_{ref} V_{ref}^2 \cdot f_{ref}$$

(7.12)

where $C_{ref}$ is the total effective switched capacitance, which is the sum of the products of the switching activities with the node capacitances, that is,

$$C_{ref} = \sum C_i \alpha_i.$$  

(7.13)

Without reducing the clock frequency, the power dissipation cannot be reduced by reducing the supply voltage. However, same throughput (number of operations per unit time) can be maintained by the parallel architecture shown in Fig. 7.5b. Here, the adder has been duplicated twice, but the input registers have been clocked at half the frequency of $f_{ref}$. This helps to reduce the supply voltage such that the critical path delay is not more than 20 ns. With the same $16 \times 16$ adder, the power supply can be reduced to about half the $V_{ref}$. Because of duplication of the adder, the capacitance increases by a factor of two. However, because of extra routing to both the adders, the effective capacitance would be about 2.2 times of $C_{ref}$.

Therefore, the estimated power dissipation of this parallel implementation is:

$$P_{par} = 2.2 C_{ref} \left( \frac{V_{ref}}{2} \right)^2 \cdot f_{ref}$$

(7.14)

$$\approx \frac{2.2}{8} \cdot P_{ref} \approx 0.277 P_{ref}.$$  

(7.15)

This shows that the power dissipation reduces significantly. The impact of parallelism is highlighted in Table 7.6. Column 2 shows parallelism for higher performance without voltage scaling having larger power dissipation, whereas column 3 corresponds to parallelism for low power with voltage scaling and without degradation of performance.
Table 7.7 Power in multi-core architecture

<table>
<thead>
<tr>
<th>Number of cores</th>
<th>Clock in MHz</th>
<th>Core supply voltage</th>
<th>Total power</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>5</td>
<td>15.0</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>3.6</td>
<td>8.94</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>2.7</td>
<td>5.20</td>
</tr>
<tr>
<td>8</td>
<td>25</td>
<td>2.1</td>
<td>4.5</td>
</tr>
</tbody>
</table>

7.3.2 Multi-Core for Low Power

The idea behind the parallelism for low power can be extended for the realization of multi-core architecture. Figure 7.6 shows a four-core multiplier architecture. Table 7.7 shows how the clock frequency can be reduced with commensurate scaling of the supply voltage as the number of cores is increased from one to four while maintaining the same throughput. This is the basis of the present-day multi-core commercial processors introduced by Intel, AMD, and other processor manufacturers. Thread-level parallelism is exploited in multi-core architectures to increase throughput of the processors.
Fig. 7.7 Pipelined realization 16-bit adder

Table 7.8 Impact of pipelining on area, power, and throughput

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Without $V_{dd}$ scaling</th>
<th>With $V_{dd}$ scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>1.15X</td>
<td>1.15X</td>
</tr>
<tr>
<td>Power</td>
<td>2.30X</td>
<td>0.28X</td>
</tr>
<tr>
<td>Throughput</td>
<td>2X</td>
<td>1X</td>
</tr>
</tbody>
</table>

Pipelining for Low Power

Instead of reducing the clock frequency, in pipelined approach, the delay through the critical path of the functional unit is reduced such that the supply voltage can be reduced to minimize the power. As an example, consider the pipelined realization of 16-bit adder using two-stage pipeline shown in Fig. 7.7. In this realization, instead of 16-bit addition, 8-bit addition is performed in each stage. The critical path delay through the 8-bit adder stage is about half that of 16-bit adder stage. Therefore, the 8-bit adder will operate at a clock frequency of 100 MHz with a reduced power supply voltage of $V_{ref}/2$. It may be noted that in this realization, the area penalty is much less than the parallel implementation leading to $C_{pipe} = 1.15C_{ref}$. Substituting these values, we get:

$$P_{pipe} = C_{pipe} \cdot V_{ref}^2 \cdot f = (1.15C_{ref}) \cdot \left(\frac{V_{ref}}{2}\right)^2 \cdot f = 0.28P_{ref}. \quad (7.16)$$

It is evident that the power reduction is very close to that of a parallel implementation with an additional bonus of a reduced area overhead. The impact of pipelining is highlighted in Table 7.8. Here, column 2 shows pipelining for improved performance with larger power dissipation, higher clock frequency, and without voltage scaling, whereas column 3 corresponds to parallelism for low power with voltage scaling and without degradation of performance.
Combining Parallelism with Pipelining

An obvious extension of the previous two approaches is to combine the parallelism with pipelining. Here, more than one parallel structure is used and each structure is pipelined. Figure 7.8 shows the realization of a 16-bit adder by combining both pipelining and parallelism. Two pipelined 16-bit adders have been used in parallel. Both power supply and frequency of operation are reduced to achieve substantial overall reduction in power dissipation:

\[ P_{\text{parpipe}} = C_{\text{parpipe}} \cdot V_{\text{ref}}^2 \cdot f_{\text{ref}/2} \]  
(7.17)

The effective switching capacitance \( C_{\text{parpipe}} \) will be more than the previous because of the duplication of functional units and more number of latches. It is assumed to be equal to 2.5 \( C_{\text{ref}} \). The supply voltage can be more aggressively reduced to about one quarter of \( V_{\text{ref}} \) and the frequency of operation is reduced to half the reference frequency \( f_{\text{ref}/2} \). Thus,

\[ P_{\text{parpipe}} = (2.5 \cdot C_{\text{ref}} \cdot (0.3V_{\text{ref}})^2 \cdot \left(\frac{f_{\text{ref}}}{2}\right)^{-1} \]
(7.18)

\[ P_{\text{parpipe}} = 0.1125P_{\text{ref}}. \]

Table 7.9 highlights the impact of combined parallelism and pipelining. Without \( V_{\text{dd}} \) scaling, the performance can be improved by four times with an increase in power
Table 7.9 Impact of parallelism and pipelining on area, power, and throughput

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Without $V_{dd}$ scaling</th>
<th>With $V_{dd}$ scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>2.5×</td>
<td>2.5×</td>
</tr>
<tr>
<td>Power</td>
<td>5.0×</td>
<td>0.1125×</td>
</tr>
<tr>
<td>Throughput</td>
<td>4×</td>
<td>1×</td>
</tr>
</tbody>
</table>

Fig. 7.9  
(a) A first-order infinite impulse response (IIR) filter; 
(b) directed acyclic graph (DAG) corresponding to the IIR filter

dissipation by five times. On the other hand, with $V_{dd}$ scaling, the power dissipation can be reduced by about 90%, without the degradation of performance.

7.4 Voltage Scaling Using High-Level Transformations

For automated synthesis of digital systems, high-level transformations such as dead code elimination, common sub-expression elimination, constant folding, in-line expansion, and loop unrolling are typically used to optimize the design parameters such as the area and throughput [4]. These high-level transformations can also be used to reduce the power consumption either by reducing the supply voltage or the switched capacitance. In this section, we discuss how loop unrolling can be used to minimize power by voltage scaling. Let us consider a first-order infinite impulse response (IIR) filter as shown in Fig. 7.9a, which can be modeled by the following equation:

$$Y_N = X_N + K \times Y_{N-1}. \tag{7.19}$$

In a high-level synthesis, the behavioral specification is commonly transformed into an intermediate graph-based representation known as directed acyclic graph (DAG). The DAG corresponding to Fig. 7.9a is shown in Fig. 7.9b.

If we simply unroll the loop to process two samples in parallel, the effective critical path remains unchanged as shown in Fig. 7.10. As the switched capacitance is also doubled, the power consumption per sample remains unchanged for
the same supply voltage. Normalized throughput and power dissipation are shown in Fig. 7.10.

However, loop unrolling facilitates other transformations, such as distributivity, constant propagation, and pipelining, to be readily applied. For example, after applying distributivity and constant propagation, the output samples can be represented by the following equations:

\[
Y_{N-1} = X_{N-1} + K \times Y_{N-2}
\]
\[
Y_N = X_N + K \times Y_{N-1} = X + K \times X_{N-1} + K^2 \times Y_{N-2}.
\] (7.20)

The DAG corresponding to these equations are shown in Fig. 7.11. It may be noted that the critical path is now reduced to 3, i.e., the computation of two samples can be...
performed in three clock cycles as opposed to four-clock cycles in case of Fig. 7.10. To maintain the same throughput level, the clock frequency can be reduced by 25\% with a corresponding reduction in the supply voltage from 5 to 3.7 V. This leads to a reduction in power by about 20\%. Alternatively, the performance can be improved by 25\%, keeping the supply voltage and the operating frequency unchanged. This, however, results in an increase in normalized power dissipation by 50\% due to an increased switched capacitance. Furthermore, by applying pipelining to this structure, computation can be performed in two stages.

Stage 1:

\[ Z = X_N + K \times X_{N-1}. \]

Stage 2:

\[
\begin{align*}
Y_{N-1} &= X_{N-1} + K \times Y_{N-2} \\
Y &= Z + K^2 \times Y_{N-2}. \tag{7.21}
\end{align*}
\]

The corresponding DAG is shown in Fig. 7.12. It may be noted that the critical path of both the stages is now reduced to two. To maintain the throughput level, the sample rate can be reduced to half with a corresponding reduction of the supply voltage to 2.9 V. This results in the reduction in power by 50\%. Alternatively, the throughput can be doubled by keeping the supply voltage and sample rate same as in the case of Fig. 7.9b. This, however, increases the normalized power dissipation by about 50\% because of the increase in effective capacitance by 50\%.

This simple example highlights the point that a particular transformation may have conflicting effects on the different parameters of power dissipation. In this case, loop unrolling leads to reduction of supply voltage but increase in effective capacitance. However, the reduction in the supply voltage more than compensates for the increase in capacitance resulting in overall reduction in power dissipation.

In the above example, an unrolling factor of 2 (two samples are computed) has been considered to demonstrate the effectiveness of the loop unrolling for the

<table>
<thead>
<tr>
<th>Vdd</th>
<th>5V</th>
<th>2.9V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>2X</td>
<td>1X</td>
</tr>
<tr>
<td>Power</td>
<td>1.5X</td>
<td>0.5X</td>
</tr>
</tbody>
</table>

Fig. 7.12 Directed acyclic graph (DAG) after unrolling and pipelining
Fig. 7.13 Speed optimization is different than power optimization

Fig. 7.14 Assignment of multiple supply voltages based on delay on the critical path

reduction of power dissipation. As the unrolling factor is increased, the speedup grows linearly (for the same supply voltage) and the capacitance also grows linearly. If the supply voltage is scaled to minimize power dissipation, then the power dissipation starts increasing with the unrolling factor after giving an optimum power dissipation for the unrolling factor of about 3, as shown in Fig. 7.13.

**Multilevel Voltage Scaling**

As high $V_{dd}$ gates have a less delay, but higher dynamic and static power dissipation, devices on time-critical paths can be assigned higher $V_{dd}$, while devices on noncritical paths shall be assigned lower $V_{dd}$, such that the total power consumption can be reduced without degrading the overall circuit performance [5]. Figure 7.14 shows that the delay on the critical path is 10 ns, whereas delay on the noncritical path is 8 ns. The gates on the critical path are assigned higher supply voltage $V_{ddH}$. The slack of the noncritical path can be traded for lower switching power by assigning lower supply voltage $V_{ddL}$ to the gates on the noncritical path.
For multiple dual-$V_{dd}$ designs, the voltage islands can be generated at different levels of granularity, such as macro level and standard cell level. In the standard-cell level, gates on the critical path and noncritical paths are clustered into two groups. Gates on the critical path are selected from the higher supply voltage ($V_{ddH}$) standard cell library, whereas gates on the noncritical path are selected from the lower supply voltage ($V_{ddL}$) standard cell library, as shown in Fig. 7.15. This approach modifies the normal distribution of path delays using a single supply voltage in a design to a distribution of path delays skewed toward higher delay with multiple supply voltages, as shown in Fig. 7.16.

Macro-based voltage island methodology targets toward an entire macro or functional block to be assigned to operate at different voltages at the time of high-level synthesis. Starting with an intermediate representation known as DAG, high-level synthesis involves two basic steps: scheduling and allocation. These are the two most important steps in the synthesis process. Scheduling refers to the allocation of the operations to the control steps. A control step is the fundamental sequencing unit in a digital system. It corresponds to a clock cycle. Allocation involves assigning the operations to the hardware, i.e., allocating functional units, storage, and communication paths. After the scheduling step, some operations will be on the critical path, whereas some operations will be on the noncritical path. The slack of the off-critical path can be utilized for the allocation of macro modules to off-critical-path operations. As shown in Fig. 7.17, a multiplier operating at lower supply voltage ($V_{ddL}$) is allocated to the off-critical-path multiplier operation *3.
A number of studies have shown that the use of multiple supply voltages results in the reduction of dynamic power from less than 10% to about 50%, with an average of about 40%. It is possible to use more than two, say three or four, supply voltages. However, the benefit of using multiple $V_{dd}$ saturates quickly. Extending the approach to more than two supply voltages yields only a small incremental benefit. The major gain is obtained by moving from a single $V_{dd}$ to a dual $V_{dd}$. It has been found that in a dual-$V_{dd}$/single-$V_t$ system, the optimal lower $V_{dd}$ is about 60–70% of the original $V_{dd}$. The optimal supply voltage depends on the threshold voltage $V_t$ of the MOS transistors as well.

**Challenges in MVS**

The overhead involved with multiple-$V_{dd}$ systems includes the additional power supply networks, insertion of level converters, complex characterization and static timing analysis, complex floor planning and routing, and power-up–power-down sequencing. As a consequence, even a simple multi-voltage design presents the designer with a number of challenges, which are highlighted in this section.
Voltage Scaling Interfaces

When signals go from one voltage domain to another voltage domain, quite often, it is necessary to insert level converters or shifter that convert the signals of one voltage level to another voltage level. Consider a signal going from a low-\(V_{\text{dd}}\) domain to a high-\(V_{\text{dd}}\) domain, as shown in Fig. 7.18. A high-level output from the low-\(V_{\text{dd}}\) domain has an output \(V_{\text{ddL}}\), which may turn on both nMOS and pMOS transistors of the high-\(V_{\text{dd}}\) domain inverter resulting in a short circuit between \(V_{\text{ddH}}\) and the GND. A level converter needs to be inserted to avoid this static power consumption. Moreover, to avoid the significant rise and fall time degradations between the voltage-domain boundaries, it is necessary to insert buffers to improve the quality of signals that go from one domain to another domain with proper voltage swing and rise and fall times. So, it may be necessary to insert buffers even when signals go from high- to low-voltage domain. This approach of clean interfacing helps to maintain the timing characteristics and improves ease of reuse.

**High-to-Low-Voltage Level Converters** The need for a level converter as a signal passes from high-\(V_{\text{dd}}\) domain to low-\(V_{\text{dd}}\) domain arises primarily to provide a clean signal having a desired voltage swing and rise and fall times. Without a level converter, the voltage swing of the signal reaching the low-\(V_{\text{dd}}\) domain is 0 to \(V_{\text{ddH}}\). This causes higher switching power dissipation and high leakage power dissipation due to GIDL effect. Moreover, because of the longer wire length between the voltage domains, the rise and fall time may be long leading to increase in short-circuit power dissipation. To overcome this problem, a level converter as shown in Fig. 7.19b may be inserted. The high-to-low level converter is essentially two inverter stages in cascade. It introduces a buffer delay and its impact on the static timing analysis is small.

**Low-to-High-Voltage Level Converters** Driving logic signals from a low-voltage domain to high-voltage domain is a more critical problem because it has significant...
degrading effect on the operation of a circuit. The logic symbol of the low-to-high-voltage level converter is shown in Fig. 7.20a. One of the most common approaches of converting a low-voltage level signal to a high-voltage level signal is using dual cascode voltage switch logic (DCVSL) as shown in Fig. 7.20b. The circuit combines two concepts: differential logic and positive feedback. Two inverter operating in the low-voltage domain provide inputs to two nMOS transistors that form the differential inputs. When the signal coming from the low-voltage domain is of low logic level, the output of the first inverter is high and the output of the second inverter is low. The high-level output of the first inverter turns the nMOS transistor M1 ON, which, in turn, turns the pMOS transistor M4 ON. As M4 turns ON, the pMOS transistor M3 is turned OFF because of the positive feedback implemented using a cross-coupled connection. This produces a low-level output at the output of the inverter (OUTH). Similarly, when the input INL is of high logic level, the output of the second inverter is high, which turns the nMOS transistor M2 ON. This sets in a regenerative action to quickly switch the OUTH output to high logic level ($V_{ddH}$). The circuit requires both $V_{ddL}$ and $V_{ddH}$ supply voltages for its operation. The low-to-high level converters introduce considerably larger delay compared to the high-to-low level converters discussed above. Moreover, these level converters are to be characterized over an extended voltage range to match different voltage domains of the low and high side of voltage domains for accurate static timing analysis.

**Converter Placement**

One important design decision in the voltage scaling interfaces is the placement of converters. As the high-to-low level converters use low-$V_{dd}$ voltage rail, it is appropriate to place them in the receiving or destination domain, that is, in the low-$V_{dd}$ domain. This not only avoids the routing of the of the low-$V_{dd}$ supply rail from the low-$V_{dd}$ domain to the high-$V_{dd}$ domain but also helps in the improvement of the rise and fall time of the signals in low-$V_{dd}$ domain. Placement of high-to-low level converter is shown in Fig. 7.21a. It is also recommended to place the low-to-
Fig. 7.21  a High-to-low converter placement;  b low-to-high converter placement

high level converters in the receiving domain, that is, in the high-\(V_{\text{dd}}\) domain. This, however, involves routing of the low-\(V_{\text{dd}}\) supply rail to the high-\(V_{\text{dd}}\) domain. As the low-to-high level converters require both low- and high-\(V_{\text{dd}}\) supply rails, at least one of the supply rails needs to be routed from one domain to the other domain. The placement of the low-to-high level converter is shown in Fig. 7.21b.

**Floor Planning, Routing, and Placement**

The multiple power domain demands multiple power grid structure and a suitable power distribution among them. As a consequence, it requires more careful floor planning, placement, and routing. It is necessary to separate low-\(V_{\text{dd}}\) and high-\(V_{\text{dd}}\) cells because they have different n-well voltages. Typically, a row-by-row separation, as shown in Fig. 7.22 is used. This type of layout provides high performance and is suitable for standard-cell and gate arrays.

**Static Timing Analysis**

Use of single voltage makes static timing analysis simpler because it can be performed for single performance point based on the characterized libraries. Electronic design automation (EDA) tools can optimize the design for worst-case process, voltage, and temperature (PVT) conditions. On the other hand, in the case of multi-voltage designs, libraries should be characterized for different voltage levels that are used in the design. EDA tool has to optimize individual blocks or subsystems and also multiple voltage domains.

**Power-Up and Power-Down Sequencing**

One of the important system-level issues is the power sequencing. To avoid power line glitches, it is not advisable to bring up all power supplies simultaneously. It is essential to plan a power sequence such that different domains can come up in a
Fig. 7.22 Placement and routing in multi-$V_{dd}$ design

well-defined order to satisfy the correct operation. Particularly, reset can be initiated only after all power domains are completely powered up. The ramp times are to be carefully controlled to avoid overshoot and undershoot on the supply lines.

**Clock Distribution**

As the clock needs to be distributed across different voltage domains, the clocks have to go through the level converters. This requires that the clock synthesis tool must understand the need for level converters and automatically insert them in appropriate places.

**Low-Voltage Swing**

As the signal is sent through a long interconnect wire having large capacitance, the voltage swing can be reduced to minimize the switching power dissipation. Power saving can be achieved based on the following two mechanisms:

- Charge needed to charge/discharge the capacitance is smaller.
- Driver size can be reduced because of smaller current requirement.
Figure 7.23 shows how low-voltage swing circuit can be realized using the low-to-high and high-to-low level converters discussed earlier. Before allowing a signal to pass through a long interconnect wire, it is converted into a low-voltage swing signal using a high-to-low level converter. At the end of the interconnect wire, the signal is translated back to high level using a low-to-high level converter as shown in Fig. 7.23. Although this approach has the potential to reduce the switching power by a significant amount, its efficacy is gradually decreasing in deep-submicron process technology due to the small supply voltage $V_{dd}$ and threshold voltage $V_t$.

### 7.7 Dynamic Voltage and Frequency Scaling

DVFS has emerged as a very effective technique to reduce CPU energy [6]. The technique is based on the observation that for most of the real-life applications, the workload of a processor varies significantly with time and the workload is bursty in nature for most of the applications. The energy drawn for the power supply, which is the integration of power over time, can be significantly reduced. This is particularly important for battery-powered portable systems.

#### 7.7.1 Basic Approach

The energy drawn from the power supply can be reduced by using the following two approaches:

**Dynamic Frequency Scaling** During periods of reduced activity, there is a scope to lower the operating frequency with varying workload keeping the supply voltage constant. As we know, digital CMOS circuits are used in a majority of microprocessors, and, for present-day digital CMOS, the sources of power dissipation can be summarized as follows:
Fig. 7.24 Energy versus workload. *DVFS* dynamic voltage and frequency scaling

\[ P_{\text{total}} = CV_d^2 f + (I_{\text{sub}} + I_{\text{diode}} + I_{\text{gate}})V, \]  

(7.22)

where the first component represents the switching power dissipation, the most dominant component of power dissipation, which is proportional to \( f \), the frequency of operation, \( C \), the switched capacitance \( \sum \alpha_i C_i \) of the circuit, and the square of the supply voltage. The static power dissipation is due to various leakage current components shown within the bracket in Eq. 7.22. By scaling frequency with varying workload, the power dissipation reduces linearly with the workload as shown in Fig. 7.24 by the solid line.

