

Ahmet Bindal

# Electronics for Embedded Systems

---

# Electronics for Embedded Systems

---

Ahmet Bindal

# Electronics for Embedded Systems



Springer

Dr. Ahmet Bindal  
Computer Engineering Department  
San Jose State University  
San Jose, CA  
USA

ISBN 978-3-319-39437-4      ISBN 978-3-319-39439-8 (eBook)  
DOI 10.1007/978-3-319-39439-8

Library of Congress Control Number: 2016960287

© Springer International Publishing Switzerland 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

This Springer imprint is published by Springer Nature  
The registered company is Springer International Publishing AG  
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*For my loving wife, Yen Fen...*

---

## Preface

This book is written for young professionals, undergraduate and graduate students who have a background in basic circuit theory and want to learn about the circuits used in printed circuit boards and embedded systems. My teaching method in this textbook caters towards a lot of schematics, block diagrams and examples at the expense of text because I believe in engineers are “shape-oriented” people and learn from figures, charts and diagrams.

The book has nine chapters. Chapter 1 analyzes the first-order passive RC and RL circuits and the second-order passive RLC circuits encountered in all Printed Circuit Boards (PCB). The general understanding of passive circuits also establishes a vital background for more complex circuits that contain active elements such as diodes and transistors.

Chapter 2 reviews rectifier diodes, light emitting diodes, Zener diodes and explains simple circuits incorporating them. This chapter is also a review chapter for bipolar transistors, specifically NPN transistors, and discusses the circuit behavior and the conditions that lead to the NPN bipolar transistor in cut-off, active and saturation regions.

Chapter 3 explains the N-channel and the P-channel MOS transistors, their current-voltage characteristics and CMOS gates, and then discusses the proper circuit design techniques that lead to transistor sizing in CMOS to meet design requirements prior to simulation.

Chapter 4 starts with Transistor-Transistor-Logic (TTL), explains the circuit operation of a TTL inverter, its logic levels and fan-out limit. However, the emphasis of this chapter is more about the CMOS-TTL interface and the various circuits used at the interface for successful logic translation.

Sensors and preliminary sensor physics are given in Chapter 5. The most common sensors such as thermocouple, photo-diode and photo-detector, Hall-effect device and piezoelectric sensors are discussed in this chapter.

Chapter 6 examines the operational amplifiers and low-frequency operation amplifier circuits used for sensor output amplification. Voltage and trans-resistance amplifiers, analog comparators, Schmitt triggers, square waveform generators are included in this chapter.

Chapter 7 reviews the theory behind data converters such as analog-to-digital converters (ADC) and digital-to-analog converters (DAC). Many different types of ADC designs, such as flash, ramp and successive approximation are explained in this chapter with numerical examples. The weighted adder-type and ladder-type DAC designs are also shown in this chapter.

Chapter 8 combines most of the sensors studied in Chapter 6 and the operational amplifier circuits in Chapter 7 to explain the electromechanical control circuits. The usage of relay switch, opto-isolator, Hall-effect device, pulse-width-modulation (PWM) circuits to operate electromechanical devices, such as DC motors, are also discussed. In the last part of this chapter, several full-scale electronics projects are presented. In each project, the characteristics of sensors used in the project are discussed, and the signal conditioning stage is designed to prepare the unit for analog-to-digital conversion. In the last stage of the design, an ADC is selected to interface with the microcontroller, integrating the design with the rest of the system.

In order to provide a reference material for all the past eight chapters, a brief review of combinational and sequential logic design principles is presented in Chapter 9. A complete design does not only contain a sensor, a signal conditioning circuit and a data converter. Combinational and sequential logic blocks, in the form of “glue logic”, support various parts of the analog data-path, and therefore they constitute the major part of a design. Some of the analog front-end projects presented at the end of Chapter 8 prove that both analog and digital components must be integrated in a design in order to achieve complete functionality. Therefore, a solid understanding in digital logic design becomes a requirement to be able to build the entire analog front-end electronics for a sensor array prior to interfacing with a microcontroller.

Dr. Ahmet Bindal  
Computer Engineering Department  
San Jose State University  
San Jose, CA, USA

---

# Contents

<b>1 Fundamentals of Passive Circuit Analysis . . . . .</b>	<b>1</b>
1.1 Laplace Transform . . . . .	1
1.2 Definitions of Passive Elements . . . . .	1
1.3 Time-Domain Analysis of First Order Passive Circuits . . . . .	3
1.3.1 RC Circuits . . . . .	3
1.3.2 RL Circuits . . . . .	6
1.4 First Order Passive Circuit Analysis Using Natural Frequencies . . . . .	12
1.5 Time-Domain Analysis of Second Order Passive Circuits . . . . .	14
1.6 Transfer Function and Circuit Stability . . . . .	19
<b>2 Diode and Bipolar Transistor Circuits . . . . .</b>	<b>33</b>
2.1 A Brief Review of Semiconductors . . . . .	33
2.2 PN Junction . . . . .	36
2.3 Rectifying Diode Circuits . . . . .	38
2.4 Zener Diode Circuits . . . . .	42
2.5 Light Emitting Diode (LED) . . . . .	42
2.6 Bipolar Transistors . . . . .	43
2.7 Bipolar Transistor Circuits . . . . .	45
<b>3 MOS Transistors and CMOS Circuits . . . . .</b>	<b>57</b>
3.1 N-Channel MOSFET (NMOSFET) . . . . .	57
3.2 P-Channel MOSFET (PMOSFET) . . . . .	59
3.3 Complementary MOS (CMOS) Inverter . . . . .	62
3.4 Two-Input CMOS NAND Logic Gate . . . . .	63
3.5 Two-Input CMOS NOR Logic Gate . . . . .	65
3.6 Complex CMOS Logic Gate Implementation . . . . .	66
3.7 Rise and Fall Times . . . . .	72
3.8 Rise and Fall Delays . . . . .	73
<b>4 TTL Logic and CMOS-TTL Interface . . . . .</b>	<b>89</b>
4.1 TTL Inverter . . . . .	89
4.2 TTL Inverter with Open Collector . . . . .	93
4.3 Two-Input TTL Nand Logic Gate . . . . .	97
4.4 Two-Input TTL NOR Logic Gate . . . . .	99
4.5 Input Current and Voltage Measurements of a Logic Gate . . . . .	102

4.6	Output Current and Voltage Measurements of a Logic Gate . . . . .	103
4.7	TTL-CMOS Interface with a Pull-up Resistor . . . . .	104
4.8	TTL-CMOS Interface with a Bipolar Transistor . . . . .	108
4.9	CMOS-TTL Interface with a Pull-Down Resistor . . . . .	115
4.10	CMOS-TTL Interface with a Bipolar Transistor . . . . .	118
<b>5</b>	<b>Physics of Sensors . . . . .</b>	<b>123</b>
5.1	Thermocouple . . . . .	123
5.2	Photodiode . . . . .	125
5.3	Solar Cell . . . . .	127
5.4	Photo-resistor . . . . .	129
5.5	Piezoelectric Materials and Accelerometers . . . . .	129
5.6	Hall-Effect Devices . . . . .	132
<b>6</b>	<b>Operational Amplifiers and Circuits . . . . .</b>	<b>135</b>
6.1	Operational Amplifier Properties . . . . .	135
6.2	Voltage Amplifier Circuits for Sensors . . . . .	135
6.3	Trans-resistance Amplifier Circuits for Sensors . . . . .	144
6.4	Analog Voltage Comparator . . . . .	146
6.5	Schmitt Trigger . . . . .	147
6.6	Square Waveform Generator . . . . .	150
<b>7</b>	<b>Data Converters . . . . .</b>	<b>157</b>
7.1	Analog-to-Digital Converter Principles . . . . .	157
7.2	Sample and Hold Principle . . . . .	160
7.3	Flash Type Analog-to-Digital Converter . . . . .	160
7.4	Ramp Type Analog-to-Digital Converter . . . . .	162
7.5	Successive Approximation Type Analog-to-Digital Converter . . . . .	165
7.6	Weighted Sum Type Digital-to-Analog Converter . . . . .	169
7.7	Ladder Type Digital-to-Analog Converter . . . . .	170
<b>8</b>	<b>Front-End Electronics for Embedded Systems . . . . .</b>	<b>175</b>
8.1	Electromechanical Device Control . . . . .	175
8.2	Pulse Width Modulation Circuits . . . . .	178
8.3	DC Motor Control . . . . .	181
8.4	Servo Control . . . . .	181
8.5	Hall-Effect Sensor Control . . . . .	182
8.6	Design Project 1: Designing Front-End Electronics for an Analog Microphone . . . . .	183
8.7	Design Project 2: Designing Front-End Electronics for a Temperature Measurement System . . . . .	185
8.8	Project 3: Designing Front-End Electronics for a Light Level Measurement System . . . . .	189
8.9	Project 4: Designing Photo Detector Circuits . . . . .	191
8.10	Project 5: Designing Front-End Electronics for an Optoelectronic Tachometer . . . . .	194
8.11	Project 6: Designing Front-End Electronics for a Hall-Effect-Based Tachometer . . . . .	196

<b>9 Review of Combinational and Sequential Logic Circuits and Design . . . . .</b>	201
9.1 Logic Gates . . . . .	202
9.2 Boolean Algebra . . . . .	208
9.3 Designing Combinational Circuits Using Truth Tables . . . . .	211
9.4 Combinational Logic Minimization—Karnaugh Maps. . . . .	214
9.5 Basic Logic Blocks . . . . .	220
9.6 Adders . . . . .	229
9.7 Subtractors . . . . .	240
9.8 Shifters. . . . .	241
9.9 Multipliers . . . . .	244
9.10 D Latch . . . . .	247
9.11 Timing Methodology Using D Latches . . . . .	249
9.12 D Flip-Flop. . . . .	250
9.13 Timing Methodology Using D Flip-Flops . . . . .	252
9.14 Timing Violations . . . . .	254
9.15 Register . . . . .	259
9.16 Shift Register . . . . .	261
9.17 Counter . . . . .	262
9.18 Moore-Type State Machine . . . . .	263
9.19 Mealy-Type State Machine . . . . .	268
9.20 Controller Design: Moore-Type State Machine Versus Counter-Decoder Scheme . . . . .	272
9.21 A Simple Memory Block . . . . .	276
9.22 A Design Example. . . . .	279
<b>Index . . . . .</b>	295

---

## About the Author



**Ahmet Bindal** received his M.S. and Ph.D. degrees in Electrical Engineering from the University of California, Los Angeles CA. His doctoral research was on the material characterization for HEMT GaAs transistors. During his graduate studies, he was a research associate and a technical consultant for Hughes Aircraft Co. In 1988, he joined the technical staff of IBM Research and Development Center in Fishkill, NY, where he worked as a device design and characterization engineer. He developed asymmetrical MOS transistors and ultra thin

Silicon-On-Insulator (SOI) technologies for IBM. In 1993, he transferred to IBM at Rochester, MN, as a senior circuit design engineer to work on the floating-point unit for the AS-400 mainframe processor. He continued his circuit design career at Intel Corporation in Santa Clara, CA, where he designed 16-bit packed multipliers and adders for the MMX unit for Pentium II processors. In 1996, he joined Philips Semiconductors in Sunnyvale, CA, where he was involved in the designs of instruction/data caches and various SRAM modules for the Trimedia processor. His involvement with VLSI architecture also started in Philips Semiconductors and led to the design of the Video-Out unit for the same processor. In 1998, he joined Cadence Design Systems as a VLSI architect and directed a team of engineers to design self-timed asynchronous processors. After approximately 20 years of industry work, he joined the Computer Engineering faculty at San Jose State University in 2002. His current research interests range from nano-scale electron devices to robotics. Dr. Bindal has over 30 scientific journal and conference publications and 10 invention disclosures with IBM. He currently holds three U.S. patents with IBM and one with Intel Corporation.

## 1.1 Laplace Transform

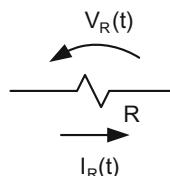
Laplace transform “transforms” a function,  $f(t)$ , in time-domain to a function,  $F(s)$ , in frequency domain. We will use Laplace transform to determine the impedance of a passive element in this section.

It is defined as:

$$\mathcal{L}[f(t)] = F(s) = \int_0^{\infty} f(t) \exp(-st) dt \quad (1.1)$$

## 1.2 Definitions of Passive Elements

Current-voltage relationship across a resistor,  $R$ , is defined as shown in Fig. 1.1.



**Fig. 1.1** Current-voltage relationship across a resistor,  $R$

This relationship is called Ohm's law and mathematically expressed as follows:

$$V_R(t) = R I_R(t) \quad (1.2)$$

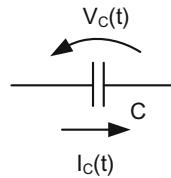
Taking Laplace transform of both sides of Eq. 1.2 yields:

$$\mathcal{L}[V_R(t)] = V_R(s) = R [I_R(t)] = R I_R(s)$$

Thus, the impedance of the resistor becomes:

$$Z_R(s) = \frac{V_R(s)}{I_R(s)} = R \quad (1.3)$$

Current-voltage relationship across a capacitor, C, is defined as shown in Fig. 1.2.



**Fig. 1.2** Current-voltage relationship across a capacitor, C

This relationship originates from the Coulomb's law and mathematically expressed as follows:

$$I_C(t) = C \frac{dV_C(t)}{dt} \quad (1.4)$$

Taking Laplace transform of both sides of Eq. 1.4 yields:

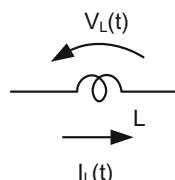
$$\mathcal{L}[I_C(t)] = I_C(s) = C \mathcal{L}\left[\frac{dV_C(t)}{dt}\right] = C[sV_C(s) - V_C(0)] = sCV_C(s)$$

where  $V_C(0)$  is assumed 0 V.

Thus, the impedance of the capacitor becomes:

$$Z_C(s) = \frac{V_C(s)}{I_C(s)} = \frac{1}{sC} \quad (1.5)$$

Current-voltage relationship across an inductor, L, is defined as shown in Fig. 1.3.



**Fig. 1.3** Current-voltage relationship across an inductor, L

This relationship is mathematically expressed as follows:

$$V_L(t) = L \frac{dI_L(t)}{dt} \quad (1.6)$$

Taking Laplace transform of both sides of Eq. 1.6 yields:

$$\mathcal{L}[V_L(t)] = V_L(s) = L \mathcal{L}\left[\frac{dI_L(t)}{dt}\right] = L[sI_L(s) - I_L(0)] = sLI_L(s)$$

where  $I_L(0)$  is assumed 0 A.

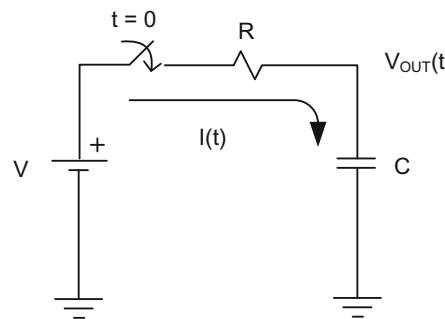
Thus, the impedance of the capacitor becomes:

$$Z_L(s) = \frac{V_L(s)}{I_L(s)} = sL \quad (1.7)$$

### 1.3 Time-Domain Analysis of First Order Passive Circuits

#### 1.3.1 RC Circuits

In this section, we will examine the circuit behavior of simple RC circuits. A circuit composed of a series combination of a resistor and a capacitor is shown in Fig. 1.4. The switch is assumed to close at  $t = 0$ .



**Fig. 1.4** A simple RC circuit where the switch closes at  $t = 0$

Applying Kirchoff's voltage law to the circuit yields:

$$V = RI(t) + V_{OUT}(t) \quad (1.8)$$

But,

$$I(t) = C \frac{dV_{OUT}(t)}{dt} \quad (1.9)$$

In order to use Eq. 1.9 in Eq. 1.8, we need to differentiate both sides of Eq. 1.8.

$$\frac{dV}{dt} = R \frac{dI(t)}{dt} + \frac{dV_{OUT}(t)}{dt} \quad (1.10)$$

Substituting Eq. 1.9 into Eq. 1.10 yields:

$$\frac{dV}{dt} = R \frac{dI(t)}{dt} + \frac{I(t)}{C} = 0 \quad (1.11)$$

$$\frac{dI(t)}{dt} + \frac{I(t)}{RC} = 0 \quad (1.12)$$

In Eq. 1.11, the right side is equal to zero because  $V$  is a constant voltage and its derivative becomes zero.

We know  $I(t)$  is composed of a general and a particular solution for this first-order constant coefficient differential equation. Thus,

$$I(t) = I_{PART}(t) + I_{GEN}(t)$$

But,  $I_{PART}(t) = 0$  since the right hand side of the differential equation in Eq. 1.12 is zero.  $I_{GEN}(t)$ , on the other hand, is expressed as follows:

$$I_{GEN}(t) = A \exp(st) \quad (1.13)$$

Thus,

$$I(t) = I_{PART}(t) + I_{GEN}(t) = A \exp(st) \quad (1.14)$$

To find  $s$ , we need to take the Laplace transform of both sides of the differential equation in Eq. 1.12.

Thus,

$$\mathcal{L}\left[\frac{dI(t)}{dt} + \frac{I(t)}{RC}\right] = sI(s) + \frac{1}{RC}I(s) = 0 \quad (1.15)$$

Since  $I(s) \neq 0$ , then Eq. 1.15 reveals  $s + \frac{1}{RC} = 0$  (characteristic equation).

Therefore,  $s = -\frac{1}{RC}$ , and  $I(t)$  becomes:

$$I(t) = A \exp\left(-\frac{t}{RC}\right) \quad (1.16)$$

To find  $A$ , we need to use the initial voltage across the capacitor. In this case, we assume  $V_{OUT}(0) = 0$ .

But, at  $t = 0$ , we have:

$$V = RI(0) + V_{\text{OUT}}(0) = RI(0) \quad (1.17)$$

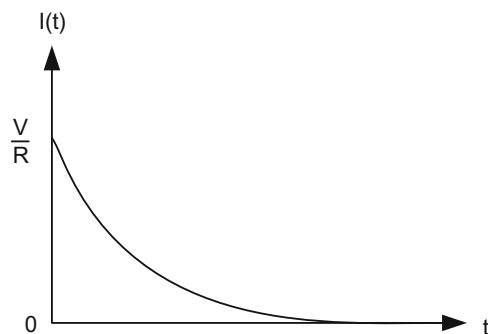
Thus,

$$I(0) = \frac{V}{R}$$

and

$$I(t) = \frac{V}{R} \exp\left(-\frac{t}{RC}\right) \quad (1.18)$$

The current in Eq. 1.18 is plotted as a function of time in Fig. 1.5.



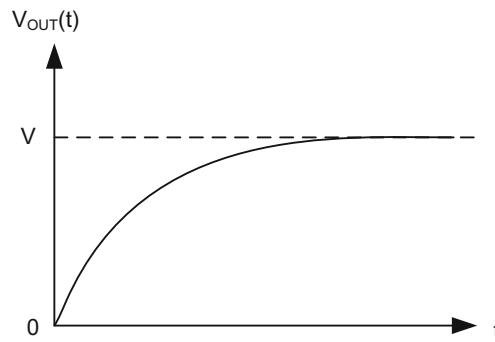
**Fig. 1.5** The current waveform of the RC circuit in Fig. 1.4 as a function of time

Substituting  $I(t)$  in Eq. 1.18 into Eq. 1.8 yields:

$$V_{\text{OUT}}(t) = V - RI(t) = V - R \frac{V}{R} \exp\left(-\frac{t}{RC}\right)$$

$$V_{\text{OUT}}(t) = V - RI(t) = V \left[ 1 - \exp\left(-\frac{t}{RC}\right) \right] \quad (1.19)$$

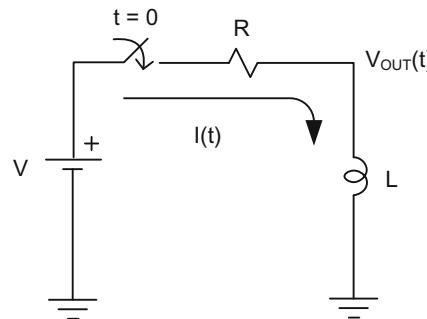
The voltage in Eq. 1.19 is plotted as a function of time in Fig. 1.6.



**Fig. 1.6** The voltage waveform of the RC circuit in Fig. 1.4 as a function of time

### 1.3.2 RL Circuits

In this section, we will examine the circuit behavior of simple RL circuits. A circuit composed of a series combination of a resistor and an inductor is shown in Fig. 1.7. The switch is assumed to close at  $t = 0$ .



**Fig. 1.7** A simple RL circuit where the switch closes at  $t = 0$

Applying Kirchoff's voltage law to the circuit yields:

$$V = RI(t) + V_{OUT}(t) \quad (1.20)$$

But, the current-voltage relationship across the inductor is

$$V_{OUT}(t) = L \frac{dI(t)}{dt} \quad (1.21)$$

Substituting Eq. 1.21 into Eq. 1.20 yields:

$$V = RI(t) + L \frac{dI(t)}{dt} \quad (1.22)$$

Rearranging the terms in Eq. 1.22 produces:

$$\frac{dI(t)}{dt} + \frac{R}{L} I(t) = \frac{V}{L} \quad (1.23)$$

Eq. 1.23 is a constant coefficient linear differential equation. Therefore,  $I(t)$  in this equation is composed of general and particular solutions,  $I_{\text{GEN}}(t)$  and  $I_{\text{PART}}(t)$ , respectively. Thus,

$$I(t) = I_{\text{PART}}(t) + I_{\text{GEN}}(t) \quad (1.24)$$

$I_{\text{PART}}(t) = K$  since the right hand side of the differential equation in Eq. 1.23 is a constant.  $I_{\text{GEN}}(t)$ , on the other hand, is expressed as follows:

$$I_{\text{GEN}}(t) = A \exp(st) \quad (1.25)$$

Combining  $I_{\text{GEN}}(t)$  and  $I_{\text{PART}}(t)$  yields:

$$I(t) = I_{\text{PART}}(t) + I_{\text{GEN}}(t) = K + A \exp(st) \quad (1.26)$$

To find  $K$ , we need to substitute  $I_{\text{PART}}(t) = K$  into Eq. 1.23.

$$\frac{dI_{\text{PART}}(t)}{dt} + \frac{R}{L} I_{\text{PART}}(t) = \frac{V}{L}$$

$$I_{\text{PART}}(t) = K = \frac{V}{R} \quad (1.27)$$

To find  $s$  in Eq. 1.26, we need to equate the right hand side of the differential equation in Eq. 1.23 to zero, and take the Laplace transform of both sides.

Therefore:

$$\frac{dI_{\text{GEN}}(t)}{dt} + \frac{R}{L} I_{\text{GEN}}(t) = 0$$

$$\mathcal{L}\left[\frac{dI_{\text{GEN}}(t)}{dt} + \frac{R}{L} I_{\text{GEN}}(t)\right] = sI_{\text{GEN}}(s) + \frac{R}{L} I_{\text{GEN}}(s) = 0$$

Since  $I_{\text{GEN}}(s) \neq 0$ , then  $s + \frac{R}{L} = 0$

$$\text{Therefore, } s = -\frac{R}{L}$$

$$I_{\text{GEN}}(t) = A \exp\left(-\frac{R}{L}t\right) \quad (1.28)$$

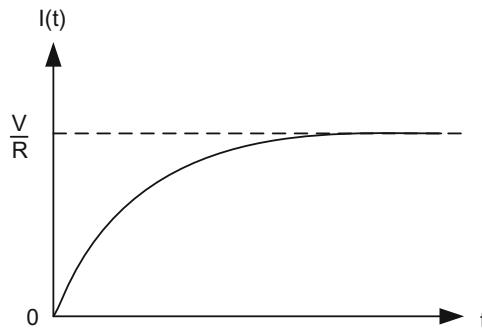
Thus,

$$I(t) = I_{\text{PART}}(t) + I_{\text{GEN}}(t) = \frac{V}{R} + A \exp\left(-\frac{R}{L}t\right) \quad (1.29)$$

To solve for A, we need to use the initial condition,  $I(0) = 0$ . Thus,

$$\begin{aligned} I(0) &= \frac{V}{R} + A = 0 \\ A &= -\frac{V}{R} \\ I(t) &= \frac{V}{R} \left[ 1 - \exp\left(-\frac{R}{L}t\right) \right] \end{aligned} \quad (1.30)$$

This current is plotted as a function of time in Fig. 1.8.



**Fig. 1.8** The current waveform of the RL circuit in Fig. 1.7 as a function of time

For the output voltage, we need to use Eq. 1.20.

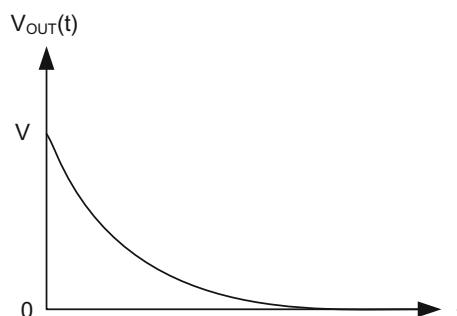
$$V = RI(t) + V_{OUT}(t)$$

$$V_{OUT}(t) = V - RI(t)$$

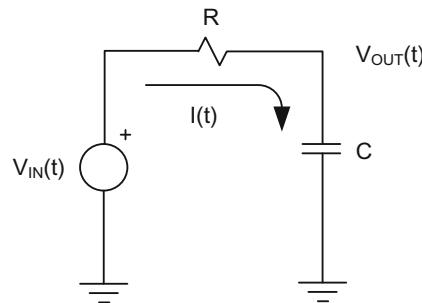
Substituting  $I(t)$  in Eq. 1.30 into  $V_{OUT}(t)$  above yields:

$$V_{OUT}(t) = V \exp\left(-\frac{R}{L}t\right) \quad (1.31)$$

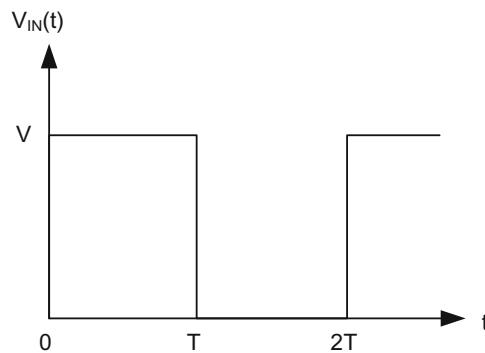
This voltage is plotted as a function of time in Fig. 1.9.



**Fig. 1.9** The voltage waveform of the RL circuit in Fig. 1.7 as a function of time



**Fig. 1.10** A simple RC circuit with a pulse input



**Fig. 1.11** Pulse input waveform

**Example 1.1** Assume the RC circuit in Fig. 1.10 receives a pulse input as shown in Fig. 1.11. Let us derive the response of the circuit at the output node,  $V_{OUT}(t)$ .

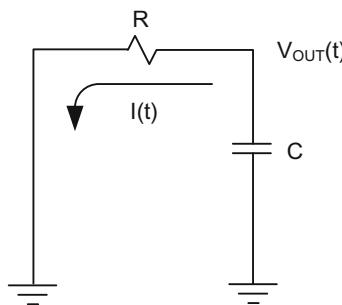
This circuit can be examined in two distinct time intervals:  $0 < t < T$  and  $T < t < 2T$ .

The circuit analysis for  $0 < t < T$  is the same as the analysis for the RC circuit in Fig. 1.4. Thus,

$$V_{OUT}(t) = V \left[ 1 - \exp\left(-\frac{t}{RC}\right) \right]$$

If the time duration,  $T$ , is long enough such that  $T \gg RC$ , the output voltage,  $V_{OUT}(t)$ , at  $t = T$  becomes:

$$V_{OUT}(T) = V \left[ 1 - \exp\left(-\frac{T}{RC}\right) \right] \approx V \quad (1.32)$$



**Fig. 1.12** The equivalent RC circuit in Fig. 1.10 during  $T < t < 2T$

Therefore,  $V_{OUT}(T) = V$  becomes the initial condition for the circuit for  $T < t < 2T$ . However, at  $t = T$ , the input transitions to 0 V, and the circuit in Fig. 1.10 transforms into the RC circuit shown in Fig. 1.12. Here, the current flows into the opposite direction of the current in Fig. 1.10.

Writing the Kirchoff's voltage law for this circuit yields:

$$V_R(t) + V_C(t) = 0 \quad (1.33)$$

Employing the Ohms's law in Eq. 1.33 yields:

$$R I(t) + V_C(t) = 0$$

Differentiating both sides of this equation, and substituting the current-voltage relationship for the capacitor produces:

$$\begin{aligned} R \frac{dI(t)}{dt} + \frac{dV_C(t)}{dt} &= 0 \\ R \frac{dI(t)}{dt} + \frac{I(t)}{C} &= 0 \\ \frac{dI(t)}{dt} + \frac{I(t)}{RC} &= 0 \end{aligned} \quad (1.34)$$

Solving the differential equation in Eq. 1.34 yields:

$$I(t) = A \exp\left(-\frac{(t-T)}{RC}\right) \quad (1.35)$$

Then the output voltage becomes:

$$V_{OUT}(t) = R I(t) = R A \exp\left(-\frac{(t-T)}{RC}\right) \quad (1.36)$$

Fetching the initial condition from Eq. 1.32, and using it in Eq. 1.36 produces:

$$R A = V$$

or

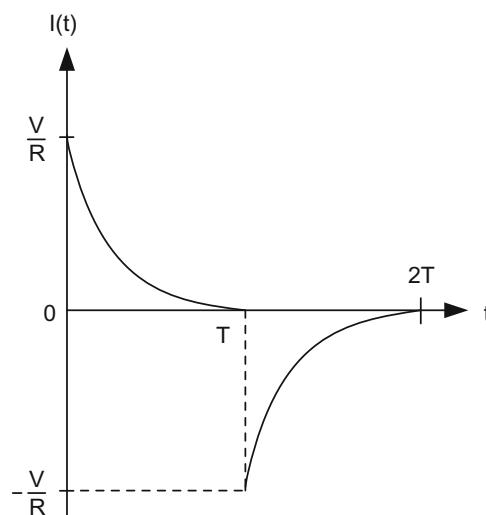
$$A = \frac{V}{R} \quad (1.37)$$

Substituting A in Eq. 1.37 into I(t) in Eq. 1.35 yields:

$$I(t) = \frac{V}{R} \exp\left[-\frac{(t-T)}{RC}\right] \quad (1.38)$$

Plotting  $I(t) = \frac{V}{R} \exp\left(-\frac{t}{RC}\right)$  during  $0 < t < T$ , and  $I(t) = -\frac{V}{R} \exp\left[-\frac{(t-T)}{RC}\right]$  during  $T < t < 2T$  yields the following waveform in Fig. 1.13. In this figure, I(t) becomes a negative quantity during  $T < t < 2T$  because the current in Fig. 1.12 flows in the opposite direction of the current in Fig. 1.10.