**Dynamic Voltage and Frequency Scaling** An alternative approach is to reduce the operating frequency along with the supply voltage without sacrificing the performance required at that instance. It has been established that CMOS circuits can operate over a certain voltage range with reasonable reliability, where frequency increases monotonically with the supply voltage. For a particular process technology, there is a maximum voltage limit beyond which the circuit operation is destructive. Similarly, there is a lower voltage limit below which the circuit operation is unreliable or the delay paths no longer vary monotonically. Within the reliable operating range, the delay increases monotonically with the decrease in supply voltage following Eq. 7.23. Therefore, the propagation delay restricts the clock frequency in a microprocessor:

\[ \text{Delay}(D) \propto \left( \frac{V_{\text{dd}}}{V_t} \right)^2 = \frac{1}{V_{\text{dd}}} \left( 1 - \frac{V_t}{V_{\text{dd}}} \right)^2 \]  

(7.23)
However, the processor can continue to operate at a lower frequency as the supply voltage is lowered. This has opened up the possibility of lowering the supply voltage, as the frequency is lowered due to the reduced workload. Because of the square law dependence of switching power dissipation on the supply voltage, there is a significant saving in energy. The goal of the DVFS technique is to adapt the supply voltage and the operating frequency of the processor to match the workload such that the performance loss of the processor is minimized, thereby saving maximum possible energy, as shown by the dotted line in Fig. 7.24.

Four different situations are illustrated in Fig. 7.25. Figure 7.25a shows the situation when the workload is 100%. Here, the processor is operating with the highest voltage and frequency and it is taking time $T_1$ to complete the task. As a consequence, it requires average power dissipation of $P_1$ and energy dissipation is $E_1 = (P_1 \times T_1)$. Now, when the workload is 50%, energy consumption is $E_2 = (P_1 \times T_2)$ without voltage and frequency scaling as shown in Fig. 7.25b. Since $T_2 = T_1/2$, $E_2 = E_1/2$. Now, let us consider the situation represented by Fig. 7.25c. Here, frequency scaling by 50% has been done such that execution time is again $T_1$. So, the energy consumption is $P_2 \times T_1 = E_2$, which is the same as the situation represented by Fig. 7.25b. Although energy consumption is the same, the power dissipation is reduced by 50%. Finally, consider the situation when the workload is 50%, as shown in Fig. 7.25d. In this case, both the voltage and frequency scaling are done and power dissipation is $P_3$, which is less than $P_2$. Here, the energy consumption is $E_3 = P_3 \times T_1$. As $P_3 = 0.25P_1$, there is a significant reduction in the energy consumption.
Table 7.10 Relationship between voltage, frequency, and power

<table>
<thead>
<tr>
<th>Frequency (f) MHz</th>
<th>Voltage $V_{dd}$</th>
<th>Relative power</th>
</tr>
</thead>
<tbody>
<tr>
<td>700</td>
<td>1.65</td>
<td>100</td>
</tr>
<tr>
<td>600</td>
<td>1.60</td>
<td>80.59</td>
</tr>
<tr>
<td>500</td>
<td>1.50</td>
<td>59.03</td>
</tr>
<tr>
<td>400</td>
<td>1.40</td>
<td>41.14</td>
</tr>
<tr>
<td>300</td>
<td>1.25</td>
<td>24.60</td>
</tr>
<tr>
<td>200</td>
<td>1.10</td>
<td>12.70</td>
</tr>
</tbody>
</table>

7.7.2 DVFS with Varying Work Load

So far, we have considered static conditions to establish the efficiency of DVFS approach. In real systems, however, the frequency and voltage are to be dynamically adjusted to match the changing demands for processing power. The implementation of the DVFS system will require the following hardware building blocks:

- Variable voltage processor $\mu(r)$
- Variable voltage generator $V(r)$
- Variable frequency generator $f(r)$

In addition to the above hardware building blocks, there is the need of a workload monitor. The workload monitor can be performed by the operating system (OS). Usually, the OS has no prior knowledge of the workload to be generated by a bursty application. In general, the future workloads are nondeterministic. As a consequence, predicting the future workload from the current situation is very difficult and errors in prediction can seriously reduce the gains of DVFS, which has been observed in several simulation studies. Moreover, the rate at which the DVFS is done has a significant bearing on the performance and energy. So, it is essential to develop suitable strategy for workload prediction and the rate of DVFS, such that processor utilization and energy saving are maximized.

7.7.2.1 Variable Voltage Processor $\mu(r)$

The need of a processor which can operate over a frequency range with a corresponding lower supply voltage range can be manufactured using the present-day process technology and several such processors are commercially available. Transmeta’s TM 5400 or “Crusoe” processor and Strong ARM processor are examples of such variable voltage processors. Transmeta’s Crusoe processor can operate over the voltage range of 1.65–1.1 V, with the corresponding frequency range of 700–200 MHz. Table 7.10 provides the relation among frequency, voltage, and power consumption for this processor. It allows the following adjustments:

- Frequency change in steps of 33 MHz
- Voltage change in steps of 25 mV
- Up to 200 frequency/voltage change per second
7.7 Dynamic Voltage and Frequency Scaling

**Fig. 7.26** Processor-voltage versus clock frequency of Strong ARM processor. CPU central processing unit

Another processor, Strong ARM, also allows voltage scaling. Experimental results performed on Strong ARM 1100 are shown in Fig. 7.26. The diagram shows the operating frequency and corresponding supply voltages for the ARM processor.

**Variable Voltage Generator V(r)**

The variable voltage power supply can be realized with the help of a direct current (DC)-to-DC converter, which receives a fixed voltage $V_F$ and generates a variable voltage $V_r$ based on the input from the workload monitoring system. Block diagram of a DC-to-DC converter based on pulse-width modulation (PWM) is shown in Fig. 7.27. The fixed voltage $V_F$ passes through a power switch, which generates a rectangular wave of variable duty cycle. The pulse-width modulator controls the duty cycle of the rectangular wave based on the input received from the comparator. The low-pass filter removes the alternating current (AC) ripple to the acceptable level and generates an average value of the PWM signal. The average value is nearly equal to the derived DC output voltage. With the availability of good quality components such as integrated gate bipolar junction transistor (IGBT) and other semiconductor devices, it is possible to realize DC-to-DC converters having high efficiency, say 90% and above. However, the efficiency drops off as the load decreases as shown in Fig. 7.28. At a lower current load, most of the power drawn
from the fixed voltage supply gets dissipated in the switch and the efficiency falls off. This also leads to a reduction in the efficiency of the dynamic voltage scaling (DVS) system.

**Variable Frequency Generator f(r)**

The variable frequency is generated with the help of a phase lock loop (PLL) system. The heart of the device is the high-performance PLL-core, consisting of a phase frequency detector (PFD), programmable on-chip filter, and voltage-controlled oscillator (VCO). The PLL generates a high-speed clock which drives a frequency divider. The divider generates the variable frequency \( f(r) \). The PLL and the divider together generate the independent frequencies related to the PLL operating frequency.

**The Model**

A generic block diagram of a variable voltage processing system is shown in Fig. 7.29. The tasks generated from various sources are represented by the task queue. Each of the sources produce events at an average rate of \( \lambda_k \), \( k = 1, 2, \ldots, n \). The task scheduler of the OS manages all these tasks and decides which process should run on the processor. The average rate of arrival of tasks at the processor is \( \lambda = \lambda_k \). The processor, in turn, provides time varying processing rate \( \mu(r) \). The OS kernel measures the CPU utilization over some observation frames and estimates the current workload \( w \). The workload monitor predicts the processing rate \( r \) based on \( w \) and a history of workloads in the previous frames. The predicted workload \( r \), in turn, is used to generate the frequency of operation \( f(r) \) and the power supply voltage \( V(r) \) for the next observation time slot. Predicting the future workload from the current situation is extremely difficult and errors can seriously reduce the gain of DVS approach.
Workload Prediction

It is assumed that the workload for the next observation interval can be predicted based on the workload statistics of the previous $N$ intervals. The workload prediction for $(n+1)$ interval can be represented by

$$W_p[n+1] = \sum_{k=0}^{N-1} h_n[k] W(n-k),$$

(7.24)

where $W[n]$ denotes the average normalized workload in the interval $(n-1)T \leq t \leq nT$ and $h_n[k]$ represents an $N$-tap, adaptable finite impulse response (FIR) filter, whose coefficients are updated in every observation interval based on the difference between the predicted and actual workloads. Uses of three possible filter types are examined below:

Moving Average Workload (MAW) In this case $h_n[k] = 1/N$, that is the filter predicts the workload in the next time slot as the average of the previous $N$ times slots. This simplistic scheme removes high-frequency workload changes and does not provide a satisfactory result for the time-varying workload statistics.

Exponential Weighted Averages (EWA) In this approach, instead of giving equal weightage to all the workload values of the previous slots, higher weightages are given to the most recent workload history. The idea is to give progressively decreasing importance to historical data. Mathematically, this can be achieved by providing the filter coefficients $h_n[k] = a^{-k}$, for all $n$, where a positive value of $a$ is chosen such that $\sum h_n[k] = 1$.
Least Mean Square In this approach, the filter coefficients are modified based on the prediction error. One of the popular adaptive filter algorithms is the least-mean-square (LMS) algorithm, where $w[n]$ and $w_p[n]$ are the actual workload and predicted workload, respectively. Then the prediction error is given by $W_e[n] = W[n] - W_e[n]$. The filter coefficients are updated based on the following rule:

$$h_n[k] + 1[k] = h_n[k] + \mu W_e[n - k],$$

where $\mu$ is the step size.

Although these types of filters learn from the history of workloads, the main problem is the stability and convergence. Selection of a wrong number of coefficients or an inappropriate step size may lead to undesirable consequences.

A comparison of the prediction performance in terms of root-mean-square error of the three filters is given in Fig. 7.30. It shows that the prediction is too noisy for a small number of taps and when the number of taps is large, it leads to excessive low pass filtering. Both result in poor prediction. It is observed that the LMS adaptive filter outperforms the other techniques and an optimum result is obtained with $N = 3$ taps.

**Discrete Processing Rate**

The operating points are determined analytically, first by finding out appropriate clock frequencies for different workloads. As PLLs along with frequency dividers are used to generate different frequencies, it is preferable to select clock periods which are multiples of the PLL frequency. This helps to generate clock frequencies with minimum latency. Otherwise, it is necessary to change the PLL frequency, which requires a longer time to stabilize. Finally, the voltages required to support each of the frequencies are found out.

Too few operating points may result in the ramping between two levels for workload profiles and too many levels may result in hunting of the power supply for a new supply voltage, most of the time. To optimize the number of operating points, it is necessary to understand the number of distinct performance levels necessary under realistic dynamic workload conditions. Most of the processors available with built-in DVS facility have a discrete set of operating frequencies, which implies that
the processing rate levels are quantized. Question arises whether the quantization has any impact on the performance. The effect of the number of discrete processing level is shown in Fig. 7.31. As the number of discrete levels increases, the energy saving ratio gets close to the perfect prediction case. For \( L = 10 \), which is provided by all practical processors, the root mean square (RMS) (ESR) degradation due to quantization noise is less than 10%.

### 7.7.6 Latency Overhead

There is a latency overhead involved in processing rate update. This is due to the finite feedback bandwidth associated with the DC-to-DC converter. Changing the processor clock frequency also involves a latency overhead, during which the PLL circuit locks. To be on the safe side, it is recommended that the voltage and frequency changes should not be done in parallel. In case of switching to higher processing rate, the voltage should be increased first, followed by the increase in frequency, and the following steps are to be followed:

- Set the new voltage.
- Allow the new voltage to settle down.
- Set the new frequency by changing the divider value, if possible. Otherwise, change the PLL clock frequency.

In case of switching to low processing rates, the frequency should be decreased first and then the voltage should be reduced to the appropriate level, and the following steps are to be followed:

- Set the new frequency by changing the divider value, if possible. Otherwise, change the PLL clock frequency.
- Set the new voltage. The CPU continues operating at the new frequency while voltage settles to the new value.
This ensures that the voltage level is always sufficient to support the required operating frequency, and help to avoid data corruption due to failure of the circuit. The voltage must be limited to the range over which delay and voltage track monotonically. Moreover, there is temperature dependence on delay; normally delay increases as temperature increases, but below a certain voltage ($2. V_T$), this relationship inverts and the delay decreases as temperature increases.

### Adaptive Voltage Scaling

The voltage scaling techniques discussed so far are open loop in nature [7]. Voltage–frequency pairs are determined at design time keeping sufficient margin for guaranteed operation across the entire range of best- and worst-case PVT conditions. As the design needs to be very conservative for successful operation, the actual benefit obtained is lesser than actually possible. A better alternative that can overcome this limitation is the adaptive voltage scaling (AVS) where a close-loop feedback system is implemented between the voltage scaling power supply and delay-sensing performance monitor at execution time. The on-chip monitor not only checks the actual voltage developed but also detects whether the silicon is slow, typical, or fast and the effect of temperature on the surrounding silicon.

The implementation of the AVS system is shown in Fig. 7.32. The dynamic voltage control (DVC) emulates the critical path characteristic of the system by using a delay synthesizer and controls the dynamic supply voltage. It consists of three major components: the pulse generator, the delay synthesizer, and the delay detector. By comparing the digitized delay value with the target value, the delay detector determines whether to increase, decrease, or keep the present supply voltage
value. The minimum operating voltage from 0.9 to 1.6 V at 5-mV step is supplied in real time by controlling off-chip DC–DC converter to adjust the digitized value to target. The dynamic frequency control (DFC) adjusts the clock frequency by monitoring the system activity. It consists of an activity monitor and a frequency adjuster. The DFC block controls the clock frequency at the required minimum value autonomously in hardware without special power-management software. The activity monitor calculates the total large scale integrated (LSI) activity periodically from activity information of embedded dynamic random-access memory (DRAM), bus, and CPU. The voltage and frequency are predicted according to the performance monitoring of the system. The dynamic voltage and frequency management (DVFM) system can track the required performance with a high level of accuracy over the full range of temperature and process deviations.

Subthreshold Logic Circuits

As the supply voltage continues to scale with each new generation of CMOS technology, subthreshold design is an inevitable choice in the semiconductor road map for achieving ultra-low power consumption [8, 9]. Figure 7.33 shows the operation of an MOS transistor in the subthreshold region. The subthreshold region is often referred to as the weak inversion region. Subthreshold drain current $I_{\text{subth}}$ is the current that flows between the source and drain of an MOSFET when the transistor is in subthreshold region, that is, for gate-to-source voltages below the threshold voltage. In digital circuits, subthreshold conduction is generally viewed as a parasitic leakage in a state that would ideally have no current. In the early years of very-large-scale integration (VLSI) technology, the subthreshold conduction of transistors has been very small, but as transistors have been scaled down to a deep submicron region, leakage from all sources has increased. For a technology generation with threshold voltage of 0.2 V, leakage can exceed to 50% of total power consumption. The reason for a growing importance of subthreshold conduction is that
the supply voltage has continually scaled down, both to reduce the dynamic power consumption and to keep electric fields inside small devices low, to maintain device reliability. The amount of subthreshold current depends on the threshold voltage, which lies between 0 V and $V_{dd}$, and so has to be reduced along with the supply voltage. That reduction means less gate voltage swing below threshold to turn the device OFF, and as subthreshold current varies exponentially with gate voltage, it becomes more and more significant as MOSFETs shrink in size.

Subthreshold circuits operate with a supply voltage $V_{dd}$ less than the threshold voltage $V_t$ of the MOS transistor. As power is quadratically related to the supply voltage, reducing it to the subthreshold level leads to significant reduction in both power and energy consumption. However, the reduction in power consumption results in an increase in delay. As a consequence, subthreshold logic will be suitable only for specific applications which do not need high performance, but requires ultra-low power consumption. Such applications include medical equipment, such as hearing aid, pacemaker, wireless sensor nodes, radio-frequency identification (RFID) tags, etc. Subthreshold circuits can also be used in applications where the circuits remain idle for extended periods of time.

One important metric is the power delay product (PDP). The PDP for $V_{dd} = 0.5$ V is about 40 times smaller than that of the PDP for $V_{dd} = 3$ V. As the reduction in power dissipation outweighs the increase in delay, the PDP is lower. It has been established that the delay of circuits operating in the subthreshold region is three orders of magnitude higher compared to that of circuits operating in the strong inversion region, but the power consumption is four orders of magnitude lower for subthreshold circuits [6]. It implies that if both the circuits operate at the same clock frequency, the subthreshold circuits consume lesser energy compared to the circuit operating in the strong inversion region. One of the major concerns for subthreshold circuit design is increased sensitivity to PVT variations. The performance and robustness comparisons of bulk CMOS and double-gate silicon-on-insulator (DGSOI) subthreshold basic logic gates with and without parameter variations have been studied [7] and it has been observed that 60–70 % improvement in PDP and roughly 50 % better tolerance to PVT variations of DGSOI subthreshold logic circuits compared to bulk CMOS subthreshold circuits at the 32 nm node.
UNIT-4
Switched Capacitance Minimization

Introduction

In the previous chapter, we have discussed various voltage scaling approaches, which provide the most effective techniques to minimize dynamic power dissipation. However, there exist other alternative approaches such as minimizing switched capacitance to achieve the same goal. In this chapter, various approaches based on switched capacitance minimization are considered. Switched capacitance can be minimized at various levels of design hierarchy. A system-level approach based
on hardware–software codesign is presented in Sect. 8.2. Section 8.3 presents the example of Transmeta’s Crusoe processor, which shows how the function of a complex hardware is replaced by a software to achieve low power. Various bus-encoding techniques such as Gray coding, one-hot encoding, bus-inverse encoding, and T0 encoding have been presented in Sect. 8.4. These bus-encoding techniques can be used to reduce the switching activity on a bus. Various aspects of clock-gating (CG) technique have been provided in Sect. 8.5. Section 8.6 presents the basic principle behind gated-clock finite-state machines (FSMs) for reducing the switching activity in an FSM. In Sect. 8.7, FSM state encoding approach has been presented to minimize switching activity. Another approach for reducing switching activity of an FSM is FSM partitioning in which a single FSM is partitioned into more than one FSM to reduce switching activity, which has been presented in Sect. 8.8. The technique of operand isolation, presented in Sect. 8.9, can be used to reduce the switching activity of a combinational circuit. Precomputation is a technique in which selective computation of output values is done in advance with the objective of using it to reduce the switching activity in the subsequent cycles. This technique is presented in Sect. 8.10. The basic approach of minimizing glitching power has been considered in Sect. 8.11. Finally, various logic styles have been considered in Sect. 8.12 for low-power logic synthesis.

**System-Level Approach: Hardware–Software Codesign**

It is well known that a particular functionality can be realized purely by hardware, or by a combination of both hardware and software. Consider Example 8.1:

**Example 8.1** Realize a microcontroller-based system to monitor the temperature of a water bath.

Solution: The above application will require an analog-to-digital converter (ADC) to convert analog data obtained from a temperature transducer to a digital form, which can be processed by the central processing unit (CPU) of the microcontroller. Assuming that the microcontroller has no inbuilt ADC, there are two alternative approaches to implement the ADC. These two approaches are discussed below:

Approach-I: In this case, an ADC chip is interfaced externally as shown in Fig. 8.1a. This approach involves the use of a costly ADC chip along with a few lines of program code to read the ADC data. The ADC chip can be selected based on the sampling rate and the precision of the digital data. The software overhead is very small. This approach can provide higher performance in terms of conversion time and sampling rate. However, it involves higher cost and higher power dissipation.

Approach-II: If the requirement analysis of the application is done, it is observed that the present application does not require fast conversion of the analog data to digital data because the temperature of the water bath changes slowly and sampling
Fig. 8.1 a Analog-to-digital converter (ADC) implemented by hardware and b ADC implemented by hardware–software mix. DAC digital to analog, EOC end of conversion

of the output of the temperature sensor can be done at the interval of few seconds. So, the ADC functionality can be implemented by software with few inexpensive external components such as a digital-to-analog converter (DAC) and a comparator, as shown in Fig. 8.1b. Any of the A/D conversion algorithms such as a successive approximation can be implemented by the software utilizing the DAC and the comparator. As a consequence, it will have higher software overhead, but lesser hardware cost of implementation.

The first approach provides a fast conversion time at the cost of higher cost or larger chip area. In the second alternative, the hardware cost and chip area are lower, but conversion time is longer. So, for a given application, there is a trade-off between how much is implemented by hardware and by software. This has led to the concept of hardware–software codesign, which involves partitioning of the system to be realized into two distinct parts: hardware and software. Choosing which functions to implement in hardware and which in software is a major engineering challenge that involves consideration of issues such as cost, complexity, performance, and power consumption. From the behavioral description, it is necessary to perform hardware/software partitioning.

8.3 Transmeta’s Crusoe Processor

Transmeta’s Crusoe processor [16] is an interesting example that demonstrated that processors of high performance with remarkably low power consumption can be implemented as hardware–software hybrids. The approach is fundamentally software based, which replaces complex hardware with software, thereby achieving large power savings. By virtualizing the x86 CPU with a hardware–software combine, the Transmeta engineers have drawn a line between hardware and software such that the hardware part is relatively simple and high-speed, but much less power-hungry very long instruction word (VLIW) engine. On the other hand, the
Fig. 8.2 A molecule can contain up to four atoms, which are executed in parallel. \textit{FADD} floating point addition, \textit{ADD} addition, \textit{LD} load, \textit{BRCC} branch if carry cleared, \textit{ALU} arithmetic logic unit

complex task of translating the x86 instructions into the instructions of the VLIW is performed by a piece of software known as the code morphing software (CMS).

8.3.1 The Hardware

The Crusoe processor is a very simple, high-performance VLIW processor with two integer units, a floating-point unit, a memory (load/store) unit, and a branch unit. The long instruction word, called a molecule, can be 64 bits or 128 bits long, containing up to four reduced instruction set computing (RISC)-like instructions called atoms. All atoms within a molecule are executed in parallel and the format of the molecule directly determines how atoms get routed to the functional units. A simple 128-bit molecule is shown in Fig. 8.2. The diagram also shows how different fields (atoms) of the molecule map to different functional units. The molecules are executed sequentially. So, there is no need for complex out-of-order hardware. The molecules are packed as fully as possible with atoms to keep the processor running in full speed. The superscalar out-of-order x86 processors, such as the Pentium II and Pentium III processors, also have multiple functional units like the Crusoe processor. These functional units can also execute RISC-like operations in parallel. However, in contrast to a software (CMS) used in Crusoe processor to translate x86 instructions into micro-operations and schedule the micro-operations to make best the use of the functional units, the x86 processors use a hardware. As the dispatch unit reorders the micro-operations to keep the functional units busy, a separate hardware, known as in-order retire unit, is necessary to effectively reconstruct the order of the original x86 instructions, and ensures that they take effect in the proper order. The schematic diagram of the traditional superscalar architecture showing different building blocks is given in Fig. 8.3. As the 8086 instruction set is quite complex, the decoding and dispatch hardware requires large quantities of power-hungry transistors, which is responsible for high power consumption in rough proportion to the number of transistors used in realizing the functional elements shown in Fig. 8.2. Table 8.1 compares the sizes of Intel mobile and Crusoe processors.
It is evident from the table that Crusoe processors using the same technology and comparable performance level require about 50% chip area. This is also reflected in the power dissipations in the form of heat by Pentium III and a Crusoe processor running a DVD software. In the case of PIII, the temperature rises to 105 °C, which may lead to the point of failure if it is not aggressively cooled. On the other hand, in the case of the Crusoe processor module TM5400, the temperature rises to only 48 °C, which requires no active cooling. A smaller power dissipation of Crusoe processors not only makes the chip less prone to failure but also cheaper in terms of packaging and cooling costs.

### 8.3.2 The Software

The success of the Crusoe processor lies in the use of an innovative software known as CMS that surrounds the VLIW hardware processor. This software layer decouples the x86 instruction set architecture (ISA) from the underlying processor hardware, which is very different from that of the conventional x86 processor architectures. The CMS is essentially a dynamic translation system of a program that compiles the instruction of the ISA into the instruction of the VLIW processors. Normally, it is the first program that starts execution. The CMS is the only program that is written directly for the VLIW processor as shown in Fig. 8.4. All the other software, such as basic input/output system (BIOS), operating system (OS), and other application programs, are written using x86 ISA. The x86 only sees the CMS, which insulates x86 programs from the native VLIW processor. As a consequence, the native processor (the ISA) can be changed arbitrarily without any need to modify the x86 software. Any change in the VLIW ISA calls for a change in the CMS itself. The proof of this concept has been demonstrated by replacing the TMS3120 processor with the TMS5400 processor simply by having a different version of the CMS.
**Decoding and Scheduling** In a conventional superscalar processor such as x86, instructions are fetched from the memory and decoded into micro-operations, which are then recorded by an out-of-order dispatch hardware and fed to the functional units for parallel execution. In contrast to this, CMS translates an entire group of x86 instructions at once and saves the resulting translation in a translation cache and in subsequent executions. The system skips the translation step and directly executes the existing optimized translation. In a superscalar architecture, the out-of-order dispatch unit has to translate and schedule instructions every time these are executed, and it must be done very quickly. In other words, power dissipation occurs each time when instructions are executed. The cost of translation is amortized in the code morphing approach over many executions. This also allows much more sophisticated translation and scheduling algorithms. The generated code is better optimized, resulting in lesser number of instructions in the generated code. This not only speeds up execution but also reduces power dissipation.

**Caching** A separate memory space is used to store the translation cache and the CMS. The memory space is not accessible to x86 code, and the size of this memory can be set at boot time. One primary advantage of caching is to reuse the translated code by making use of the locality of reference property. In real-life applications, it is very common to execute a block of code many times over and over after it has been translated once. Because of the high repeat rates of many applications, code morphing has the opportunity to optimize execution and amortize the initial translation overhead. As a result, the approach of caching translations provides excellent opportunity of reuse in many real-life applications.
**Filtering**  It is of common knowledge that a small percentage (may be less than 10 %) of the code accounts for 95 % of the execution time. This opens up the possibility of applying different amounts of translation efforts to different portions of a program. The code morphing has a built-in feature of a wide choice of execution modes of x86 code, starting from interpretation, which has no translation overhead at all, but the execution is slower, to highly optimized code, which has a large overhead for code generation, but that runs the fastest once it is translated. A set of sophisticated heuristics helps to choose from several execution modes based on the dynamic feedback information gathered during the actual execution of the code.

**Prediction and Path Selection** The CMS can gather feedback information about the x86 program with the help of an additional code present in the translator whose sole purpose is to collect information about block execution frequencies or branch history. Decision can be made about how much effort is required to optimize a code based on how often a piece of x86 code is executed. For example, whether a conditional branch instruction, as shown in Fig. 8.5, is balanced (50 % probability of taken) or biased in a particular direction, decision can be made about how much effort to put to optimize that code. In a conventional hardware-only x86 implementation, it would be extremely difficult to make similar kinds of decisions.

**Example 8.2** The translation of a piece of x86 code into an equivalent of code for Crusoe VLIW processor is illustrated with the help of the following example. The x86 code, consisting of four instructions, to be translated is given below:

1. addl %eax,(%esp) // load data from stack, add to %eax
2. addl %ebx,(%esp) // ditto, for %ebx
3. movl %esi,(%ebp) // load %esi from memory
4. subl %ecx,5 // subtract 5 from %ecx register

The translation is performed in three phases. In the first pass, the front end of the translator system decodes the x86 instructions and translates them into a simple sequence of atoms of VLIW ISA. Using %r30 and %r31 as temporary registers for the memory load operations, the output of the first pass is given below:
ld %r30,(%esp)  // load from stack, into temporary
add.c %eax,%eax,%r30  // add to %eax, set condition codes.
ld %r31,[%esp]
add.c %ebx,%ebx,%r31
ld %esi,[%ebp]
sub.c %ecx,%ecx,5

In the second pass, the translator performs typical compiler optimizations such as common subexpression elimination, dead code elimination, etc. This helps in eliminating some of the atoms. For example, the output of the second pass for the above code is given below:

ld %r30,[%esp]  // load from stack only once
add %eax,%eax,%r30
add %ebx,%ebx,%r31  // reuse data loaded earlier
ld %esi,[%ebp]
sub.c %ecx,%ecx,5  // only this last condition code needed

In the final pass, the scheduler reorders the atoms and groups them into individual molecules, which is somewhat similar to the function of the dispatch hardware. The output, consisting of only two molecules of the final phase, is given below:

1. ld %r30,[%esp]; sub.c %ecx,%ecx,5
2. ld %esi,[%ebp]; add %eax,%eax,%r30; add %ebx,%ebx,%r30

The above VLIW instructions are executed in-order by the hardware and the molecules explicitly encode the instruction-level parallelism. As a consequence, the VLIW engine is very simple.

### Bus Encoding

The intrinsic capacitances of system-level busses are usually several orders of magnitude larger than that for the internal nodes of a circuit. As a consequence, a considerable amount of power is dissipated for transmission of data over input/output (I/O) pins. It is possible to save a significant amount of power, reducing the number of transactions, i.e., the switching activity, at the processors’ I/O interface. One possible approach for reducing the switching activity is to suitably encode [4, 9] the data before sending over the I/O interface. A decoder is used to get back the original data at the receiving end as shown in Fig. 8.6.

Encoding can be used either to remove undesired correlation among the data bits, as it is done while doing encryption, or to introduce controlled correlation. Coding to reduce the switching activity falls under the second category. To reduce the switching activity, it is necessary to increase the correlation among the signal values transmitted over the I/O pins.
Coding scheme can be broadly divided into two broad categories—nonredundant and redundant. In case of nonredundant coding, an $n$-bit code is translated into another $n$-bit code ($m = n$) and the $2^n$ code elements of $n$-bit are mapped among themselves. Although there is no additional lines for sending off chip data, the nonredundant coding is done based on the statistics of the sequence of words such that the switching activity is reduced.

In case of redundant coding technique, additional lines are used for sending data, i.e., $m > n$. Here, $2^n$ different $n$-bit words are mapped to a larger set of $m$-bit $2^m$ data words. Here, there are two alternatives—the encoder/decoder may be memory-less or may use memory elements for the purpose of encoding and/or decoding. If an $n$-bit word has a unique $m$-bit code word, then neither encoder nor decoder requires memory. Another possibility is to use a static one-to-many mapping of $2^n$ unique words to $2^m$ code words. The current state of the encoder selects a particular $m$-bit code word for transmission out of several alternatives for a particular $n$-bit data word. In this scheme, an encoder requires memory, but the decoder can be implemented without memory.

### 8.4.1 Gray Coding

One popular example of the nonredundant encoding scheme is the Gray coding. Gray coding produces a code word sequence in which adjacent code words differ only by 1 bit, i.e., Hamming distance of 1 as shown in Table 8.2. The number of transitions for binary representation is 30. On the other hand, the number of transitions for Gray code will always have 16. As a consequence, the transition activity, and hence the power dissipation, reduces by about 50%, and it is very useful when the data to be transmitted is sequential and highly correlated. The encoder and decoder are memory-less as shown in Fig. 8.7. Each encoder and decoder requires $(n-1)$ two-input exclusive OR (EX-OR) gates.
Gray coding has been applied to the address lines for both instruction and data access for the reduction of switching activity. The simulation results show that there is significant reduction in switching activity on the address bus because instructions are fetched from sequential memory locations. However, when a branch instruction is executed, the temporal transition is not sequential which leads to more than one transition even when Gray coding is used. The frequency of occurrence of branch instructions in a typical code is 1 in 15.

As data accesses are not usually sequential in nature, there is no noticeable reduction in the switching activity for Gray coding compared to uncoded binary data accesses. Approximately, equal transition activity has been found for random data

![Encoder and Decoder for Gray code](image)

**Fig. 8.7** Encoder and decoder for Gray code
Table 8.3 Bit transitions per second for different benchmark programs

<table>
<thead>
<tr>
<th>Benchmark program</th>
<th>Instruction address</th>
<th>Data address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Binary coded</td>
<td>Gray coded</td>
</tr>
<tr>
<td>Fastqueens</td>
<td>2.46</td>
<td>1.03</td>
</tr>
<tr>
<td>Qsort</td>
<td>2.64</td>
<td>1.33</td>
</tr>
<tr>
<td>Reducer</td>
<td>2.57</td>
<td>1.71</td>
</tr>
<tr>
<td>Circuit</td>
<td>2.33</td>
<td>1.47</td>
</tr>
<tr>
<td>Semigroup</td>
<td>2.68</td>
<td>1.99</td>
</tr>
<tr>
<td>Nand</td>
<td>2.42</td>
<td>1.57</td>
</tr>
<tr>
<td>Boyer</td>
<td>2.76</td>
<td>2.09</td>
</tr>
<tr>
<td>Browse</td>
<td>2.51</td>
<td>1.64</td>
</tr>
<tr>
<td>Chat</td>
<td>2.43</td>
<td>1.54</td>
</tr>
</tbody>
</table>

Fig. 8.8 One-hot encoding

patterns. Table 8.3 shows the switching activities in terms of the number of bit transitions per instruction (BPI) for different situations for a set of benchmark programs.

**One-Hot Coding**

Nonredundant coding provides reduction in the switching activity when the data sent over the lines are sequential and highly correlated. This can be ensured only in many situations. As we have mentioned in the previous section, when data sequences are random in nature, the Gray coding does not provide reduction in the switching activity. If we want reduction of the switching activity in all possible situations, we have to go for nonredundant coding. One-hot encoding is one such nonredundant coding that always results in reduction in switching activity. Here, an $n$-bit data word maps to a unique code word of $m$-bit in size, where $m = 2^n$. In this case, two devices are connected by using $m = 2^n$ signal lines as shown in Fig. 8.8. The encoder
receives the $n$-bit data as input and produces “1” on the $i$-th line $0 \leq i \leq 2^n-1$, when $i$ is the binary value of the $n$-bit data words, and “0” on the remaining lines. In this case, both encoder and decoder are memory-less.

The most important advantage of this approach is that the number of transitions for transmission of any pair of data words one after the other is two: one 0-to-1 and one 1-to-0. The reduction in dynamic power consumption can be computed. Although one-hot encoding provides a large reduction in switching activity, the number of signal lines increases exponentially ($2^n$) with $n$. For example, for $n = 8$, the number of signal lines is $m = 256$, and reduction in switching activity is 75%.

**Bus-Inversion Coding**

Another redundant coding scheme is bus-inversion coding, which requires only one redundant bit i.e., $m = n + 1$ for the transmission of data words as shown in Fig. 8.9. In case of the Hamming distance between $Data(t)$ and $Bus(t-1)$ is more than $n/2$, complement of $Data(t)$ is sent by the encoder over the bus as $Bus(t)$. On the other hand, when the Hamming distance between $Data(t)$ and $Bus(t-1)$ is less than or equal to $n/2$, then the complement of $Data(t)$ is sent by the encoder as $Bus(t)$. The redundant bit $P$ is added to indicate if $Bus(t)$ is an inverted version of $Data(t)$ or not. The encoder and decoder for bus-inversion encoding are shown in Fig. 8.10. By using this encoding method, the number of transitions is reduced by 10–20% for data busses. It may be noted that this approach is not applicable to address busses.

**T0 Coding**

The Gray coding provides an asymptotic best performance of a single transition for each address generated when infinite streams of consecutive addresses are
Data(t)

B(t)

B(t-1)

Data bus

Decoder

Encoder

CLK

P(t)

P(t-1)

G(X)

INV/pass

Fig. 8.10 Encoder and decoder of bus-inversion encoding. CLK clock signal, INV invalid

Fig. 8.11 T0 encoding

c onsidered. However, the code is optimum only in the class of irredundant codes, i.e., codes that employ exactly \( n \)-bit patterns to encode a maximum of \( 2^n \) words. By adding some redundancy to the code, better performance can be achieved by adapting the T0 encoding scheme [1], which requires a redundant line increment (INC) as shown in Fig. 8.11. When the INC signal value is 1, it implies that a consecutive stream of addresses is generated. If INC is high, all other bus lines are frozen to avoid unnecessary transitions. The receiver computes the new addresses directly. On the other hand, when two addresses are not consecutive, the INC signal is LOW and the remaining bus lines are used as the raw address. The T0 encoding scheme can be formally specified as

\[
(B(t), \text{INC}(t)) = \begin{cases} 
B(t - 1), 1, & \text{if } t > 0, b(t) = b(t - 1) + S \\
\{b(t), 0, & \text{otherwise}
\end{cases}
\]
The corresponding decoding scheme can be formally specified as follows:

\[ b(t) = \begin{cases} b(t-1) + S & \text{if } INC = 1 \text{ and } t > 0 \\ B(t) & \text{if } INC = 0 \end{cases} \]

where \( B(t) \) is the value on the encoded bus lines at time \( t \), \( INC(t) \) is the redundant bus line value at time \( t \), \( b(t) \) is the address value at time \( t \), and \( S \) is the stride between two successive addresses. The encoder and decoder for T0 encoding are shown in Fig. 8.12.

The T0 code provides zero-transition property for infinite streams of consecutive addresses. As a result, it outperforms the Gray coding, because of 1-bit line switching per a pair of consecutive addresses is performed in Gray coding. On an average, 35% reduction in address bus switching activity is achieved by this encoding scheme.

**Clock Gating**

A clock is commonly used in majority of logic blocks in a very-large-scale integration (VLSI) chip. As the clock switches in every cycle, it has a switching activity of 1. It has been found that 50% of the dynamic power originates from clock-related circuits. One of the most common and widely used techniques for reducing the dynamic power dissipation is clock gating (CG) [15]. It is based on the observation
Fig. 8.13 Power reduction using clock gating

![Clock gate diagram](image)

Fig. 8.14 Clock-gating mechanism. EN enable, CLK global clock, CLKG gated clock

that a large portion of signal transitions are unnecessary and therefore they can be suppressed to reduce power dissipation without affecting the functionality of the circuit. CG involves dynamically preventing the clock from propagating to some portions of the clock path under a certain condition computed by additional circuits as shown in Fig. 8.13. As shown in Fig. 8.13, only a leakage power dissipation takes place when a circuit is clock gated. The reduction in dynamic power dissipation takes place due to the reduction of switched capacitance in the clock network and the switching activity in the logic block fed by the clock-gated storage elements.

### 8.5.1 CG Circuits

The mechanism of CG is shown in Fig. 8.14. A combinational circuit is used to detect the condition when a specific functional unit is in use and generate an enable (EN) signal. When the idle condition of the functional unit is true, the clock to the functional unit shuts off with the help of a CG circuit. Here CLK is the global clock and CLKG is the gated clock generated by the CG unit. Several implementations have been proposed to realize the CG function. The simplest one is implemented with the help of an AND gate as shown in Fig. 8.15a. When the output of the CG function $F_{cg}$ is 0, the output of the AND gate of Fig. 8.15a is forced to 0, thus suppressing unnecessary transitions on the gated-clock CLKG. However, if the CG function does not stabilize to 0 level before the clock rises to high level, the glitch generated by the $F_{cg}$ propagates through the AND gate as shown in Fig. 8.15c. A slightly better approach is to use an OR gate as shown in Fig. 8.15b. In this case, when the output of the CG function $F_{cg}$ is 0, the output of the OR gate is forced to 1, thereby suppressing unnecessary transitions on the gated-clock CLKG. As long as the EN signal stabilizes before the high-to-low transition of the clock, the glitch
Fig. 8.15  
(a) Clock gating using AND gate, (b) clock gating using OR gate, (c) glitch propagation through the AND gate, and (d) glitch propagation through the OR gate. EN enable, CLK global clock, CLKG gated clock.

Fig. 8.16  
(a) Clock gating using a level-sensitive, low-active latch along with an AND gate and (b) clock gating using a level-sensitive, low-active latch along with an OR gate. EN enable, CLK global clock, CLKG gated clock.

Generated by the $F_{cg}$ cannot propagate through the OR gate as shown in Fig. 8.15d. This glitches generated by the CG function $F_{cg}$ can be filtered by using a level-sensitive, low-active latch and an AND gate as shown in Fig. 8.16a. This method freezes the latch output at the rising edge of the clock (CLK), and ensures that the new EN signal at the input of the AND gate is stable when the clock is high. This also allows the entire clock period to propagate the enable (EN) signal to the latch, thereby providing more flexible timing constraint. Similarly, a level-sensitive, low-active latch along with an OR gate can be used for a better glitch suppression as shown in Fig. 8.16b.
8.5.2 CG Granularity

Granularity of the circuit block at which CG is applied is an important issue. It greatly affects the power savings that can be achieved by CG. Global CG can turn a larger clock load off, but this approach has lesser opportunity of disabling the clock. Based on the granularity, the CG can be categorized into the following three levels:

- Module-level CG
- Register-level CG
- Cell-level CG

Module-Level CG This involves shutting off an entire block or module in the design leading to large power savings. Typical examples are blocks which are explicitly used for some specific purpose such as transmitter/receiver, ADC, MP3 player, etc. For example, the transmitter can be turned off when the receiver is in use and vice versa. Normally, this is to be identified by the designer and incorporated in the register transfer language (RTL) code. Although global- or module-level CG can provide large saving in power when used, its opportunity is limited. Moreover, as large blocks are turned off and on, this approach may result in higher \( \frac{di}{dt} \) issue.

Example 8.3 The CG of the register file of a processor is shown in Fig. 8.17. It is known that the arithmetic logic unit (ALU) is not used when load or store instructions are executed in a processor having load/store architecture. As shown in Fig. 8.17, the register bank can be clock-gated to prevent unnecessary loading of operands to the ALU when load/store instruction is executed. In a similar manner, memory bank can also be clock-gated when ALU instructions are executed.

Register-Level Gating The clock to a single register or a set of registers is gated in register-level CG. Figure 8.18a shows a synchronous load-enabled register bank typically implemented using clocked D flip-flops and a recirculating multiplexer. Here, the register bank is clocked in every cycle, irrespective of whether a new
value (D_in) is loaded into it or not. The clock-gated version of the same register bank is shown in Fig. 8.18b. In this clock-gated version, the register does not get the clock in the cycles when no new data is to be loaded. The elimination of the multiplexer circuit (MUX) from the clock-gated version also saves power. As the CG circuit has its associated power dissipation, it is not cost-effective to do it for a single-bit register. However, the penalty of the CG circuit can be amortized over a large number of registers, saving the flip-flop clocking power and the power of all the multiplexers by a single CG circuit.

The power saving per clock gate in register-level CG is much less compared to the module-level CG, but there is much more scope for shutting off the clock compared to module-level CG. It also lends itself to automated insertion of CG in the design and can result in massively clock-gated designs.

**Cell-Level Gating** Cell designer introduces cell-level CG by incorporating CG circuit as part of a cell, thereby removing the burden of the circuit designer. For example, a register bank can be designed such that it receives clock only when new data need to be loaded. Similarly, memory banks can be clocked only during active memory access cycles. Although it does not involve any flow issue and design is simple, it may not be efficient in terms of area and power. There is area overhead and all registers are to be predesigned with inbuilt CG feature, and it does not allow sharing of CG logic with many registers.