$V_{OUT}(t)$  becomes  $V_{OUT}(t) = R I(t)$ .

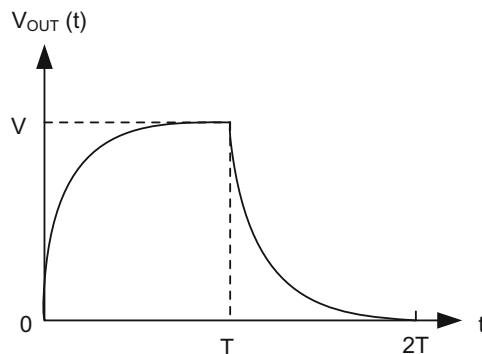


**Fig. 1.13** The current through the RC circuit in Fig. 1.10

Thus,

$$V_{\text{OUT}}(t) = R I(t) = V \exp\left[-\frac{(t-T)}{RC}\right] \quad (1.39)$$

Plotting  $V_{\text{OUT}}(t) = V \left[1 - \exp\left(-\frac{t}{RC}\right)\right]$  during  $0 < t < T$ , and  $V_{\text{OUT}}(t) = V \exp\left[-\frac{(t-T)}{RC}\right]$  during  $T < t < 2T$  yields the following waveform in Fig. 1.14.



**Fig. 1.14** The voltage at the output of the RC circuit in Fig. 1.10

#### 1.4 First Order Passive Circuit Analysis Using Natural Frequencies

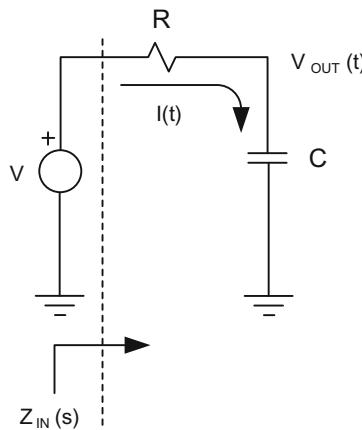
Natural frequencies method is an easy way to find the current through a passive element and the voltage value across it. This method only uses the initial and final conditions on the element and the natural frequencies of the circuit.

There are three steps involved to evaluate the current or the voltage value as a function of time:

- Step 1: Evaluate the initial ( $t = 0$ ) current or voltage values at a passive element
- Step 2: Evaluate the final ( $t \rightarrow \infty$ ) current or voltage values at a passive element
- Step 3: Evaluate the natural frequencies of the circuit when there is no current or voltage source applied to the circuit.

Once these three steps are completed, we can determine the current and voltage values for a passive element and plot the corresponding waveforms.

To show how this method works, assume the simple RC circuit in Fig. 1.15 with a fixed voltage source (the input voltage source can also be time-dependent, however this does not alter the analysis) at its input.



**Fig. 1.15** A simple RC circuit with a fixed voltage source,  $V$ , at its input

Assume that  $V_{\text{OUT}}(0) = 0 \text{ V}$ . The initial output voltage may have a different value; however, this does not alter the circuit analysis. The initial current,  $I(0)$ , becomes:

$$I(0) = \frac{V}{R} \quad (1.40)$$

This completes step 1.

For step 2, we need to identify the value of the current through the circuit, and the voltage across the capacitor at  $t \rightarrow \infty$ .

We know that as a constant current flows through a capacitor, and charges it completely, the capacitor becomes an open circuit. Thus, the current through this circuit becomes 0 A, and voltage across the capacitor becomes equal to  $V$  at  $t \rightarrow \infty$ .

For step 3, the natural frequencies of the circuit must be calculated. When the input voltage source is separated from the circuit in Fig. 1.15, the equivalent input impedance of the circuit,  $Z_{\text{IN}}(s)$ , becomes:

$$Z_{\text{IN}}(s) = \frac{V_{\text{IN}}(s)}{I_{\text{IN}}(s)} = R + \frac{1}{sC} = \frac{1 + sRC}{sC} \quad (1.41)$$

The natural frequencies method dictates to “terminate” all independent current and voltage sources in a circuit. Therefore, when the input voltage source,  $V_{\text{IN}}$ , is terminated or forced to become 0 V, the numerator of  $Z_{\text{IN}}(s)$  automatically becomes zero, revealing  $(sRC + 1) = 0$ .

Thus,

$$s = -\frac{1}{RC} \quad (1.42)$$

This is the natural frequency of the circuit, which is inversely proportional to  $t$ .

Therefore, the initial current starts at  $V/R$  but reduces to 0 A with a time constant  $\tau = RC$  as  $t$  approaches  $\infty$ . Similarly, the output voltage starts at 0 V, and increases towards the

value,  $V$ , with the same time constant,  $\tau = RC$ . Here, the time constant is inversely proportional to the natural frequency as mentioned previously.

Then, the equations for  $I(t)$  and  $V_{OUT}(t)$  are calculated as follows:

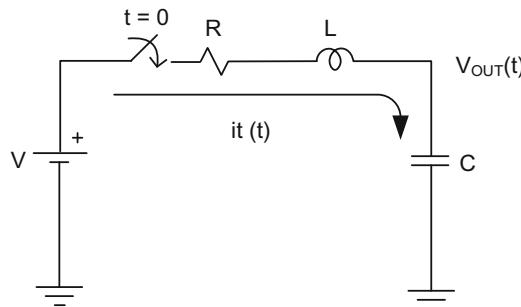
$$I(t) = \frac{V}{R} \exp(st) = \frac{V}{R} \exp\left(-\frac{t}{RC}\right) \quad (1.43)$$

Similarly,

$$V_{OUT}(t) = V[1 - \exp(st)] = V\left[1 - \exp\left(-\frac{t}{RC}\right)\right] \quad (1.44)$$

## 1.5 Time-Domain Analysis of Second Order Passive Circuits

An RLC circuit composed of a series combination of a resistor, inductor and capacitor is considered a second order passive network as shown in Fig. 1.16. In this circuit, assume the switch is closed at  $t = 0$ .



**Fig. 1.16** A simple second order RLC circuit

The Kirchoff's voltage law for the circuit yields:

$$V = RI(t) + V_L(t) + V_{OUT}(t) \quad (1.45)$$

But,

$$V_L(t) = L \frac{dI(t)}{dt} \quad (1.46)$$

$$I(t) = C \frac{dV_{OUT}(t)}{dt} \quad (1.47)$$

To be able to substitute Eqs. 1.46 and 1.47 into Eq. 1.45, we need to differentiate both sides of Eq. 1.45. Thus,

$$0 = R \frac{dI(t)}{dt} + \frac{dV_L(t)}{dt} + \frac{dV_{OUT}(t)}{dt} = R \frac{dI(t)}{dt} + L \frac{d^2I(t)}{dt^2} + \frac{I(t)}{C}$$

Reorganizing the terms yields:

$$\frac{d^2I(t)}{dt^2} + \frac{R}{L} \frac{dI(t)}{dt} + \frac{I(t)}{LC} = 0 \quad (1.48)$$

But,

$$I(t) = I_{GEN}(t) + I_{PART}(t) \quad (1.49)$$

Here,  $I_{PART}(t) = 0$  and  $I_{GEN}(t) = A \exp(s_1 t) + B \exp(s_2 t)$  since the differential equation in Eq. 1.48 is a second order constant coefficient differential equation.

Thus,

$$I(t) = A \exp(s_1 t) + B \exp(s_2 t) \quad (1.50)$$

$s_1$  and  $s_2$  are the natural frequencies of this circuit and computed by solving the characteristic equation. Taking Laplace transform of Eq. 1.48 yields:

$$\mathcal{L} \left[ \frac{d^2I(t)}{dt^2} + \frac{R}{L} \frac{dI(t)}{dt} + \frac{I(t)}{LC} \right] = s^2 + \frac{R}{L}s + \frac{1}{LC} = 0 \quad (1.51)$$

Thus,

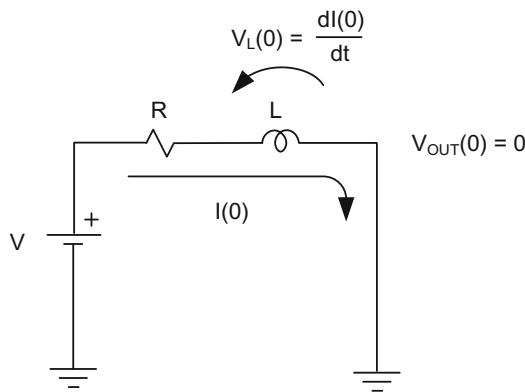
$$s_{1,2} = \frac{-\frac{R}{L} \pm \sqrt{\left(\frac{R}{L}\right)^2 - \frac{4}{LC}}}{2} = -\frac{R}{2L} \pm \sqrt{\left(\frac{R}{2L}\right)^2 - \frac{1}{LC}} \quad (1.52)$$

Let  $\alpha = \frac{R}{2L}$  and  $\omega_0 = \frac{1}{\sqrt{LC}}$  in  $s_{1,2}$ . Then,

$$s_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \omega_0^2} \quad (1.53)$$

We still need two initial conditions to solve A and B. The first initial condition,  $I(0) = 0$ , because the initial current stored in the inductor is assumed 0 A.

The second initial condition,  $\frac{dI(0)}{dt}$ , requires evaluating the RLC circuit at  $t = 0$ , and constructing its equivalent circuit as shown in Fig. 1.17.



**Fig. 1.17** The equivalent RLC circuit in Fig. 1.16 at  $t = 0$

Therefore,

$$V = RI(0) + V_L(0) = RI(0) + L \frac{dI(0)}{dt} = L \frac{dI(0)}{dt}$$

Here,  $I(0) = 0$ . Then,

$$\frac{dI(0)}{dt} = \frac{V}{L} \quad (1.54)$$

Applying the first initial condition,  $I(0) = 0$ , to Eq. 1.50 yields:

$$I(0) = 0 = A + B$$

$$A = -B$$

$$I(t) = A(\exp(s_1 t) - \exp(s_2 t)) \quad (1.55)$$

Applying the second initial condition,  $\frac{dI(0)}{dt} = \frac{V}{L}$ , to Eq. 1.55 yields:

$$\begin{aligned} \frac{dI(0)}{dt} &= A(s_1 - s_2) = \frac{V}{L} \\ A &= \frac{V}{(s_1 - s_2)L} \end{aligned} \quad (1.56)$$

Substituting Eq. 1.56 into Eq. 1.55 produces:

$$I(t) = \frac{V}{(s_1 - s_2)L} (\exp(s_1 t) - \exp(s_2 t)) \quad (1.57)$$

where,  $s_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \omega_0^2}$ .

From Eq. 1.45, we also have:

$$V_{\text{OUT}}(t) = V - R I(t) - V_L(t) = V - R I(t) - L \frac{dI(t)}{dt}$$

But,

$$\frac{dI(t)}{dt} = \frac{V}{(s_1 - s_2)L} (s_1 \exp(s_1 t) - s_2 \exp(s_2 t))$$

Thus,

$$V_{\text{OUT}}(t) = V - \frac{VR}{(s_1 - s_2)L} (\exp(s_1 t) - \exp(s_2 t)) - \frac{V}{(s_1 - s_2)} (s_1 \exp(s_1 t) - s_2 \exp(s_2 t))$$

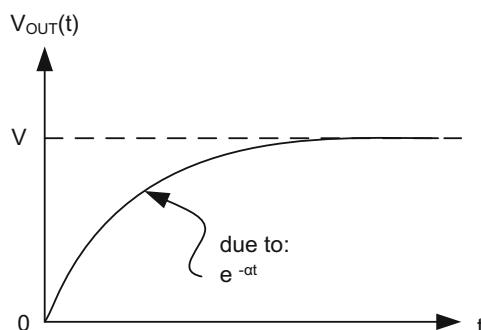
$$V_{\text{OUT}}(t) = V \left[ 1 - \frac{\exp(s_1 t)}{(s_1 - s_2)} \left( \frac{R}{L} + s_1 \right) + \frac{\exp(s_2 t)}{(s_1 - s_2)} \left( \frac{R}{L} + s_2 \right) \right] \quad (1.58)$$

If  $\alpha > \omega_0$ , substituting  $s_1 - s_2 = -2\sqrt{\alpha^2 - \omega_0^2}$  into  $V_{\text{OUT}}(t)$  in Eq. 1.58 yields:

$$V_{\text{OUT}}(t) = V \left[ 1 + \frac{\exp(-\alpha t) \exp(-\sqrt{\alpha^2 - \omega_0^2} t)}{2\sqrt{\alpha^2 - \omega_0^2}} \left( \frac{R}{L} - \alpha - \sqrt{\alpha^2 - \omega_0^2} \right) \right. \\ \left. - \frac{\exp(-\alpha t) \exp(\sqrt{\alpha^2 - \omega_0^2} t)}{2\sqrt{\alpha^2 - \omega_0^2}} \left( \frac{R}{L} - \alpha + \sqrt{\alpha^2 - \omega_0^2} \right) \right]$$

$$V_{\text{OUT}}(t) = V \left\{ 1 + \frac{\exp(-\alpha t)}{\sqrt{\alpha^2 - \omega_0^2}} \left[ \left( \alpha - \frac{R}{L} \right) \sinh \sqrt{\alpha^2 - \omega_0^2} t - \sqrt{\alpha^2 - \omega_0^2} \cosh \sqrt{\alpha^2 - \omega_0^2} t \right] \right\} \quad (1.59)$$

This equation can be plotted in Fig. 1.18.



**Fig. 1.18** The output voltage waveform when  $\alpha > \omega_0$

However, for  $\alpha < \omega_0$

$$s_1 - s_2 = -2j\sqrt{\omega_0^2 - \alpha^2}$$

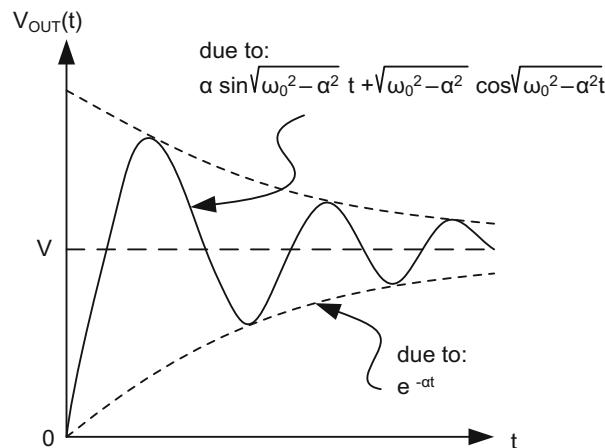
Thus,

$$\begin{aligned} V_{\text{OUT}}(t) = & \left\{ 1 - j \frac{\exp(-\alpha t)}{2\sqrt{\omega_0^2 - \alpha^2}} \left[ \left( \frac{R}{L} - \alpha \right) \left( \exp\left(-j\sqrt{\omega_0^2 - \alpha^2}t\right) - \exp\left(j\sqrt{\omega_0^2 - \alpha^2}t\right) \right) \right. \right. \\ & \left. \left. - j\sqrt{\omega_0^2 - \alpha^2} \left[ \exp\left(-j\sqrt{\omega_0^2 - \alpha^2}t\right) + \exp\left(j\sqrt{\omega_0^2 - \alpha^2}t\right) \right] \right\} \end{aligned}$$

Using  $\exp(j\theta) = \cos \theta + j \sin \theta$  (Euler's equation) in  $V_{\text{OUT}}(t)$  yields:

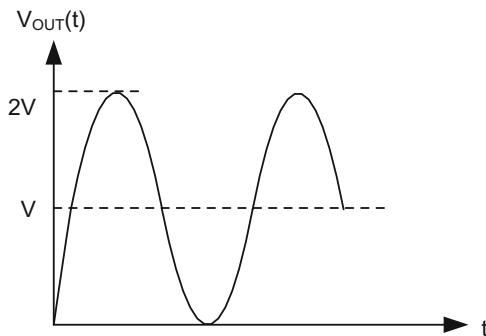
$$V_{\text{OUT}}(t) = V \left[ 1 - \frac{\exp(-\alpha t)}{\sqrt{\omega_0^2 - \alpha^2}} \left( \alpha \sin \sqrt{\omega_0^2 - \alpha^2}t + \sqrt{\omega_0^2 - \alpha^2} \cos \sqrt{\omega_0^2 - \alpha^2}t \right) \right] \quad (1.60)$$

This equation produces quite a different result from Fig. 1.18, and contains the component of oscillation from sine and cosine terms in Eq. 1.60 as shown in Fig. 1.19.



**Fig. 1.19** The output voltage waveform when  $\alpha < \omega_0$

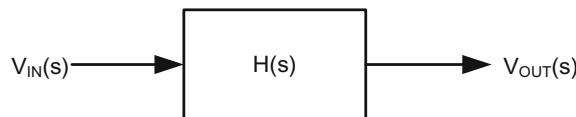
When the resistor in component in this circuit is completely eliminated, then  $V_{\text{OUT}}(t)$  will reduce to  $V_{\text{OUT}}(t) = V (1 - \cos \omega_0 t)$  and oscillate with an angular frequency of  $\omega_0$ . The resultant waveform is plotted in Fig. 1.20.



**Fig. 1.20** The output voltage waveform when  $R = 0$

## 1.6 Transfer Function and Circuit Stability

Transfer function,  $H(s)$ , defines the relationship between the input and output of a passive circuit in frequency domain. The passive network has to be defined in terms of individual impedances of  $R$ ,  $L$  and  $C$  to produce  $H(s)$  as shown in Fig. 1.21.



**Fig. 1.21** Input-output relationship of a passive network in frequency (s) domain

$$V_{\text{OUT}}(s) = H(s) V_{\text{IN}}(s)$$

Thus,

$$H(s) = \frac{V_{\text{OUT}}(s)}{V_{\text{IN}}(s)} \quad (1.61)$$

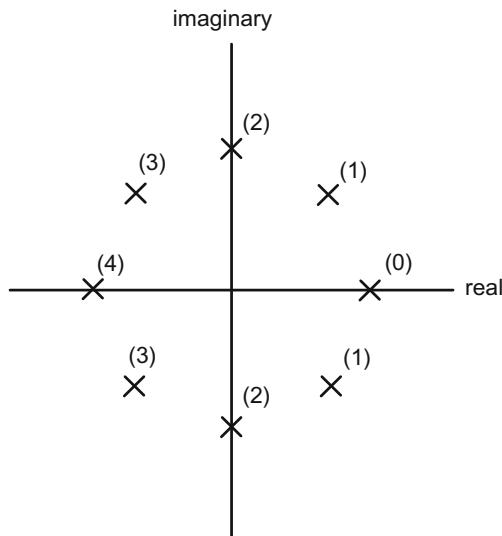
The transfer function,  $H(s)$ , in Eq. 1.61 can be described as:

$$H(s) = \frac{N(s)}{D(s)}$$

In this equation, if  $N(s) = 0$ , the roots of the polynomial gives the zeros of  $H(s)$ . Similarly, if  $D(s) = 0$ , the roots of the polynomial produces the poles of  $H(s)$ , which describes the stability of the circuit. The poles of the circuit are the same as the natural frequencies, and follow the form in Eq. 1.62:

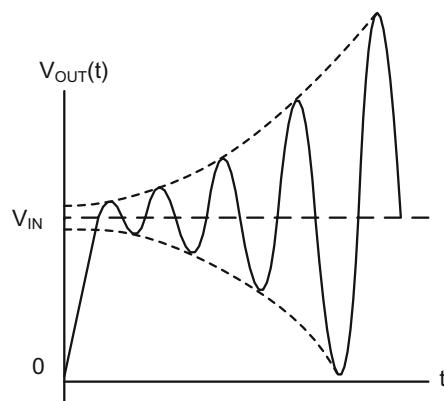
$$s = \alpha + j\omega \quad (1.62)$$

Here,  $\alpha$  is the real component and  $\omega$  is the imaginary component of a pole. The root(s) from  $D(s) = 0$  can be plotted on a complex plane as shown in Fig. 1.22.



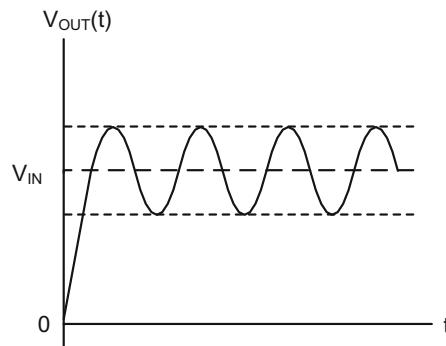
**Fig. 1.22** Possible locations of poles in a complex plane

In this figure, if the pole(s) is at position (0) or in complex conjugate form at position (1), this situation creates instability in the circuit. The resulting waveform grows in amplitude as a function of time similar to Fig. 1.23.



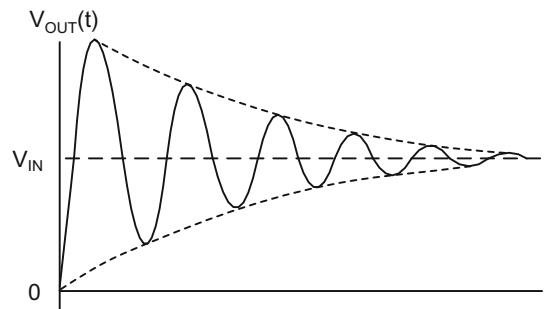
**Fig. 1.23** Output of a passive network with complex poles at position (1) in Fig. 1.22

If the poles lay on the imaginary axis or in position (2) on the complex plane, the output waveform becomes oscillatory as shown in Fig. 1.24.



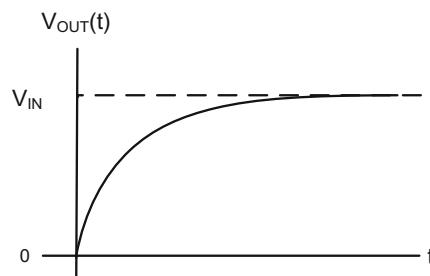
**Fig. 1.24** Output of a passive network with complex poles at position (2) in Fig. 1.22

If the poles shift to the left hand side of the complex plane in Fig. 1.22 such as in position (3), the output of the circuit still becomes oscillatory but reaches to a stable voltage as shown in Fig. 1.25.



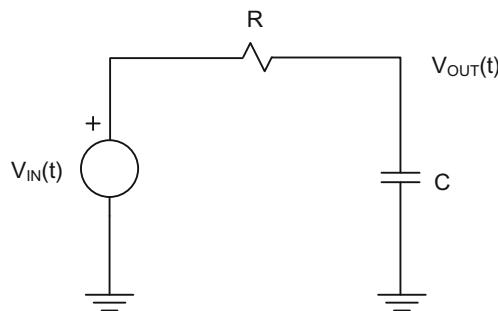
**Fig. 1.25** Output of a passive network with complex poles at position (3) in Fig. 1.22

If the pole(s) lay on the negative real axis of the complex plane such as position (4), then the oscillation at the output disappears, and the waveform reaches to a stable value as shown in Fig. 1.26.



**Fig. 1.26** Output of a passive network with real pole(s) at position (4) in Fig. 1.22

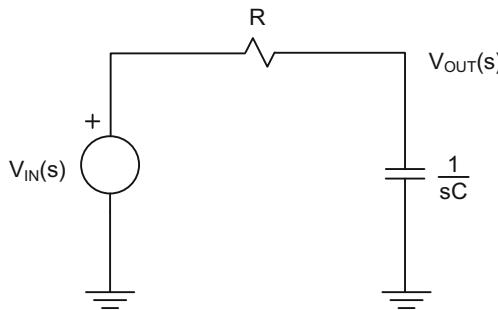
**Example 1.2** Assume the following RC circuit in Fig. 1.27 to evaluate its stability.



**Fig. 1.27** The RC circuit for stability analysis

First, each time-domain component in the circuit is transformed to a frequency-domain component using Laplace transform.

Thus, the circuit transforms into Fig. 1.28.



**Fig. 1.28** The frequency-domain equivalent of the RC circuit in Fig. 1.27

Then the relationship between the input and output becomes:

$$H(s) = \frac{V_{OUT}(s)}{V_{IN}(s)} = \frac{\frac{1}{sC}}{R + \frac{1}{sC}} = \frac{1}{sRC + 1} = \frac{1}{RC\left(s + \frac{1}{RC}\right)} \quad (1.63)$$

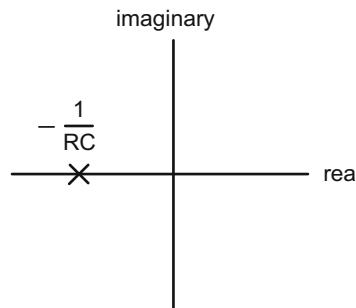
To find the pole(s), the denominator needs to be zero.

$$D(s) = RC \left( s + \frac{1}{RC} \right) = 0$$

Thus, the pole becomes:

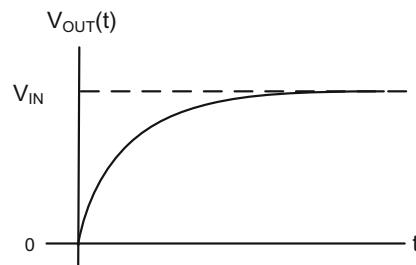
$$s = -\frac{1}{RC} \quad (1.64)$$

Plotting this single pole on complex plane yields the following in Fig. 1.29.



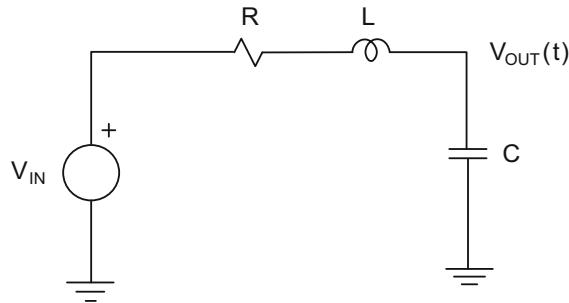
**Fig. 1.29** A single pole representation of the RC circuit in Fig. 1.27

From the position of the pole, we know that the circuit is stable, and  $V_{OUT}(t)$  approaches to a stable value as shown in Fig. 1.30.

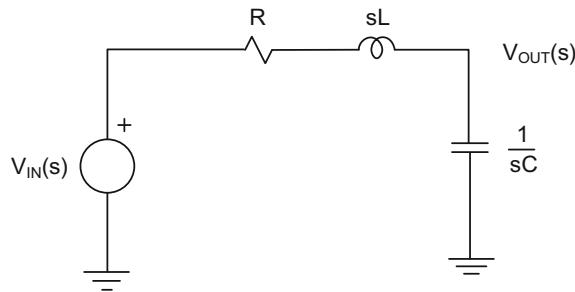


**Fig. 1.30** The output response of the RC circuit in Fig. 1.27

**Example 1.3** Now, let us assume the following RLC circuit in Fig. 1.31 to evaluate its stability. Again, we need to convert the time-domain components in this circuit to frequency-domain components. Taking the Laplace transform of  $V_{IN}(t)$ , and each passive component in the circuit transforms Fig. 1.31 to Fig. 1.32.



**Fig. 1.31** The RLC circuit for stability analysis



**Fig. 1.32** The frequency domain equivalent of the RLC circuit in Fig. 1.31

The transfer function,  $H(s)$ , is evaluated as follows:

$$H(s) = \frac{V_{\text{OUT}}(s)}{V_{\text{IN}}(s)} = \frac{\frac{1}{sC}}{R + sL + \frac{1}{sC}} = \frac{1}{s^2CL + sRC + 1} \quad (1.65)$$

To find the pole(s), the denominator of  $H(s)$  in Eq. 1.65 needs to be zero. Thus,

$$D(s) = s^2CL + sRC + 1 = 0 \quad (1.66)$$

Solving the polynomial in Eq. 1.66 reveals:

$$s_{1,2} = \frac{-RC \pm \sqrt{(RC)^2 - 4LC}}{2LC} = -\frac{R}{2L} \pm \sqrt{\left(\frac{R}{2L}\right)^2 - \left(\frac{1}{\sqrt{LC}}\right)^2} \quad (1.67)$$

The roots,  $s_1$  and  $s_2$ , can be simplified as follows:

$$s_1 = -\alpha_0 + \sqrt{\alpha_0^2 - \omega_0^2}$$

$$s_2 = -\alpha_0 - \sqrt{\alpha_0^2 - \omega_0^2}$$

where,

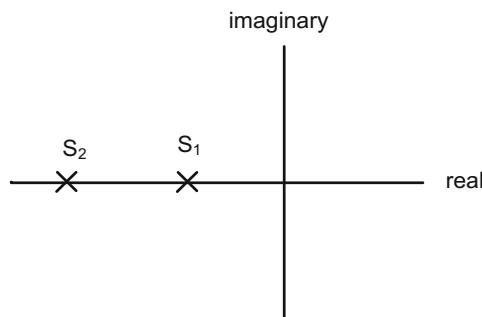
$$\alpha_0 = \frac{R}{2L}$$

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Assuming that L, R and C are all non-zero and non-negative, there are three possible cases depending on the passive component values.

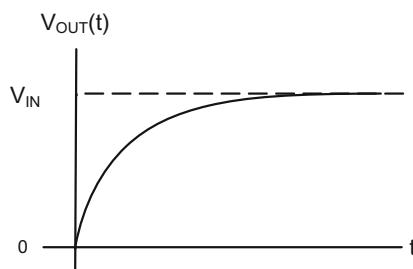
**Case 1:** when  $\alpha_0 > \omega_0$  then both poles become real.