**CG Challenges** Although CG helps to reduce dynamic power dissipation, it introduces several challenges in the application-specific integrated circuit (ASIC) design flow. Some of the important issues are as follows:

- Clock latency
- Effect of clock skew
- Clock tree synthesis
- Physical CG
- Testability concern
Gated-Clock FSMs

In digital systems, FSMs are very common components. These are widely used for a variety of purposes such as sequence generator, sequence detector, controller of data paths, etc. The basic structure of an FSM is shown in Fig. 8.19. It has a combinational circuit that computes the next state (NS) as a function of the present primary input (PI) and previous state (PS). It also computes the primary output (PO) as a function of the PS and PI for Mealy machines, and PO is a function of only the PS for a Moore machine. An FSM can be characterized by a quintuple, \( M = (PI, PO, S, \delta, \lambda) \), where \( PI \) is the finite nonempty set of inputs, \( PO \) is the finite nonempty set of outputs, \( S \) is the finite nonempty set of states, \( \delta \) is the state transition function (\( \delta: S \times PI \to NS \)), and \( \lambda \) is the output transition function (\( \lambda: S \times PI \to PO \)).

For a given FSM, there are situations when the NS and output values do not change. If these situations can be identified, the clock network can be shut off, thereby saving unnecessary power dissipations. This is the basic idea of CG. As shown in Fig. 8.20, an activation function \( F_{cg} \) is synthesized that evaluates to logic 1 when the clock needs to be shut off. Automated gated-clock synthesis approach for Moore-type FSMs was reported in [13]. The procedure for synthesizing \( F_{cg} \) is based on the observation that for a Moore FSM, a self-loop in the state-transition graph represents the idle condition when the clock to the FSM can be synthesized. For a Moore machine (\( \lambda: S \to PO \)), a self-loop corresponds to the idle condition. The self-loop for each state
Si can be defined as \( \text{Self} \, S_i : \Pi \rightarrow \{0, 1\} \), such that \( \text{Self} \, S_i (p_i) = 1 \iff \delta(x, S_i) = S_i, \) \( p_i \in \Pi \). The activation can be written as \( F_{cg} = \sum \text{Self} \, S_i \cdot x_i \), where \( x_i \) is the decoded state variable corresponding to state \( s_i \), i.e., \( x_i = 1 \) if and only if the FSM is in \( S_i \).

The circuitry to realize the activation function to identify these idle conditions may involve a large area, delay, and power overhead. Here, the challenge is to identify a subset of the idle conditions at a significantly lower overhead such that the complexity of implementation of \( F_{cg} \) is reduced. The gated-clock FSM implementation is illustrated with help of Example 8.4.

**Example 8.4** The state-transition diagram of an FSM is shown in Fig. 8.21. The FSM interacts with a timer-counter section, which generates a delay of thousands of cycles, and at the end of timeout (indicated by the ZERO signal), a short but complex operation is performed in the DO_IT state of the machine. The clock can be shut off during the long countdown time of the timer-counter (WAIT state of the machine). The idle situation can be easily identified by the signals TC_SET=0 and TC_EN=1. \( F_{cg} \) can be realized accordingly as shown in Fig. 8.22.
Table 8.4 State assignments using Gray code and binary code for modulo 6 counter

<table>
<thead>
<tr>
<th>States</th>
<th>Binary code</th>
<th>Gray code</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>000</td>
<td>000</td>
</tr>
<tr>
<td>S2</td>
<td>001</td>
<td>001</td>
</tr>
<tr>
<td>S3</td>
<td>010</td>
<td>011</td>
</tr>
<tr>
<td>S4</td>
<td>011</td>
<td>010</td>
</tr>
<tr>
<td>S5</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>S6</td>
<td>101</td>
<td>100</td>
</tr>
<tr>
<td>State locus</td>
<td>10</td>
<td>6</td>
</tr>
</tbody>
</table>

**FSM State Encoding**

State assignment is an important state in the synthesis of an FSM [3]. In this step, each state is given a unique code. State assignment strongly influences the complexity of the combinational logic part of the FSM. Traditionally, the state assignment has been used to optimize the area and delay of the circuit. It can also be used to reduce the switching activity for the reduction of the dynamic power. Given the state-transition diagram of a machine, the state assignment can be done such that the objective function $\gamma$, as specified below, is minimized.

$$\gamma = \sum p_{ij}w_{ij}$$ for all transitions, where $p_{ij}$ is the probability transition from state $S_i$ to state $S_j$ and $w_{ij}$ is corresponding weight representing the activity factor.

This is illustrated with the help of the following two examples:

**Example 8.5** Consider the state-transition diagram of a modulo 6 counter shown in Fig. 8.23. Counter is a special type of FSM having no PI or PO. In this case, the probability of all transitions $p_{ij}$ is 1. One metric known as state locus, which is the sum of bit distances for all transitions between states of a sequential machine, can be found out. The state assignment which provides minimum state locus can be used for state assignment. Two state assignments, one using a Gray code and another using binary code assignments, is shown in the Table 8.4. The state locus of the Gray code is 6, whereas the same for the binary code is 10. Therefore, the FSM implementation using the Gray code will have lesser switching activity compared to that of binary code implementation.

**Example 8.6** Consider the state-transition diagram of a sequence detector that produces “1” when five 1’s appear sequentially at the input, as shown in Fig. 8.24. In this case, the probability of each transition is 0.5 and weight $w_{ij}$ is the Hamming
distance between the codes of $S_i$ and $S_j$. The objective function is computed for assignment-1 as well as for assignment-2. The objective function of the assignment-1 is 10, whereas the same for assignment-2 is 5.5. Therefore, the FSM implementation using the assignment-2 will have lesser switching activity compared to that of assignment-1 (Table 8.5).

### FSM Partitioning

The idea is to decompose a large FSM into several smaller FSMs with smaller number of state registers and combinational blocks. Only an active FSM receives clock and switching inputs, and the others are idle and consume no dynamic power. Consider a large FSM that includes a small subroutine as shown in Fig. 8.25a. The FSM is decomposed into two FSMs as shown in Fig. 8.25b. Here, two additional wait states SW22 and TW0 have been added between the entry and exit points of the two FSMs. In this case, when one of the FSMs is operating, the other one is in the wait state. When an FSM is in the wait state, its input and clock can be gated to minimize switched capacitance. There can be significant savings in switched capacitance if it is possible to isolate a very small subset of states where the initial FSM remains most of the time.
Operand Isolation

Operand isolation is a technique for power reduction in the combinational part of the circuit. Here the basic concept is to “shut off” the logic blocks when they do not perform any useful computation. Shutting off is done by not allowing the inputs to toggle in clock cycles when the block output is not used. Consider the circuit of Fig. 8.26a. Here, the output of the adder is loaded into the latch only when the signal S_1 is 1 and the signal S_2 is 0. Addition can be performed only when the output of the adder is loaded into the latch, an activation signal (AS) is generated to isolate the operands using AND gates as shown in Fig. 8.26b. It may be noted that although operand isolation reduces power, it introduces timing, area, and power overhead.
Precomputation

Precomputation is a technique in which selective computation of output values is done in advance using a much simpler circuit than the original circuit. Precomputed values are used to reduce the switching activity in the subsequent cycles. Consider a combinational logic block sandwiched between two registers R1 and R2 as shown in Fig. 8.27. Figure 8.28 shows a generalized schematic diagram for performing precomputation. Here, the predictor functions, g1 and g2, are realized, which are far simpler than the original combinational logic block g. The combinational circuits g1 and g2 are realized such that g1=1=>f=1 and g2=1=>f=0. During clock cycle n, if either g1 or g2 evaluates to 1, the register R1 is disabled from loading. In this cycle, output is taken from either g1 or g2. However, when both g1 and g2 evaluates to 0, then computation is done by the logic block g as shown in Fig. 8.28. In this case, complete disabling of all inputs takes place, resulting in power-consumption savings. The extra logic circuits added to implement precomputation incurs area, delay, and power overheads. The area and power overheads can be minimized by making g1 and g2 as simple as possible and maximizing the signal probabilities of g1 and g2. The delay overhead can be reduced by applying this approach to subcircuits in the noncritical path.
An \quad \quad B_n

An-1...A_1 \quad \quad B_n-1...B_1

\hspace{2cm} \hspace{2cm}

\hspace{2cm} \hspace{2cm}

\hspace{2cm} \hspace{2cm}

Fig. 8.29 Precomputation to realize comparator function

**Example 8.7** A simple and realistic example is an $n$-bit digital comparator. In this case, the most significant bits are sufficient to predict the output when they are not equal. Therefore, the predictor functions can be realized using $A_n$ and $B_n$, the most significant bits. The expressions for the predictor functions are: $g_1 = A_n \cdot B_n'$ and $g_2 = A_n' \cdot B_n$. When either of the predictor functions evaluates to 1, all inputs to the comparator except the most significant bits can be disabled. The realization of the precomputation circuit is shown in Fig. 8.29. In this example, disabling of a subset of inputs takes place.

**Glitching Power Minimization**

In case of a static complementary metal–oxide–semiconductor (CMOS) circuit, the output node or internal nodes can make undesirable transitions before attaining a stable value. As shown in Fig. 8.30a, if the input changes from 010 to 111 at time $t=0$, the output at $O_1$ switches from 1 to 0 after one gate delay duration. This, in
turn, leads to a low-level output of one gate delay duration after two gate delays. A node, internal or output, can have multiple transitions inside a single clock cycle before settling down to a final value. These spurious signals are known as glitches [2]. The Glitches are due to converging combinational paths with different propagation delays. The number of these undesired transitions is dependent on the logic depth, signal skew of different paths, as well as input signal pattern. At the logic synthesis phase, selective collapsing and logic decomposition can be carried out before technology mapping to minimize the level difference between the inputs of the nodes driving high-capacitive nodes. Alternatively, delay insertion and pin reordering can be done after technology mapping to make delays of all the paths equal. Here, the issue is to insert minimum number of delay elements to achieve maximum reduction in glitching activity. On highly loaded nodes, buffers can be inserted to balance delays and cascaded implementation can be avoided, if possible, to minimize glitching power. Figure 8.30b shows a cascaded realization, which is prone to high glitching activity compared to the tree realization of Fig. 8.30c that balances the path delays.

**Logic Styles for Low Power**

There are two basic approaches to realize a digital circuit by metal–oxide–semiconductor (MOS) technology: gate logic and switch logic [20]. A gate logic is based on the implementation of digital circuits using inverters and other conventional gates such as NAND, NOR, etc. Moreover, depending on how circuits function, they can also be categorized into two types—static and dynamic gates. In case of a dynamic gate, no clock is necessary for their operation and the output remains steady as long as the power supply is maintained. Dynamic circuits are realized by making use of the information storage capability of the intrinsic capacitors of the MOS devices, which are to be refreshed at regular intervals before information is lost because of the typical leakage of charge stored in the capacitors. On the other hand, the switch logic is based on the use of pass transistors or transmission gates, just like relay contacts, to steer logic signals. In this section, we shall briefly introduce three potential logic styles and mention their advantages and disadvantages.

It has been observed that the choice of logic style affects the power dissipation significantly. In present-day VLSI circuits, static CMOS has emerged as the technology of choice because of its robustness, ease of fabrication, and availability of sophisticated design tools and techniques. However, there exist other logic styles such as dynamic CMOS, pass-transistor logic (PTL) circuits, etc., which have the potential for the realization of high-performance and low-power circuits. In the context of low-power logic synthesis, comparison of PTL with the static CMOS logic styles has been done by a limited number of researchers [5, 8, 10, 17, 22, 23, 25]. Some of the reported results are conflicting in nature. For example, Yano et al. [23] have established that pass-transistor realization is superior to the static CMOS realization, whereas Zimmerman and Fichtner [25] have demonstrated that static CMOS is superior to PTL circuits. No such investigation has been done to compare
static CMOS and dynamic CMOS styles. To compare the three logic styles on an equal basis, it is necessary to develop efficient tools and techniques for logic synthesis using dynamic CMOS and PTL styles, which already exists for static CMOS style, and then compare the performance of the three logic styles.

Based on this motivation, efficient logic synthesis tools have been developed for PTL circuits [22] and dynamic CMOS [21]. This section reports the comparative study of these logic styles in terms of area, delay, and power with the help of a large number of International Symposium on Circuits and Systems (ISCAS) benchmark circuits, each having large number of inputs and outputs. An overview of the efficient logic synthesis techniques reported in [22] and [21] are given in Sects. 8.10.4 and 8.10.5. Section 8.4 presents the implementation details. Finally, experimental results based on the realization of ISCAS benchmark circuits are given in Sect. 8.5.

**Static CMOS Logic**

The static CMOS or full complementary circuits require two separate transistor networks: pull-up pMOS network and pull-down nMOS network, as shown in Fig. 8.31a, for the realization of digital circuits. Realization of the function \( f = A + B \cdot C \) is shown in Fig. 8.31b. Both the nMOS and pMOS networks are, in general, a series–parallel combination of MOS transistors. In realizing complex functions using full complementary CMOS logic, it is necessary to limit the number (the limit is in the range of four to six transistors) of MOS transistors in both the pull-up network (PUN) and pull-down network (PDN), so that the delay remains within acceptable limit.

**Advantages of Static CMOS Logic**

- Ease of fabrication
- Availability of matured logic synthesis tools and techniques
- Good noise margin
Fig. 8.32 Dynamic complementary metal–oxide–semiconductor (CMOS) gate with a n-block and b p-block

- Good robustness property against voltage scaling and transistor sizing
- Lesser switching activity
- No need for swing restoration
- Good I/O decoupling
- Easy to implement power-down circuit
- No charge sharing problem

Disadvantages of Static CMOS Logic
- Larger number of transistors (larger chip area and delay)
- Spurious transitions due to finite propagation delays from one logic block to the next, leading to extra power dissipation and incorrect operation
- Short-circuit power dissipation
- Weak output driving capability
- Large number of standard cells requiring substantial engineering effort for technology mapping

**Dynamic CMOS Logic**

Dynamic CMOS circuits are realized based on pre-charge logic. There are two basic configurations of a dynamic CMOS circuit as shown in Fig. 8.32a and b. In the first case, an n-block is used as the PDN as shown in Fig. 8.32a. In this case, the output is charged to “1” in precharge phase, and in the evaluation phase, the output either discharges to “0” through the PDN, if there is discharge path depending on the input combination. Otherwise, the output maintains the “1” state. In the second case, a p-block is used as PUN as shown in Fig. 8.32b. In the pre-charge phase, output is discharged to “0” level, and in the evaluation phase, it is charged to “1” level through the PUN, if there is charging path depending on the input combination. Otherwise, the output remains at “0” level.
Advantages of Dynamic CMOS Logic

• Combines the advantage of low power of static CMOS and lower chip area of pseudo-nMOS
• The number of transistors is substantially lower compared to static CMOS, i.e., \( N + 2 \) versus \( 2N \)
• Faster than static CMOS
• No short-circuit power dissipation occurs in dynamic CMOS, except when static pull-up devices are used to reduce charge sharing.
• No spurious transitions and glitching power dissipation, since any node can undergo at the most one power-consuming transition per clock cycle

Disadvantages of Dynamic CMOS Logic:

• Higher switching activity
• Not as robust as static CMOS
• Clock skew problem in cascaded realization
• Suffers from charge sharing problem
• Suffers from charge leakage problem requiring precharging at regular interval
• Difficult to implement power-down circuits
• Matured synthesis tools not available

Special type of circuits, namely domino and NORA CMOS circuits, overcomes the clock skew problem. The general structure of the domino logic is shown in Fig. 8.33a. The same function \( f \) is realized in the domino logic as shown in Fig. 8.33b. NORA stands for NORAce. In the NORA technique, logic functions are implemented by using nMOS and pMOS blocks alternately for cascading as shown in Fig. 8.34a. The function \( f \) realized in the NORA logic is shown in Fig. 8.34b. Here, we follow the domino logic style for the synthesis of dynamic circuits. Unfortunately, domino logic circuit can implement only non-inverting logic. Hence, complements of internal signals need to be realized through the separate cones of logic using complements of PIs, resulting in a significant area overhead.
There are various members in the PTL family such as complementary pass-transistor logic (CPL), swing-restored pass-transistor logic (SRPL), double pass-transistor logic (DPL), and single-rail pass-transistor logic (LEAP). Among the different PTL families, LEAP cells based on single-rail PTL requires minimum number of transistors. All the PTL families except SRPL provide good driving capability using inverters. SRPL logic style also does not provide I/O decoupling. Additional inverters and weak pMOS transistors are required for all the PTL styles except DPL that does not require swing restoration. We have used three LEAP-like cells for technology mapping stage of logic synthesis.

Advantages of PTL

- Lower area due to smaller number of transistors and smaller input loads.
- As the PTL is ratioless, minimum dimension transistor can be used. This makes pass-transistor circuit realization very area efficient.
- No short-circuit current and leakage current, leading to lower power dissipation.

Disadvantages of PTL

- When a signal is steered through several stages of pass transistors, the delay can be considerable.
- There is a voltage drop as we steer signal through nMOS transistors. To overcome this problem, it is necessary to use swing restoration logic at the gate output.
- Pass-transistor structure requires complementary control signals. Dual-rail logic is usually necessary to provide all signals in complementary form.
- Double intercell wiring increases wiring complexity, and capacitance by a considerable amount.
- There is possibility of sneak path.

Efficient techniques for the logic synthesis using dynamic CMOS and PTL have been developed. An overview of the existing approaches is given in the following section.
8.12.4 Synthesis of Dynamic CMOS Circuits

In recent years, some works [18, 19, 21, 24] have been reported on the synthesis of dynamic CMOS circuits. In [19], an approach to perform technology-independent optimization and technology mapping for synthesizing dynamic CMOS circuits using domino cells has been proposed. This approach assumes a circuit in multilevel decomposed form, converts the circuit into unate using standard bubble pushing algorithm and then optimizes the unate circuit using unate optimization procedure based on extraction/factoring, substitution, elimination, and don’t-care optimization. As a task of technology mapping, the approach follows the general tree-by-tree mapping approach and without using any standard cell library. The approach proposed in [19] has been found to give good results in terms of delay and area. However, the said approach adopts computationally expensive bubble pushing algorithm and is not applicable for large circuits. Another approach has been reported in [24] to synthesize domino circuits. This approach starts with an optimized circuit based on output phase assignment technique. The optimized circuit is decomposed into forest of trees to perform technology mapping. The approach proposed on-the-fly domino cell mapping, satisfying the width (the number of transistors in parallel) and length (the number of transistors in series) as it was done in. Further, the approach [19] proposed multi-output domino cells and dynamic programming framework to reduce the total number of cells to be used. According to the results reported in [24], some circuits have an area overhead; this is because of the logic replications to obtain inverters, as necessary. However, there is no reported work on the NORA logic synthesis.

A different approach is adopted to synthesize the dynamic CMOS circuits. It is based on the two-level unate decomposition of logic functions [18]. Two-level unate decomposition of a switching function $f$ can be expressed as, $f = \sum P_i \cdot N_i$,

where $P_i$ is a positive unate function (all literals are in non-complemented form) and $N_i$ is a negative unate function (all literals are in complemented form). The generic structures of dynamic CMOS circuits based on two-level decomposition with domino logic and NORA logic are shown in Fig. 8.35a and b, respectively. The keeper transistors are not shown for simplicity. Details of the algorithm to obtain two-level unate decomposition of a function can be understood from [18]. The domino and NORA circuits based on this unate decomposition are shown in Fig. 8.36a and b, respectively. As stated earlier, we see that dynamic NORA circuit is completely inverter-free.

**Example 8.8** Let us consider a Boolean function

$$f_1 = \sum (3, 5, 7, 8, 11, 12, 14, 15)$$

Following the algorithm in [21], the two-level unate decomposition of the function can be given as

$$f_1 = (x_4 + x_3 x_1)(\bar{x}_4 + \bar{x}_2 \bar{x}_1) + (x_2 x_1 + x_4 x_3 x_2) \cdot 1$$
Fig. 8.35 Dynamic CMOS circuits based on two-level unate decomposition: a domino CMOS circuit and b NORA CMOS circuit

Two-level dynamic CMOS realization for functions of large number of variables using the unate decomposition approach as mentioned above may not be suitable for practical implementation, because of existence of a large number of transistors in series and parallel in gates. There is a constraint (to realize circuits of reasonable delay) on the length and width of a practical gate. By length, we mean the maximum number of series transistors in a gate. When the number of transistors
in series increases (this is the case when a product term consists of a large number of variables), on-resistance of gate increases and thereby affecting the delay of the gate. Increase in (width) the number of transistors in parallel (this is the case when a large number of product terms are there in the logic) proportionally increases the drain capacitance of a gate, which in turn increases the delay time as well as the dynamic power consumption. It is known that the delay of a gate drastically increases when there are more than four transistors in series. In order to reduce the delay, there should be four transistors, at the most, in series. Similarly, the number of transistors in parallel is restricted to five in order to check the delay as well as dynamic power consumption due to load capacitance. To satisfy length and width constraints, another decomposition step is proposed, where each node in the two-level decomposed circuit is further broken down into intermediate nodes till each node satisfies the length and width constraints. This decomposition is termed as the cell-based multilevel decomposition. To minimize the number of nodes, all kernels are extracted and then kernel-based decomposition is performed. After the kernel-based decomposition is over, each node is examined, whether it satisfies the length and width constraints or not. If the length and width constraints are not satisfied, then we carry out the cell-based multilevel decomposition. The cell-based multilevel decomposition is better illustrated with Example 8.9. This cell-based multilevel decomposition accomplishes the task of technology mapping. It can be noted that, this technology mapping is not based on any library of standard gates; instead, it does “on the fly” generation of cells, satisfying length and width constraints.