Plotting these poles in Fig. 1.33 yields:



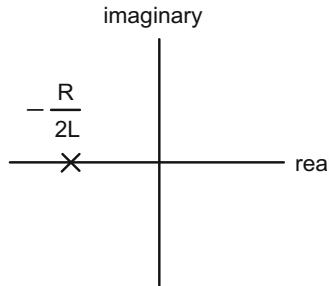
**Fig. 1.33** The real poles of the RLC circuit in Fig. 1.31 when  $\alpha_0 > \omega_0$

The output of the circuit, therefore, approaches to a stable value as  $t \rightarrow \infty$  as shown in Fig. 1.34.



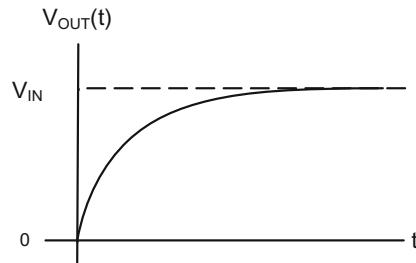
**Fig. 1.34** The output of the RLC circuit in Fig. 1.31 when  $\alpha_0 > \omega_0$

**Case 2:** when  $\alpha_0 = \omega_0$ , then both poles are still real but they collapse on top of each other when plotted on complex plane as shown in Fig. 1.35.



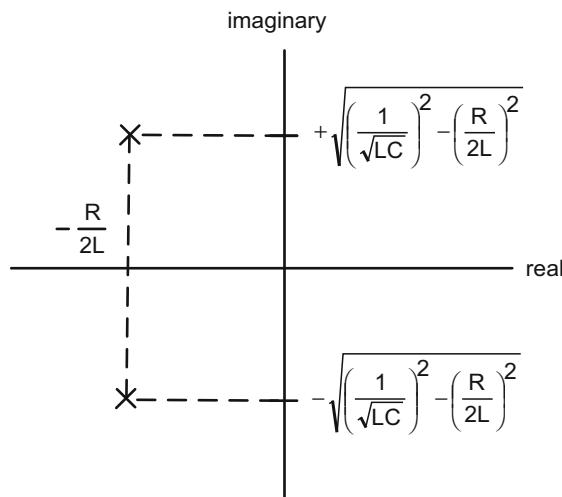
**Fig. 1.35** The real poles of the RLC circuit in Fig. 1.31 when  $\alpha_0 = \omega_0$

The output of the circuit still produces a stable value as  $t \rightarrow \infty$  in Fig. 1.36.



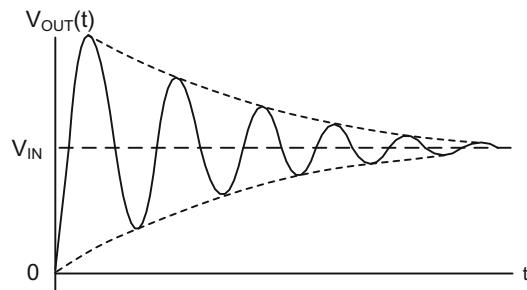
**Fig. 1.36** The output of the RLC circuit in Fig. 1.31 when  $\alpha_0 = \omega_0$

**Case 3:** when  $\alpha_0 < \omega_0$ , then both poles become complex conjugate as shown in Fig. 1.37.



**Fig. 1.37** The complex conjugate poles of the RLC circuit in Fig. 1.31 when  $\alpha_0 < \omega_0$

The output of the circuit produces an oscillatory waveform but reaches to a stable value as  $t$  approaches infinity in Fig. 1.38.



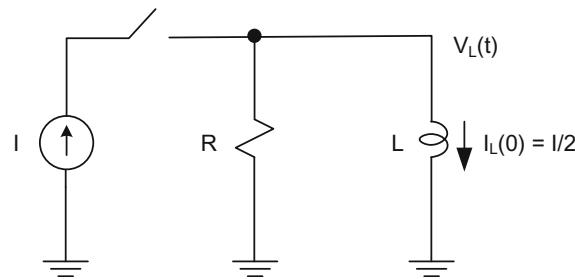
**Fig. 1.38** The output of the RLC circuit in Fig. 1.31 as a result of complex conjugate poles when  $\alpha_0 < \omega_0$

## Review Questions

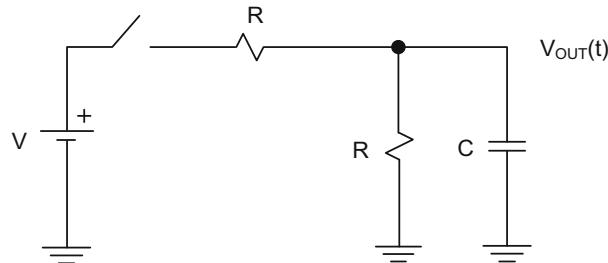
1. The following RL circuit is given:

In this circuit, the current,  $I$ , represents a constant current value. The current in the inductor  $L$  at  $t = 0$  s is  $I/2$  as shown in the schematic. At  $t = 0$ , the switch closes. Using the natural frequencies method

- (a) Find the current through the inductor and plot it.
- (b) Find the voltage across the inductor and plot it.



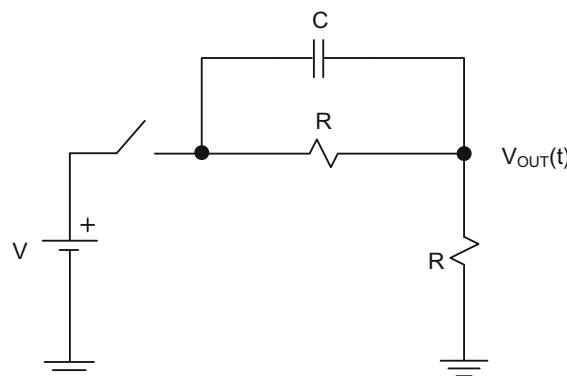
2. The following RC circuit is given.



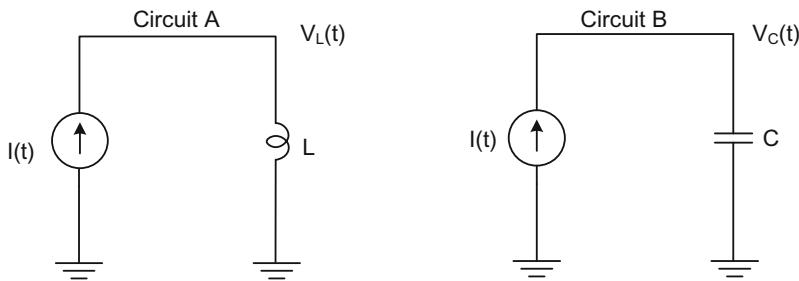
At  $t = 0$ , the switch closes.  $V$  is a constant voltage and  $V_{OUT}(0) = 0$  V.

Using natural frequencies method, find the output voltage,  $V_{OUT}(t)$  and plot it.

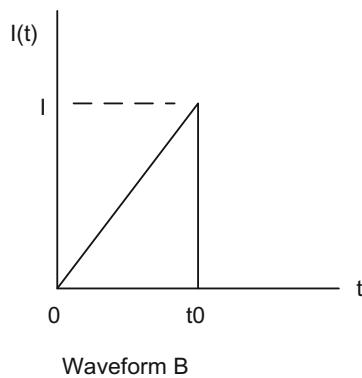
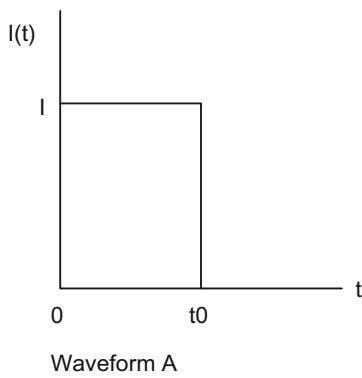
Now, change the location of the capacitor,  $C$ , as shown below. Assume the voltage across the capacitor is 0 V at  $t = 0$ . How does  $V_{OUT}(t)$  change? Plot the waveform for  $V_{OUT}(t)$ .



3. The following circuits are given:

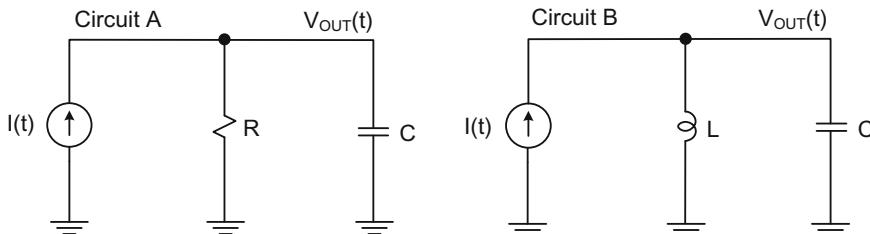


Apply the following waveforms to circuits A and B.



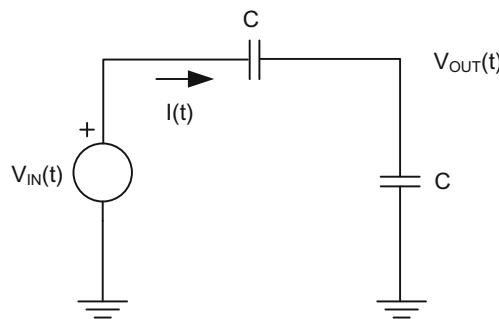
- (a) Find the expression for  $V_L(t)$  in circuit A, and plot it for both input waveforms A and B.
- (b) Find the expression for  $V_C(t)$  in circuit B, and plot it for both input waveforms A and B.

4. The following circuits are given:

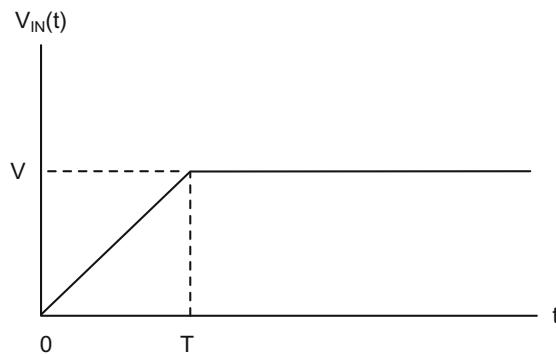


- (a) Find the transfer functions,  $H(s) = \frac{V_{OUT}(s)}{I(s)}$ , of circuits A and B.
- (b) Draw the pole-zero diagrams and find out the stability of each circuit. From stability criteria, how does  $V_{OUT}(t)$  change with time?

5. The following capacitive circuit is given.

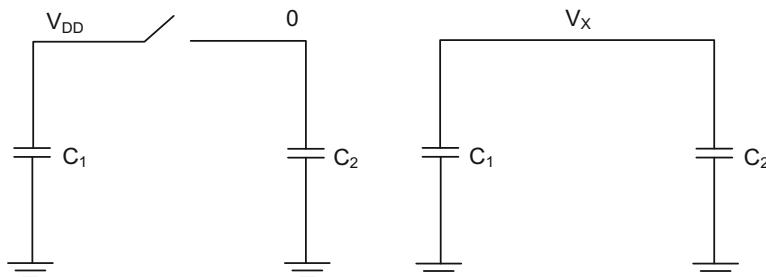


Apply the waveform below to the input of this circuit.

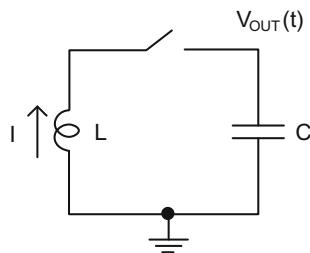


Solve  $I(t)$  and  $V_{OUT}(t)$  using time-domain analysis and plot them as a function of time.

6. The capacitors,  $C_1$  and  $C_2$ , are connected in parallel with a switch as shown below. The capacitor,  $C_1$ , has an initial voltage of  $V_{DD}$ , and capacitor,  $C_2$ , has 0 V. When the switch closes at  $t = 0$ , what will be the resultant voltage,  $V_X$ ?

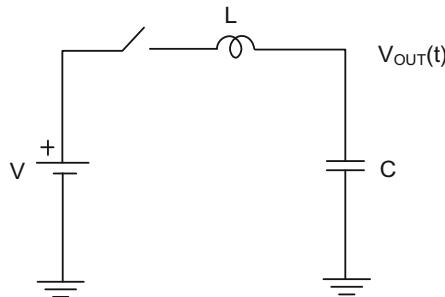


7. An LC circuit is given below:

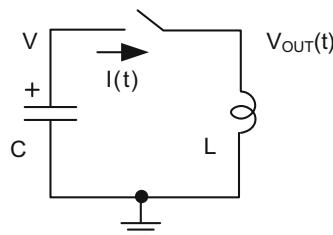


The initial current stored in the inductor is  $I$  in the direction indicated in the circuit diagram, and the voltage across the capacitor is 0 V at  $t = 0$  s before the switch is closed. Find the expression for  $V_{OUT}(t)$  and draw its waveform using time-domain analysis.

8. A constant voltage source,  $V$ , is applied to the circuit below. If the stored current in the inductor is 0 A, and the stored voltage across the capacitor is 0 V before the switch is closed, calculate the output voltage,  $V_{OUT}(t)$ , for  $t > 0$ .

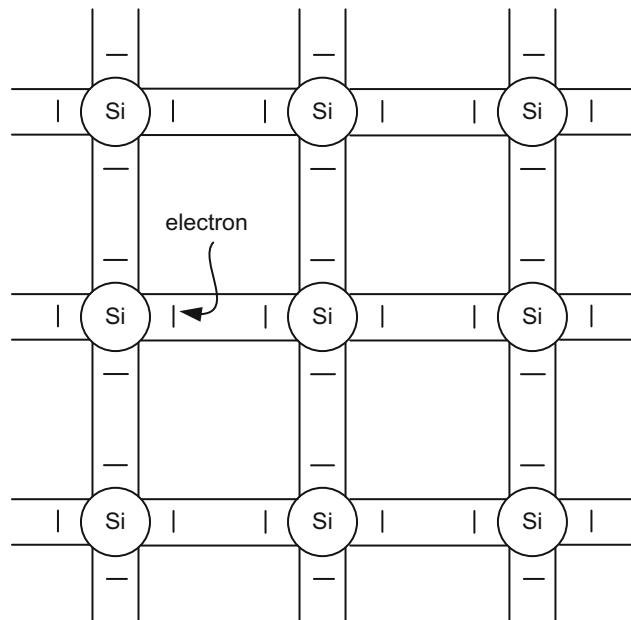


9. Assume the LC circuit below. The initial voltage stored across the capacitor is  $V$ . When the switch closes at  $t = 0$ , the current flows from the capacitor into the inductor and back into the capacitor. Assuming the stored current in the inductor is 0 A, find the current through the circuit,  $I(t)$ , the voltage across the inductor,  $V_{OUT}(t)$ . Plot  $I(t)$  and  $V_{OUT}(t)$  as a function of time.



## 2.1 A Brief Review of Semiconductors

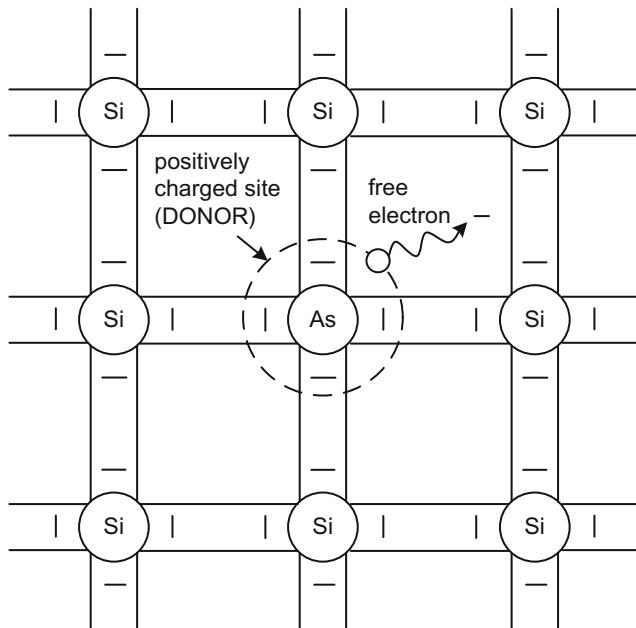
Semiconductors are crystalline structures in which each atom shares its valence electrons with the neighboring atoms. The simple two-dimensional crystalline structure in Fig. 2.1 shows every silicon atom sharing four valence electrons with neighboring four atoms. This is an intrinsic semiconductor which does not contain any impurities.



**Fig. 2.1** Intrinsic semiconductor

However, if a different type of atom, such as arsenic, with five valence band electrons is placed in intrinsic silicon, the crystalline structure changes: As atom occupies a site by sharing four of the five valence electrons with Si atoms, and releases the last valence electron to the crystal as a free electron as shown in Fig. 2.2. As soon as this free electron departs from the As site, the site becomes positively charged because of the extra proton in the As atom. This crystalline structure is called N-type extrinsic semiconductor because the structure contains extra electrons and negatively charged.

If an atom with three electrons, such as boron, is introduced in intrinsic semiconductor instead of As, B atom shares all of its three valence electrons with neighboring Si atoms. The crystalline structure ultimately forms a “hole” in the boron site, signifying absence of electrons as shown in Fig. 2.3. This is called P-type semiconductor where the B site is charged negatively when the hole is occupied by a free electron.



**Fig. 2.2** N-type extrinsic semiconductor

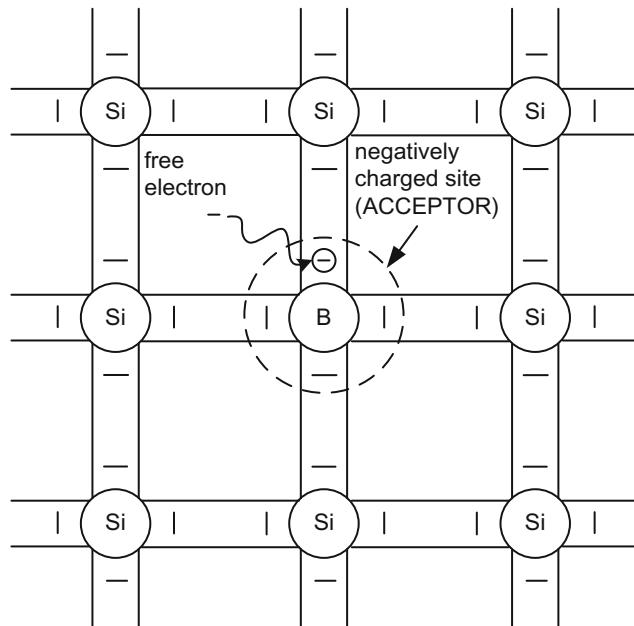
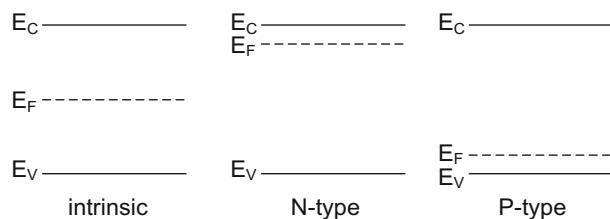
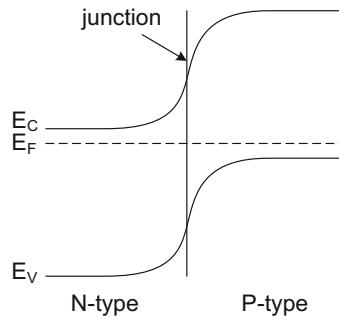
**Fig. 2.3** P-type extrinsic semiconductor

Figure 2.4 shows the simplified energy band diagrams of all three types of semiconductors. An electron at the valance band,  $E_V$ , needs to have enough energy to transition to the conduction band,  $E_C$ , in order to be a free electron in the crystalline structure. This energy difference between  $E_C$  and  $E_V$  is called band gap energy,  $E_G$ . There is also a virtual energy level called Fermi level,  $E_F$ , where there is only 50% probability of finding an electron. In N-type semiconductors, the probability of finding free electrons is higher. Therefore,  $E_F$  is closer to  $E_C$ . However, this probability decreases if the semiconductor is the P-type where there are not many free electrons. As a result  $E_F$  approaches to  $E_V$ .

**Fig. 2.4** Intrinsic, N-type and P-type semiconductor energy band diagrams

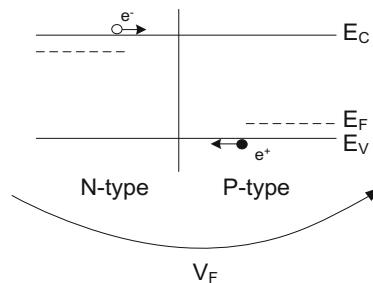
## 2.2 PN Junction

When N-type and P-type semiconductors are joined there will be an initial flow of electrons from the N-type semiconductor to the P-type until the Fermi energy levels on both sides become equal, and the structure becomes stable. Both the conduction and valence bands close to the junction bend as the result of the initial electron flow as shown in Fig. 2.5.



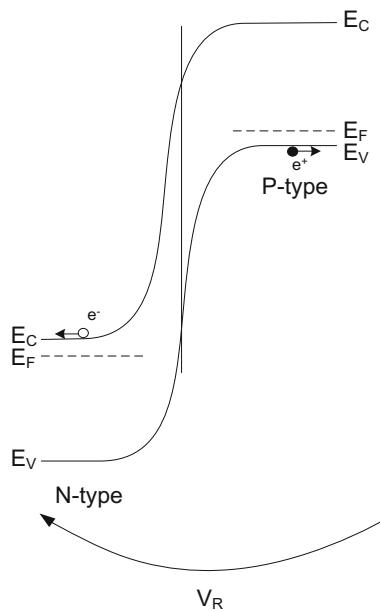
**Fig. 2.5** PN junction

If a forward voltage,  $V_F$ , is applied to the PN junction in the direction shown in Fig. 2.6, the energy supplied to the junction,  $E = -qV_F$ , moves the Fermi level of the N-type semiconductor upwards relative to the P-type until the conduction and valence bands almost become flat across the junction. The applied voltage pushes the electrons in the N-type semiconductor and the holes in the P-type semiconductor towards each other, resulting in a continuous flow of current.



**Fig. 2.6** PN junction under forward bias

When a reverse voltage,  $V_R$ , is applied to the junction, on the other hand, electrons in the N-side and holes in the P-side are extracted from the PN junction, resulting in no conduction current as shown in Fig. 2.7.



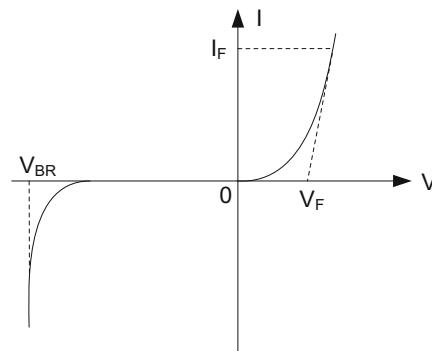
**Fig. 2.7** PN junction under reverse bias

When we plot the forward PN junction current as a function of forward voltage, the current increases slightly before the applied voltage reaches  $V_F$  as shown in Fig. 2.8. However, this current keeps increasing exponentially in the neighborhood of  $V_F$  as it abides by the following equation:

$$I_F = I_0 \left[ \exp\left(\frac{qV_F}{kT}\right) - 1 \right] \quad (2.1)$$

In this equation,  $q$  is called the electronic charge,  $k$  is the Boltzmann constant,  $T$  is temperature, and  $I_0$  is the leakage current.

Applying reverse voltage across the diode produces only a small leakage current until the junction breaks down at  $V_{BR}$  and becomes a short circuit.



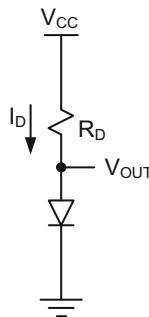
**Fig. 2.8** PN junction diode current-voltage (I-V) characteristics

### 2.3 Rectifying Diode Circuits

The simplest diode circuit incorporates a series resistor with the diode as shown in Fig. 2.9. In this circuit, the output voltage,  $V_{OUT}$ , clamps to  $V_F$  which is approximately equal to 0.7 V for silicon. The current,  $I_D$ , becomes equal to:

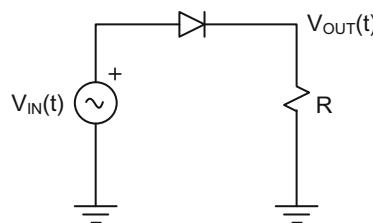
$$I_D = \frac{(V_{CC} - 0.7)}{R_D} = I_F \quad (2.2)$$

We need to use an appropriate  $R_D$  to limit the forward current through the diode according to the manufacturer's specifications.

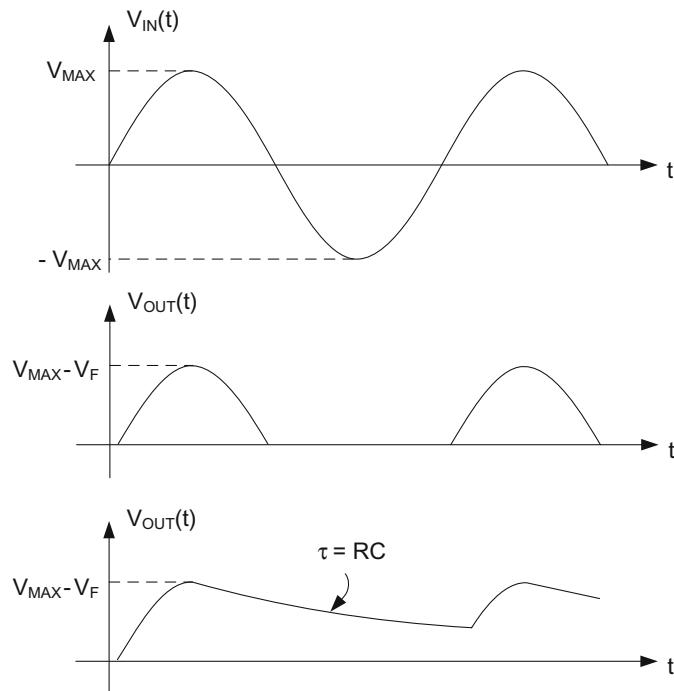


**Fig. 2.9** Biasing circuit for a rectifying diode

A half-wave rectifier shown in Fig. 2.10 is an extension of the simple diode circuit in Fig. 2.9. When a sinusoidal waveform is applied to the input of this circuit as shown in the top part of Fig. 2.11, the diode will only conduct if the input voltage becomes greater than or equal to its forward voltage,  $V_F \approx 0.7$  V. Therefore, when the input reaches its peak value,  $V_{MAX}$ , the output can only produce  $(V_{MAX} - 0.7)$  as shown by the center waveform in Fig. 2.11.

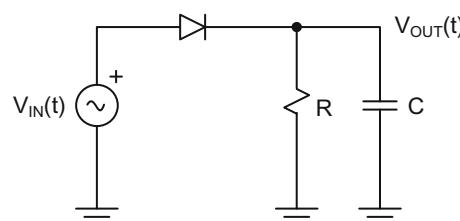


**Fig. 2.10** Half-wave rectifier circuit



**Fig. 2.11** Sinusoidal input (top), half-wave rectifier output (mid), output with capacitor (bottom)

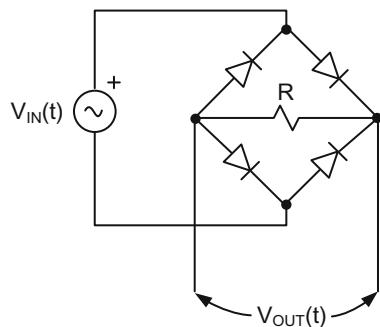
However, the real purpose of this circuit is to produce an “almost” constant voltage by eliminating the half sinusoids as shown by the bottom waveform in Fig. 2.11. This can be achieved by connecting a parallel capacitor to the output resistor as shown in Fig. 2.12. When the input climbs from 0 V towards  $V_{MAX}$ , the output closely follows the input as in the bottom waveform of Fig. 2.11. However, as the input starts decreasing from  $V_{MAX}$  to 0 V, the voltage across the capacitor can no longer follow this change. The previously stored voltage across the capacitor ( $V_{MAX}$  in this case) becomes larger than the input voltage, reverse-biasing the diode, and stopping the current conduction. As a result, the capacitor starts discharging through the resistor with a time constant,  $RC$ , as shown in the bottom part of Fig. 2.11.



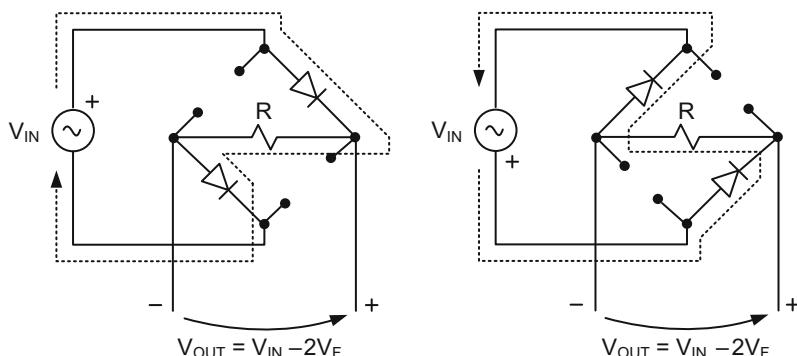
**Fig. 2.12** Half-wave rectifier circuit with output capacitor

The full-wave rectifier composed of four diodes is shown in Fig. 2.13. The circuit output is measured across the resistor, R. When the polarity of the sinusoidal input voltage is in its positive phase, the current conducts through two diodes in series and the output resistor as shown by the equivalent circuit on the left hand side of Fig. 2.14. As a result, the output voltage follows the input, but becomes  $(V_{IN} - 2V_F)$  as shown by the center waveform in Fig. 2.15.

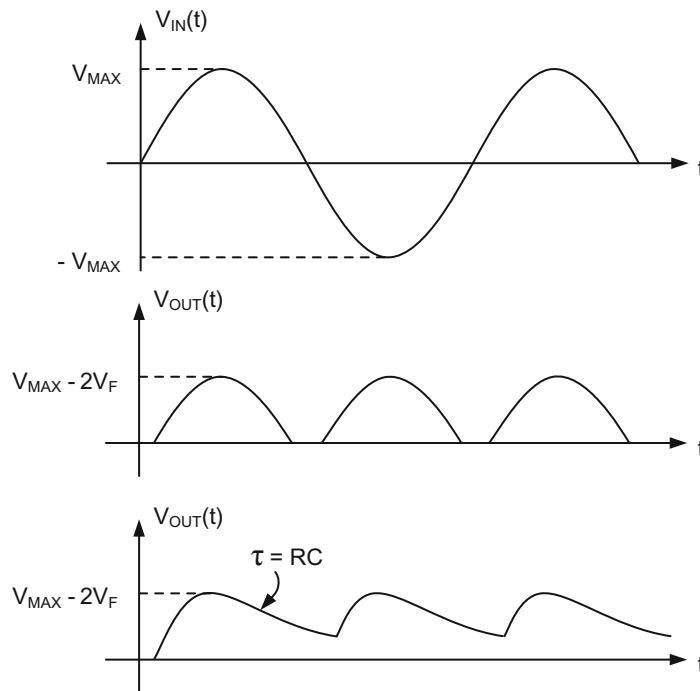
When the sinusoidal input voltage changes its phase to negative, current conduction through the circuit still exists. But, this time the other two diodes start conducting as shown by the equivalent circuit on the right hand side of Fig. 2.14. The voltage drop across the output resistor still becomes equal to  $(V_{IN} - 2V_F)$ .