Fig. 8.36 Realization of dynamic circuits for $f_1$ a using domino logic and b using NORA logic
Example 8.9 Consider the function \( f_2 \)

\[
f_2 = x_1 x_2 x_3 x_4 x_{12} + x_6 x_7 x_8 + x_1 x_9 + x_2 x_7 + x_4 x_8 x_9 + x_2 x_5 x_6 x_9 x_{10} x_{11} + x_1 x_8 x_{11}
\]

Further assume that the length and width constraints for our technology are 3 and 4, respectively. After the decomposition satisfying the length and width constraints, we can obtain a netlist as shown below:

\[
A = x_1 x_2 x_3 \quad B = x_2 x_5 x_6 \quad C = x_9 x_{10} x_{11} \quad D = x_6 x_7 x_8 + x_1 x_9 + x_2 x_7 + x_4 x_8 x_9
\]

\[
f_2 = A \cdot x_4 x_{12} + D + B \cdot C + x_1 x_8 x_{11}
\]

Domino circuits corresponding to this netlist can be readily obtained. On the other hand, for NORA circuits, we need a decomposition into odd number of levels, so that alternate n-block, p-block, and n-block (for PDN of NORA) or p-block, n-block, and p-block (for PUN of NORA) can be realized. As in the current example, number of levels is two, so at the final level, we can do one more decomposition as

\[
f_{21} = A \cdot x_4 x_{12} + D \quad f_{22} = B \cdot C + x_1 x_8 x_{11} \quad f_2 = f_{21} + f_{22}
\]

The objective of the technology mapping with cell-based decomposition is to decompose a node such that it can be realized with cells of permissible size. At the same time, in order to ensure lesser number of levels, we are to create intermediate nodes in terms of PIs as far as possible.

Figure 8.37 shows the various steps involved in our synthesis procedure. We assume that input circuits are in the Berkeley logic interchange format (BLIF). The first step in our approach is partitioning. This step is required because the unate decomposition algorithm is based on minterm table, and it has been observed that if the number of input variables is not more than 15, then the runtime of the algorithm lie within an acceptable range. To ensure this, the partitioning of input circuit graph is proposed into a number of partitions so that the number of inputs of a partition does not exceed upper limit 15. The partitioning is done in two phases as shown in Fig. 8.37. In the first phase, an initial partition is created, and in the second phase, the initial partition is refined to reduce the total number of partitions. To create the initial partition, the entire graph is traversed in breadth-first fashion and based on the maximum size (number of inputs) of a partition; the circuit graph is partitioned into smaller subgraphs as shown in Fig. 8.38b. The first phase partitions the input graph into variable-sized partitions. We usually end up with partitions having too small sizes, which can be merged with other partitions to form a partition with reasonably larger number of inputs. For example, as shown in Fig. 8.38c, a node in the partition \( P_x \) can be moved to \( P_y \) and a node in partition \( P_y \) can be moved to \( P_z \) without violating the size limit of partitions \( P_y \) and \( P_z \). The second phase allows us to merge or refine those partitions. We move a node from its current position to
the new one, if the number of inputs to the new partition is not exceeding the upper limit on the size of partition. This is similar to the refinement phase in the standard Fiduccia–Mattheyses (FM) bipartitioning algorithm [14]. It has been observed that, by adopting this divide-and-conquer approach, it is possible to handle VLSI circuits involving hundreds of input variables.

In the earlier works, binary decision diagrams (BDDS) of the complete functions called monolithic BDDs were constructed and then technology mapping onto PTL cells were performed. The problem with the monolithic BDD-based approach is that the size of monolithic BDDs grows exponentially with the number of input vari-
Table 8.6 Ratio parameter table of $f_3$

<table>
<thead>
<tr>
<th>Minterm $x_i$</th>
<th>$x_4$</th>
<th>$x_3$</th>
<th>$x_2$</th>
<th>$x_1$</th>
<th>$f_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RP</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Variables of the function, making the approach unsuitable for functions of large number of variables. Decomposed BDD approach has been proposed to overcome this problem. In decomposed BDD approach, instead of creating a single big monolithic BDD, compact BDDs in terms of intermediate variables are constructed. These smaller BDDs can be optimized with ease, and the approach allows synthesis of any arbitrary function with a large number of variables. Liu et al. [17] and Chaudhury et al. [10] have proposed performance-oriented and area-oriented optimization techniques based on the decomposed BDDs. Here, decomposed BDDs have been realized using LEAP-like cells in the PTL circuit synthesis approach. In addition to using decomposed BDDs, we have used new heuristic, called ratio parameters (RP) [18], for optimal variable ordering. The RP is purely a functional property, which can be obtained from the completely specified minterm table consisting of only the true vectors of Boolean functions. The RP can be defined as follows:

For an $n$-variable function $f$, the set of parameters $N_i/D_i$, $N_{n-1}/D_{n-1}$, ..., $N_1/D_1$ are called the RP of $f$, where $N_i$ and $D_i$ are the number of 1’s and 0’s, respectively, in the $x_i$-th column of the minterm table of $f$.

8.12.5 Synthesis of PTL Circuits

**Example 8.10** Consider a function $f_3 = \sum (0, 1, 4, 6, 8, 10, 11)$. The RP of $f_3$ can be obtained as shown in Table 8.6.

To explain the working methodology of the RP-based heuristic, let us consider a Boolean function $f$ with $n$ Boolean variables $x_1, x_2, x_3, \ldots, x_i, \ldots, x_n$. Selection of variable at any stage is governed by three cases:

- **Case 1:** $N_i/D_i = 2^{n-1}$ or 0 for a variable $x_i$ in $f$. Then the Shannon’s expansion [22] will be:
  
  (a) $f = x \cdot C + x \cdot \bar{C}$ if $N_i = 2^{n-1}$ or 0
  
  or (b) $f = x \cdot \hat{f} + x \cdot \hat{C}$ if $D_i = 2^{n-1}$ or 0,

  where $C = 1$ (0), if $N_i(D_i) = 2^{n-1}$ (0).

- **Case 2:** $N_i(D_i) = 2^{n-2}$ and for some $j \neq i$, $N_j(D_j) \geq 2^{n-2}$ and simultaneous occurrence of $x_i \cdot x_j = 2$. Shannon’s expansion in this case will look like as:
\( (a) \ f = x \cdot x + x \cdot f, \) if \( N = 2^{n-2}, \ N \geq 2^{n-2} \)

or \( (b) \ f = x - x \cdot f, \) if \( D = 2^{n-2}, \ N \geq 2^{n-2} \)

or \( (c) \ f = x \cdot x \cdot f, \) if \( N = 2^{n-2}, \ D \geq 2^{n-2} \)

or \( (d) \ f = x \cdot x \cdot f, \) if \( D = 2^{n-2}, \ D \geq 2^{n-2} \)

Case 3: There is a \( k \)-group: \( x_{i_1} \cdot x_{i_2} \cdot x_{i_3} \ldots x_{i_k}, \ k \geq 3 \)

BDD can be obtained by cofactoring \( f \) with respect to \( x_{i_1} \cdot x_{i_2} \cdot x_{i_3} \ldots x_{i_k} \), where \( m = 2^k \).

Note that case 3 is the generalization of case 2.

The RP-based heuristic helps in obtaining BDDs with a smaller number of nodes. However, BDDs generated by this approach may not have the same ordering of variables at the same level along different paths. Or in other words, we propose reduced unordered BDDs (RUBDDs), in contrast to reduced ordered BDDs (ROBDDs) commonly used in the existing approaches.

There are four major steps in our approach of synthesizing PTL circuits:

- Decompose large circuits into a number of smaller sub circuits. We termed each decomposed subcircuit as a Boolean node.
- For each Boolean node in the decomposed graph, we create a BDD using RP heuristic.
- Map the entire circuit with the PTL cells.

A circuit in the BLIF is itself in a multilevel decomposed form where most components of the network are, in general, very simple gates. It is an overkill to apply PTL circuit synthesis at this level, since BDD representation of such a small gate is seldom advantageous. We, therefore, propose the concept of merging smaller components into a set of super Boolean nodes using the removal of reconvergent and partial collapsing.

Our second step in the algorithm is to compose BDD for each node in the decomposed circuit obtained in the previous step. We propose a heuristic based on RP to select a variable for the expansion of function at a current level. The heuristic is to select a variable at a current stage in such a way that the maximum number of closed inputs is available at a minimum path length from the current stage.

After the construction of BDD for each Boolean node in the network, we are to map the BDDs to the PTL cell library. Comprising all the steps mentioned above, the algorithm \( ptlRP \) for the realization of PTL circuits is outlined as below:
8.12.6 Implementation and Experimental Results

In order to verify the performances of the three logic styles, we have realized static CMOS, dynamic CMOS, and PTL circuits for a number of ISCAS benchmark circuits. The implementation details are discussed in the following sections.

8.12.6.1 Implementation of Static CMOS Circuits

For the synthesis of the static CMOS circuits, we have employed the Berkeley sequential interactive synthesis (SIS) software tool [6, 7, 21] to make the netlist for a given circuit. For a particular benchmark problem, it is first optimized with script. rugged code in SIS and then using the standard library 44–1.genlib in SIS; gate mapping is done with the option of minimum area. In the output file generated by SIS tool, the circuit is described by the equations employing NAND, NOR, and NOT operations. In order to simplify the analysis, we have taken the first two types of gates with two to four inputs. The mapped circuit is then translated into a bidirected graph using a parser and a graph generator program.
Implementation of Dynamic CMOS Circuits

The synthesis of dynamic CMOS circuits starts with partitioning an input circuit in BLIF into a number of smaller subcircuits. Each subcircuit is then transformed into PLA form (in terms of minterms), and disjoint decomposition is carried out to contain nonoverlapping minterms only. After preprocessing the input, our tool decomposes a Boolean function into a set of positive and negative unate subfunctions based on the algorithm MUD (multiuser detection) [3]. It should be noted that, algorithm MUD can handle a single logic function at a time, whereas a circuit is, in general, multi-output in nature. This problem has been sorted out by extracting one function at a time in a partition and then obtaining its unate decomposed version (in the form of a netlist) and storing the result in a temporary file. This process is to be repeated for all functions in the partition, and finally giving the netlist for a partition. Our tool then performs cell-based multilevel decomposition for nodes in the netlist in order to satisfy the length and width constraints. During the multilevel decomposition, for the constraints of a cell, we have chosen length=4 and width=5. After the multilevel decomposition is over, our tool produces the final netlist of the synthesized circuit.

Implementation of PTL Circuits

To synthesize a PTL circuit, the circuit under test is optimized with script.rugged command using Berkeley SIS tool [7]. A parser is written to store the optimized circuit in multilevel decomposed form as directed acyclic graph (DAG). Then the graph is transformed by replacing all reconvergent nodes [11] to their corresponding single nodes. After that, partial collapsing [11] is performed to get Boolean nodes of reasonably larger sizes. For each node in the graph so obtained, BDD is constructed with our algorithm ptlRP [11]. After the construction of BDD, we have mapped the BDD to PTL cell library [11].

Experimental Results

Switching the power and delay of the realized static CMOS, dynamic CMOS and PTL circuits are estimated using the estimation models presented in [3, 10, 11], respectively. The accuracy of these models is verified with Spice and Cadence tools. Value of transistor’s parameters is extracted using BSIM3V3 model and for 0.18 μm process technology as a particular instance. In our experiment, for all nMOS pass transistors, we have assumed the effective channel length as 0.18 μm and channel widths are 0.54 and 1.08 μm for nMOS and pMOS transistors, respectively, and the thickness of gate oxide is 40 Å. Further, we have assumed 1.0 V as the supply voltage (V_{dd}) and 0.2 V as the threshold voltage. The input transition probability for each PI is assumed to be 0.3. The operating temperature is assumed to be 110 °C. Operating frequency is assumed as 100 MHz.
<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Area (#Transistor)</th>
<th>Delay (ns)</th>
<th>Power (μW)</th>
<th>Area (ns)</th>
<th>Delay (ns)</th>
<th>Power (μW)</th>
<th>Area (ns)</th>
<th>Delay (ns)</th>
<th>Power (μW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C432</td>
<td>692</td>
<td>3.32</td>
<td>122</td>
<td>581</td>
<td>1.82</td>
<td>88</td>
<td>546</td>
<td>2.08</td>
<td>91</td>
</tr>
<tr>
<td>C499</td>
<td>1880</td>
<td>2.23</td>
<td>367</td>
<td>1506</td>
<td>1.69</td>
<td>248</td>
<td>1428</td>
<td>1.62</td>
<td>167</td>
</tr>
<tr>
<td>C880</td>
<td>1412</td>
<td>2.21</td>
<td>293</td>
<td>1249</td>
<td>1.79</td>
<td>166</td>
<td>988</td>
<td>1.18</td>
<td>267</td>
</tr>
<tr>
<td>C1355</td>
<td>1880</td>
<td>2.61</td>
<td>400</td>
<td>1603</td>
<td>1.51</td>
<td>280</td>
<td>1203</td>
<td>1.04</td>
<td>379</td>
</tr>
<tr>
<td>C1908</td>
<td>1756</td>
<td>2.91</td>
<td>367</td>
<td>1689</td>
<td>1.80</td>
<td>251</td>
<td>1088</td>
<td>1.57</td>
<td>298</td>
</tr>
<tr>
<td>C2670</td>
<td>1804</td>
<td>2.94</td>
<td>493</td>
<td>1584</td>
<td>1.74</td>
<td>395</td>
<td>1010</td>
<td>1.54</td>
<td>449</td>
</tr>
<tr>
<td>C3540</td>
<td>4214</td>
<td>4.57</td>
<td>409</td>
<td>2815</td>
<td>2.63</td>
<td>314</td>
<td>2782</td>
<td>2.58</td>
<td>294</td>
</tr>
<tr>
<td>C5315</td>
<td>7058</td>
<td>3.65</td>
<td>830</td>
<td>5970</td>
<td>2.69</td>
<td>515</td>
<td>5364</td>
<td>1.62</td>
<td>778</td>
</tr>
<tr>
<td>C6288</td>
<td>11,222</td>
<td>11.84</td>
<td>409</td>
<td>8716</td>
<td>4.64</td>
<td>504</td>
<td>6060</td>
<td>4.69</td>
<td>445</td>
</tr>
<tr>
<td>C7552</td>
<td>8214</td>
<td>2.99</td>
<td>1604</td>
<td>7328</td>
<td>1.98</td>
<td>1173</td>
<td>5682</td>
<td>1.66</td>
<td>1328</td>
</tr>
</tbody>
</table>

Average % reduction compared to static CMOS circuits

-16  -37  -25  -33  -47  -17
CMOS complementary metal–oxide–semiconductor, PTL pass-transistor logic

The experimental results on the synthesis of circuits with three different logic styles are shown in Table 8.7. Only dynamic CMOS circuits using domino logic is reported. In Table 8.7, Column 1 represents the benchmark circuits.

Area of a circuit is approximated as the number of transistors; requirements of area in static CMOS, dynamic CMOS, and PTL circuits are shown in column 2, column 5, and column 8, respectively. It is observed that on an average, dynamic CMOS circuit and PTL circuit require 16 and 33 % less area, respectively, compared to circuit with static CMOS logic. The area requirement is compared as shown in Fig. 8.39.

The delay of the circuits with the three logic styles are shown in column 3 (static CMOS), column 6 (dynamic CMOS), and column 9 (PTL). A comparison of the delay among the circuits with three logic styles is shown in Fig. 8.40. It is observed that, dynamic CMOS and PTL circuits are 37 and 47 % faster, respectively, compared to static CMOS circuit.

The power dissipation (sum of the switching power and leakage power) of circuits is shown in column 4, column 7, and column 10 for static CMOS, dynamic CMOS, and PTL circuits, respectively. Comparison of the power dissipations among three circuits is shown in Fig. 8.41. We may observe that static CMOS circuits dissipates 25 and 17 % more power than the dynamic CMOS and PTL circuits, respectively.
Fig. 8.39 Area (#Transistor) for static CMOS, dynamic CMOS, and PTL circuit. *CMOS* complementary metal–oxide–semiconductor, *PTL* pass-transistor logic

Fig. 8.40 Delay for static CMOS, dynamic CMOS, and PTL circuits. *CMOS* complementary metal–oxide–semiconductor, *PTL* pass-transistor logic

minimum area and smaller delay compared to the other two logic styles. The power dissipation of the dynamic CMOS and PTL styles are comparable and are substantially lower than the static CMOS circuits. Based on this study, we may conclude that the dynamic CMOS and PTL styles have better performance for low-power and high-performance applications.
Some Related Techniques for Dynamic Power Reduction

Power reduction techniques are applied throughout the design process from system level to layout level, gradually refining or detailing the abstract specification or model of the design. Power optimization approaches at different levels result in different percentage of reduction. Power optimization approaches at the high level are significant since research results indicate that higher levels of abstraction have greater potential for power reductions. Apart from the techniques mentioned in the preceding sections to reduce dynamic power, there remain few other methods which can also help to reduce the power dissipation. These are mainly system-level and gate-level optimization procedures that helps in significant power reduction. These techniques are discussed below:

Operand Isolation  Operand isolation as discussed earlier reduces dynamic power dissipation in data path blocks controlled by an EN signal. Thus, when EN is inactive, the data path inputs are disabled so that unnecessary switching power is not wasted in the data path. Operand isolation is implemented automatically in synthesis by enabling an attribute before the elaboration of the design. Operand isolation logic is inserted during elaboration, and evaluated and committed during synthesis based on power savings and timing. Figure 8.42 illustrates the concept and how it contributes to the power savings.

In the digital system (Fig. 8.42), before operand isolation, register C uses the result of the multiplier when the EN is on. When the EN is off, register C uses only
the result of register B, but the multiplier continues its computations. Because the multiplier dissipates the most power, the total amount of power wasted is quite significant. One solution to this problem is to shut down (isolate) the function unit (operand) when its results are not used, as shown by the digital system after operand isolation. The synthesis tool inserts AND gates at the inputs of the multiplier and uses the enable logic of the multiplier to gate the signal transitions. As a result, no dynamic power is dissipated when the result of the multiplier is not needed.

**Memory Splitting** In many systems, the memory capacity is designed for peak usage. During the normal system activity, only a portion of that memory is actually used at any given time. In many cases, it is possible to divide the memory into two or more sections, and selectively power down unused sections of the memory. With increasing memory capacity, reducing the power consumed by memories is increasingly important.

**Logic Restructuring** This is a gate-level dynamic power optimization technique. Logic restructuring can, for example, reduce three stages to two stages through logic equivalence transformation, so the circuit has less switching and fewer transitions. It helps mainly in reducing the glitching power of a circuit. An example of logic restructuring is shown in Fig. 8.43.

**Logic Resizing** By removing a buffer to reduce gate counts, logic resizing reduces dynamic power. An example of logic resizing is shown in Fig. 8.44. In Fig. 8.44, there are also fewer stages; both gate count and stage reduction can reduce power and also, usually, the area.

**Transition Rate Buffering** In transition rate buffering, buffer manipulation reduces dynamic power by minimizing switching times. An example of transition rate buffering is shown in Fig. 8.45.
**Fig. 8.43** Logic restructuring technique

**Fig. 8.44** Logic resizing technique

**Fig. 8.45** Transition rate buffering technique
**Pin Swapping** Figure 8.46 illustrates the pin-swapping approach. The pins are swapped so that, most frequently, switching occurs at the pins with lower capacitive load. Since the capacitive load of pin A (upper pin) is lower, there is less power dissipation.

![Pin Swapping (CA>C'C)](image)

**Fig. 8.46** Pin-swapping technique

- Nonredundant bus-encoding technique such as Gray coding technique for address bus has been explained.
- Redundant bus-encoding techniques such as one-hot encoding, bus-inversion encoding, and T0 encoding techniques have been explained with examples.
- Clock gating technique to reduce dynamic power dissipation has been discussed.
- Clock gating at different levels of granularity has been considered.
- Synthesis of gated-clock FSM to reduce power consumption has been introduced.
- State encoding of FSM to reduce power consumption has been explained.
- How FSMs can be partitioned to reduce power consumption has been explained.
- FSM isolation to minimize power consumption has been explained.
- Realization logic functions using dynamic CMOS and PTL styles have been considered to reduce power consumption compared to static CMOS realization of functions.
UNIT-5
Leakage Power Minimization

Introduction

Due to aggressive device-size scaling, the very-large-scale integration (VLSI) technology has moved from the millimetre to nanometre era by providing increasingly higher performance along the way. Performance improvement has been continuously achieved primarily because of the gradual decrease of gate capacitances. However, as the supply voltage must continue to scale with device-size scaling to maintain a constant field, the threshold voltage of the metal–oxide–semiconductor
(MOS) transistors should also be scaled at the same rate to maintain gate overdrive \((V_{cc}/V_t)\) and hence performance. Unfortunately, the reduction of \(V_t\) leads to an exponential increase in the subthreshold leakage current. As a consequence, the leakage power dissipation has gradually become a significant portion of the total power dissipation. For example, for a 90-nm technology, the leakage power is 42% of the total power and for a 65-nm technology, the leakage power is 52% of the total power. This has led to vigorous research work to develop suitable approaches for leakage power minimization. Various components of the leakage power at the transistor level have been presented in Sect. 6.5. In this chapter, we discuss different approaches for leakage power reduction.

As the supply voltage is scaled down, the delay of the circuit increases as per the relationship given in Eq. (7.2). Particularly, there is a dramatic increase in delay as the supply voltage approaches the threshold voltage. The delay can be kept constant if the threshold voltage is scaled at the same ratio as the supply voltage; i.e. the ratio of \(V/V_{dd}\) is kept constant. Unfortunately, as the threshold voltage is scaled down, the subthreshold leakage current increases drastically, as shown in Fig. 9.1a. Moreover, the delay increases with an increase in threshold voltage when the supply voltage is kept constant as shown in Fig. 9.1b. The threshold voltage is the parameter of importance for the control leakage power. As leakage power has an exponential dependence on the threshold voltage and all the leakage power reduction techniques are based on controlling the threshold voltage either statically or dynamically, we first discuss various parameters on which the threshold voltage of a MOS transistor depends. Various approaches for the fabrication of multiple-threshold-voltage transistors are presented in Sect. 9.2. The leakage power reduction techniques can be categorized into two broad types—standby and run-time leakage. When a circuit or a part of it is not in use, it is kept in the standby mode by a suitable technique such as clock gating. The clock gating helps to reduce the dynamic power dissipation, but leakage power dissipation continues to take place even when the circuit is not in use. There are several approaches such as transistor stacking, variable-threshold-voltage complementary metal–oxide–semiconductor (VTCMOS), and multiple-threshold-voltage complementary metal–oxide–
semiconductor (MTCMOS), which can be used to reduce the leakage power when a circuit is in the standby condition. Variable-threshold-voltage CMOS (VTCMOS) approach for leakage power minimization has been discussed in Sect. 9.3. Transistor stacking approach based on the stack effect has been highlighted in Sect. 9.4. On the other hand, there are several approaches for the reduction of the leakage power when a circuit is in actual operation. These are known as run-time leakage power reduction techniques. It may be noted that run-time leakage power reduction techniques also reduce the leakage power even when the circuit is in standby mode. As leakage power is a significant portion of the total power, importance of run-time leakage power reduction is becoming increasingly important.

Classification on leakage power reduction techniques is also possible based on whether the technique is applied at the time of fabrication of the chip or at run time. The approaches applied at fabrication time can be classified as static approaches. On the other hand, the techniques that are applied at run time are known as dynamic approaches. Run-time leakage power reduction based on multi-threshold-voltage CMOS (MTCMOS) has been discussed in Sect. 9.5. Section 9.5 has addressed the power-gating technique to minimize leakage power. The isolation strategy has been highlighted in Sect. 9.7. A state retention strategy has been introduced in Sect. 9.8. Power-gating controllers have been discussed in Sect. 9.8. Power management techniques have been considered in Sect. 9.10. The dual-\( V_t \) assignment technique has been introduced in Sect. 9.11. The delay-constrained dual-\( V_t \) technique has been presented in Sect. 9.12 and energy constraint has been considered in Sect. 9.13. The dynamic \( V_t \) scaling technique has been introduced in Sect. 9.14.

**Fabrication of Multiple Threshold Voltages**

The present-day process technology allows the fabrication of metal–oxide–semiconductor field-effect transistors (MOSFETs) of multiple threshold voltages on a single chip. This has opened up the scope for using dual-\( V_t \) CMOS circuits to realize high-performance and low-power CMOS circuits. The basic idea is to use high-\( V_t \) transistors to reduce leakage current and low-\( V_t \) transistors to achieve high performance. Before we discuss dual-\( V_t \) (or multiple \( V_t \)) circuit design techniques, we shall explore various fabrication techniques [1] used for implementing multiple threshold voltages in a single chip.

**Multiple Channel Doping**

The most commonly used technique for realizing multiple-\( V_t \) MOSFETs is to use different channel-doping densities based on the following expression:

\[
V = V_{th} + 2\tau_B \left[ \frac{q^2 \epsilon_s \cdot Na(2\tau_B + V_{tn})}{C_{ox}} \right],
\]  

(9.1)
where $V_{fb}$ is the flat-band voltage, $N_a$ is the doping density in the substrate, and
\[ \tau_B = \frac{kT}{q} \left( \frac{L_x}{N_a} \right). \]

Based on this expression, the variation of threshold voltage with channel-doping density is shown in Fig. 9.2. A higher doping density results in a higher threshold voltage. However, to fabricate two types of transistors with different threshold voltages, two additional masks are required compared to the conventional single-$V_t$ fabrication process. This makes the dual-$V_t$ fabrication costlier than single-$V_t$ fabrication technology. Moreover, due to the non-uniform distribution of the doping density, it may be difficult to achieve dual threshold voltage when these are very close to each other.

**Multiple Oxide CMOS**

The expression for the threshold voltage shows a strong dependence on the value of $C_{ox}$, the unit gate capacitance. Different gate capacitances can be realized by using different gate oxide thicknesses. The variation of threshold voltage with oxide thickness ($t_{ox}$) for a 0.25-μm device is shown in Fig. 9.3. Dual-$V_{th}$ MOSFETs can be realized by depositing two different oxide thicknesses. A lower gate capacitance due to higher oxide thickness not only reduces subthreshold leakage current but also provides the following benefits:

a. Reduced gate oxide tunnelling because the oxide tunnelling current exponentially decreases with the increase in oxide thickness.
Reduced dynamic power dissipation due to reduced gate capacitance, because of higher gate oxide thickness. Although the increase in gate oxide thickness has the above benefits, it has some adverse effects due to an increase in short-channel effect. For short-channel devices as the gate oxide thickness increases, the aspect ratio (AR), which is defined by AR=lateral dimension/vertical dimension, decreases:

$$\text{AR} = \frac{L}{\left[ t_{\text{ox}} \left( \frac{\varepsilon_{\text{si}}}{\varepsilon_{\text{ox}}} \right)^{1/3} \right]^{1/3}}$$

(9.2)

where $\varepsilon_{\text{si}}$ and $\varepsilon_{\text{ox}}$ are silicon and oxide permittivities, $L$, $t_{\text{ox}}$, $W_{\text{dm}}$, and $X_{j}$ are channel length, gate oxide thickness, depletion depth, and junction depth, respectively.

Immunity to the short-channel effect decreases as the AR value reduces. Figure 9.4 shows the channel lengths for different oxide thicknesses to maintain AR. A sophisticated process technology is required for fabricating multiple oxide CMOS circuits.

**Multiple Channel Length**

In the case of short-channel devices, the threshold voltage decreases as the channel length is reduced, which is known as $V_{th}$ roll-off. This phenomenon can be exploited to realize transistors of dual threshold voltages. The variation of the threshold voltage with channel length is shown in Fig. 9.5. However, for transistors with feature sizes close to 0.1 μm, halo techniques have to be used to suppress the short-channel
effects. As the $V_{th}$ roll-off becomes very sharp, it turns out to be a very difficult task to control the threshold voltage near the minimum feature size. For such technologies, longer channel lengths for higher $V_{th}$ transistors increase the gate capacitance, which leads to more a dynamic power dissipation and delay.

**Multiple Body Bias**

The application of reverse body bias to the well-to-source junction leads to an increase in the threshold voltage due to the widening of the bulk depletion region, which is known as body effect. This effect can be utilized to realize MOSFETs having multiple threshold voltages. However, this necessitates separate body biases to be applied to different nMOS transistors, which means the transistors cannot share the same well. Therefore, costly triple-well technologies are to be used for this purpose. Another alternative is to use silicon-on-insulator (SoI) technology, where the devices are isolated naturally.

In order to get the best of both the worlds, i.e. a smaller delay of low-$V_{th}$ devices and a smaller power consumption of high-$V_{th}$ devices, a balanced mix of both low-$V_{th}$ and high-$V_{th}$ devices may be used. The following two approaches can be used to reduce leakage power dissipation in the standby mode.

**VTCMOS Approach**

We have observed that low supply voltage along with low-threshold voltage provides a reduced overall power dissipation without a degradation in performance. However, the use of low-$V_{th}$ transistors inevitably leads to increased subthreshold leakage current, which is of major concern when the circuit is in standby mode. In many recent applications, such as cell phones, personal digital assistants (PDAs), etc., a major part of the circuit remains in standby mode most of the time. If the standby current is not low, it will lead to a shorter battery life.

VTCMOS [2, 3] circuits make use of the body effect to reduce the subthreshold leakage current, when the circuit is in normal mode. We know that the threshold voltage is a function of the voltage difference between the source and the substrate. The substrate terminals of all the n-channel metal–oxide–semiconductor (nMOS) transistors are connected to the ground potential and the substrate terminals of all the p-channel metal–oxide–semiconductor (pMOS) transistors are connected to $V_{dd}$, as shown in Fig. 9.6. This ensures that the source and drain diffusion regions always remain reversed-biased with respect to the substrate and the threshold voltages of the transistors are not significantly influenced by the body effect. On the other hand, in the case of VTCMOS circuits, the substrate bias voltages of nMOS and pMOS transistors are controlled with the help of a substrate bias control circuit, as shown in Fig. 9.7.
When the circuit is operating in the active mode, the substrate bias voltages for nMOS and pMOS transistors are $V_{Bn} = 0V$ and $V_{Bp} = V_{dd}$, respectively. Assuming $V_{dd} = 1V$, the corresponding threshold voltages for the nMOS transistors and pMOS transistors are $V_{tn} = 0.2V$ and $V_{tp} = -0.2V$, respectively. On the other hand, when the circuit is in standby mode, the substrate bias control circuit generates a lower substrate bias voltage of $V_{Bn} = -V_{B}$ for the nMOS transistor and a higher substrate bias voltage $V_{Bp} = V_{dd} + V_{B}$ for the pMOS transistors. This leads to an increase in threshold voltages for nMOS and pMOS transistors to $V_{in} = 0.5V$ and $V_{ip} = -0.5V$, respectively. This, in turn, leads to substantial reduction in subthreshold leakage currents because of the exponential dependence of subthreshold leakage current on the threshold voltage. It has been found that for every 100-mV increase in threshold voltage, the subthreshold leakage current reduces by half.

Although the VTCMOS technique is a very effective technique for controlling threshold voltage and reducing subthreshold leakage current, it requires a twin-well or triple-well CMOS fabrication technology so that different substrate bias voltages can be applied to different parts of the chip. Separate power pins may also be required if the substrate bias voltage levels are not generated on chip. Usually, the additional area required for the substrate bias control circuitry is negligible compared to the overall chip area.

### 9.4 Transistor Stacking

When more than one transistor is in series in a CMOS circuit, the leakage current has a strong dependence on the number of turned off transistors. This is known as the stack effect [4–6]. The mechanism of the stack effect can be best understood by
considering the case when all the transistors in a stack are turned off. Figure 9.8a shows four nMOS devices of a four-input NAND gate in a stack. The source and drain voltages of the MOS transistors obtained by simulation are shown in the figure. These voltages are due to a small drain current passing through the circuit. The source voltages of the three transistors on top of the stack have positive values. Assuming all gate voltages are equal to zero, the gate-to-source voltages of the three transistors are negative. Moreover, the drain-to-source potential of the MOS transistors is also reduced. The following three mechanisms come into play to reduce the leakage current:

i  Due to the exponential dependence of the subthreshold current on gate-to-source voltage, the leakage current is greatly reduced because of negative gate-to-source voltages.

ii The leakage current is also reduced due to body effect, because the body of all the three transistors is reverse-biased with respect to the source.

iii As the source-to-drain voltages for all the transistors are reduced, the subthreshold current due to drain-induced barrier lowering (DIBL) effect will also be lesser. As a consequence, the leakage currents will be minimum when all the transistors are turned off, which happens when the input vector is 0000. The leakage current passing through the circuit depends on the input vectors applied to the gate and it will be different for different input vectors. For example, for a three-input NAND gate shown in Fig. 9.8b, the leakage current contributions for different input vectors are given in Table 9.1. It may be noted that the highest leakage current is 99 times the lowest leakage current. The current is lowest when all the...
transistors in series are OFF, whereas the leakage current is highest when all the transistors are ON.

For a single gate, it is a trivial problem to find out the input vector that produces the minimum leakage current when the circuit is in standby mode. However, when the circuit is complex, it is a nontrivial problem to find out the input vector that produces the minimum leakage current. Many researchers have used models and algorithms to estimate the nominal leakage current of a circuit. The minimum and maximum leakage currents of a circuit have been estimated using suitable greedy heuristic. The set of input vectors which can put the circuits in the low-power standby mode with standby leakage power reduced by more than 50% are selected. Given a primary input combination, the logic value of each internal node can be obtained simply by starting from primary inputs and simulating the circuit level by level (level of a node is equal to the maximum of the level of its fan-in nodes plus 1, with the level of all primary inputs being 0). By applying the minimum-leakage producing input combination to the circuit when it is in the idle mode, we can significantly reduce the leakage power dissipation of the circuit. Consequently, the identification of a minimum leakage vector (MLV) is an important problem in low power design of VLSI circuits.

In view of the above discussion, researchers have attempted to make use of the stack effect to minimize leakage power dissipation in the standby condition. The following three basic approaches have been proposed by the researchers to select an appropriate input vector.

An input vector that maximizes the stack effect is determined so that the leakage power is minimized. This method is suitable for regular structure data path circuits such as adders, multipliers, etc. The best input vector is identified by using a suitable algorithm such as genetic algorithm.

Input vectors are generated randomly and applied to the circuit and leakage power dissipation is reduced. An input vector that minimizes the leakage current is selected.

A probabilistic approach eliminates the need to do simulations over all the $2^n$ input combinations, where $n$ is the number of circuit inputs. A small subset of all the possible states is evaluated for leakage.

<table>
<thead>
<tr>
<th>State (ABC)</th>
<th>Leakage current (nA)</th>
<th>Leaking transistors</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0.095</td>
<td>$Q_1$, $Q_2$, $Q_3$</td>
</tr>
<tr>
<td>001</td>
<td>0.195</td>
<td>$Q_1$, $Q_2$</td>
</tr>
<tr>
<td>010</td>
<td>0.195</td>
<td>$Q_1$, $Q_3$</td>
</tr>
<tr>
<td>011</td>
<td>1.874</td>
<td>$Q_1$</td>
</tr>
<tr>
<td>100</td>
<td>0.184</td>
<td>$Q_2$, $Q_3$</td>
</tr>
<tr>
<td>101</td>
<td>1.220</td>
<td>$Q_2$</td>
</tr>
<tr>
<td>110</td>
<td>1.140</td>
<td>$Q_3$</td>
</tr>
<tr>
<td>111</td>
<td>9.410</td>
<td>$Q_4$, $Q_5$, $Q_6$</td>
</tr>
</tbody>
</table>
MTCMOS Approach

Mutoh and his colleagues [7] introduced the MTCMOS approach to implement high-speed and low-power circuits operating from a 1-V power supply. In this approach, MOSFETs with two different threshold voltages are used in a single chip. It uses two operational modes—active and sleep for efficient power management. A basic MTCMOS circuit scheme is shown in Fig. 9.9. The realization of a two-input NAND gate is shown in the figure. The CMOS logic gate is realized with transistors of low-threshold voltage of about 0.2–0.3 V. Instead of connecting the power terminal lines of the gate directly to the power supply lines $V_{dd}$ and GND, here these are connected to the ‘virtual’ power supply lines (VDDV and GNDV). The real and virtual power supply lines are linked by the MOS transistor $Q_1$ and $Q_2$. These transistors have a high-threshold voltage in the range 0.5–0.6 V and serve as sleep control transistors. Sleep control signals $SL$ and $\overline{SL}$ are connected to $Q_1$ and $Q_2$, respectively, and used for active/sleep mode control. In the active mode, when $SL$ is set to LOW, both $Q_1$ and $Q_2$ are turned ON connecting the real power lines to VDDV and GNDV. In this mode, the NAND gate operates at a high speed corresponding to the low-threshold voltage of 0.2 V, which is relatively low compared to the supply voltage of 1.0 V. In the sleep mode, $SL$ is set to HIGH to turn both $Q_1$ and $Q_2$ OFF, thereby isolating the real supply lines from VDDV and GNDV. As the sleep transistors have a high-threshold voltage (0.6 V), the leakage current flowing through these two transistors will be significantly smaller in this mode. As a consequence, the leakage power consumption during the standby period can be dramatically reduced by sleep control. The performance of the MTCMOS approach is shown in Fig 9.10. In Fig. 9.10a, delay MTCMOS NAND gate as a function of supply voltage is compared with respect to conventional full high-$V_t$ and full low-$V_t$ logic gates. It is demonstrated that the delay of the MTCMOS gate is much smaller than that of a conventional CMOS gate with high $V_t$ and it is very close to the conventional CMOS with low-$V_t$ realization. With a power supply voltage of 1 V, the MTCMOS gate delay time is reduced by 70% as compared with the conventional CMOS gate with high-$V_t$. Moreover, the supply voltage dependence of an MTC-
MOS gate delay is much smaller than that of a conventional CMOS gate with high $V_t$ and MTCMOS gate operates as fast as the full low-$V_t$ gate. On the other hand, the variation of normalized power-delay product (NPDP) with the supply voltage is shown in Fig. 9.10b, where power consumption is normalized by frequency. At low voltage, say at 1 V, the NPDP of the MTCMOS gate is much lower than that of the conventional high-$V_t$ gates, reflecting the improved speed performance at lower voltage. The curve also shows that the power reduction effect is proportional to the square of the supply voltage overcoming speed degradation in a low-voltage operation. It is also claimed that the standby current is reduced by three or four orders of magnitude due to sleep control.

Two other factors that affect the speed performance of an MTCMOS circuit are: the width of the sleep control transistors and the capacitances of the virtual power line. The sleep transistors should have their widths large enough so that the ON resistances are small. It has been established by simulation that $W_t/W_L$ of 5 and $C_V/C_0$ of 5 leads to the decrease in $V_{eff}$ within 10% of $V_{dd}$ and the degradation in gate delay time within 15% compared to a pure low-$V_t$ CMOS as shown in Fig. 9.11. The area penalty is relatively small because this is shared by all the logic gates on a chip. So far as $C_V$ is concerned, the condition is satisfied by the intrinsic capacitances present and no external capacitance needs to be added.
The main advantage of MTCMOS is that it can be easily implemented using existing circuits, without modification of the cell library. But the MTCMOS approach suffers from the disadvantage that only the standby power is reduced and large inserted sleep transistors increase both area and delay. Some of the problems are overcome in the dual-$V_t$ approach presented in the following section.

**Power Gating [8]**

The basic approach of the MTCMOS implementation has been generalized and extended in the name of power management or power gating. Here, the basic strategy is to provide two power modes: an *active mode*, which is the normal operating mode of the circuit, and a low-power mode when the circuit is not in use. The low-power mode is commonly termed as *sleep mode*. At an appropriate time, the circuit switches from one mode to the other in an appropriate manner such that the energy drawn from the power source is maximized with minimum or no impact on the performance.

### 9.6.1 Clock Gating Versus Power Gating

We have discussed the use of the clock-gating approach for the reduction of switching power. The typical activity profile for a subsystem using a clock-gating strategy is shown in Fig. 9.12a. As shown in the figure, no dynamic power dissipation takes place when the circuit is clock-gated. However, leakage power dissipation takes place even when the circuit is clock-gated. In the early generation of CMOS circuits (above 250 nm), the leakage power was an insignificant portion of the total power. So, the power dissipation of a clock-gated subsystem was negligible. However, the leakage power has grown with every generation of CMOS process technology and it is essential to use power gating to reduce leakage power when the circuit is not in use. The activity profile for the same subsystem using a power-gating strategy is shown in Fig. 9.12b. Power gating can be implemented by inserting power-gating transistors in the stack between the logic transistors and either power or ground, thus creating a virtual supply rail or a virtual ground rail, respectively. The logic block contains all low-$V_{th}$ transistors for fastest switching speeds while the switch transistors, header or footer, are built using high-$V_{th}$ transistors to minimize the leakage power. Power gating can be implemented without using multiple thresholds, but it will not reduce leakage as much as if implemented with multiple thresholds. MTCMOS refers to the use of transistors with multiple threshold voltages in power-gating circuits. The most common implementations of power gating use a footer switch alone to limit the switch area overhead. High-$V_{th}$ NMOS footer switches are about half the size of equivalent-resistance high-$V_{th}$ PMOS header switches due to differences in majority carrier mobilities. Power gating reduces leakage by reducing the gate-to-source voltage, which in turn, drives the logic transistors deeper into
the cutoff region. This occurs because of the stack effect. The source terminal of the bottom-most transistor in the logic stack is no longer at ground, but rather at a voltage somewhat above ground due to the presence of the power-gating transistor. Leakage is reduced due to the reduction of the $V_{gs}$.

### 9.6.2 Power-Gating Issues

The clock-gating approach discussed earlier does not affect the functionality of the circuit and does not require changes in the resistor–transistor logic (RTL) representation. But, power gating is much more invasive because, as we shall see later, it affects inter-block interfaces and introduces significant time delays in order to safely enter and exit power-gated modes. The most basic form of power-gating control, and the one with the lowest long-term leakage power, is an externally switched power supply. An example is an on-chip CPU that has a dedicated off-chip power
supply, that is, the supply provides power only to the CPU. We can, then, shut down this power supply and reduce the leakage in the CPU to essentially zero. This approach, though, also takes the longest time and requires the most energy to restore power to a gated block. Internal power gating, where internal switches are used to control power to selected blocks, is a better solution when the blocks have to be powered down for shorter periods of time. Figure 9.13 illustrates an system on chip (SoC) that uses internal power gating.