**Fig. 2.13** Full-wave rectifier circuit

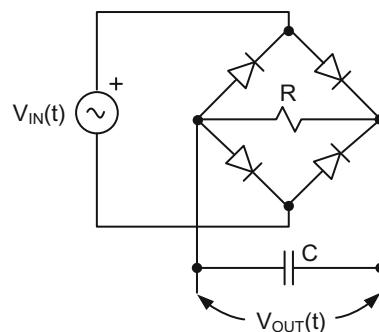


**Fig. 2.14** Full-wave rectifier circuit in positive phase (*left*), in negative phase (*right*)



**Fig. 2.15** Sinusoidal input (top), full-wave rectifier output without capacitor (mid), output with capacitor (bottom)

The Direct Current (DC) rectification is achieved by adding a parallel capacitor at the output of the full-wave rectifier shown in Fig. 2.16. This circuit reveals a more efficient mechanism in generating a DC output compared to a half-wave rectifier. The circuit behavior in full-wave rectifier is similar to half-wave rectifier: the output closely follows the input in the first and third quadrants of the sinusoidal input signal, and the current conduction stops during the second and the fourth quadrants when the capacitor discharges through the output resistor with a time constant,  $RC$ , as shown by the bottom waveform of Fig. 2.15.

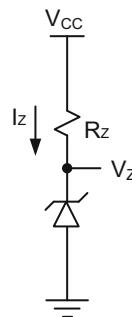


**Fig. 2.16** Full-wave rectifier circuit with output capacitor

## 2.4 Zener Diode Circuits

Zener diode is another two terminal device where a precise constant voltage can be obtained across its terminals. Zener diode manufacturers specify the nominal and maximum Zener voltage,  $V_Z$ , and the reverse current,  $I_Z$ , in their datasheets. Zener diode is reverse biased with a series resistor that adjusts  $I_Z$  as shown in Fig. 2.17. The resistance value becomes:

$$R_Z = \frac{(V_{CC} - V_Z)}{I_Z} \quad (2.3)$$



**Fig. 2.17** Zener diode circuit

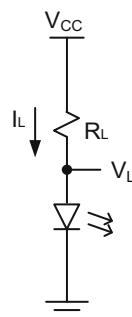
---

## 2.5 Light Emitting Diode (LED)

Light Emitting Diode (LED) is an optoelectronic device that emits light at a specific wavelength when an appropriate bias current,  $I_L$ , is applied as shown in Fig. 2.18.  $R_L$  is adjusted in order to compute  $I_L$ , and comply with the required luminosity.

$$R_L = \frac{(V_{CC} - V_L)}{I_L} \quad (2.4)$$

The voltage drop across LED,  $V_L$ , is also specified in the manufacturer's datasheet.

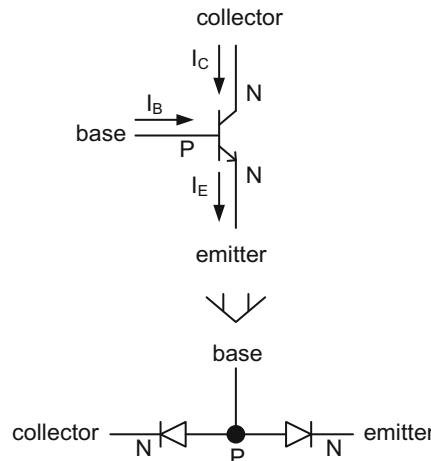


**Fig. 2.18** LED diode circuit

## 2.6 Bipolar Transistors

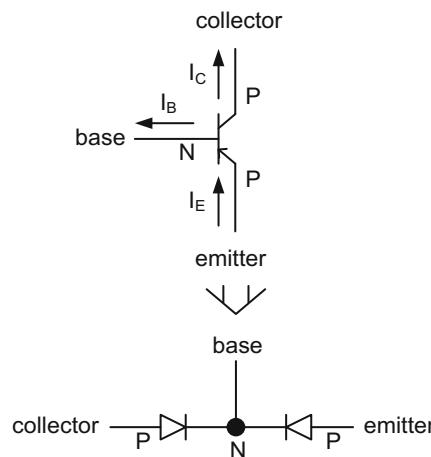
There are two kinds of bipolar junction transistors (BJT) used in electronic circuits: NPN and PNP. The equivalent circuit of an NPN transistor shown in Fig. 2.19 basically consists of two back-to-back diodes. Its emitter current is the sum of base and collector currents as shown in Eq. 2.5

$$I_E = I_B + I_C \quad (2.5)$$



**Fig. 2.19** NPN bipolar transistor and its equivalent circuit

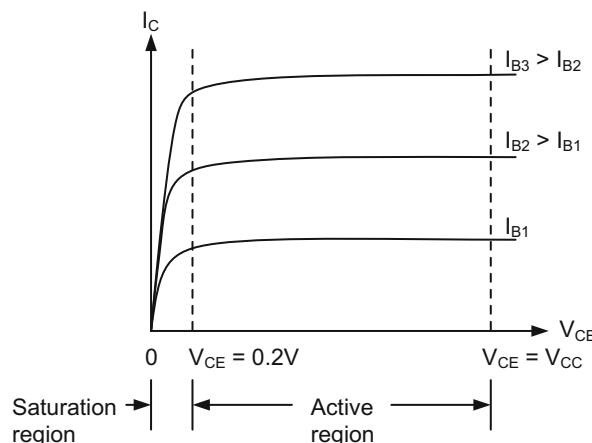
The second type of BJT is the PNP transistor shown in Fig. 2.20. Its emitter current is still the sum of collector and base currents. However, the currents in this transistor flow in completely opposite directions compared to the currents in an NPN transistor.



**Fig. 2.20** PNP bipolar transistor and its equivalent circuit

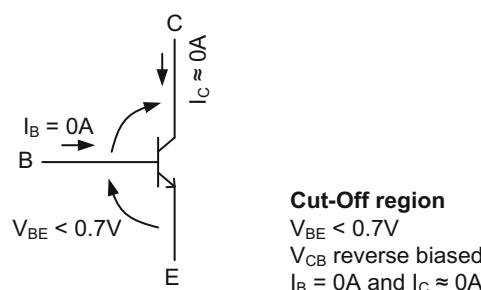
The current-voltage relationship of a typical NPN transistor is shown in Fig. 2.21. In this figure, collector current,  $I_C$ , is plotted as a function of collector-emitter voltage,  $V_{CE}$ , for a set of base currents,  $I_B$ . For each  $I_B$ ,  $I_C$  initially increases exponentially with  $V_{CE}$  until it reaches a point after which it essentially remains constant.

The current-voltage characteristics in Fig. 2.21 show three different regions of transistor operation. The first region is the cut-off region where the transistor does not conduct at all. The second region is the active region where  $I_C$  remains almost constant between  $V_{CE} \approx 0.2$  V and  $V_{CC}$ . The third region is the saturation region where  $I_C$  exponentially increases with  $V_{CE}$  between 0 and 0.2 V.



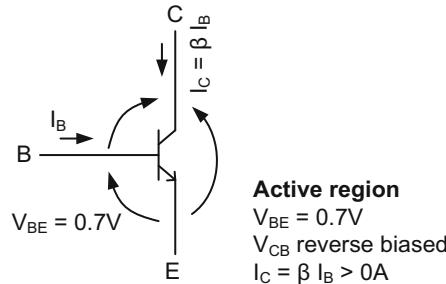
**Fig. 2.21** Current-voltage characteristics of an NPN transistor for different base current values

In cut-off mode of operation, the transistor stops conducting because the voltage across its base-emitter junction drops below 0.7 V, and its collector-base junction becomes reverse biased as shown in Fig. 2.22. As a result, only leakage currents exist in the transistor;  $I_B$  and  $I_C$  essentially become 0 A.



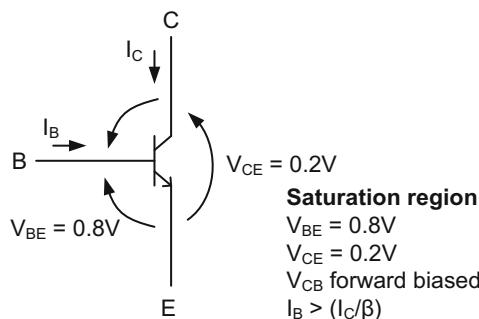
**Fig. 2.22** Transistor bias currents and voltages in cut-off mode of operation

When the transistor operates in active mode, its base-emitter junction voltage,  $V_{BE}$ , becomes approximately equal to 0.7 V although its collector-base junction still remains reverse biased. The base and collector currents become proportional to each other with a current gain of  $\beta$  (or  $h_{FE}$  according to some transistor manufacturers) as shown in Fig. 2.23.



**Fig. 2.23** Transistor bias currents and voltages in active mode of operation

When the transistor finally reaches saturation region, its collector-base and emitter-base junctions become forward biased as shown in Fig. 2.24. The transistor remains in saturation as long as  $V_{BE}$  is greater than 0.7 V. A typical  $V_{BE}$  and  $V_{CE}$  are approximately 0.8 and 0.6 V, respectively. This produces 0.2 V between collector and emitter terminals as shown in the figure. In this mode,  $I_B$  and  $I_C$  can no longer hold the linear relationship ( $I_C = \beta I_B$ ). A new relationship forms between the base saturation current,  $I_{BSAT}$ , and the base active-region current,  $I_{BACT}$ :  $I_{BSAT} \gg I_{BACT}$ . This, in turn, produces  $I_{BSAT} \gg (I_{CACT}/\beta)$ . Here,  $I_{CACT}$  is the active-region collector current.

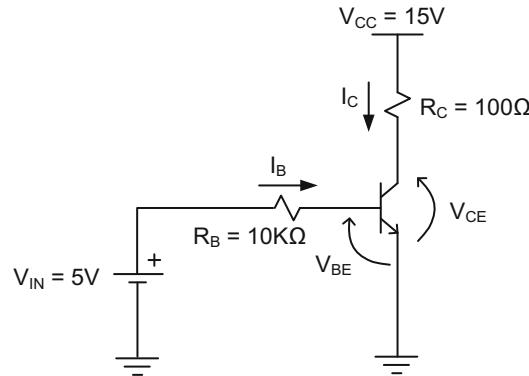


**Fig. 2.24** Transistor bias currents and voltages in saturation mode of operation

## 2.7 Bipolar Transistor Circuits

This section is dedicated to using BJTs in simple circuits, and analyzing the circuit behavior when the transistor is in active and saturation regions.

**Example 2.1** A simple NPN bipolar transistor circuit that consists of an NPN bipolar transistor,  $R_B$  and  $R_C$  is given in Fig. 2.25. If  $V_{CC} = 15$  V,  $V_{IN} = 5$  V and  $\beta = 100$ , calculate  $I_B$ ,  $I_C$ . What mode will this transistor operate in?



**Fig. 2.25** A simple NPN bipolar circuit

The first task is to write Kirchoff's Voltage Law (KVL) around the BE and CE junctions as shown in Eqs. 2.6 and 2.7. Therefore,

$$V_{IN} = R_B I_B + V_{BE} \quad (2.6)$$

$$V_{CC} = R_C I_C + V_{CE} \quad (2.7)$$

Test for cut-off mode:

Set  $I_B = I_C = 0$  A. This produces  $V_{BE} = 5$  V and  $V_{CE} = 15$  V, and it proves that the transistor does not operate in cut-off region.

Test for active mode, set  $V_{BE} = 0.7$  V and solve  $I_B$  from Eq. 2.6. Thus,

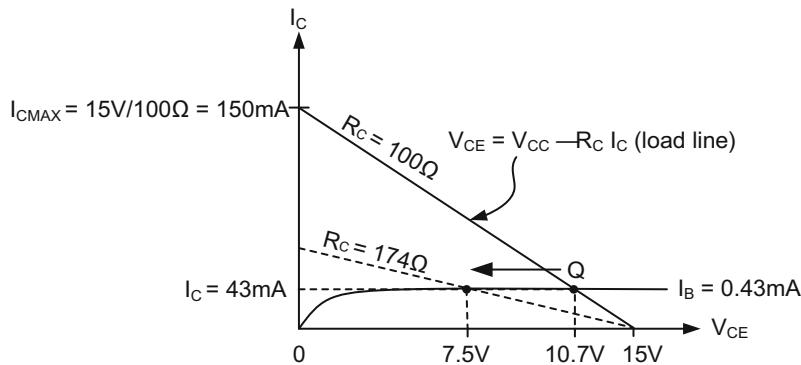
$$I_B = \frac{(V_{IN} - V_{BE})}{R_B} = \frac{(5 - 0.7)}{10 \text{ K}\Omega} = 0.43 \text{ mA}$$

Then,

$I_C = \beta I_B = 100 (0.43 \text{ mA}) = 43 \text{ mA}$ . Substitute  $I_C = 43 \text{ mA}$  into Eq. 2.7 and solve for  $V_{CE}$ . Thus,

$$V_{CE} = V_{CC} - R_C I_C = 15 - 0.1 \text{ K}\Omega (43 \text{ mA}) = 15 - 4.3 = 10.7 \text{ V}$$

This value is greater than 0.2 V and proves that the transistor operates in the active region as shown in Fig. 2.26. In this figure, Eq. 2.7 is superimposed on top of the transistor  $I_C$ - $V_{CE}$  characteristics in order to show where exactly this transistor operates at (quiescent point, Q).



**Fig. 2.26** The quiescent point of the transistor

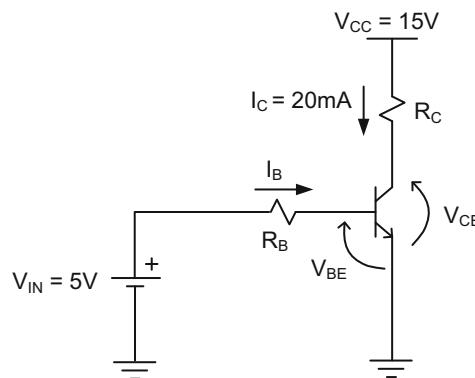
However, the location of the  $Q$  point is not optimal in this figure, and as it needs to be closer to the midpoint between 0 and  $V_{CC}$  such as  $V_{CEQ} = V_{CC}/2 = 7.5$  V to be more suitable for analog applications.

From Fig. 2.26, when  $I_B = 0.43$  mA and  $V_{CE} = 7.5$  V,  $I_C$  becomes 43 mA. Thus,

$$R_C = \frac{(V_{CC} - V_{CEQ})}{I_C} = \frac{(15 - 7.5)}{0.043} = 174\Omega \text{ instead of } 100\Omega.$$

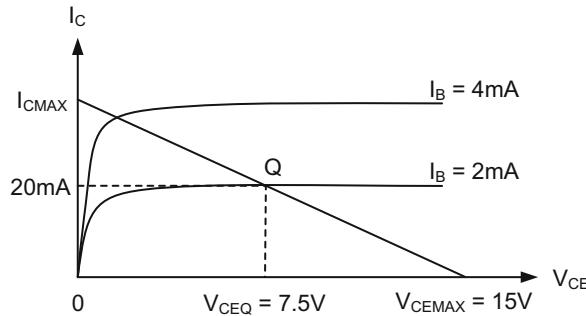
The new load line is plotted in Fig. 2.26 with dashed lines.

**Example 2.2** Another bipolar transistor circuit is given in Fig. 2.27. In this schematic,  $I_C$  is selected to be 20 mA. The circuit still uses  $V_{IN} = 5$  V and  $V_{CC} = 15$  V for its operation. What are the values of  $R_B$  and  $R_C$  such that this transistor operates in the middle of the active region at  $V_{CEQ} = 7.5$  V?



**Fig. 2.27** A bipolar circuit whose  $I_C = 20$  mA, but its  $R_C$  and  $R_B$  are unknown

For this design question, one must refer to the manufacturer's datasheet and fetch the transistor  $I_C$ - $V_{CE}$  characteristics as shown in Fig. 2.28.



**Fig. 2.28** Manufacturer's I-V characteristics of the NPN transistor

The load line from Eq. 2.7 is superimposed on top of the  $I_C$ - $V_{CE}$  characteristics, and  $I_B$  becomes 2 mA at  $V_{CEQ} = 7.5$  V when  $I_C = 20$  mA. Since the transistor operates in active region,  $\beta = I_C/I_B$  becomes 10. Rewriting Eq. 2.7 yields:

$$R_C = \frac{(V_{CC} - V_{CEQ})}{I_C} = \frac{(15 - 7.5)}{0.020} = 375 \Omega$$

Rewriting Eq. 2.6 yields

$$R_B = \frac{(V_{IN} - V_{BE})}{I_B} = \frac{(5 - 0.7)}{0.002} = 2.15 \text{ K}\Omega$$

However, there may be an instance where the load line may not necessarily intersect any of the  $I_C$ - $V_{CE}$  curves at  $V_{CEQ} = 7.5$  V provided by the manufacturer, and the situation becomes similar to the case in Fig. 2.29. In this instance,  $R_C$  is still calculated using Eq. 2.7 since  $V_{CEQ}$  and  $I_{CQ}$  have not changed.

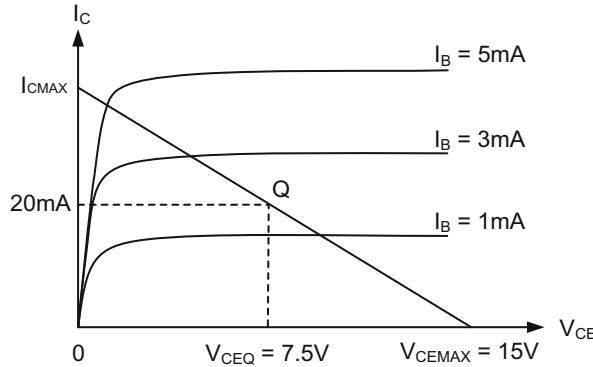
$$R_C = \frac{(V_{CC} - V_{CEQ})}{I_C} = \frac{(15 - 7.5)}{0.020} = 375 \Omega$$

However, to calculate  $R_B$  we need to use  $\beta = 10$  calculated earlier. Thus,

$$I_B = \frac{I_C}{\beta} = \frac{20 \text{ mA}}{10} = 2 \text{ mA}$$

Then using Eq. 2.6,  $R_B$  can be calculated as

$$R_B = \frac{(V_{IN} - V_{BE})}{I_B} = \frac{(5 - 0.7)}{0.002} = 2.15 \text{ K}\Omega$$



**Fig. 2.29** An instance where load line does not intersect any I-V characteristics

**Example 2.3** Suppose the bipolar transistor circuit in Fig. 2.27 needs to operate in the saturation region where  $I_{CSAT} = 40 \text{ mA}$  and  $V_{CESAT} = 0.2 \text{ V}$ . What are the values of  $R_B$  and  $R_C$ ?

From Eq. 2.7, we have:

$$R_C = \frac{(V_{CC} - V_{CESAT})}{I_{CSAT}} = \frac{(15 - 0.2)}{0.040} = 370 \Omega$$

To find  $R_B$ , we need to use Eq. 2.6. Thus,

$$R_B = \frac{(V_{IN} - V_{BE})}{I_{BSAT}} = \frac{(5 - 0.8)}{I_{BSAT}} = \frac{4.2}{I_{BSAT}}$$

In saturation, the circuit operates with the load line no.1 as shown in Fig. 2.30, producing  $I_C = 40 \text{ mA}$  at  $Q_{SAT}$ . At this point however,  $I_{BSAT}$  is not known other than  $I_{BSAT} \gg I_{BACT}$ . The ratio between  $I_{BSAT}$  and  $I_{BACT}$  can be set as high as 10, although this number is very conservative. A more practical the multiplication factor may be in the neighborhood of 2 or 3 depending on  $I_C$ - $V_{CE}$  characteristics of the transistor.

Now,  $R_B$  becomes:

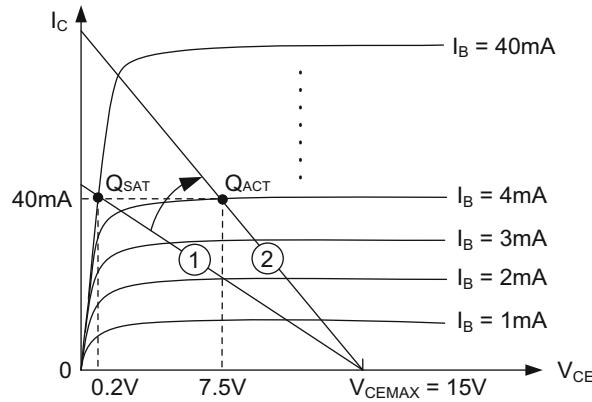
$$R_B = \frac{4.2}{10I_{BACT}}$$

To find  $I_{BACT}$  in the equation above, we need to operate the transistor in the active region with the same collector current when the transistor was in saturation region. That means that  $I_{CACT}$  needs to be 40 mA, which rotates the load line clockwise around  $V_{CEMAX} = 15 \text{ V}$  to position no. 2 so that the transistor operates in the middle of active region at  $V_{CE} = 7.5 \text{ V}$ . At this new point,  $Q_{ACT}$ , load line no. 2 intersects only one  $I_B$ -curve:  $I_B = 4 \text{ mA}$ . Therefore,  $I_{BACT}$  becomes 4 mA, and  $I_{BSAT} = 10I_{BACT} = 40 \text{ mA}$ .

Thus,

$$R_B = \frac{4.2}{I_{BSAT}} = \frac{4.2}{0.040} = 105 \Omega$$

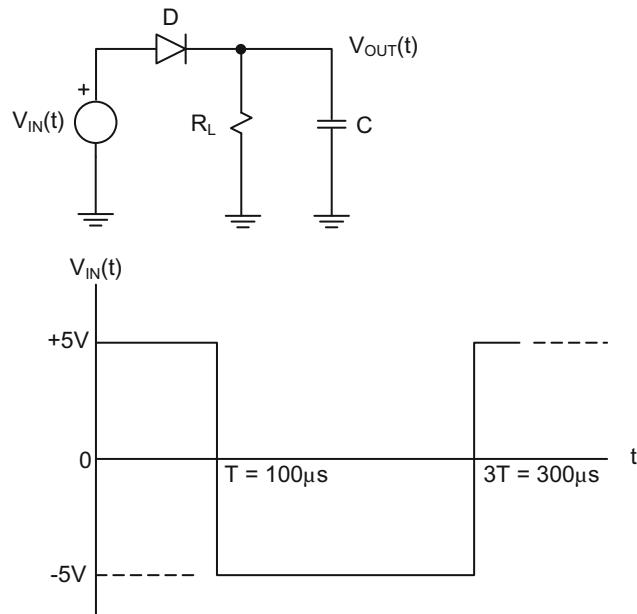
With  $R_C = 370 \Omega$  and  $R_B = 105 \Omega$ , the collector and the base currents both become 40 mA, and the transistor operates in saturation region with load line no. 1.



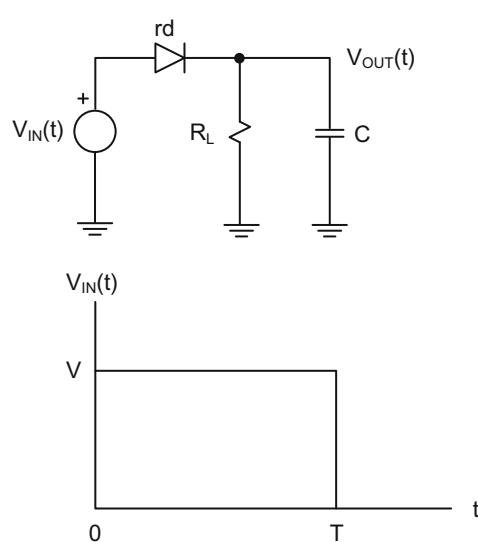
**Fig. 2.30** Load line rotation to make  $I_{CSAT} \approx I_{CACT}$  to allow  $I_{BSAT} \gg I_{BACT}$

## Review Questions

1. The following rectifier circuit is given:

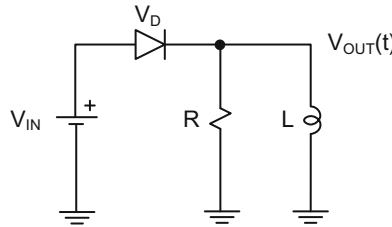


- (a) If the forward-biased diode voltage is 1 V,  $C = 10 \mu F$  and  $R_L = 10 \Omega$  (load resistance), calculate the expression for  $V_{OUT}(t)$  and plot the output voltage.
  - (b) If  $R_L = 100 \Omega$ , what is the value of  $C$  if  $V_{OUT}(t)$  drops by 1 V between  $T = 100 \mu s$  and  $3T = 300 \mu s$ ?
2. The following diode circuit is given:

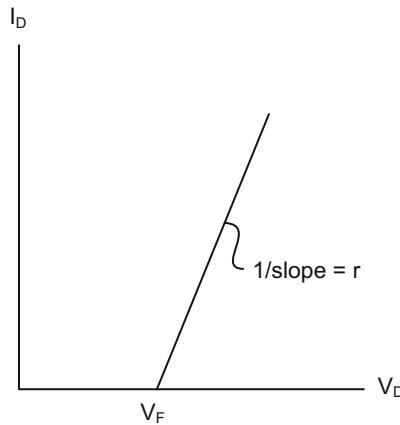


In this circuit, the equivalent resistance of the diode is  $r_d$  when the diode turns on at a forward bias voltage of  $V_F$ , where  $V_F$  is much smaller compared to  $V$  for  $0 < t < T$ . Assuming that  $V_{OUT}(0) = 0$  V and the time constant of the circuit is much smaller than  $T$ , determine the output voltage,  $V_{OUT}(t)$ , for  $0 < t < T$  and  $t > T$ . Natural frequencies method described in Chapter 1 may be used to obtain the equation for  $V_{OUT}(t)$ . Plot the output waveform.

3. The following diode circuit is given:

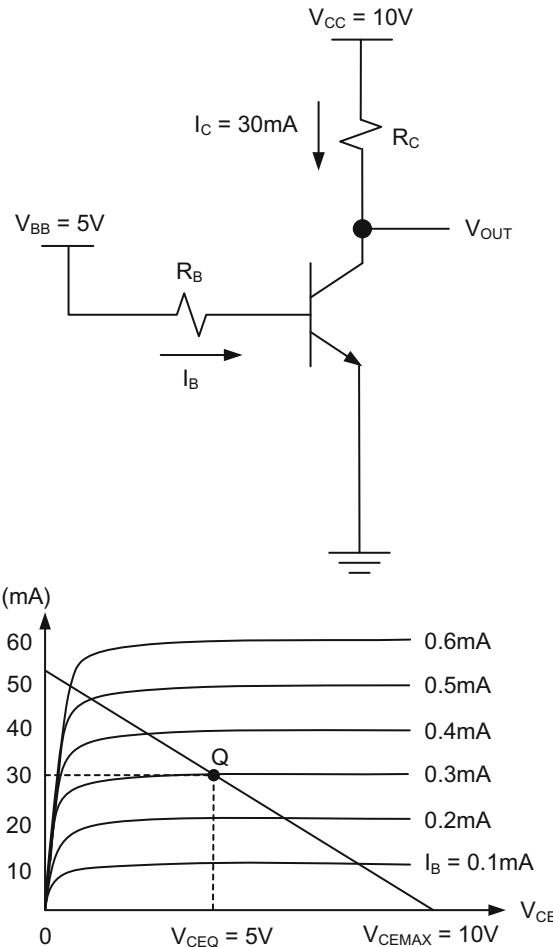


In this circuit above, when  $V_{IN}$  is reduced below  $V_F$  the diode turns off. The inverse slope in the I-V characteristics below gives the diode internal resistance,  $r$ .



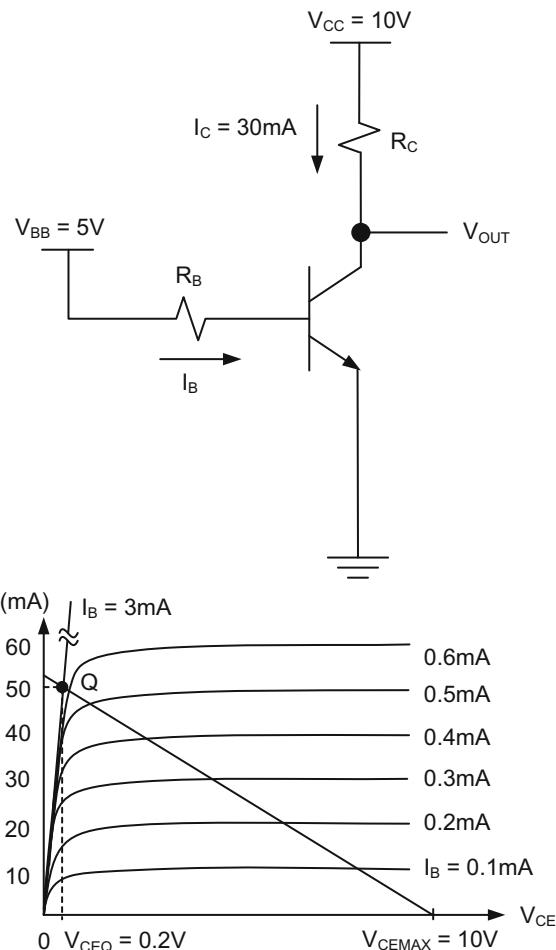
- (a) Compute and plot the current through the inductor if the initial inductor current is 0 A. Either time domain analysis or natural frequencies method may be used to compute the result.
- (b) Compute and plot the voltage across the output,  $V_{OUT}$ . Either time domain analysis or natural frequencies method may be used to compute the result.
4. The NPN bipolar transistor is used to operate in the middle of active region. The transistor requires a collector current,  $I_C = 30$  mA, when the output of the circuit,  $V_{OUT}$ , is at 5 V. The transistor develops  $V_{BE} = 0.7$  V in active region and maintains a current gain of 100, i.e.  $\beta = I_C/I_B = 100$ .