Unlike a block that is always powered on, the power-gated block receives its power through a power-switching network. This network switches either VDD or VSS to the power-gated block. In this example, VDD is switched and VSS is provided directly to the entire chip. The switching fabric typically consists of a large number of CMOS switches distributed around or within the power-gated block. Various issues involved in the design of power-gated circuits are listed below:

- Power-gating granularity
- Power-gating topologies
- Switching fabric design
- Isolation strategy
- Retention strategy
- Power-gating controller design

9.6.2.1 Power-Gating Granularity

Two levels of granularity are commonly used in power gating. One is referred to as fine-grained power gating and the other one is referred to as coarse-grained power gating. In the case of fine-grained power gating, the power-gating switch is placed
locally as part of the standard cell. The switch must be designed to supply the worst-case current requirement of the cell so that there is no impact on the performance. As a consequence, the size of the switch is usually large (2 × to 4 × the size of the original cell) and there is significant area overhead.

In the case of coarse-grained power gating, a relatively larger block, say a processor, or a block of gates is power switched by a block of switch cells. Consider two different implementations of a processor chip. In the first case, a single sleep control signal is used to power down the entire chip. In the second case, separate sleep control signals are used to control different building blocks such as instruction decoder, execution unit, and memory controller. The former design is considered as coarse-grained power gating, whereas the latter design may be categorized as fine-grained power gating.

The choice of granularity has both logical and physical implications. A power domain refers to a group of logic with a logically unique sleep signal. Each power domain must be physically arranged to share the virtual ground common to that particular group (except for the boundary case of the switching-cell topology in which there are no shared virtual grounds). The motivation for fine-grained power gating is to reduce run-time leakage power, that is, the leakage power consumed during normal operation. While the coarse-grained example mentioned above will reduce leakage during standby, it will not affect run-time leakage since with a single sleep domain the power supply is either completely connected (active mode) or completely disconnected (standby mode). However, with fine-grained switching, portions of the design may be switched off while the other portions continue to operate. For example, in a multicore implementation with four cores, if only three of the cores are active, the fourth one may be put to sleep until such time as it is scheduled to resume computation.

Another advantage of fine-grained power gating is that the timing impact of the current I passing through a through a switch with equivalent resistance R resulting in IR drop across the switch can be easily characterized and it may be possible to use the traditional design flow to deploy fine-grained power gating. On the other hand, the sizing of the coarse-grained switched network is more difficult because the exact switching activity of the logic block may not be known at design time. In spite of the advantages of fine-grained power gating, the coarse-grained power gating is preferred because of its lesser area overhead.

**Power-Gating Topologies**

Another issue closely related to granularity of power gating is the power-gating topologies. Power-gating topologies can be categorized into three types:

- Global power gating
- Local power gating
- Switch in cell gating
Global Power Gating

Global power gating refers to a logical topology in which multiple switches are connected to one or more blocks of logic, and a single virtual ground is shared in common among all the power-gated logic blocks as shown in Fig. 9.14. This topology is effective for large blocks (coarse-grained) in which all the logic is power gated, but is less effective, for physical design reasons when the logic blocks are small. It does not apply when there are many different power-gated blocks, each controlled by a different sleep enable signal.

Local Power Gating

Local power gating refers to a logical topology in which each switch singularly gates its own virtual ground connected to its own group of logic. This arrangement results in multiple segmented virtual grounds for a single sleep domain as shown in Fig. 9.15.
Switch in Cell

Switch in cell may be thought of as an extreme form of local power-gating implementation. In this topology, each logic cell contains its own switch transistor as shown in Fig. 9.16. Its primary advantages are that delay calculation is very straightforward. The area overhead is substantial in this approach.

Switching Fabric Design

Although the basic concept of using sleep transistors for power gating is simple, the actual implementation of the switching fabric involves many highly technology-specific issues. First and foremost among the issues is the architectural issue to decide whether to use only header switch using pMOS transistors or use only footer switch using nMOS transistors or use both. Some researchers have advocated the use of both types of switches. However, for designs at 90 nm or smaller than 90 nm, either the header or footer switch is recommended due to the tight voltage margin, significant IR drop and large area, and delay penalties when both types of transistors are used. Various issues to be addressed for switching fabric design are:

- Header-versus-footer switch
- Power-gating implementation style

Header-Versus-Footer Switch

Figure 9.17a shows a header switch used for power gating. High-$V_t$ PMOS transistors are used to realize the header switch. Similarly, a footer switch used for power gating is shown in Fig. 9.17b, where high-$V_t$ NMOS transistors are used to realize the switch. Key issues affecting the choice are: architectural constraint, area overhead, and IR drop constraints. When external and internal power gating are used together, the header switch is the most appropriate choice. The header switch
is also suitable if multiple power rails and/or voltage swings are used on the chip. The switch efficiency of sleep transistors is defined as the ratio of drain current in the ON and OFF states (I\text{on}/I\text{off}). The maximum value of switch efficiency is desirable to achieve a high current drive in normal operation and low leakage in sleep condition. It has been found that for the same drive current, the header switch using pMOS transistors results in 2.67 times more leakage current than the footer switches realize using nMOS transistors.

**Implementation Styles**

Implementation of power-gating switches can be broadly categorized into two types: *ring* and *grid* styles. In ring-style implementation, the switches are placed external to the power-gated block by encapsulating it by a ring of switches as shown in Fig. 9.18. The switches connect VDD to the virtual VVDD of the power-gated block. This is the only style that can be used to supply power to an existing hard block by placing the switches outside it. On the other hand, the switches are distributed throughout the power-gated region as shown in Fig. 9.19. Any one of the styles can be used for the implementation of coarse-grain power gating.

**9.7 Isolation Strategy**

Ideally, the outputs and internal nodes of a header-style power-gated block should collapse down towards ground level. Similarly, the outputs and internal nodes of a footer-style power-gated block should collapse down towards supply rail. However, in practice, the outputs and internal nodes may neither discharge to ground level nor fully charge to supply voltage level, because of finite leakage currents.
Fig. 9.18 Ring-style switching fabric

Fig. 9.19 Grid-style switching fabric
Fig. 9.20 Output of a power-gated block driving a power-up block

passing through the off switches. So, if the output of power-down block drives a power-up block, there is possibility of short-circuit power (also known as crowbar power) in the power-up block as shown in Fig. 9.20. It is necessary to ensure that the floating output of the power-down block does not result in spurious behaviour of the power-up block. This can be achieved by using an isolation cell to clamp the output of power-down block to some well-defined logic level. Isolation cells can be categorized into three types: cells those clamp the output to ‘0’ logic level, cells those clamp to ‘1’ logic level and cells those clamp the output to the most recent value. To which value the output should be clamped is based on the inactive state of the power-up block. In the case of active high logic, the output is clamped to logic ‘0’ and in case of low-active logic, the output is clamped to logic ‘1’ level. When the power-down block is driving a combinational logic circuit, the output can be clamped to a particular value that reduces the leakage current using stack effect (refer to Sect. 9.4). Figure 9.21 shows how the output of a power-gated block is clamped with the help of an AND gate. Isolation cell to clamp the output to ‘0’ logic

Fig. 9.21 AND gate to clamp the output to LOW level
level can be accomplished using an AND gate as shown in Fig. 9.22a. The output is clamped to ‘0’ as long as the isolation (ISOLN) signal is active.

Similarly, isolation cell to clamp the output to ‘1’ logic level can be accomplished using an OR gate as shown in Fig. 9.22b. It can be realized using a NOR gate and an inverter. The output is forced to ‘1’ as long as the ISOL signal is held high. An alternative approach is to use pull-up or pull-down transistors to avoid full gate delay as shown in Fig. 9.23. If we want to clamp the output to the last value, it is necessary to use a latch to hold the last value.

**State Retention Strategy**

Given a power switching fabric and an isolation strategy, it is possible to power gate a block of logic. But unless a retention strategy is employed, all state information is lost when the block is powered down. To resume its operation on power up, the block must either have its state restored from an external source or build up its state from the reset condition. In either case, the time and power required can be significant. One of the following three approaches may be used:

- A software approach based on reading and writing registers
- A scan-based approach based on the reuse of scan chains to store state off chip
- A register-based approach that uses retention registers

**Software-Based Approach** In the software approach, the always-ON CPU reads the registers of the power-gated blocks and stores in the processor’s memory. During power-up sequence, the CPU writes back the registers from the memory. Bus traffic slows down the power-down and power-up sequence and bus conflicts may make powering down unviable. Software must be written and integrated into the system’s software for handling power down and power up.
Scan-Based Approach  Scan chains used for built-in self-test (BIST) can be reused. During power-down sequence, the scan register outputs are routed to an on-chip or off-chip memory. In this approach, there can be significant saving of chip area.

Retention Registers  In this approach, standard registers are replaced by retention registers. A retention register contains a shadow register that can preserve the registers state during power down and restore it at power up. High-$V_t$ transistors are used in the slave latch, the clock buffers, and the inverter that connects the master latch to the slave latch as shown in Fig. 9.24. In addition to area penalty, this approach requires more complex power controller.

Power-Gating Controller

A key concern in controlling the switching fabric is to limit the in-rush of current when power to the block is switched on. An excessive in-rush current can cause voltage spikes on the supply, possibly corrupting registers in the always-on blocks, as well as retention registers in the power-gated block. One representative approach is to daisy-chain the control signal to the switches. The result of this daisy chaining is that it takes some time from the assertion of a ‘power-up’ signal until the block is powered up. A more aggressive approach to turning on the switching fabric is to use several power-up control signals in sequence. Regardless of the specific control method, during the power-up sequence, it is important to wait until the switching fabric is completely powered up before enabling the power-gated block to resume normal operation. A realistic activity profile for power gating is shown in Fig. 9.25.

Power-gating control without retention is shown in Fig. 9.26. The following sequence is followed at the time of switching OFF:

- Flush through any bus or external operation in progress
- Stop the clock at appropriate phase
Fig. 9.25 Activity profile with realistic power gating

Fig. 9.26 Power-gating control without retention

- Assert the isolation control signal
- Assert reset to the block
- Assert the power-gating control signal to power down the block

At the time of switching ON, the same signals are sent in reverse order as follows:

- De-assert the power-gating control signal
- De-assert reset to the block
- De-assert the isolation control signal to restore all outputs
- Restart the clocks, without glitches and without violating minimum pulse width design constraints

Power-gating control with retention is shown in Fig. 9.27. The following sequence is followed at the time of switching OFF:

- Flush through any bus or external operation in progress
- Stop the clock at appropriate phase
- Assert the isolation control signal
- Assert the state retention save condition
- Assert reset to the block
- Assert the power-gating control signal to power down the block
At the time of switching ON, the same signals are sent in reverse order as follows:

- De-assert the power-gating control signal to power back up the block
- De-assert reset to ensure clean initialization following the gated power-up
- Assert the state retention restore condition
- De-assert the isolation control signal to restore all outputs
- Restart the clocks, without glitches and without violating minimum pulse width design constraints

### Power Management

The basic idea of power management stems from the fact that all parts of a circuit are not needed to function all the time. The power management scheme can identify conditions under which either certain parts of the circuit or the entire circuit can remain idle and shut them down to reduce power consumption. For example, the most conventional approach used in the X86-compatible processors is to regulate the power consumption by rapidly altering between running the processor at full speed and turning the processor off. A different performance level is achieved by varying the on/off time (duty cycle) of the processor. The mobile X86 power management model provides one ‘ON’ state corresponding to normal (100 %) activity level and multiple states as shown in Table 9.2. It is based on the advanced configuration and power interface (ACPI), a standard developed by Microsoft, Intel, and Toshiba [9].

The relationship between the apparent performance level and the power consumption is shown in Fig. 9.28. By decreasing the amount of time the processor spends in the active ‘ON’ state, the performance level as well as the power consumption level can be decreased linearly as shown in the figure.

However, this approach delivers performance in discrete bursts, which tends to be unsuitable for smooth multimedia content such as software-based DVD or MP3 playback. This may create artefacts, such as dropped frames during movie playback.
## Table 9.2 Traditional power management states

<table>
<thead>
<tr>
<th>State name</th>
<th>ACPI state name</th>
<th>State description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>C0</td>
<td>In the normal state, the processor executes instruction with 100% activity</td>
</tr>
<tr>
<td>Auto halt</td>
<td>C1</td>
<td>The processor enters auto halt state by executing HLT instruction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In the auto halt state, the processor stops its internal clocks in response to HLT instruction</td>
</tr>
<tr>
<td>Quick start</td>
<td>C2</td>
<td>The processor stops its internal clocks in response to the STP-CLK signal from the south bridge</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The processor maintains cache coherence (the processor caches continue to snoop system memory transaction)</td>
</tr>
<tr>
<td>Deep sleep</td>
<td>C3</td>
<td>The south bridge stops the external clock input to the processor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The system enforces cache coherency (the processor caches do not need to snoop system memory transactions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>It realizes the maximum saving in processor power savings without losing the processor content (internal state)</td>
</tr>
</tbody>
</table>

*ACPI* advanced configuration and power interface, *HLT* halt, *STPCLK* stop clock

![Fig. 9.28 Linear power savings of conventional power management](image)

that are perceptible to a user. In such a situation, the dynamic voltage and frequency scaling (DVFS) technique can be used to match the desired performance, giving power reduction to follow cubic relationship as shown in Fig. 9.29. This approach always drives the processor frequency down to the point where the traditional sleep states almost disappear.

### 9.10.1 Combining DVFS and Power Management

However, the DVFS ranges hit the limits of physics and the curve flattens out, when the minimum frequency–voltage point is reached. At this point, it is more efficient
to switch over to traditional power management, i.e. alternating between normal and sleep states according to the performance demands of the circuit as shown in Fig. 9.30.

9.11 Dual-$V_t$ Assignment Approach (DTCMOS) [10]

The availability of two or more threshold voltages on the same chip has opened up a new opportunity of making tradeoffs between power and delay. Unlike the MTCMOS approach, where additional sleep transistors are used to reduce leakage power in standby mode, two or more threshold voltages may be assigned to transis-
tors realizing functional gates in the circuit. This not only avoids additional area and delay due to the inserted sleep transistors, but power reduction takes place both in standby and active mode of operation of the circuit.

As we know, the delay of a logic gate increases with the threshold voltage of the transistors shown in Fig. 9.2b. The static power due to the subthreshold leakage current, on the other hand, decreases with the increase in the threshold voltage as shown in Fig. 9.2a. So, by the judicious use of both low and high-\(V_t\) transistors in a circuit, one can try to realize digital circuits with speed corresponding to the low-\(V_t\) transistors and power consumption corresponding to the high-\(V_t\) transistors. The key idea is to use low-\(V_t\) transistors for realizing gates on the critical path so that there is no degradation in performance and high-\(V_t\) transistors in realizing gates off the critical path such that leakage power consumption is significantly lower.

This is illustrated with the help of an example circuit shown in Fig. 9.31. Figure 9.31a is the original single-\(V_t\) circuit, where the supply voltage is 1 V and threshold voltage is 0.2 V. Gates on the critical paths are shown by dark shade. Figure 9.31b shows a dual-\(V_t\) circuit, in which a high \(V_t\) of 0.25 V is assigned to all the nodes on the off-critical path. The same circuit is shown in Fig. 9.32c and d with a high-\(V_t\) of 0.396 V and 0.46 V, respectively. Note that to maintain the delay less than or equal to the critical path, some nodes on the off-critical path are assigned with low \(V_t\). Figure 9.32 shows the standby leakage power of the example circuit with different high-\(V_t\) values in the range of 0.2–0.5 V. The diagram shows that there exists an optimum high-\(V_t\) value for which the standby leakage power is minimum. This has opened up a problem known as dual-\(V_t\) assignment problem. The goal is to obtain an optimum value of high \(V_t\) such that it can be assigned to the maximum number of gates to realize circuits requiring minimum standby leakage power.

Depending on the application for which the circuit is designed, there are two alternative approaches—delay-constrained and energy-constrained realizations. In the first case, the circuit is designed for high-performance applications by optimizing power dissipation with constraint on delay, i.e. without compromise in performance. This problem has been addressed in several recent research publications [10–14]. The basic approach used in these works is to realize circuits using low-threshold voltage transistors (low-\(V_t\)) for gates on the critical path and high-threshold voltage transistors (high-\(V_t\)) for gates on the off-critical path of the circuit under realization. It has been demonstrated that significant savings in the leakage power for the optimized circuit is possible based on this approach.

In the second case, the circuit is designed for battery-operated, hand-held and portable systems, where the energy requirement for the circuit is minimized at the expense of performance. A preliminary work on this was reported in [13]. Based on the observation that a small compromise in performance (increase in delay) leads to significant reduction in leakage power, a novel approach has been developed for the synthesis of dual-\(V_t\) CMOS circuit. This may be termed as the energy-constrained approach.

In both the cases, the key issue is how efficiently the two threshold voltages are assigned to the MOS transistors of different gates to minimize power (energy) consumption of the realized circuits.
Fig. 9.31  a Darker gates on the critical path, b high $V_t = 0.25$ assigned to all gates in the off-critical path, c high $V_t = 0.396$ assigned to some gates in the off-critical path, and d high $V_t = 0.46$ assigned to some gates in the off-critical path.
Fig. 9.32 Standby leakage power for different $V_{th2}$

![Graph showing standby leakage power vs. $V_{th2}$]

Fig. 9.33 Dual-$V_t$ CMOS circuit

![Diagram of a circuit with critical path highlighted]

Assigned with high-$V_t$ transistors

9.12 Delay-Constrained Dual-$V_t$ CMOS Circuits [12]

The basic approach in dual-$V_t$ CMOS circuits is to use two threshold voltages ($0.2V_{dd} \leq V_t \leq 0.5V_{dd}$) instead of single threshold voltage so that both low power and high performance circuits can be realized. In a dual-$V_t$ CMOS circuit, as shown in Fig. 9.33, low $V_t$ is assigned to the transistors for the gates on the critical paths to obtain a smaller delay and a high $V_t$ is assigned to all the transistors for the gates on the off-critical path to get a smaller power dissipation. It may be noted that all gates on the off-critical paths cannot be assigned with high $V_t$ because critical paths might change and hence the delay. However, the savings in leakage power in the said approach depends on how many transistors can be assigned with high $V_t$; larger the number of transistors assigned with high-$V_t$, lower is the leakage power. But the assignment of dual-$V_t$ to get maximum savings in leakage power is an NP-complete problem.

In the synthesis of delay-constrained dual-$V_t$ CMOS circuits, the problem is to reduce the standby leakage power subject to the constraint of not increasing the delay. In [12], a heuristic-based algorithm has been presented which, for a given circuit with a fixed supply voltage (1 V) and a fixed low-threshold voltage $V_L$ (0.2V) for gates on the critical path, finds a ‘static power optimum’ high-threshold voltage $V_H$, and selects a subset of the gates on the off-critical path which can be switched to $V_H$. 

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{dd}$</td>
<td>1V</td>
</tr>
<tr>
<td>$L_{eff}$</td>
<td>0.32μm</td>
</tr>
<tr>
<td>$W_{peff}$</td>
<td>10.5μm</td>
</tr>
<tr>
<td>$W_{neff}$</td>
<td>3μm</td>
</tr>
<tr>
<td>$Tox$</td>
<td>9.8nm</td>
</tr>
</tbody>
</table>
This approach demonstrated significant savings in the leakage power for the dual-$V_t$ CMOS circuits without degradation in performance. However, this technique employed simple backward breadth-first search (BFS) strategy to identify the subset of gates that can be switched to $V_t^H$ examining slacks. So far as the computational effort is concerned, this heuristic-based approach is fast but selects fewer gates for assigning $V_t^H$ on the off-critical path. Another approach has been proposed, called delay balancing method, which is based on the insertion of buffers (specific delay fictitious (SDF) buffer) into the circuit by transforming a circuit graph $G$ into its functionally equivalent circuit $G'$ so that every wire has the zero edge-slag in $G'$ and it then assigns high-threshold voltage to gates in order to eliminate all these SDF buffers. However, in the dual-$V_t$ assignment, an algorithm is formulated with an integer linear programming framework and thus takes an inordinate amount of CPU time for large circuits. Moreover, this technique is believed to give good results for synthesizing CMOS circuits when arbitrary numbers of threshold voltages are allowed instead of two threshold voltages used in dual-$V_t$ CMOS circuits. Here, the same problem has been addressed and developed a better heuristic-based algorithm such that $V_t^H$ is assigned to more number of gates leading to larger savings in leakage power.

Algorithms
A directed acyclic graph (DAG) representation $G(V,E)$ of a gate-level combinatorial circuit is assumed, where $V$ is a set of nodes representing the gates and $E$ is a set of edges representing the interconnection among the gates. The graph is first initialized by assigning the values of $t_p(x)$, the propagation delay of a node $x$, $T_f(x)$, the arrival time (this is the propagation delay of each fan-in path of a node $x$), $T_d(x)$, the departure time, and $T_{\max}(x)$, the maximum propagation delay time for each node $x$ in $G$. For each primary input $x$, $t_p(x) = 0$, $T_f(x) = 0$, $T_d(x) = 0$, and $T_{\max}(x) = 0$. For all other nodes in the graph, $T_{\max}(x)$ and $T_f(x)$ can be computed as given below:

$$T_{\max}(x) = \max \{ T_a(x) | i \in \text{fanin}(x) \}$$

$$T_f(x) = T_{\max}(x) + t_p(x).$$

The delay of a circuit is defined as the longest propagation time through a critical path. In dual-$V_t$ CMOS circuits, the minimum delay occurs when all transistors in the circuit are assigned with a low-threshold voltage ($V_t^L = 0.2V_d$). Here, $V_t^H$ is initially assigned to the transistors of all gates. In this initial condition, the circuit consumes minimum power but incurs maximum delay. In order to reduce the delay, the procedure selects a gate on the critical path in depth-first fashion such that it can be assigned with $V_t^L$. Assign $V_t^L$ to all transistors in that gate. As this assignment may alter the critical path, all critical paths are computed and then iterate low-threshold voltage assignment until there does not exist any gate on the critical path(s) not assigned with $V_t^L$. When all nodes in the critical paths are assigned with $V_t^L$, the circuit eventually has minimum delay. The pseudo code for assigning an arbitrary value of high-threshold voltage ($V_t^H$) to all transistors in the majority of gates on the off-critical paths is given below.
Algorithm Dual\(_{V_T}\)Assignment (\(V^i\), \(V^L\))

1. Initialize all the nodes with \(V^i\).
2. Compute the critical path(s) in the circuit.
3. Select a node on the critical path(s) in depth first fashion which is not assigned with \(V^i\). If there is no such node then go to Step 6.
4. Assign the node with \(V^L\).
5. Go to Step 2.
6. Stop.