Use the output  $I_C$ - $V_{CE}$  characteristics below to compute  $R_B$  and  $R_C$  that satisfy the proper transistor operation.



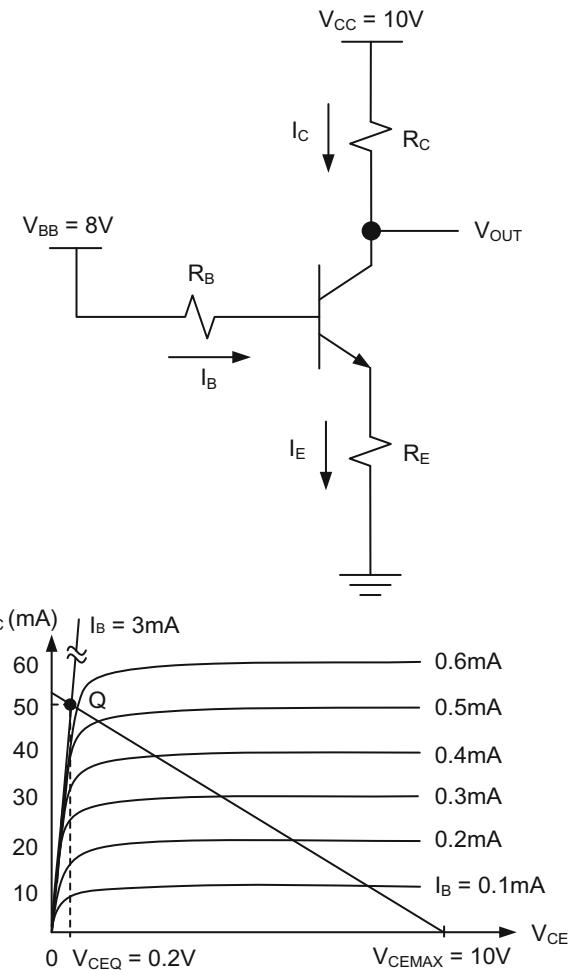
5. The same NPN bipolar transistor in question 4 now operates in saturation region where it develops  $V_{BESAT} = 0.8$  V and  $V_{CESAT} = 0.2$  V. It also produces a base current,  $I_{BSAT}$ , 10 times larger than the active-region base current,  $I_{BACT}$ .

If the transistor requires a collector current,  $I_{CSAT} = 30 \text{ mA}$ , to operate in saturation region, find  $R_B$  and  $R_C$  to satisfy the proper transistor operation. Use the  $I_C$ - $V_{CE}$  characteristics below for calculations.



- In the following circuit, the NPN bipolar transistor is used in such a way that it operates in the saturation region. The circuit requires a collector current,  $I_{CSAT} = 30 \text{ mA}$ , when its output,  $V_{OUT}$ , is at 5 V. The transistor develops  $V_{BESAT} = 0.8 \text{ V}$  and  $V_{CESAT} = 0.2 \text{ V}$  in saturation region when its base current,  $I_{BSAT}$ , is set 10 times higher compared to the base current when it is in active region,  $I_{BACT}$ .

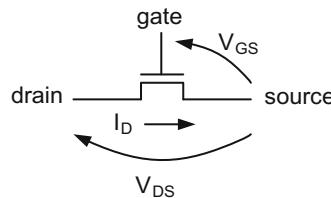
Find  $R_C$ ,  $R_B$  and  $R_E$  that satisfy the proper transistor operation.



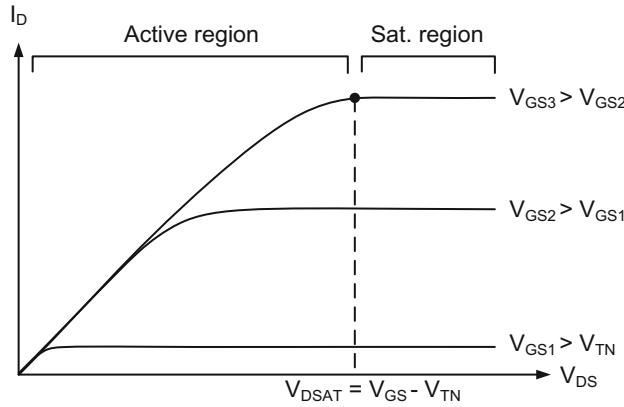
### 3.1 N-Channel MOSFET (NMOSFET)

N-channel Metal-Oxide-Semiconductor-Field-Effect-Transistor (NMOSFET) is a three-terminal device that produces a drain current,  $I_D$ , when positive voltages are applied to the gate and drain terminals with respect to the source terminal as shown in Fig. 3.1.

When the conduction starts, the device produces  $I_D$ - $V_{DS}$  characteristics for different values of  $V_{GS}$  as shown in Fig. 3.2. Each curve in this figure is obtained by increasing  $V_{DS}$  and measuring  $I_D$  at a constant  $V_{GS}$ . Each I-V curve consists of two distinct regions: an active region and a saturation region. The device transitions from the active region into the saturation region when  $V_{DS}$  reaches a saturation potential,  $V_{DSAT} = V_{GS} - V_{TN}$ . Here,  $V_{TN}$  is the threshold voltage applied to the gate to turn on the transistor.



**Fig. 3.1** NMOSFET circuit symbol



**Fig. 3.2** NMOSFET current-voltage characteristics

The I-V curve in Fig. 3.2 can be explained using the following equation:

$$I_D = \frac{\mu_N C_{OX} W_N}{L_N} \left[ (V_{GS} - V_{TN}) V_{DS} - \frac{V_{DS}^2}{2} \right] \quad (3.1)$$

where,

$\mu_N$  is the mobility of electrons in the drain-source channel in  $\text{cm}^2/\text{Vsec}$

$C_{OX}$  is the gate oxide capacitance in  $\text{F/cm}^2$

$W_N$  is the width of the NMOS transistor

$L_N$  is the length of the NMOS transistor

$V_{TN}$  is the threshold voltage of the NMOS transistor

For small values of  $V_{DS}$ , Eq. 3.1 can be approximated as:

$$I_D = \frac{\mu_N C_{OX} W_N}{L_N} (V_{GS} - V_{TN}) V_{DS} \quad (3.2)$$

This equation represents the active region of operation where the NMOS transistor reveals a linear relationship between  $I_D$  and  $V_{DS}$ . Therefore, in this region, NMOSFET can be represented as a resistor as shown in Fig. 3.3. In this figure, the power supply voltage,  $V_{DD}$ , is the maximum allowable voltage applied to the gate of the transistor.

$$R_N = \frac{V_{DS}}{I_D} = \frac{L_N}{\mu_N C_{OX} (V_{DD} - V_{TN}) W_N} \quad (3.3)$$

The constant terms in Eq. 3.3 can be combined and equated to  $K_N$  as shown below.

$$K_N = \frac{L_N}{\mu_N C_{OX} (V_{DD} - V_{TN})}$$

Then Eq. 3.3 becomes:

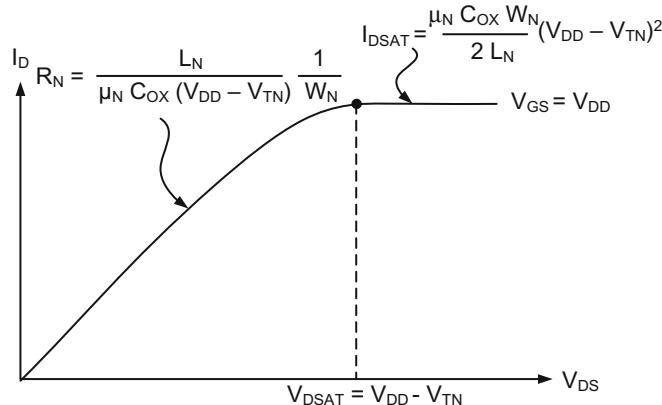
$$R_N = \frac{K_N}{W_N} \quad (3.4)$$

Equation 3.4 becomes approximately equal to the inverse slope of the  $I_D$ - $V_{DS}$  curve in Fig. 3.3 for  $V_{DS}$  smaller than  $V_{DSAT} = V_{DD} - V_{TN}$ .

For  $V_{DS} > V_{DSAT} = V_{DD} - V_{TN}$ , however, the NMOS transistor goes into the saturation region as shown in Fig. 3.3.  $I_{DSAT}$  can be obtained by substituting  $V_{GS} = V_{DD}$  and  $V_{DSAT} = V_{DD} - V_{TN}$  in Eq. 3.1. Thus,

$$I_D = \frac{\mu_N C_{OX} W_N}{2 L_N} (V_{DD} - V_{TN})^2 \quad (3.5)$$

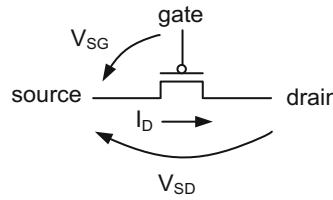
Equation 3.5 is a constant and independent of  $V_{DS}$ . Therefore, the NMOS transistor can be represented as a constant current when it is in saturation region.



**Fig. 3.3** NMOSFET  $I_D$ - $V_{DS}$  curve when  $V_{DS} = V_{DD}$  and the equivalent circuits

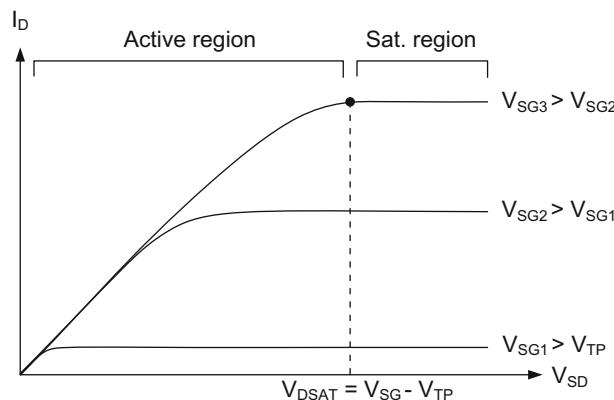
## 3.2 P-Channel MOSFET (PMOSFET)

As a complementary transistor to NMOSFET, P-channel MOSFET (PMOSFET) is also a three-terminal device which produces a drain current,  $I_D$ , when negative voltages are applied to the gate and drain terminals with respect to the source as shown in Fig. 3.4.



**Fig. 3.4** PMOSFET circuit symbol

The device produces  $I_D$ - $V_{SD}$  characteristics for different values of  $V_{SG}$  as shown in Fig. 3.5. Each curve in this figure is obtained at a constant  $V_{SG}$ , and composed of the active and saturation regions. As in NMOS transistor, the device transitions from the active region to the saturation region when  $V_{SD}$  reaches a saturation potential,  $V_{DSAT} = V_{SG} - V_{TP}$ , where  $V_{TP}$  is the threshold voltage to turn on the transistor.



**Fig. 3.5** PMOSFET current-voltage characteristics

The mathematical expression in Eq. 3.6 below identifies the  $I_D$ - $V_{SD}$  characteristics above.

$$I_D = \frac{\mu_P C_{OX} W_P}{L_P} \left[ (V_{SG} - V_{TP}) V_{SD} - \frac{V_{SD}^2}{2} \right] \quad (3.6)$$

where,

$\mu_P$  is the mobility of holes in the drain-source channel in  $\text{cm}^2/\text{Vsec}$

$C_{OX}$  is the gate oxide capacitance in  $\text{F/cm}^2$

$W_P$  is the width of the PMOS transistor

$L_P$  is the length of the PMOS transistor

$V_{TP}$  is the threshold voltage of the PMOS transistor

For small values of  $V_{SD}$ , Eq. 3.6 becomes:

$$I_D = \frac{\mu_P C_{OX} W_P}{L_P} (V_{SG} - V_{TP}) V_{SD} \quad (3.7)$$

Like its NMOS counterpart, Eq. 3.7 represents the active region of operation for the PMOS transistor. This equation can be approximated as a resistor in Eq. 3.8 when the power supply voltage,  $V_{DD}$ , is applied between the gate and source terminals.

$$R_P = \frac{V_{SD}}{I_D} = \frac{L_P}{\mu_P C_{OX} (V_{DD} - V_{TP}) W_P} \quad (3.8)$$

Combining all constant terms in Eq. 3.8 yields:

$$K_P = \frac{L_P}{\mu_P C_{OX} (V_{DD} - V_{TP})}$$

Then Eq. 3.8 becomes:

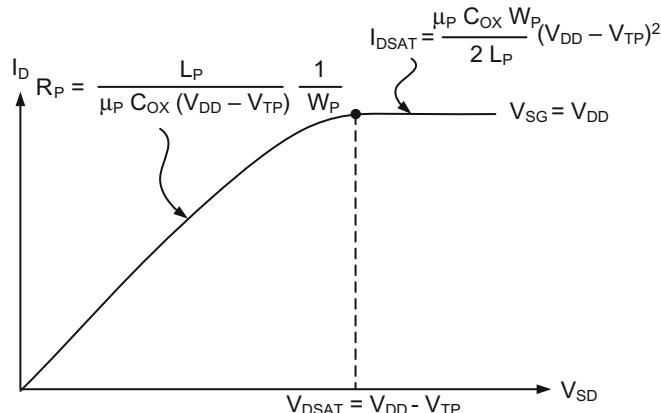
$$R_P = \frac{K_P}{W_P} \quad (3.9)$$

This equation is approximately equal to the inverse slope of the  $I_D$ - $V_{SD}$  curve in Fig. 3.6 when  $V_{SG}$  is smaller than  $V_{DD} - V_{TP}$ .

For  $V_{SD} > V_{DSAT} = V_{DD} - V_{TP}$ , the PMOS transistor goes into the saturation region. The current in this region can be obtained by substituting  $V_{SG} = V_{DD}$  and  $V_{DSAT} = V_{DD} - V_{TP}$  into Eq. 3.6. Thus,

$$I_D = \frac{\mu_P C_{OX} W_P}{2 L_P} (V_{DD} - V_{TP})^2 \quad (3.10)$$

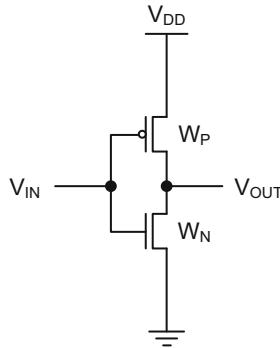
Equation 3.10 is also constant and independent of  $V_{SD}$ . Therefore, the PMOS transistor is represented as a constant current when it is in saturation region.



**Fig. 3.6** PMOSFET equivalent circuit

### 3.3 Complementary MOS (CMOS) Inverter

The first Complementary-Metal-Oxide-Semiconductor (CMOS) logic gate we will examine is the inverter. This gate consists of a single NMOS transistor in series with a PMOS transistor as shown in Fig. 3.7.



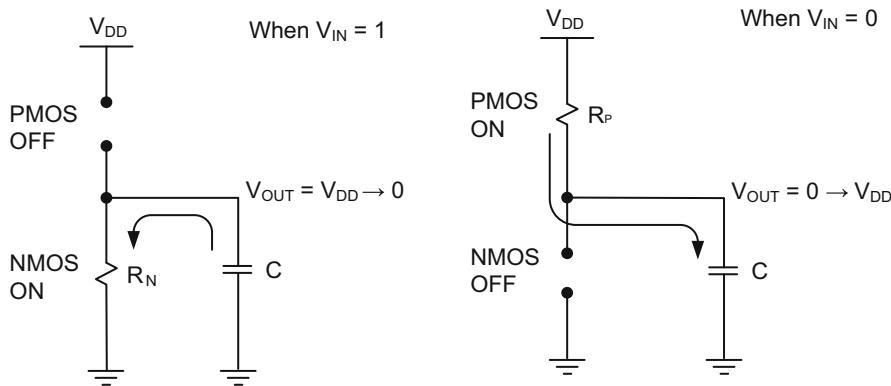
**Fig. 3.7** CMOS inverter

In this circuit, when the input voltage,  $V_{IN}(t)$ , becomes equal to  $V_{DD}$  (logic 1), the NMOS transistor turns on (because  $V_{GS} = V_{DD} > V_{TN}$ ), and the PMOS transistor turns off (because  $V_{SG} = 0$  V). The capacitance,  $C_L$ , at the output terminal is then discharged through the equivalent NMOS resistance,  $R_N$ , as shown in the left side of Fig. 3.8, and the output voltage decays towards 0 V with a RC time constant of  $R_N C_L$ . In other words,  $V_{OUT}$  becomes:

$$V_{OUT}(t) = V_{DD} \exp\left(-\frac{t}{R_N C}\right) \quad (3.11)$$

When  $V_{IN}(t) = 0$  V, on the other hand, NMOS transistor turns off (because  $V_{GS} = 0$  V), and PMOS turns on (because  $V_{SG} = V_{DD} > V_{TP}$ ). Consequently, a charge path forms through the equivalent PMOS resistor to charge the output capacitance from 0 V towards  $V_{DD}$  as shown in the right side of Fig. 3.8. In other words,

$$V_{OUT}(t) = V_{DD} \left[ 1 - \exp\left(-\frac{t}{R_P C}\right) \right] \quad (3.12)$$

**Fig. 3.8** CMOS inverter discharge path (*left*) and charge path (*right*)

### 3.4 Two-Input CMOS NAND Logic Gate

The two-input CMOS NAND gate is shown in Fig. 3.9. This logic gate is composed of two sections: an NMOS tree (between the output terminal and the ground), and a PMOS tree (between the power supply voltage and the output terminal). To construct the NMOS tree, two NMOS transistors are connected in series between the output terminal and the ground. The reason for this configuration is that if one of the NMOS transistors turns off, there will be no discharge path to lower  $V_{OUT}$ , and  $V_{OUT}$  stays at  $V_{DD}$ . However, if both NMOS transistors are turned on,  $V_{OUT}$  transitions to 0 V, which complies with the truth table for two-input NAND gate.

The next phase is to form the PMOS tree. The PMOS tree is configured in a complementary manner to the NMOS tree. This means that any series combination of NMOS transistors in the NMOS tree must be transformed into a parallel combination of PMOS transistors in the PMOS tree, and vice versa. Therefore, we need to connect two PMOS transistors in parallel to form the PMOS tree, and place it between  $V_{OUT}$  and  $V_{DD}$  as shown in Fig. 3.9.

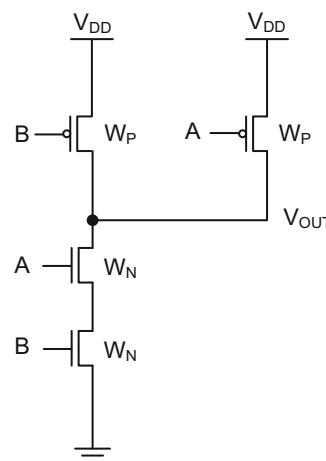
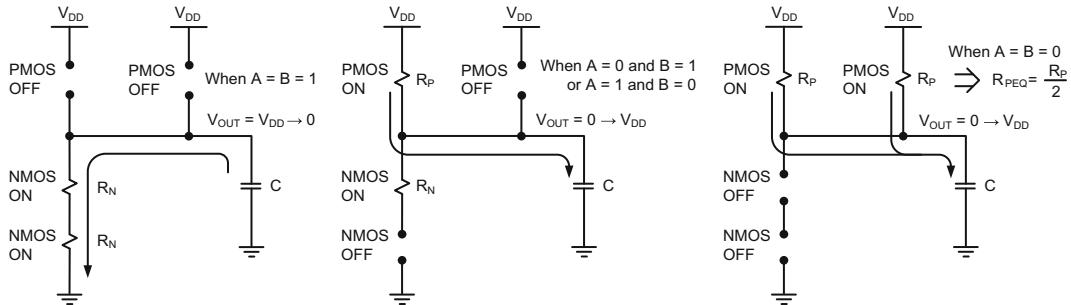
**Fig. 3.9** Two-input CMOS NAND gate

Figure 3.10 shows the discharge and charge paths for the two-input NAND gate. The only discharge path in this figure is through the series combination of two equivalent NMOS resistors as shown by the leftmost figure in Fig. 3.10. This yields the following output voltage:

$$V_{OUT}(t) = V_{DD} \exp\left(-\frac{t}{2R_N C}\right) \quad (3.13)$$

The worst-case charge path, on the other hand, forms as a result of turning on only one of the PMOS transistors in the PMOS tree shown in the middle part of Fig. 3.10, and results in the following output voltage:

$$V_{OUT}(t) = V_{DD} \left[ 1 - \exp\left(-\frac{t}{R_P C}\right) \right] \quad (3.14)$$



**Fig. 3.10** Two-input CMOS NAND gate discharging (*left*) and charging (*middle* and *right*) paths

The best-case charge path turns on both PMOS transistors in the PMOS tree, and results in charging the output node in a faster pace as shown by the rightmost figure in Fig. 3.10. Equation 3.15 shows this behavior:

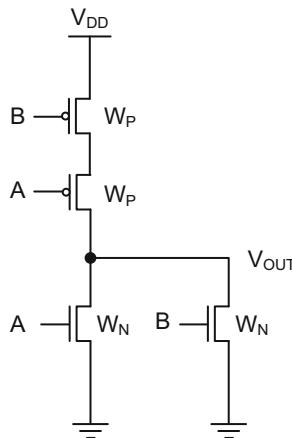
$$V_{OUT}(t) = V_{DD} \left[ 1 - \exp\left(-\frac{2t}{R_P C}\right) \right] \quad (3.15)$$

In Eqs. 3.13 to 3.15, the capacitance term,  $C$ , is associated with the intrinsic capacitance at the output node of the two-input NAND gate in Fig. 3.9. If there is an external capacitance at the output node, it must be added to the intrinsic capacitance before calculating the value of  $V_{OUT}$ .

### 3.5 Two-Input CMOS NOR Logic Gate

The two-input CMOS NOR gate is shown in Fig. 3.11. To configure this logic gate, an NMOS tree composed of two parallel NMOS transistors is formed first. This arrangement ensures  $V_{OUT}$  to be 0 V if one or more NMOSFETs in the NMOS tree are turned on. When both devices are turned off, then  $V_{OUT}$  becomes  $V_{DD}$  as dictated by the truth table of this gate.

The PMOS tree is formed in a complementary fashion to the NMOS tree, and results in two PMOS transistors in series between  $V_{DD}$  and  $V_{OUT}$  as shown in Fig. 3.11.



**Fig. 3.11** Two-input CMOS NOR gate

Figure 3.12 shows a discharge path and two charge paths for the two-input NOR gate. The worst-case discharge path in this figure is through an NMOS transistor as shown by the leftmost figure in Fig. 3.12. This yields the following output voltage:

$$V_{OUT}(t) = V_{DD} \exp\left(-\frac{t}{R_N C}\right) \quad (3.16)$$

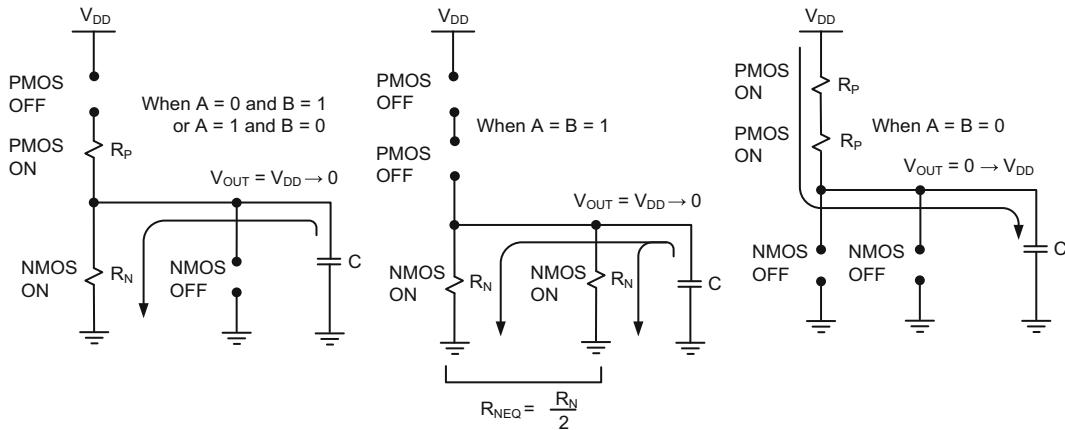
The best-case discharge path is when both NMOS transistors are turned on as shown by the center figure in Fig. 3.12. This produces a faster discharge rate as described by Eq. 3.17 below.

$$V_{OUT}(t) = V_{DD} \exp\left(-\frac{2t}{R_N C}\right) \quad (3.17)$$

The only charge path is to turn on both PMOS transistors in the PMOS tree as shown by the rightmost figure in Fig. 3.12. It results in the following output voltage:

$$V_{OUT}(t) = V_{DD} \left[ 1 - \exp\left(-\frac{t}{2R_pC}\right) \right] \quad (3.18)$$

In Eqs. 3.16 to 3.18, the capacitance term, C, is associated with the intrinsic capacitance at the output node of the two-input NOR gate in Fig. 3.11. If there is an external capacitance at the output node, this capacitance must be added to the intrinsic capacitance before calculating the value of  $V_{OUT}$ .



**Fig. 3.12** Two-input CMOS NOR gate discharging (*left and middle*) and charging (*right*) paths

### 3.6 Complex CMOS Logic Gate Implementation

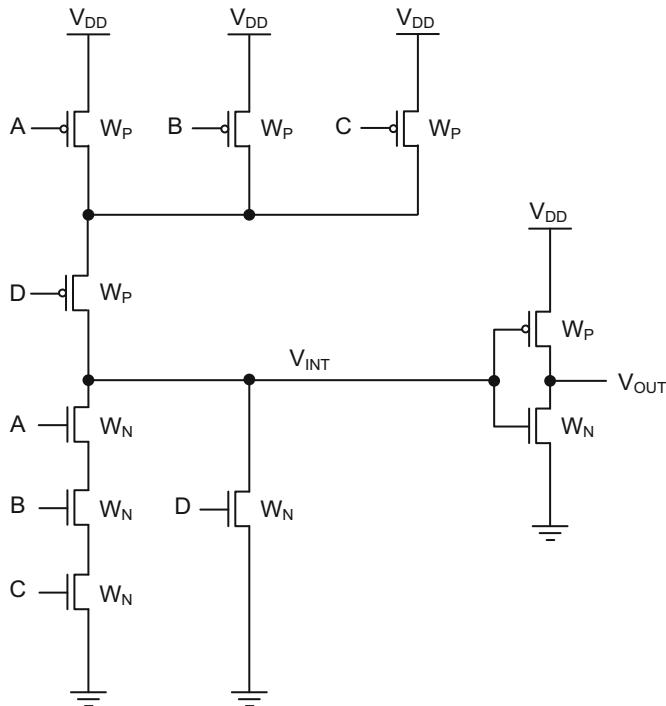
The best way to show how to build a complex CMOS logic gate is to give a comprehensive example, outlining each step in the implementation one at a time, and discussing the design trade-offs when they become necessary. For this particular reason, implementing a logic function, such as  $out = A \cdot B \cdot C + D$ , serves these two criteria. First, this logic gate is composed of AND and OR logic functions. Second, it calls for numerous design trade-offs during its implementation. There are basically two ways to implement this logic function.

The first method uses discrete CMOS gates. This produces a three-input AND gate with the A, B and C inputs, and a two-input OR gate, combining the D input with the output of the AND gate.

The second method integrates all the AND and OR functions of this complex gate in a single gate. This method forms a combined NMOS tree first, and then builds a complementary logic function in the PMOS tree. When forming the NMOS tree, we must remember connecting all NMOS transistors in series if we need to implement an AND function, and in parallel to implement an OR function.

Therefore, to implement  $\text{out} = A \cdot B \cdot C + D$ , we start with connecting three NMOS transistors with the A, B and C inputs in series to form the AND function. Then, we connect another NMOS transistor with the D input in parallel to this combination to complete the NMOS tree as shown in Fig. 3.13.

PMOS tree is formed in a complementary fashion: the PMOS transistors with the A, B and C inputs have to be connected in parallel because NMOS transistors with the same inputs were connected in series in the NMOS tree. This parallel combination is then connected in series with a PMOS transistor with the D input because the NMOS transistor with D input was connected in parallel in the NMOS tree. However, this process only produces an output function equal to  $\text{out} = \overline{(A \cdot B \cdot C + D)}$  if the implementation stops here. We need to add an inverter at the output of the logic gate to further convert the complemented logic function to its uncomplemented form,  $\text{out} = A \cdot B \cdot C + D$ .



**Fig. 3.13** Implementation of  $\text{out} = A \cdot B \cdot C + D$

The worst-case discharge and charge paths of this circuit are shown in Fig. 3.14. Both charging and discharging the output node takes two steps.

To be able to charge the output node, three equivalent NMOS resistors connected in series need to discharge the intermediate node,  $V_{INT}$ , as shown by the top figure in Fig. 3.14. The total capacitance at  $V_{INT}$  is approximately the input capacitance of the inverter,

$C_{INV}$ , which is the combination of NMOS and PMOS transistor gate capacitances. Therefore,

$$C_{INV} = C_{GN} + C_{GP} = C_{ox}(W_N + W_P)L \text{ where } L = L_N = L_P \quad (3.19)$$

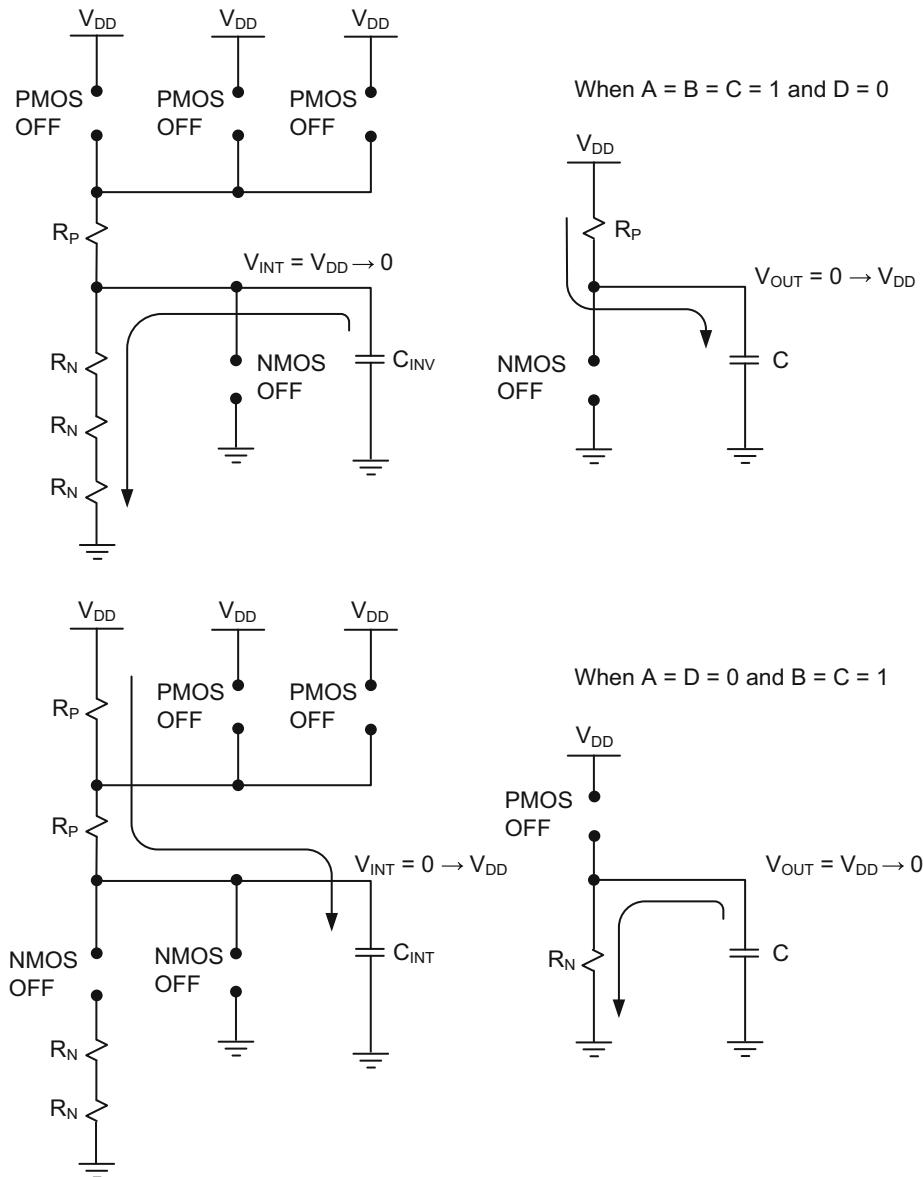
Once the charge across  $C_{INV}$  is drained away, and  $V_{INT}$  is lowered to 0 V, then in the second step, the PMOS transistor in the inverter turns on to charge the output node towards  $V_{DD}$ .

To discharge the output node, the intermediate node needs to be charged by two equivalent PMOS resistors connected in series as shown by the bottom figure in Fig. 3.14. Once  $V_{INT}$  reaches  $V_{DD}$ , then the equivalent NMOS resistor in the inverter discharges the output node.

Another method to form  $out = A \cdot B \cdot C + D$  calls for the complemented inputs,  $\bar{A}$ ,  $\bar{B}$ ,  $\bar{C}$  and  $\bar{D}$ , applied to the input of the logic gate if they are available. This method follows manipulating the logic function using De Morgan's theorem.

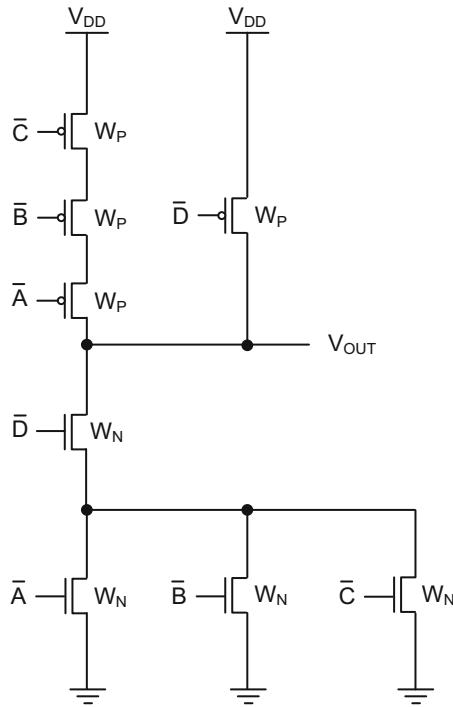
In other words,

$$out = A \cdot B \cdot C + D = \overline{\overline{(A \cdot B \cdot C + D)}} = \overline{(\bar{A} + \bar{B} + \bar{C}) \cdot \bar{D}} \quad (3.20)$$



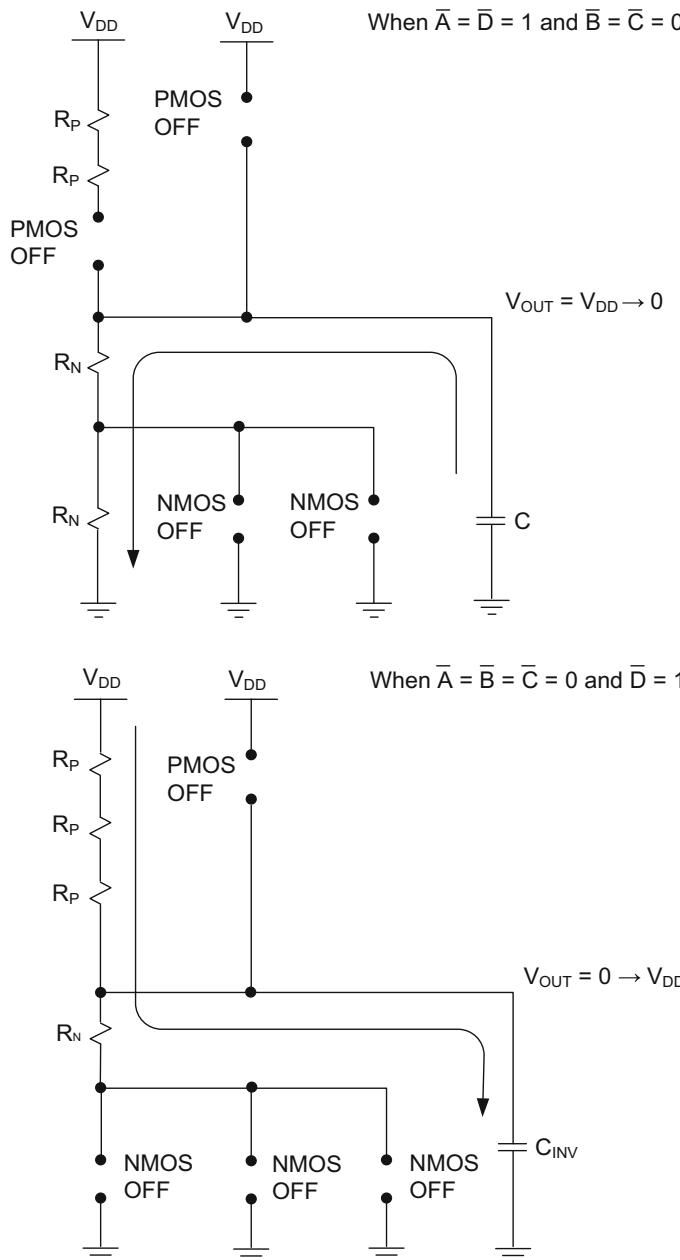
**Fig. 3.14** Worst-case charge and discharge paths for  $\text{out} = \text{A} \cdot \text{B} \cdot \text{C} + \text{D}$

The logic function in Eq. 3.20 has already been complemented, and does not require an additional inverter at the output as shown in Fig. 3.15. To form the NMOS tree of this circuit, three NMOS transistors with complemented inputs,  $\bar{A}$ ,  $\bar{B}$  and  $\bar{C}$ , are connected in parallel. Then this combination is connected in series with another NMOS transistor with  $\bar{D}$  input as shown in Fig. 3.15. The PMOS tree requires three PMOS transistors in series with  $\bar{A}$ ,  $\bar{B}$  and  $\bar{C}$  inputs, and a single PMOS transistor with  $\bar{D}$  input in parallel to this combination.



**Fig. 3.15** Implementation of  $out = A \cdot B \cdot C + D = (\bar{\bar{A}} + \bar{\bar{B}} + \bar{\bar{C}}) \cdot \bar{D}$  (no output inverter is needed but all inputs need to be inverted)

Figure 3.16 shows the worst-case discharge and charge paths for the second implementation in Fig. 3.15. To discharge the output node, two equivalent NMOS resistors drain the output voltage towards 0 V as shown by the top figure in Fig. 3.16. Charging the output node, on the other hand, requires all three PMOS transistors to be turned on as shown by the bottom figure in Fig. 3.16.



**Fig. 3.16** Worst-case charge and discharge paths for  $out = A \cdot B \cdot C + D = (\bar{A} + \bar{B} + \bar{C}) \cdot \bar{D}$

### 3.7 Rise and Fall Times

Rise time,  $T_R$ , is defined as the time interval during which the output rises from 20 to 80% of its final value as shown in Fig. 3.17. This value can be calculated analytically if the term  $R_P$  in Eq. 3.12 is replaced by  $R_{PEQ}$ , signifying a total equivalent PMOS resistance in the worst-case charge path. Therefore,

$$V_{OUT}(t) = V_{DD} \left[ 1 - \exp\left(-\frac{t}{R_{PEQ}C}\right) \right] \quad (3.21)$$

Here,  $C$  is the sum of all intrinsic and load capacitances at the output node. From this equation, 20% of  $V_{OUT}$  at  $t = T_1$  becomes:

$$0.2V_{DD} = V_{DD} \left[ 1 - \exp\left(-\frac{T_1}{R_{PEQ}C}\right) \right]$$

or

$$T_1 = -R_{PEQ}C \ln(0.8) \quad (3.22)$$

Similarly, 80% of  $V_{OUT}$  at  $t = T_2$  becomes:

$$0.8V_{DD} = V_{DD} \left[ 1 - \exp\left(-\frac{T_2}{R_{PEQ}C}\right) \right]$$

or

$$T_2 = -R_{PEQ}C \ln(0.2) \quad (3.23)$$

Then, using Eqs. 3.22 and 3.23 we can calculate the rise time,  $T_R$ :

$$T_R = T_2 - T_1 \approx 1.4 R_{PEQ}C \quad (3.24)$$

Similar to rise time, the fall time is defined as the time interval during which the output falls from 80% to 20% of its final value as shown in Fig. 3.17. Rewriting Eq. 3.11, and replacing  $R_N$  in this equation by  $R_{NEQ}$  to signify the total equivalent NMOS resistance in the worst-case discharge path yields:

$$V_{OUT}(t) = V_{DD} \exp\left(-\frac{t}{R_{NEQ}C}\right) \quad (3.25)$$

Thus, 80% of  $V_{OUT}$  at  $t = T_1$  becomes:

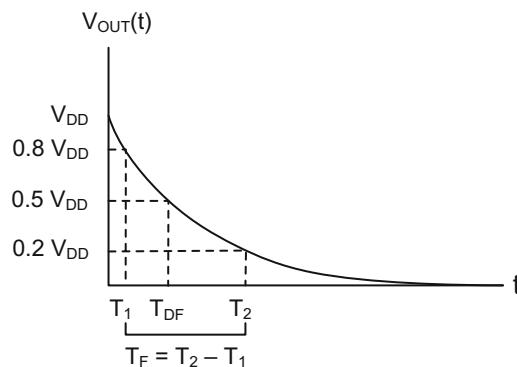
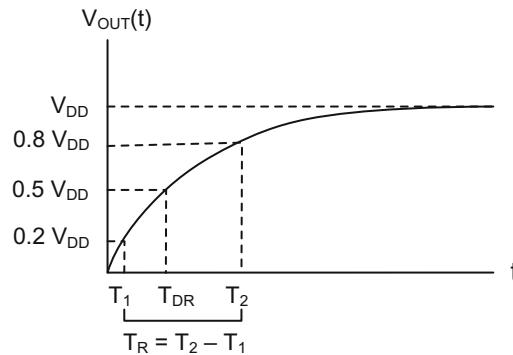
$$0.8V_{DD} = V_{DD} \exp\left(-\frac{T_1}{R_{NEQ}C}\right) \quad (3.26)$$

Similarly, 20% of  $V_{OUT}$  at  $t = T_2$  becomes:

$$0.2V_{DD} = V_{DD} \exp\left(-\frac{T_2}{R_{NEQ}C}\right) \quad (3.27)$$

Extracting  $T_1$  from Eq. 3.26 and  $T_2$  from Eq. 3.27, and computing the fall time,  $T_F$ , yields:

$$T_F = T_2 - T_1 \approx 1.4 R_{NEQ}C \quad (3.28)$$



**Fig. 3.17** Rise and fall times, rise and fall delays of a CMOS gate

### 3.8 Rise and Fall Delays

Rise delay,  $T_{DR}$ , is defined as the time interval between 50% mark of the input voltage and 50% mark of the rising output as shown in the top part of Fig. 3.17.

If the input voltage is assumed to make an abrupt transition from  $V_{DD}$  to 0 V at  $t = 0$ , then the 50% mark of the input voltage still resides at  $t = 0$ . Thus,

$$T_1 = 0 \quad (3.29)$$

For  $T_2$ , the output rises 50% of its final value. Therefore, we must refer to Eq. 3.12.

$$0.5V_{DD} = V_{DD} \left[ 1 - \exp\left(-\frac{T_2}{R_{PEQ}C}\right) \right]$$

or

$$T_2 = -R_{PEQ}C \ln(0.5) \quad (3.30)$$

Thus, the rise delay,  $T_{DR}$ , becomes:

$$T_{DR} = T_2 - T_1 \approx 0.7 R_{PEQ}C \quad (3.31)$$

Similar to  $T_{DR}$ , the fall delay,  $T_{DF}$ , is computed by using Eq. 3.11. Graphically,  $T_{DF}$  is also shown in the bottom part of Fig. 3.17.

Again,  $T_1 = 0$  because the input abruptly transitions at  $t = 0$ . For  $T_2$ , however, one can write:

$$0.5V_{DD} = V_{DD} \exp\left(-\frac{T_2}{R_{NEQ}C}\right)$$

Thus,

$$T_{DF} = T_2 - T_1 \approx 0.7 R_{NEQ}C \quad (3.32)$$

**Example 3.1** Assume that rise and fall times, and rise and fall delays of the circuit in Fig. 3.15 need to be computed.

From Eq. 3.24,

$$T_R \approx 1.4 R_{PEQ}C = 1.4 \times 3 R_p C = 4.2 R_p C \quad (3.33)$$

From Eq. 3.28,

$$T_F \approx 1.4 R_{NEQ}C = 1.4 \times 2 R_N C = 2.8 R_N C \quad (3.34)$$

From Eq. 3.31,

$$T_{DR} \approx 0.7 R_{PEQ}C = 0.7 \times 3 R_p C = 2.1 R_p C \quad (3.35)$$

From Eq. 3.32,

$$T_{DF} \approx 0.7 R_{NEQ}C = 0.7 \times 2 R_N C = 1.4 R_N C \quad (3.36)$$

The values of  $R_N$  and  $R_P$  can be calculated using the technology-dependent NMOS and PMOS transistor parameters, respectively.  $C$  is the value of the output capacitance.

**Example 3.2** Again assume the complex gate in Fig. 3.15. Size each transistor width in this logic gate in terms of a minimum transistor width,  $W$ , to ensure  $T_R = T_F$ .

Since  $T_R = T_F$ , then  $4.2 R_P C = 2.8 R_N C$

Thus,

$$1.5 R_P = R_N \quad (3.37)$$

Substituting the values for  $R_N$  and  $R_P$  from Eq. 3.4 and Eq. 3.9 yields:

$$1.5 \frac{K_P}{W_P} = \frac{K_N}{W_N} \quad (3.38)$$

But,

$$K_P = \frac{L_P}{\mu_P C_{OX}(V_{DD} - V_{TP})}$$

and

$$K_N = \frac{L_N}{\mu_N C_{OX}(V_{DD} - V_{TN})}$$

Assume that  $\mu_N = 3 \mu_P$  (due to the device technology),  $L = L_P = L_N$  and  $V_{TN} = V_{TP} \approx 0.2 V_{DD}$

Then,

$$K_P = \frac{L}{\mu_P C_{OX}(0.8V_{DD})} \quad (3.39)$$

and

$$K_N = \frac{L}{3\mu_P C_{OX}(0.8V_{DD})} \quad (3.40)$$

Substituting Eqs. 3.39 and 3.40 into Eq. 3.38 yields:

$$1.5 \frac{L}{\mu_P C_{OX}(0.8V_{DD}) W_P} = \frac{L}{3\mu_P C_{OX}(0.8V_{DD}) W_N}$$

$$\frac{1.5}{W_P} = \frac{1}{3W_N}$$

or

$$W_P = 4.5W_N$$

Assuming  $W_N = W$  (minimum geometry) yields:

$$W_P = 4.5W$$

Therefore, all NMOS transistors are sized  $W$ , and all PMOS transistors are sized  $4.5W$  in order to ensure that the rise and fall times, computed through the worst-case critical paths, are equal to each other.

**Example 3.3** Suppose the transistor widths of a two-input AND gate is given by the top figure in Fig. 3.18. The hole and electron mobilities are measured to be  $\mu_p = 100 \text{ cm}^2/\text{Vsec}$  and  $\mu_n = 300 \text{ cm}^2/\text{Vsec}$ , respectively. All transistor lengths are  $L = 0.1 \mu\text{m}$ ,  $V_{DD} = 3 \text{ V}$ ,  $V_{TN} = V_{TP} = 0.6 \text{ V}$  and  $C_{OX} = 10^{-8} \text{ F/cm}^2$ .

The fall gate delay,  $T_{DF}$ , and the rise gate delay,  $T_{DR}$ , are computed as  $(T_{DR1} + T_{DF2})$  and  $(T_{DF1} + T_{DR2})$  respectively, and they are shown by the center and the bottom figures in Fig. 3.18. The worst-case gate delay is simply the larger of these two values.

First, let us compute  $(T_{DR1} + T_{DF2})$ . Since  $C_L$  is given, the computation always starts from the last stage. We know that:

$$R_N = \frac{K_N}{W_N} \text{ where } K_N = \frac{L}{\mu_n C_{OX}(V_{DD} - V_{TN})}$$

$$R_P = \frac{K_p}{W_p} \text{ where } K_p = \frac{L}{\mu_p C_{OX}(V_{DD} - V_{TP})}$$

Substituting  $W_N = 10 \mu\text{m}$  and  $W_P = 30 \mu\text{m}$  from Fig. 3.18 and the physical device parameters given above yields  $R_N = R_P \approx 1.4 \text{ K}\Omega$ .

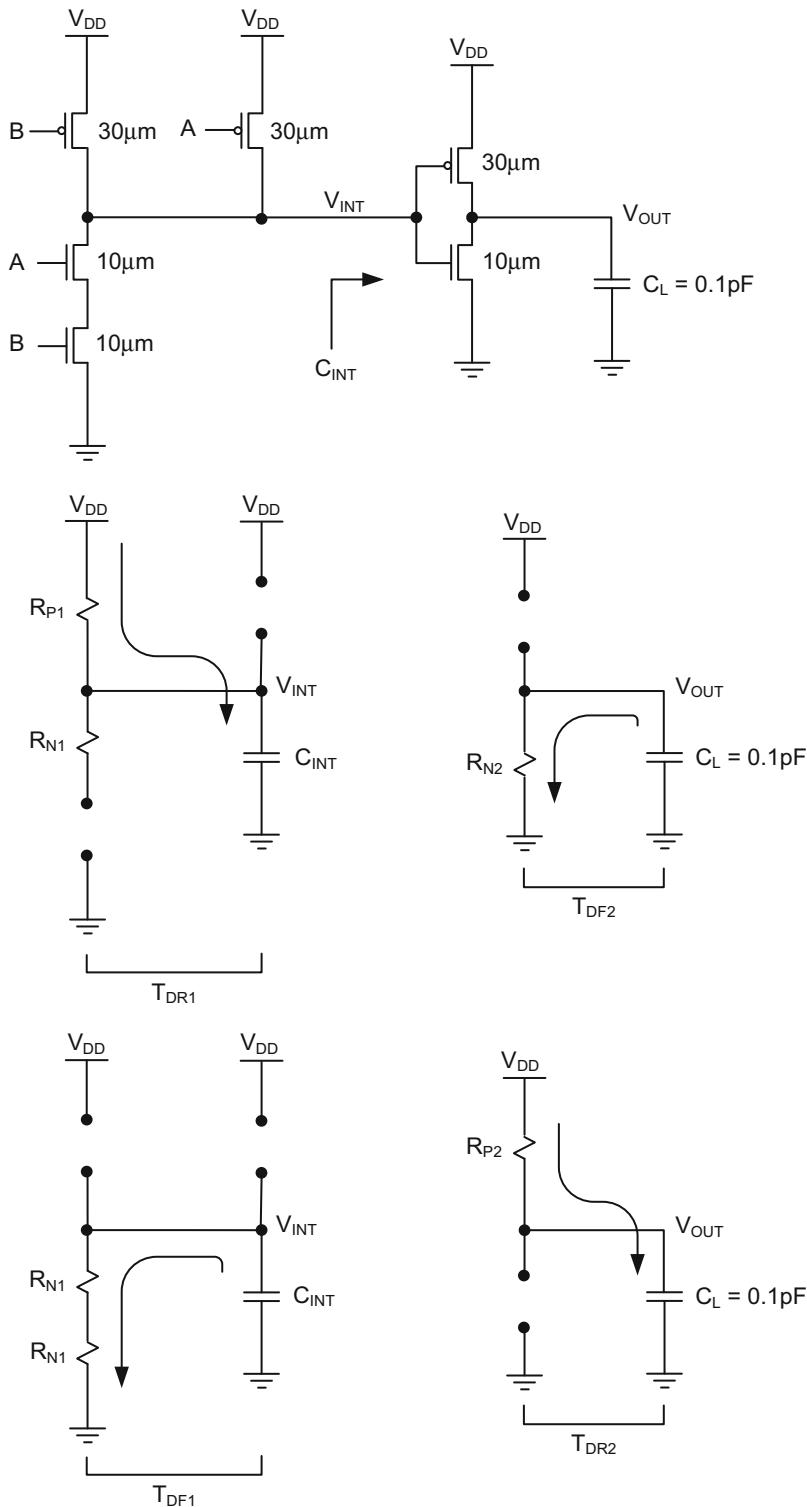
Then,

$$T_{DF2} = 0.7 R_{N2} C_L = 0.7(1400) 0.1 \times 10^{-12} = 98 \text{ ps}$$

and

$$T_{DR1} = 0.7 R_{P1} C_{INT}$$

Here,  $C_{INT}$  is the combination of NMOS and PMOS input gate capacitances of the inverter in Fig. 3.18.



**Fig. 3.18** Computation of worst-case gate delay of 2-input AND gate

Thus,

$$C_{INT} = C_{OX} L (W_N + W_P) = 10^{-8} \times 0.1 \times 10^{-4} \times (10 + 30) \times 10^{-4} = 0.4 \text{fF}$$

$$T_{DR1} = 0.7(1400)4 \times 10^{-15} = 3.9 \text{ps}$$

$$\text{Consequently, } T_{DR1} + T_{DF2} = 0.4 + 98 \approx 98 \text{ ps}$$

The computation of  $(T_{DF1} + T_{DR2})$  starts from calculating  $T_{DR2}$  first.

$$T_{DR2} = 0.7 R_{P2} C_L = 0.7(1400) 0.1 \times 10^{-12} = 98 \text{ ps}$$

$$T_{DF1} = 0.7 2R_{N1} C_{INT} = 0.7(2 \times 1400) 0.4 \times 10^{-15} = 0.8 \text{ ps}$$

$$\text{Consequently, } T_{DF1} + T_{DR2} = 98 + 0.8 \approx 99 \text{ ps}$$

Therefore, the worst-case gate delay for this AND gate is  $T_D = 99 \text{ ps}$ .

## Review Questions

**1.** Implement the following functions using CMOS circuits

- (a) Implement the following function with a two-stage CMOS circuit (with an output inverter).

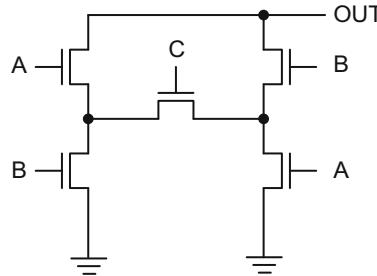
$$\text{OUT} = (A + B + C) \cdot \overline{D}$$

- (b) Implement the following function with a single stage CMOS circuit (without an output inverter).

$$\text{OUT} = (A + B + C) \cdot \overline{D}$$

**2.** The NMOS tree of a CMOS circuit is given below.

- (a) Find the logic function of this circuit at the out terminal.  
 (b) Determine the PMOS tree for this circuit. Explain the configuration steps clearly.



**3.** Design the CMOS gate that implements the following function:

$$\text{OUT} = \overline{A \oplus B}$$

- (a) Draw the circuit schematic without any output inverter.  
 (b) Size all the transistors in this circuit such that  $T_R = 3T_F$ . The minimum geometry is W.

**4.** The following function is given:

$$\text{OUT} = A + B \cdot \overline{(C + D + E)}$$

Design a single-stage CMOS circuit with  $T_{DR} = 2T_{DF}$ . Determine all transistor widths in terms of minimum geometry, W.

5. The following function is given:

$$OUT = (A + B + C)$$

Design a two-stage CMOS circuit that maintains  $T_R = 2T_F = 500$  ps at all nodes (internal or output) for a load capacitor of 100fF. Find all transistor widths in  $\mu\text{m}$ .

Use the technology below:

$$C_{OX} = 10^{-8} \text{ F/cm}^2$$

$$L = 0.25 \mu\text{m}$$

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = V_{TP} = 0.5 \text{ V}$$

$$\mu_N = 200 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 100 \text{ cm}^2/\text{Vsec}$$

6. Implement one-bit SUM and  $C_{OUT}$  (carry-out) functions in an integrated CMOS logic gate with uninverted inputs, A, B, and  $C_{IN}$  (carry-in):

$$SUM = A \oplus B \oplus C_{IN}$$

$$C_{OUT} = A \cdot B + C_{IN} \cdot (A + B)$$

The design criterion requires that  $T_{DR}$  and  $T_{DF}$  are equal to 400 ps in each stage of this CMOS circuit. Size the transistors in the CMOS circuit using the following technology:

$$C_{LOAD} = 100\text{fF} = 100 \times 10^{-15}\text{F} \text{ (placed both at the sum and carry-out nodes)}$$

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = V_{TP} = 0.5 \text{ V}$$

$$\mu_N = 300 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 100 \text{ cm}^2/\text{Vsec}$$

$$L = 0.25 \mu\text{m}$$

$$C_{OX} = 10^{-8} \text{ F/cm}^2$$

7. Design the following circuit that drives  $C_L = 500\text{fF}$ . In order to be able to drive such a large capacitance, the 2-1 MUX output must have an inverter stage. Start the design from the last stage. Inverted and uninverted inputs may be used during the design in order to minimize the number of transistors. Size the transistors such that the rise and fall times at all output nodes are 400 ps. Use the device technology specs below:

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = 0.5 \text{ V}$$

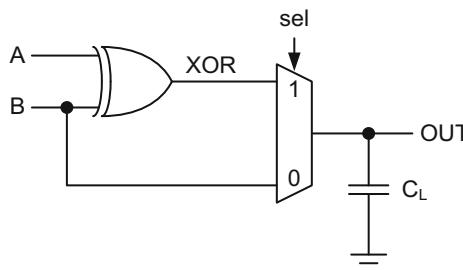
$$V_{TP} = 0.5 \text{ V}$$

$$C_{OX} = 10^{-8} \text{ F/cm}^2$$

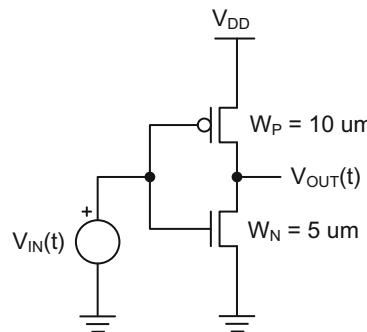
$$L = 0.25 \mu\text{m}$$

$$\mu_N = 600 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 200 \text{ cm}^2/\text{Vsec}$$



8. An inverter is given below.



The NMOS and PMOS transistors are fabricated using the following technology:

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = V_{TP} = 0.6 \text{ V}$$

$$\mu_N = 1200 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 600 \text{ cm}^2/\text{Vsec}$$

$$C_{ox} = 6.9 \times 10^{-7} \text{ F/cm}^2$$

$$L = 0.25 \mu\text{m}$$

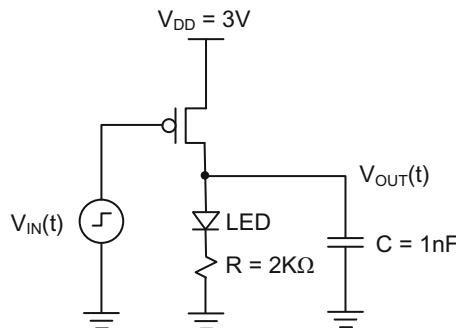
The current through a MOSFET is defined as:

$$I_D = \frac{\mu C_{ox} W}{L} \left[ (V_{GS} - V_T)V_{DS} - \frac{V_{DS}^2}{2} \right]$$

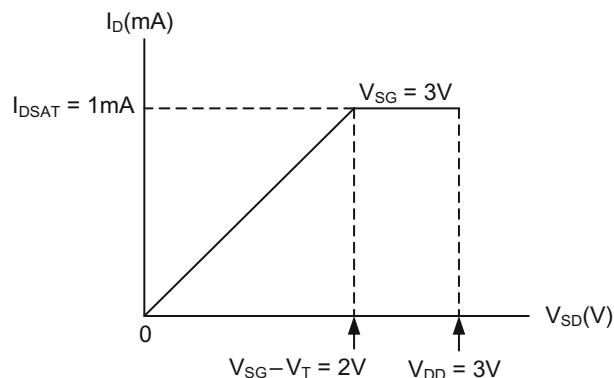
- (a) Compute the static DC current when  $V_{IN} = 2 \text{ V}$ .
- (b) Compute the output voltage when  $V_{IN} = 2 \text{ V}$ .