The algorithm Dual\(_{V_T}\)Assignment is based on depth-first search with iteration heuristic and assigns \(V^L\) to transistors for gates on the critical paths and a high-threshold voltage \(V^i\) to transistors for gates on off-critical paths. It has been observed [10] that for a given circuit and with a \(V^L = 0.2V_{DD}\), the standby leakage power varies with the value of \(V^L\). Or, in other words, there is an optimal value of \(V^i\) for which maximum number of nodes can be assigned with high \(V^i\) so that minimum leakage power is possible (Fig. 9.32). To select the optimal value for \(V^H\), the dual-\(_{V_T}\) assignment procedure is repeated with various values of \(V^i\) (0.2\(V_{DD}\) < \(V^i\) ≤ 0.5\(V_{DD}\)), as given in the algorithm Delay Constrained Dual\(_{V_T}\)Assignment.

Algorithm Delay Constrained Dual \(_{V_T}\)Assignment

1. Let \(V^i_T = 0.2V_{DD}\)
2. Initialize all the nodes with \(V^i_T\)
3. Calculate the leakage power \(P_{min}\)
4. While \(V^i_T \leq 0.5V_{DD}\) do
   Dual\(_{V_T}\)Assignment (\(V^i\), 0.2\(V_{DD}\))
   Calculate the leakage power \(P\)
   If \(P < P_{min}\) then \(P_{min} = P\), \(V^H_T = V^i_T\)
   Increment the value of \(V^i_T\) by small amount
5. Dual\(_{V_T}\)Assignment (\(V^H_T\), 0.2\(V_{DD}\))
6. Stop

The delay-constrained dual-\(_{V_T}\) assignment algorithm as proposed in [12] produces better results than the algorithm proposed in [10]. The algorithm in [10] proceeds in a specific order (BFS-based backtracking from the primary outputs) of searching the nodes which can be assigned a higher threshold voltage. In fact, it distributes the inherited slack over a fewer number of gates than the algorithm proposed in [12]. The modified algorithm searches the critical path (which keeps on changing) for a node with maximum saving and hence does not follow any particular order of searching. Figure 9.34 shows the variation of leakage power for different threshold voltages. It may be noted that there is an optimal threshold voltage for which the
leakage power is minimum. Now, let us consider Fig. 9.35 as an example. The slack present with the node $y$ is to be distributed among the nodes in its transitive fan-in cone. The algorithm in [10] would assign the slack to node $x$ itself while checking, if $x$ can be assigned a high value for $V_t$. On the other hand, the Dual-$V_t$ Assignment assigns the slack not to $x$, but rather to each of $i$, $j$, $k$, and $l$. Thus, the algorithm proposed in [12] assigns the higher threshold to four gates as opposed to one in [10]. However, the time complexity of algorithm in [10] is $O(n)$. Actually, the order is that of BFS on a graph, and since the graph has maximum of $8n$ (maximum of four edges for forward and the same for backward) edges, the complexity of $O(|V|+|E|)$ degenerates to $O(|V|)$. On the other hand, the time complexity of the algorithm [12] is $O(n^3)$. This is because finding the critical path takes $O(n)$ time and in each iteration the critical path has to be found, which keeps on changing. As a consequence, the proposed algorithm takes longer computation time. With the availability of faster computers, it does not appear to be a serious limitation. Moreover, experimental results show that even for a reasonably large benchmark circuit, the overall time requirement is about double that of [10].
9.13 Energy-Constrained Dual-$V_t$ CMOS Circuits[13]

To understand how the leakage energy changes with threshold voltage, let us have a look at Fig. 9.36. The graph in Fig. 9.36 shows the change in leakage energy with the delay, which depends on the threshold voltage of the transistors. For the entire range of threshold voltage variations, the curve looks like a typical bathtub curve. Initially, the energy decreases rapidly and for a small increase in delay, there is a large reduction in leakage power. After this initial part, the leakage energy remains more or less constant. This is due to the fact that the reduction in leakage power and increase in delay take place at the same ratio. Subsequently, when the threshold voltage approaches the supply voltage, there is sharp increase in delay with a very small reduction in leakage power leading to an increase in leakage power dissipation. From the leakage energy versus delay graph, a decrease in energy is observed, and there is a point beyond which there is very small decrease in leakage for a large increase in delay. This point in the graph corresponds to the slope $\frac{\Delta E}{\Delta T} = -1$ and leakage energy at this point is very near to minimum leakage energy possible as shown in Fig. 9.36. As reported in [13], simulation study reveals the fact that, for most of the circuits, the increase in delay is within 4–9 % for which the decrease in leakage energy is around 70 %. Thus, if the delay is increased by a small percentage than the minimum delay, then a significant amount of leakage energy can be saved. In this alternative approach of synthesizing dual-$V_t$ CMOS circuits, performance by a small amount is traded for near minimal energy in dual-$V_t$ CMOS circuits. The pseudo code for energy-constrained dual-$V_t$ CMOS circuits is given below.
Algorithm Energy Constrained Dual V<sub>T</sub> Assignment

1. Initialize all nodes with V<sub>T</sub><sup>L</sup> = 0.2V<sub>DD</sub>.
2. Calculate Leakage energy E<sub>1</sub> and delay T<sub>1</sub>.
3. Let T<sub>2</sub> = T<sub>1</sub> + Δ
4. Calculate V<sub>T</sub><sup>L</sup> corresponding to this delay T<sub>2</sub>.
5. Assign all nodes with V<sub>T</sub><sup>L</sup>.
6. Calculate leakage energy E<sub>2</sub>.
7. Calculate slope \[ \frac{\Delta E}{\Delta T} = \frac{E_2 - E_1}{T_2 - T_1} \]
8. T<sub>1</sub> = T<sub>2</sub>, E<sub>1</sub> = E<sub>2</sub>
9. If \[ \frac{\Delta E}{\Delta T} \neq -1 \] then Go to Step 3.
10. Let V<sub>T</sub><sup>i</sup> = V<sub>T</sub><sup>L</sup>
11. Initialize all the nodes with V<sub>T</sub><sup>i</sup>
12. Calculate the leakage energy E<sub>min</sub>
13. While V<sub>T</sub><sup>i</sup> ≤ 0.5V<sub>DD</sub> do
Dual V<sub>T</sub> Assignment (V<sub>T</sub><sup>i</sup>, V<sub>T</sub><sup>L</sup>)
\[ \text{Calculate leakage energy } E \]
\[ \text{If } E < E_{\text{min}} \text{ then } E_{\text{min}} = E, \quad V_{T}^{H} = V_{T}^{i} \]
Increment the value of V<sub>T</sub><sup>i</sup> by small amount
14. Dual V<sub>T</sub> Assignment (V<sub>T</sub><sup>H</sup>, V<sub>T</sub><sup>L</sup>)
15. Stop

The algorithm Energy Constrained Dual-V<sub>T</sub> Assignment consists of two phases. In the first phase, it calculates the value of low-threshold voltage V<sub>T</sub><sup>L</sup> for which \[ \frac{\Delta E}{\Delta T} = -1 \] and in the next phase, taking this as low-threshold voltage, the dual-V<sub>T</sub> assignment procedure is iterated to find an optimal value of high-threshold voltage (V<sub>T</sub><sup>H</sup>), which provides minimum leakage energy.

Implementation
The algorithms were implemented in a C programming environment on a SUN Ultra Spark 10 workstation with 256 MB cache memory. The Berkeley SIS software tool has been employed to make the net-list for a given circuit. For a particular benchmark circuit, it is first optimized with script.rugged code in SIS and then using the standard library 44–1.genlib in SIS, gate mapping is done with the option of minimum area. In the output file generated by SIS tool, the circuit is realized by
Table 9.3 Leakage power dissipation in delay-constrained dual-$V_t$ CMOS circuits

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Active mode (μW)</th>
<th>Standby mode (μW)</th>
<th>% Redn.</th>
<th>% Redn.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single $V_t$</td>
<td>Dual $V_t$</td>
<td></td>
<td>Single $V_t$</td>
</tr>
<tr>
<td>C432</td>
<td>48.88</td>
<td>7.23</td>
<td>85.21</td>
<td>0.191</td>
</tr>
<tr>
<td>C499</td>
<td>96.44</td>
<td>40.06</td>
<td>58.46</td>
<td>0.177</td>
</tr>
<tr>
<td>C880</td>
<td>97.26</td>
<td>9.68</td>
<td>86.60</td>
<td>0.252</td>
</tr>
<tr>
<td>C1355</td>
<td>120.08</td>
<td>54.91</td>
<td>54.27</td>
<td>1.51</td>
</tr>
<tr>
<td>C1908</td>
<td>119.39</td>
<td>20.02</td>
<td>83.23</td>
<td>0.609</td>
</tr>
<tr>
<td>C2670</td>
<td>167.08</td>
<td>15.59</td>
<td>90.67</td>
<td>0.364</td>
</tr>
<tr>
<td>C3540</td>
<td>209.70</td>
<td>24.64</td>
<td>88.25</td>
<td>0.620</td>
</tr>
<tr>
<td>C5315</td>
<td>406.56</td>
<td>50.09</td>
<td>87.68</td>
<td>1.29</td>
</tr>
<tr>
<td>C6288</td>
<td>520.24</td>
<td>104.63</td>
<td>79.89</td>
<td>2.65</td>
</tr>
<tr>
<td>C7552</td>
<td>521.47</td>
<td>75.69</td>
<td>85.48</td>
<td>2.01</td>
</tr>
</tbody>
</table>

using only NAND, NOR, and NOT gates. In order to simplify the analysis, the first two types of gates with two to four inputs are taken. The mapped circuit is then translated into a bi-directed graph using a parser and a graph generator program.

To calculate the delay, leakage power, and dynamic power, the estimation models as proposed in [11] is used. Energy is measured as the product of power and delay. For the calculation of power, the transition activity has been assumed to be 0.3 for all inputs. Although it is assumed that transition activity is 0.3, it can be as low as 0.1 in many situations, leading to smaller switching power requirement. The temperature is assumed as 110°C for the active mode and it is assumed to be 25°C for the standby mode of operation of the circuit. The supply voltage is taken as the 1.0 V and zero-biased threshold voltage used in our experiments is 0.2 V. It has been observed that the threshold voltage increases by about 0.8 mV for every 1 °C decrease in temperature. The threshold voltage at standby mode is taken as 68 mV higher than the threshold voltage at active mode. The values of various transistor parameters have been extracted using the BSIM3V3 model and for the TSMC 0.18 μm process technology. The effective channel length of the transistor is taken as the 0.18 μm and the gate oxide thickness is taken as 40 Å. For simplicity, all transistors are assumed to have the same channel length of 0.18 μm while the channel widths for nMOSFETs and pMOSFETs are assumed to be 0.54 and 1.62 μm, respectively. The subthreshold swing coefficient is taken as 1.44 and the body effect coefficient and DIBL coefficients are 0.03 and 0.21 for nMOSFETs and 0.02 and 0.11 for pMOSFETs, respectively. The value of $\alpha$ is taken as 1.3 (for short-channel MOSFETs). However, our experiments are not limited to such assumptions.

Experimental Results

Experimental results for the synthesis of delay-constraint dual-$V_t$ CMOS circuits based on the algorithm Delay Constrained Dual-$V_t$ Assignment are presented in Tables 9.3 and 9.4. Comparative charts are presented, as shown in Fig. 9.37, to depict how the reduction of leakage power occurs in static CMOS circuits synthesized
Table 9.4 Total power dissipation during active mode

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Delay (ns)</th>
<th>Switching power at load (µW)</th>
<th>Switching power at internal nodes</th>
<th>Total switching power (µW)</th>
<th>Total power</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Single $V_t$</td>
<td>Dual $V_t$</td>
<td>Single $V_t$</td>
</tr>
<tr>
<td>C432</td>
<td>3.32</td>
<td>98.88</td>
<td>23.51</td>
<td>20.34</td>
<td>122.39</td>
</tr>
<tr>
<td>C499</td>
<td>2.23</td>
<td>308.34</td>
<td>58.98</td>
<td>56.55</td>
<td>367.32</td>
</tr>
<tr>
<td>C880</td>
<td>2.21</td>
<td>237.61</td>
<td>55.59</td>
<td>47.22</td>
<td>293.2</td>
</tr>
<tr>
<td>C1355</td>
<td>2.61</td>
<td>325.20</td>
<td>75.18</td>
<td>71.93</td>
<td>400.38</td>
</tr>
<tr>
<td>C1908</td>
<td>2.91</td>
<td>302.59</td>
<td>64.49</td>
<td>58.02</td>
<td>367.08</td>
</tr>
<tr>
<td>C2670</td>
<td>2.94</td>
<td>409.47</td>
<td>83.82</td>
<td>71.53</td>
<td>493.29</td>
</tr>
<tr>
<td>C3540</td>
<td>4.57</td>
<td>307.21</td>
<td>101.43</td>
<td>86.83</td>
<td>408.64</td>
</tr>
<tr>
<td>C5315</td>
<td>3.65</td>
<td>673.23</td>
<td>157.20</td>
<td>129.53</td>
<td>830.43</td>
</tr>
<tr>
<td>C6288</td>
<td>11.84</td>
<td>335.30</td>
<td>73.69</td>
<td>67.21</td>
<td>408.99</td>
</tr>
<tr>
<td>C7552</td>
<td>2.99</td>
<td>1361.33</td>
<td>243.32</td>
<td>213.54</td>
<td>1604.65</td>
</tr>
</tbody>
</table>

Fig. 9.37 Reduction of leakage power in active mode in delay-constrained realization comparing leakage power for all low-$V_t$, dual-$V_t$, and all high-$V_t$ circuits

for three cases: all low $V_t$, dual $V_t$, and all high $V_t$. The dissipation of leakage power in the active mode and standby mode are presented in Table 9.3. The leakage power dissipations for single-$V_t$ CMOS circuits are shown in column 2 and column 5 of Table 9.3 in active mode and standby mode, respectively. In Table 9.3, column 3 and column 6 present the leakage power requirements in active mode and standby mode, respectively. The percentage reduction of leakage power in dual-$V_t$ circuits with respect to single-$V_t$ realization in active mode and standby mode operations is shown in column 4 and column 7 of Table 9.4, respectively. It is observed that, on an average, percentage reduction of leakage power in delay-constrained dual-$V_t$ CMOS circuits are 79.97 and 82.82 % in active mode and standby mode, respectively compared to single low-$V_t$ CMOS circuits.
Table 9.5 Leakage energy dissipation in energy-constrained dual-V<sub>t</sub> CMOS circuits

| Benchmark | Active mode (fJ) | | | Standby mode (fJ) | | | | |
| --- | --- | --- | --- | --- | --- | | | --- | --- | --- | --- | --- | --- | |
| | Single V<sub>t</sub> | Dual V<sub>t</sub> | % Redn. | Single V<sub>t</sub> | Dual V<sub>t</sub> | % Redn. | | | | | | | |
| C432 | 162.73 | 11.82 | 92.74 | 5.05 | 0.243 | 95.19 | | | | | | | | |
| C499 | 215.43 | 42.86 | 80.10 | 6.72 | 0.918 | 86.34 | | | | | | | | |
| C880 | 160.37 | 10.56 | 93.41 | 4.93 | 0.214 | 95.66 | | | | | | | | |
| C1355 | 313.49 | 69.11 | 77.95 | 9.85 | 1.51 | 84.67 | | | | | | | | |
| C1908 | 347.38 | 47.35 | 86.37 | 10.72 | 1.04 | 90.30 | | | | | | | | |
| C5315 | 1484.99 | 88.85 | 94.02 | 45.71 | 1.82 | 96.32 | | | | | | | | |
| **C6288** | 6164.91 | 612.11 | 90.07 | 192.62 | 12.22 | 93.66 | | | | | | | | |
| **C7552** | 1562.71 | 132.26 | 91.54 | 48.77 | 2.80 | 94.26 | | | | | | | | |

In Table 9.6, the total power (sum of the switching power and leakage power) requirement in delay-constrained dual-V<sub>t</sub> CMOS circuits is presented. The delay of the circuits is shown in column 2 of Table 9.4. The total switching power dissipated at the load capacitances is shown in column 3 of Table 9.4. The switching power dissipations at internal node capacitances in single-V<sub>t</sub> and dual-V<sub>t</sub> CMOS circuits are shown in column 4 and column 5 of Table 9.4, respectively. The total switching power is the sum of the switching power at load capacitances and at internal node capacitances as shown in column 6 and column 7 of Table 9.4 for single-V<sub>t</sub> and dual-V<sub>t</sub> CMOS circuits, respectively. The total power consumption, which includes total switching power and leakage power of the circuits with single-V<sub>t</sub> and delay-constrained dual-V<sub>t</sub> realizations, are shown in column 8 and column 9 of Table 9.4, respectively. It can be observed that on an average, percentage reduction of total power in delay-constrained dual-V<sub>t</sub> CMOS circuits with respect to single-V<sub>t</sub> CMOS circuits is 24.92%. 

Table 9.6 Total energy requirement during active mode

| Benchmark | Delay (ns) | % increase in delay | Switching energy at load (fJ) | Switching energy at internal nodes | Total dynamic energy (fJ) | Total energy (fJ) |
| --- | --- | --- | --- | --- | --- | --- | |
| C432 | 3.55 | 6.93 | 329.00 | 78.29 | 64.66 | 407.29 | 393.66 | 570.02 | 405.48 |
| C499 | 2.38 | 6.72 | 688.76 | 131.76 | 118.53 | 820.52 | 807.29 | 1035.95 | 972.37 |
| C880 | 2.37 | 7.24 | 527.30 | 122.18 | 100.05 | 649.48 | 627.35 | 1035.95 | 972.37 |
| C1355 | 2.78 | 6.51 | 849.01 | 196.28 | 179.08 | 1045.29 | 1028.09 | 1358.78 | 1297.21 |
| C1908 | 3.10 | 6.53 | 880.43 | 187.66 | 161.11 | 1068.09 | 1041.54 | 1415.47 | 1358.78 |
| C2670 | 3.14 | 4.76 | 1204.89 | 246.65 | 200.98 | 1451.54 | 1405.87 | 1943.21 | 1892.11 |
| C3540 | 4.88 | 4.59 | 1404.61 | 463.74 | 379.07 | 1868.35 | 1783.68 | 2827.11 | 2696.56 |
| C5315 | 3.90 | 6.85 | 2458.98 | 574.18 | 452.01 | 3033.16 | 2910.99 | 4518.15 | 4299.84 |
| C6288 | 12.65 | 6.84 | 3973.30 | 873.31 | 759.96 | 4846.61 | 4733.26 | 11011.52 | 10045.37 |
| C7552 | 3.20 | 7.02 | 4079.53 | 729.18 | 610.84 | 4808.71 | 4690.37 | 6371.42 | 5839.56 |
Reductions in energy in energy-constrained dual-$V_t$ CMOS circuits are compared with its delay-constrained counterparts, which are shown in Table 9.7. From Table 9.7, we observe that the percentage reduction in energy in the active mode of operation in delay-constrained and energy-constrained dual-$V_t$ CMOS circuits is on the average 24.92 and 28.11 %, respectively. In standby mode of the operation, percentage reductions of energy in energy-constrained and delay-constrained dual-$V_t$ CMOS circuits on the average are 92.96 and 82.82 %, respectively.

Moreover, it has been observed that there is small reduction in dynamic power in dual-$V_t$ CMOS circuits. However, it has been reported that as the device size becomes smaller and smaller, leakage power will become more and more predominant component. This will make the proposed logic synthesis approach more relevant for future-generation VLSI circuits.

Dynamic $V_{th}$ Scaling

DVFS has been found to be very effective in reducing dynamic power dissipation. For sub-100-nm generations, where leakage power is a significant portion of the total power at run time, a dynamic $V_{th}$ scheme (DVTS) can be used to reduce runtime leakage power just like the DVFS scheme is used to reduce dynamic power. When the workload is less than the maximum value, the processor can be operated
at a lower clock frequency and instead of reducing the supply voltage, the DVTS hardware is used to raise the threshold voltage using reverse body biasing. This increases the delay to match the lower clock frequency and in turn will reduce the run-time leakage power. The processor is controlled to deliver just enough throughput that is required for sustaining the current workload by dynamically adjusting the threshold voltage in an optimal manner such that the leakage power reduction is maximized.

A simpler scheme, called $V_{th}$ hopping, dynamically switches between two threshold voltages: low $V_t$ and high $V_t$, as per the workload requirement. Figure 9.38 shows a detailed implementation of the $V_{th}$-hopping scheme. As shown in Fig. 9.39, the frequency controller generates either $f_{CLK}$ or $f_{CLK}/2$ clock frequencies based on the workload requirement and the $V_{TH}$ controller adjusts the threshold voltage by selecting appropriate body bias voltages for both the nMOS and pMOS transistors.