Consider the equivalent circuits of NMOS and PMOS transistors at this bias condition.

9. The circuit below operates a Light Emitting Diode (LED):



The approximate I-V characteristics of the transistor are given below:

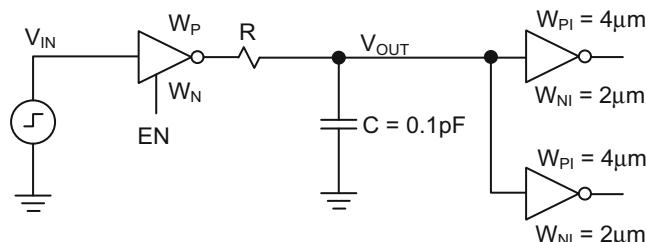


The LED needs to have 0.5 mA current and 1 V across its terminals to turn on. Otherwise, it does not conduct current. The input voltage,  $V_{IN}(t)$ , is initially at 3 V, but switches to 0 V at  $t = 0$ . The initial condition at  $V_{OUT}$  is 0 V.

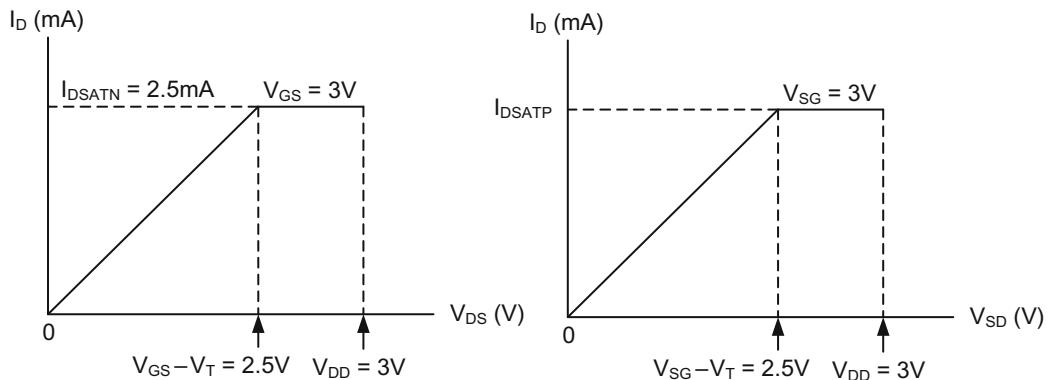
Using the I-V characteristics of the transistor and the component values in the circuit schematic above, derive the expression for  $V_{OUT}(t)$ . Plot  $V_{OUT}(t)$  as function of time.

10. The circuit below illustrates a tri-state inverter driving a long wire with an equivalent resistance of  $R$  and capacitance of  $C$ .

At the end of the wire, two inverters, composed of a PMOS transistor with  $W_{PI} = 4 \mu\text{m}$  and NMOS transistor with  $W_{NI} = 2 \mu\text{m}$ , create a capacitive load.



NMOS and PMOS transistor I-V characteristics are shown below:



- If rise and fall times at  $V_{OUT}$  are 2 ns, find the actual wire resistance in ohms.
- What is the saturation current of the PMOS transistor in the tri-state inverter?

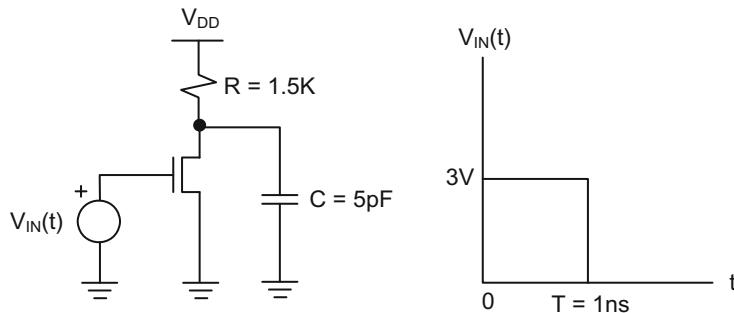
Use the following technology:

$$V_{DD} = 3 \text{ V}$$

$$C_{OX} = 10^{-7} \text{ F/cm}^2$$

$$L = 0.25 \mu\text{m}$$

- The following circuit is given:



Assuming that NMOS transistor is in the linear region when it is turned on, find the output voltage of the circuit and plot it using the following device parameters:

$$\mu_N = 100 \text{ cm}^2/\text{Vsec}$$

$$C_{OX} = 6.9 \times 10^{-7} \text{ F/cm}^2$$

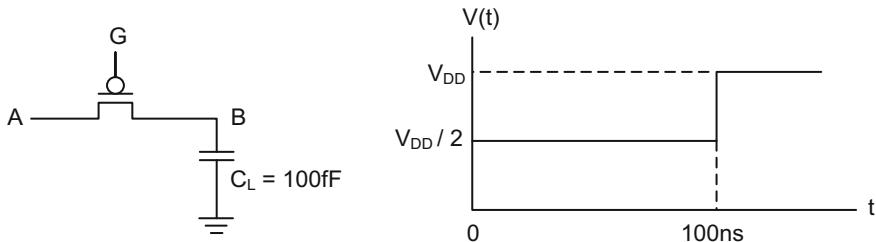
$$L = 0.25 \mu\text{m}$$

$$W = 10 \mu\text{m}$$

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = 0.6 \text{ V}$$

12. The waveform,  $V(t)$ , is applied to G terminal of the circuit below.



A constant voltage,  $V_{DD}$ , is applied to A.

PMOS  $I_D$ - $V_{SD}$  characteristics are given below:

$$I_D = \frac{\mu_P C_{OX} W}{L} (V_{SG} - V_T) V_{SD} \text{ if } V_{SD} < (V_{SG} - V_T)$$

$$I_D = \frac{\mu_P C_{OX} W}{2L} (V_{SG} - V_T)^2 \text{ if } V_{SD} > (V_{SG} - V_T)$$

If  $V_{DD} = 4$  V,  $V_T = 1$  V,  $W = 5$   $\mu\text{m}$ ,  $L = 0.25$   $\mu\text{m}$  and  $\mu_P C_{OX} = 10^{-6}$  A/V<sup>2</sup>, plot the waveform at B.

13. A Set-Reset (SR) latch is given below. This latch should be designed to drive a 100fF capacitive load. The rise and fall times at each node in the circuit are 500 ps. The device technology specs are as follows:

$$V_{DD} = 3 \text{ V}$$

$$V_{TN} = 0.5 \text{ V}$$

$$V_{TP} = 0.5 \text{ V}$$

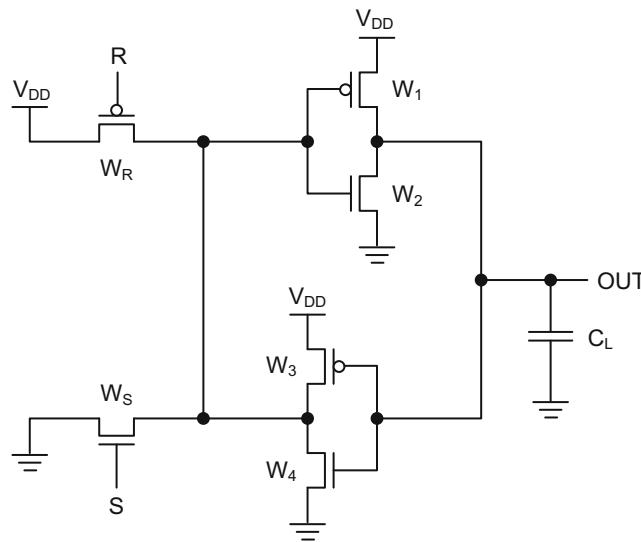
$$C_{OX} = 10^{-8} \text{ F/cm}^2$$

$$L = 0.25 \mu\text{m}$$

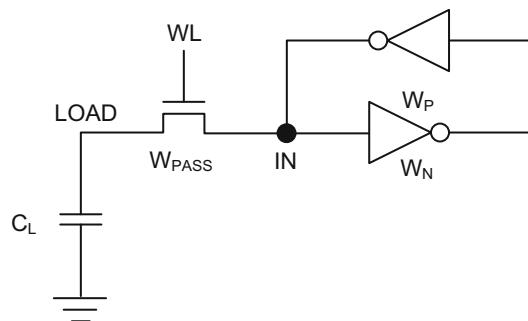
$$\mu_N = 600 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 200 \text{ cm}^2/\text{Vsec}$$

Size all the transistors in the circuit. To prevent contention between any two transistors, make sure to adjust the resistance of the transistor being sized to be about 20% of the resistance of the contending transistor.



14. A single ended SRAM cell is given below:



The device technology parameters for the NMOS and PMOS transistors are given below.

$$C_{OX} = 7 \times 10^{-8} \text{ F/cm}^2$$

$$V_{DD} = 3 \text{ V}$$

$$L = 0.25 \mu\text{m}$$

$$\mu_N = 200 \text{ cm}^2/\text{Vsec}$$

$$\mu_P = 100 \text{ cm}^2/\text{Vsec}$$

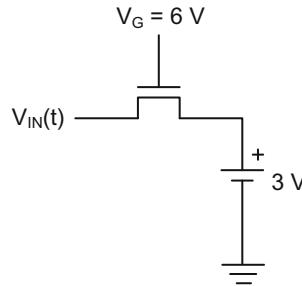
$$V_{TN} = V_{TP} = 0.6 \text{ V}$$

When  $WL = 0 \text{ V}$  and the pass-gate NMOS transistor,  $W_{PASS}$ , is off, the voltage at  $LOAD$  node is  $3 \text{ V}$ , and the voltage at  $IN$  node is  $0 \text{ V}$  (cell stores logic 0). Transistor sizes in both inverters are  $W_P = 2 \mu\text{m}$  and  $W_N = 1 \mu\text{m}$  in the SRAM cell.

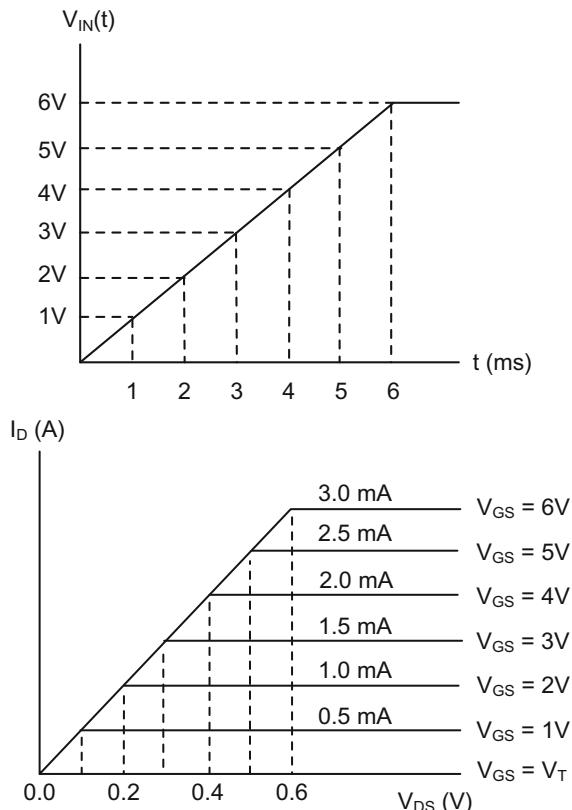
- (a) Assuming that the pass-gate NMOS transistor acts as a switch with a channel resistance of  $0 \Omega$  when it is on, calculate the maximum allowed load capacitance value  $C_L$  such that the stored bit in the SRAM cell does not accidentally change from logic 0 to 1.

- (b) What is the relationship(s) between transistor sizes to keep the logic state the same in the cell when the pass-gate NMOS transistor has a finite channel resistance when it is turned on? Assume identical rise and fall times at the internal nodes.

**15.** An NMOS transistor establishes the pass-gate functionality in the circuit below:

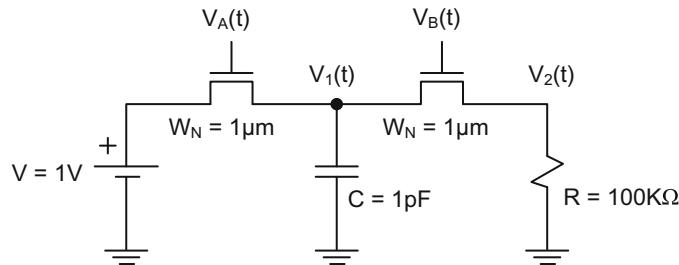


The input signal,  $V_{IN}(t)$ , and  $I_D$ - $V_{DS}$  for this NMOS transistor are shown below.

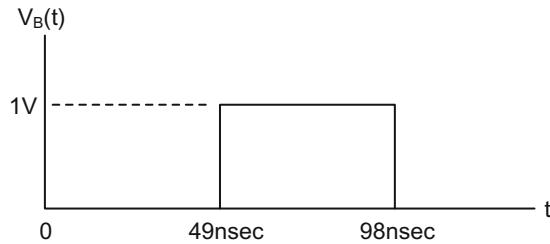
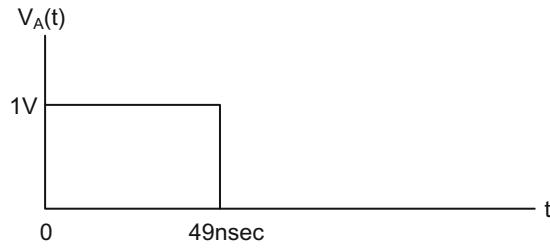


Calculate  $I_D$  from the data given above and plot the absolute value of  $I_D$  as a function of time.

16. The circuit given below includes two pass-gate NMOS transistors connected in series.



Here,  $V_A(t)$  and  $V_B(t)$  are the voltages applied at the gates of the NMOS transistors as shown below:



Compute the nodal voltages,  $V_1(t)$  and  $V_2(t)$ , as a function of time using the device technology below:

$$V_{DD} = 1 \text{ V}$$

$$V_{TN} = 0.3 \text{ V}$$

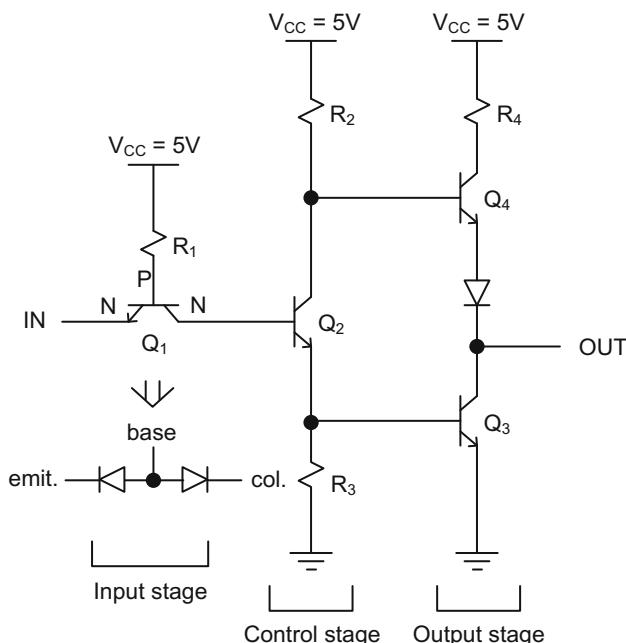
$$L = 0.1 \mu\text{m}$$

$$\mu_N = 200 \text{ cm}^2/\text{Vsec}$$

$$C_{OX} = 7 \times 10^{-8} \text{ F/cm}^2$$

## 4.1 TTL Inverter

Transistor-Transistor-Logic (TTL) is still used in limited capacity in discrete ICs due to its ability to deliver (or sink) large currents. However, power consumption, slow speed and the cost of fabrication are its main drawbacks. All TTL logic gates are composed of three stages. The simplest TTL logic gate, TTL inverter, is shown in Fig. 4.1.



**Fig. 4.1** A simplified TTL inverter

The input stage of this gate contains only a single transistor, Q<sub>1</sub>. This transistor is equivalent to two back-to-back diodes as shown in the inset of Fig. 4.1, and this

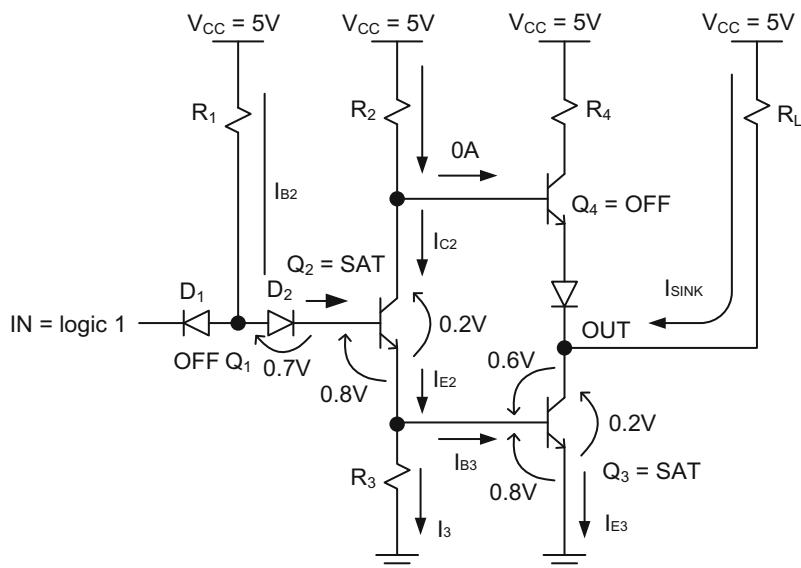
configuration directs the current flow either towards the IN terminal or towards the control stage to activate  $Q_2$  depending on the logic value at the input.

The control stage contains  $Q_2$ . Its function is to supply part of its emitter current to saturate  $Q_3$  or turn off this transistor.

The output stage, also called totem-pole configuration, consists of  $Q_3$  and  $Q_4$ .  $Q_3$  absorbs all the current (sink current) from an output load.  $Q_4$  supplies current (source current) to an output load. Supply voltage for any commercially available TTL stage is usually 5 V.

### TTL Inverter When IN = 5 V

When the input voltage at the IN node is raised to 5 V, there will be no voltage drop across the diode,  $D_1$  (corresponding to the emitter-base junction of  $Q_1$ ). As a result,  $D_1$  turns off as shown in Fig. 4.2.



**Fig. 4.2** Currents and voltages of the TTL inverter when IN = logic 1

Therefore, the current from the voltage supply,  $V_{CC}$ , directly flows into the base of  $Q_2$  through  $D_2$  (corresponding to the collector-base junction of  $Q_1$ ), forming the base current for  $Q_2$ ,  $I_{B2}$ , and driving this transistor into saturation.

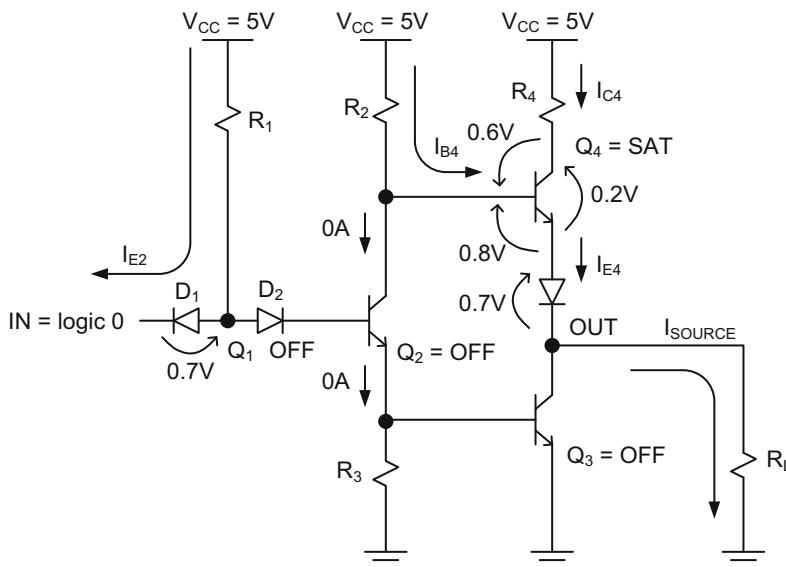
When  $Q_2$  is in the saturation region, its collector current,  $I_{C2}$ , combines with  $I_{B2}$ , and forms the emitter current for  $Q_2$ ,  $I_{E2} = I_{B2} + I_{C2}$ . The majority of  $I_{E2}$  flows into the base of  $Q_3$  and drives this transistor into saturation. The remaining part of  $I_{E2}$  flows through  $R_3$ , and biases the base-emitter junction of  $Q_3$  with 0.8 V.

As a result of this scenario, the voltage drop between the base of  $Q_4$  and the output terminal becomes  $0.2 + 0.6 = 0.8$  V. However, this potential difference is not sufficient to turn on  $Q_4$  and the diode at the output stage simultaneously since they require at least  $0.8 + 0.7 = 1.5$  V. Thus,  $Q_4$  turns off.

As a result,  $I_{C3}$  becomes the “sink” current and absorbs all the output current from an external resistor,  $R_L$ , as shown in Fig. 4.2. This current is also called  $I_{OL}$ , corresponding to the low output current when the output is equal to  $V_{OL}$ .

### TTL Inverter When IN = 0 V

When input voltage at IN is lowered to 0 V, current flows through the diode,  $D_1$  (corresponding to the emitter-base junction of  $Q_1$ ), and turns on the diode. The voltage developed across this diode, 0.7 V, then becomes equal to the sum of the voltage drops across  $D_2$  (corresponding base-collector junction of  $Q_1$ ), the base-emitter junction of  $Q_2$  and the base-emitter junction of  $Q_3$ . However, 0.7 V is not enough to saturate the transistors,  $Q_2$  and  $Q_3$ , and turn on the diode at the same time as shown in Fig. 4.3.



**Fig. 4.3** Currents and voltages of the TTL inverter when  $IN = \text{logic 0}$

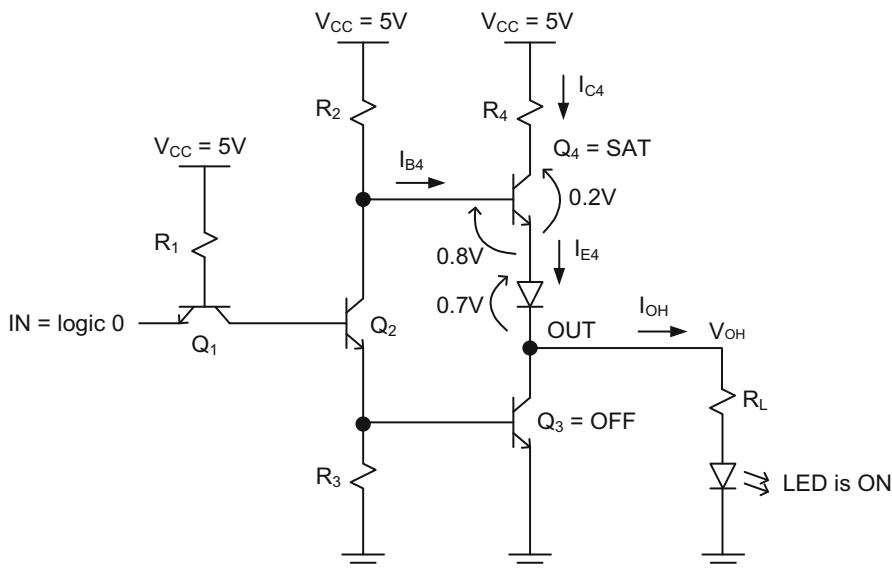
Since  $Q_2$  stays off, the supply current,  $I_{B4}$ , flows into the base of  $Q_4$  and saturates this transistor. The collector current of  $Q_4$ ,  $I_{C4}$ , combines with  $I_{B4}$  to form the emitter current for  $Q_4$ ,  $I_{E4}$ . This current becomes the “source” current and flows into an external load resistor,  $R_L$ , as shown in Fig. 4.3. This current is also called  $I_{OH}$ , corresponding to the high output voltage,  $V_{OH}$ . However,  $V_{OH}$  never reaches the full supply voltage of 5 V, but stays around 3.5 V due to the combined voltage drop between  $V_{CC}$  and the OUT terminal, namely  $R_4 I_{C4} + 0.2 + 0.7 \approx 1.5$  V.

**Example 4.1** There are two ways for a TTL inverter to drive an LED. The circuit configuration depends on the current requirement of the LED, and the relative values of  $I_{OH}$  and

$I_{OL}$  given in the manufacturer's datasheet. If the LED requires high current, and  $I_{OH}$  happens to be much higher than  $I_{OL}$  in the datasheet, then the best way to connect the LED is between the output terminal and the ground as shown in Fig. 4.4. As a result,  $I_{OH}$  is used to turn on the LED.

The series resistance,  $R_L$ , is computed as follows:

$$R_L = \frac{(V_{OH} - V_{LED})}{I_{OH}} \quad (4.1)$$



**Fig. 4.4** TTL inverter driving LED if  $I_{OH}$  is much greater than  $I_{OL}$

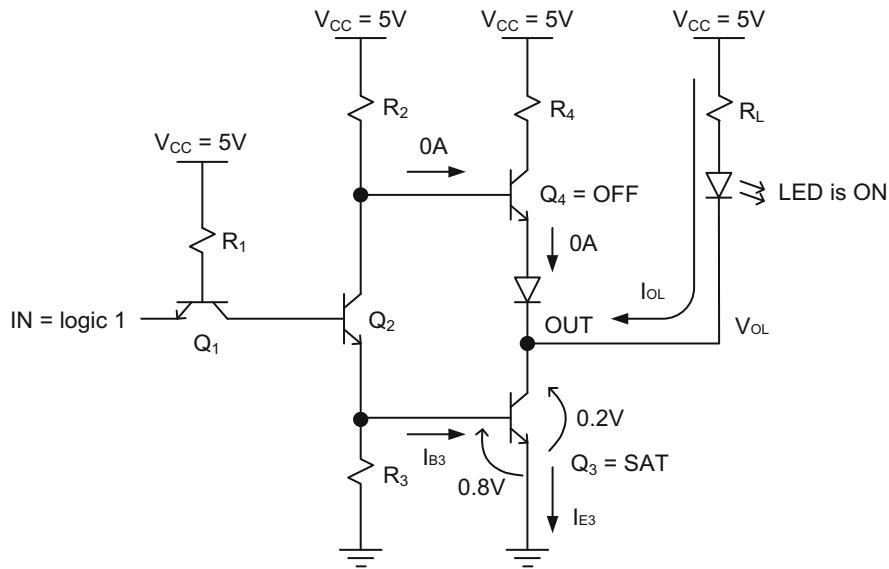
In this equation,  $V_{LED}$  is the voltage drop across the LED.

If the LED requires high current and the TTL manufacturer's datasheet specifies that  $I_{OL}$  is much higher than  $I_{OH}$ , then the better circuit configuration would be to connect the LED between the supply voltage and the OUT terminal as shown in Fig. 4.5. That way,  $I_{OL}$  of the TTL inverter supplies all the current the LED needs.

This time,  $R_L$  becomes:

$$R_L = \frac{(V_{CC} - V_{LED} - V_{OL})}{I_{OL}} \quad (4.2)$$

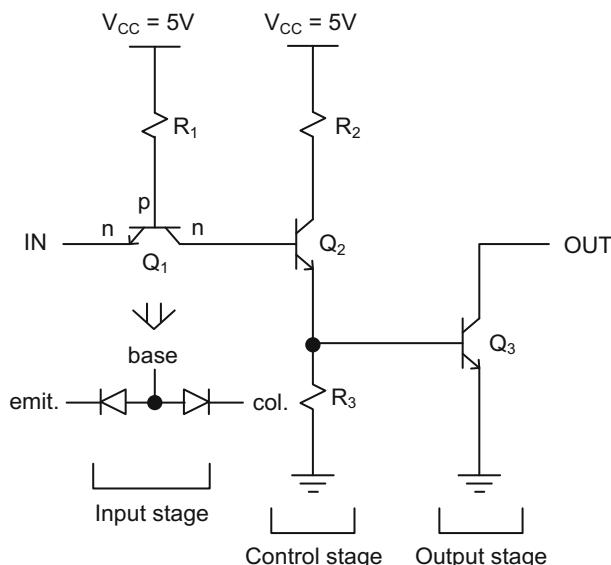
If the LED's current requirement is low and comparable to  $I_{OH}$  or  $I_{OL}$ , then either circuit configuration in Figs. 4.4 or 4.5 can be used to turn on the LED.



**Fig. 4.5** TTL inverter driving LED if  $I_{OL}$  is much greater than  $I_{OH}$

## 4.2 TTL Inverter with Open Collector

If the TTL inverter's  $I_{OL}$  is much higher than  $I_{OH}$ , the output transistor  $Q_4$  may not be necessary at all to operate an LED or some other two-terminal device. This reduces the overall device count in the inverter and forms a much simpler circuit called open-collector inverter as shown in Fig. 4.6. With open-collector configuration, the high logic level also reaches to the full voltage supply value of 5 V instead of staying at 3.5 V.

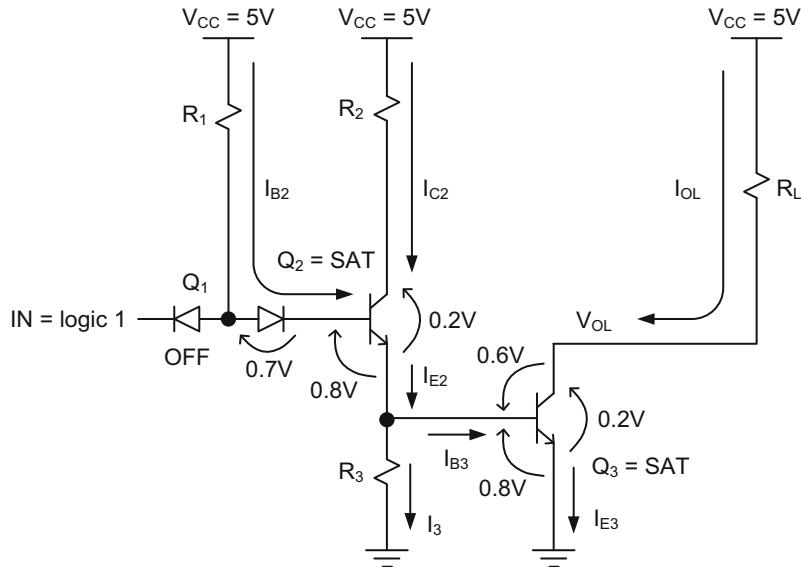


**Fig. 4.6** Open-collector TTL inverter

The operation of open-collector inverter is similar to the operation of a common TTL inverter. However, the open collector of  $Q_3$  will always need a pull-up resistor to operate properly.

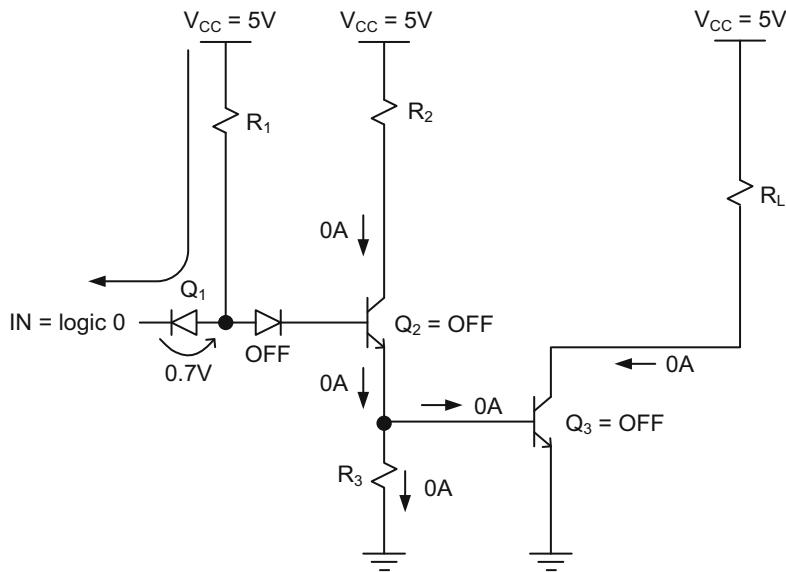
Figure 4.7 describes the operation of this logic gate with a pull-up resistor when IN is connected to 5 V. The diode corresponding to the emitter-base junction of  $Q_1$  turns off because there is 0 V across this junction.  $Q_2$  goes into the saturation region with a current flowing in its base,  $I_{B2}$ . A large portion of  $Q_2$ 's emitter saturation current,  $I_{E2}$ , also flows into the base of  $Q_3$  and causes this transistor to go into saturation provided that  $Q_3$  is biased properly with an external load resistance,  $R_L$ . The output voltage becomes  $V_{OL}$ , which is in the order of 0.2 V.  $I_{OL}$  becomes the collector current of  $Q_3$ .  $R_L$  is determined in terms of  $V_{OL}$  and  $I_{OL}$  according to the following equation:

$$R_L = \frac{(V_{CC} - V_{OL})}{I_{OL}} \quad (4.3)$$



**Fig. 4.7** Open-collector TTL inverter driving a CMOS gate when IN = logic 1

When IN = 0 V, the diode corresponding to the emitter-base junction of  $Q_1$  turns on as shown in Fig. 4.8. However, the base of  $Q_1$ , now standing at 0.7 V, will not be sufficient to turn on the transistors,  $Q_2$ ,  $Q_3$  and the diode corresponding to the collector-base of  $Q_1$ . The output node is pulled up to  $V_{CC}$ .  $I_{OH}$  through  $R_L$  becomes 0 A.



**Fig. 4.8** Open-collector TTL inverter driving a CMOS gate when IN = logic 0

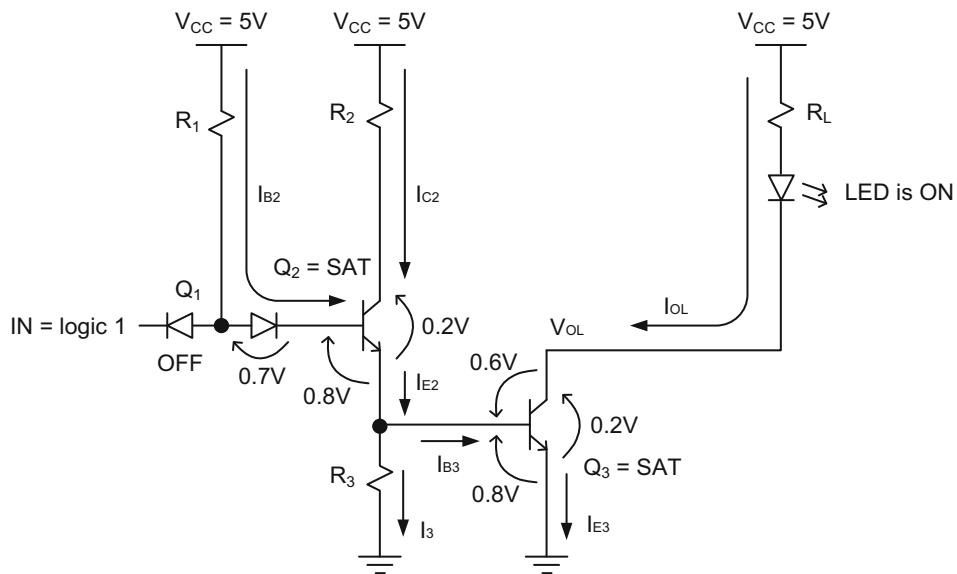
**Example 4.2** Assume the output of the open-collector inverter in Fig. 4.7 includes an LED in series with  $R_L$ . The load resistor now serves two purposes: it limits the current through LED and biases  $Q_3$  when this transistor needs to be driven into saturation. Applying  $IN = 5\text{ V}$  in Fig. 4.9 causes both  $Q_2$  and  $Q_3$  to be in saturation region which produces  $OUT = V_{OL}$ . The resulting  $I_{OL}$  turns on the LED.

$$I_{OL} = \frac{(V_{CC} - V_{LED} - V_{OL})}{R_L} \quad (4.4)$$

Then,  $R_L$  is calculated as

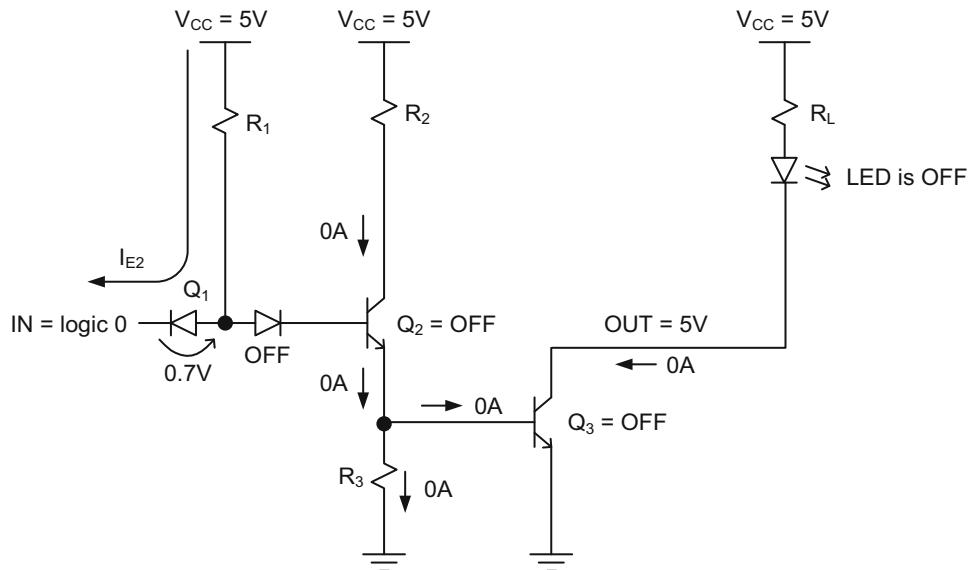
$$R_L = \frac{(V_{CC} - V_{LED} - V_{OL})}{I_{LED}} \quad (4.5)$$

where  $I_{LED} = I_{OL}$ .



**Fig. 4.9** Open-collector TTL inverter driving an LED when  $IN = \text{logic 1}$

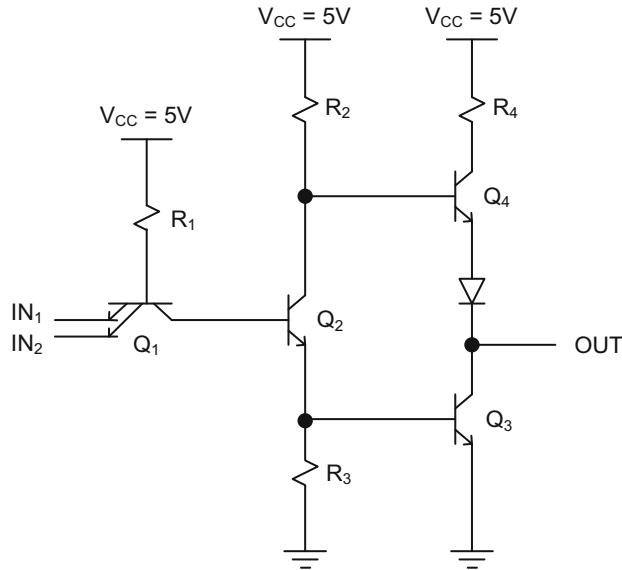
When  $IN = 0\text{ V}$  is applied to the open-collector TTL inverter input, both  $Q_2$  and  $Q_3$  turn off, resulting no current flow through the LED as shown in Fig. 4.10. The LED turns off.



**Fig. 4.10** Open-collector TTL inverter driving an LED when  $IN = \text{logic 0}$

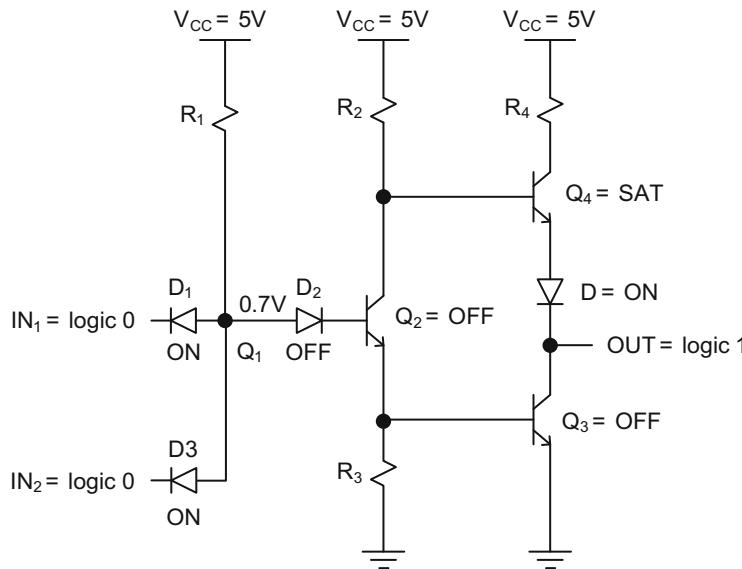
### 4.3 Two-Input TTL Nand Logic Gate

Two-input TTL NAND gate is shown in Fig. 4.11. In this figure,  $Q_1$  produces two emitter ports each of which is connected to separate external inputs,  $IN_1$  and  $IN_2$ .



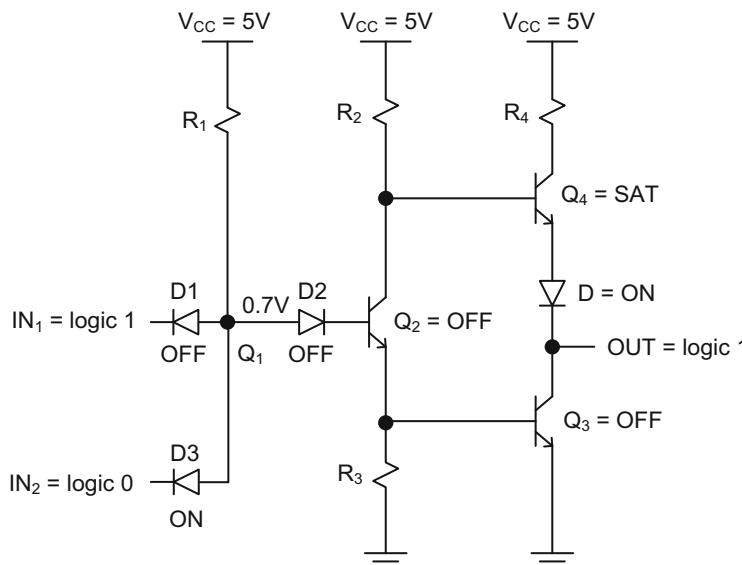
**Fig. 4.11** Two-input TTL NAND gate

Figure 4.12 shows both inputs of the two-input TTL NAND gate connected to 0 V. In this figure,  $Q_1$  has been transformed into three discrete diodes,  $D_1$ ,  $D_2$  and  $D_3$ , to make the circuit analysis simpler. When  $IN_1 = IN_2 = 0$  V, both  $D_1$  and  $D_3$  turn on. However, the voltage drop across either diode will not be sufficient to turn on  $D_2$ ,  $Q_2$  and  $Q_3$  as discussed earlier.  $Q_4$  is driven into saturation and supplies  $I_{OH}$  to an output load. The output of the NAND gate transitions to logic 1.



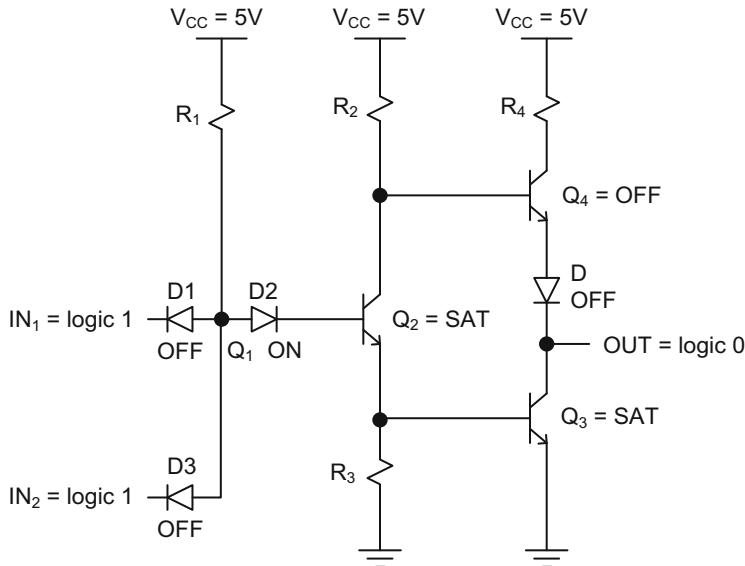
**Fig. 4.12** Two-input TTL NAND gate when  $IN_1 = IN_2 = \text{logic 0}$

When one of the inputs of the NAND gate is connected to ground while the other tied to logic 1, only  $D_3$  turns on. Both  $Q_2$  and  $Q_3$  turn off as shown in Fig. 4.13. As a result,  $Q_4$  is driven into saturation and the output transitions to logic 1.



**Fig. 4.13** Two-input TTL NAND gate when one of the inputs is at logic 0 ( $IN_2 = \text{logic 0}$ )

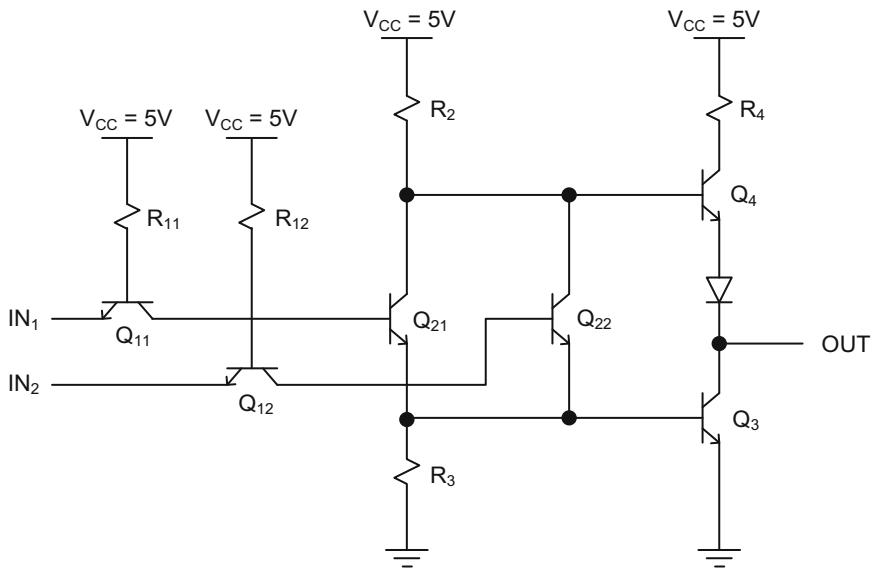
When both inputs of the NAND gate are tied to logic 1, both  $D_1$  and  $D_3$  turn off as shown in Fig. 4.14. The current through  $R_1$  saturates  $Q_2$ , and  $Q_2$ 's emitter current saturates  $Q_3$ . The sum of voltage drops across the collector-emitter junction of  $Q_2$  and collector-base junction of  $Q_3$  now reaches about 0.8 V. However, this voltage becomes insufficient to turn on  $Q_4$  and the output diode. As a result, the output terminal is pulled low to about 0.2 V and sinks any current from an external load.



**Fig. 4.14** Two-input TTL NAND gate when  $IN_1 = IN_2 = \text{logic 1}$

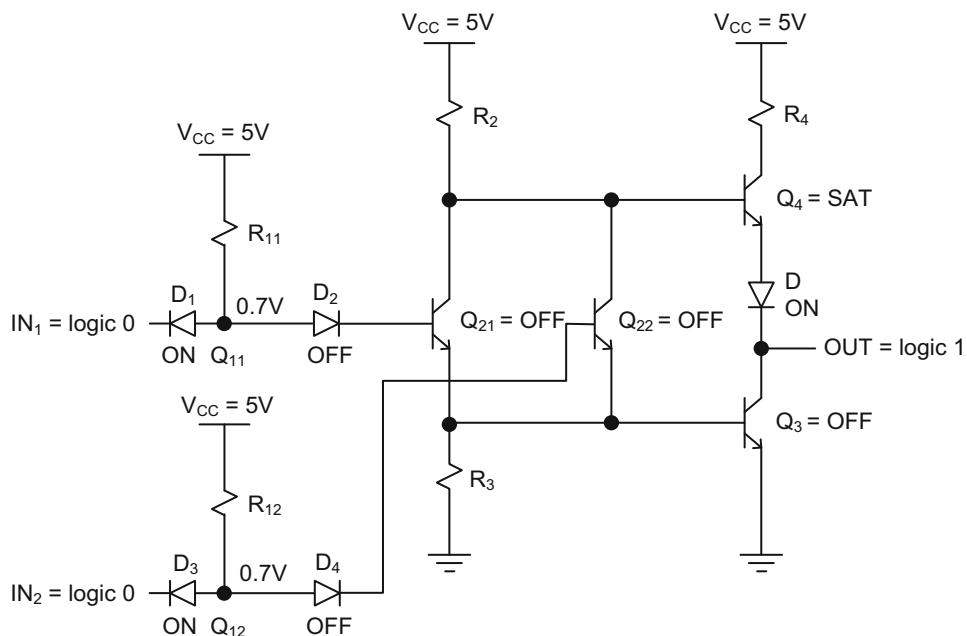
#### 4.4 Two-Input TTL NOR Logic Gate

Two-input TTL NOR gate is shown in Fig. 4.15. This circuit contains two extra transistors,  $Q_{11}$  and  $Q_{12}$ , at the input, and another two,  $Q_{21}$  and  $Q_{22}$ , at the control stage, to produce the NOR behavior.



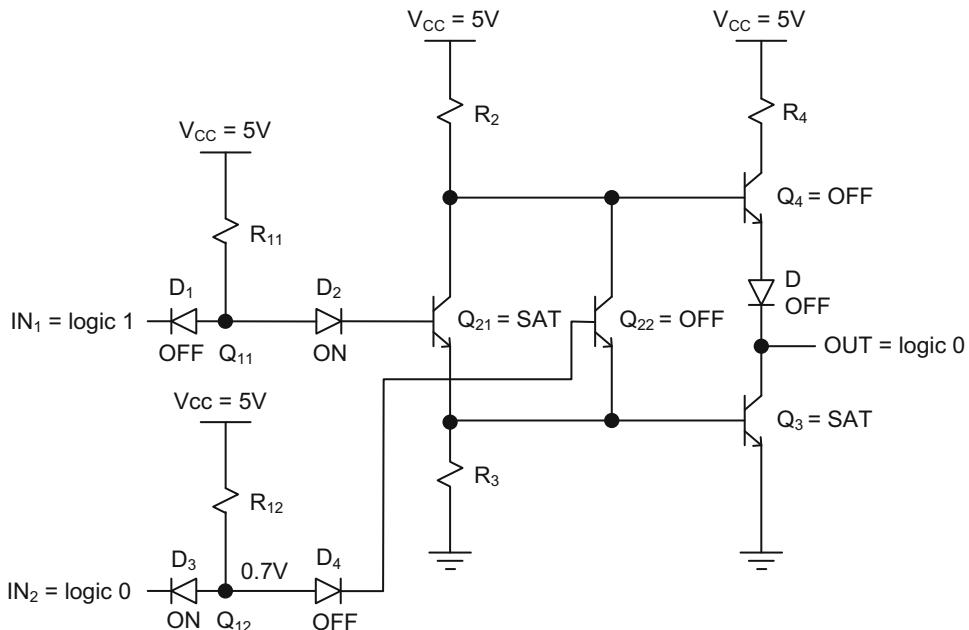
**Fig. 4.15** Two-input TTL NOR gate

When both  $IN_1$  and  $IN_2$  are grounded, the diodes,  $D_1$  and  $D_3$  corresponding to the emitter-base junctions of  $Q_{11}$  and  $Q_{12}$ , turn on as shown in Fig. 4.16. The voltage drop across either of these diodes will not be sufficient to turn on  $D_2$ ,  $D_4$ ,  $Q_{21}$ ,  $Q_{22}$  and  $Q_3$ . The transistor,  $Q_4$ , goes into saturation because a large current is supplied to the base of this transistor through  $R_2$ . As a result, the emitter current of  $Q_4$  produces a source current to an external load, and the output of the gate is pulled up to logic 1.



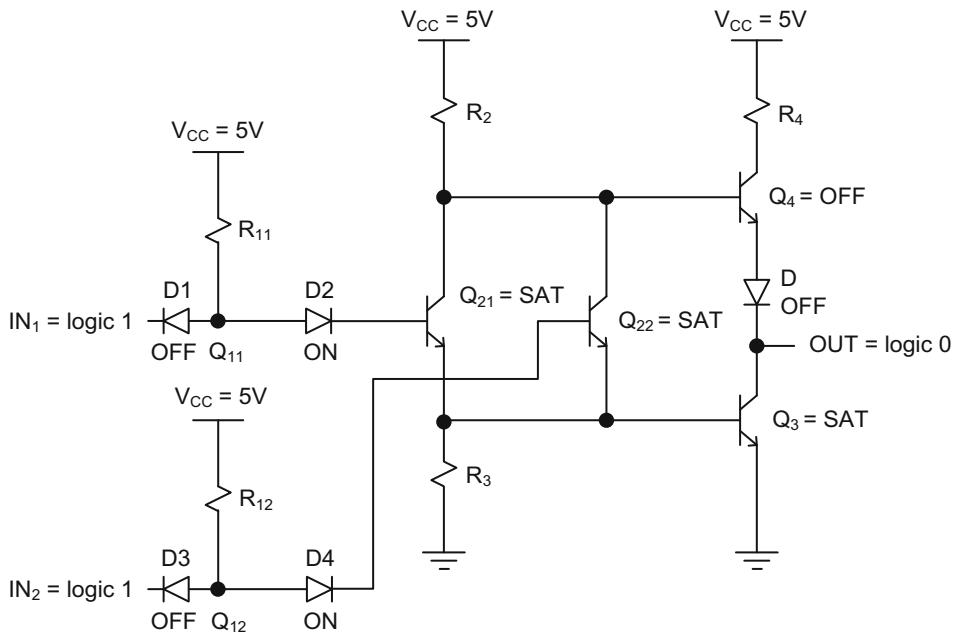
**Fig. 4.16** Two-input TTL NOR gate when  $IN_1 = IN_2 = \text{logic 0}$

When  $IN_1$  is tied to logic 1 and  $IN_2$  at logic 0 (or vice versa), the diode,  $D_1$  corresponding to the emitter-base of  $Q_{11}$ , turns off because there is 0 V across it as shown in Fig. 4.17.  $Q_{21}$  goes into saturation because a large current is supplied to the base of this transistor through  $R_{11}$  but  $Q_{22}$  stays off. A large portion of  $Q_{21}$ 's emitter saturation current flows into the base of  $Q_3$  and saturates this transistor. In the mean time, the output diode  $D$  and  $Q_4$  turn off, leaving  $Q_3$  in charge of sinking all current from an external load and pulling the output terminal to logic 0.



**Fig. 4.17** Two-input TTL NOR gate when one of the inputs is at logic 0 ( $IN_2 = \text{logic 0}$ )

When  $IN_1$  and  $IN_2$  are at logic 1, the diodes,  $D_1$  and  $D_3$  corresponding to the emitter-base junctions of  $Q_{11}$  and  $Q_{12}$ , turn off as shown in Fig. 4.18. However, both  $Q_{21}$  and  $Q_{22}$  go into saturation because of large currents supplied to the base of these transistors through  $R_{11}$  and  $R_{12}$ , respectively. The sum of emitter saturation currents from  $Q_{21}$  and  $Q_{22}$  drive  $Q_3$  into saturation and lower the output voltage to logic 0. Now,  $Q_3$  is able to sink any current from an external load.

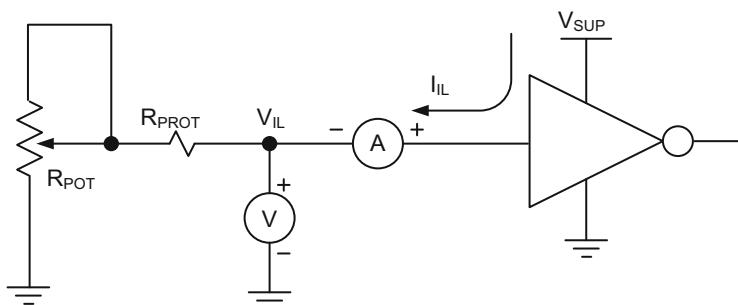


**Fig. 4.18** Two-input TTL NOR gate when  $IN_1 = IN_2 = \text{logic 1}$

#### 4.5 Input Current and Voltage Measurements of a Logic Gate

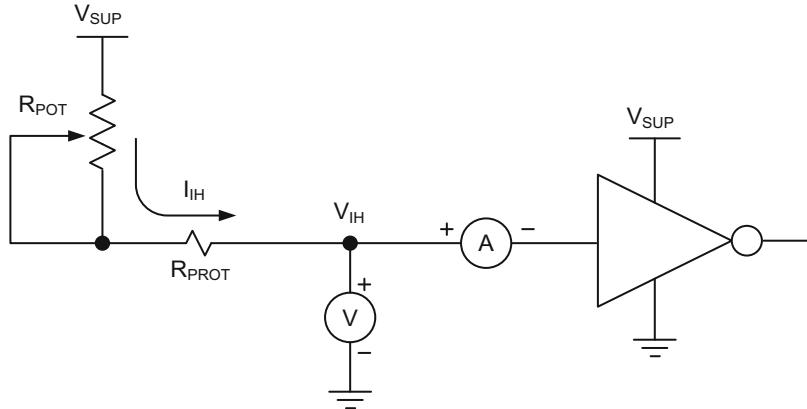
It is imperative to know the typical values of input and output terminal currents and voltages of a logic gate in order to eliminate potential electrical problems when interconnecting the gate with other gates to implement a logic block. This especially applies to logic gates fabricated using different technologies, i.e. a TTL gate interfacing a CMOS gate or vice versa.

Figure 4.19 shows the schematic to measure the input terminal current,  $I_{IL}$ , and voltage,  $V_{IL}$ , when the input is lowered. The potentiometer connected at the input constantly adjusts the value of  $V_{IL}$  while the resistor,  $R_{PROT}$ , limits the magnitude of  $I_{IL}$ . According to this figure, the minimum and maximum values of  $V_{IL}$  become equal to  $R_{PROT} I_{IL}$  and  $(R_{POT} + R_{PROT}) I_{IL}$ , respectively where  $R_{POT}$  is the maximum potentiometer value.



**Fig. 4.19** Input current ( $I_{IL}$ ) and voltage ( $V_{IL}$ ) measurements when the input voltage is low

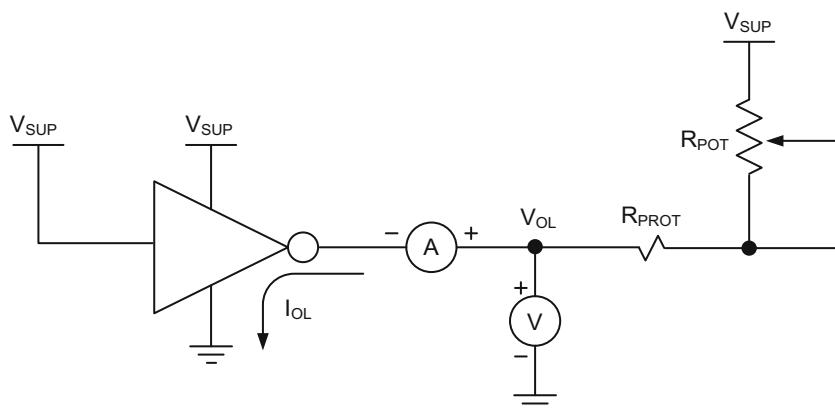
Figure 4.20 shows the setup to measure the input current,  $I_{IH}$ , and the voltage,  $V_{IH}$ , when the input voltage is high. The potentiometer constantly adjusts the value of  $V_{IH}$  while  $R_{PROT}$  limits the magnitude of  $I_{IH}$ . The minimum and maximum values of  $V_{IH}$  become equal to  $V_{SUP} - (R_{POT} + R_{PROT}) I_{IH}$  and  $V_{SUP} - R_{PROT} I_{IH}$ , respectively.



**Fig. 4.20** Input current ( $I_{IH}$ ) and voltage ( $V_{IH}$ ) measurements when the input voltage is high

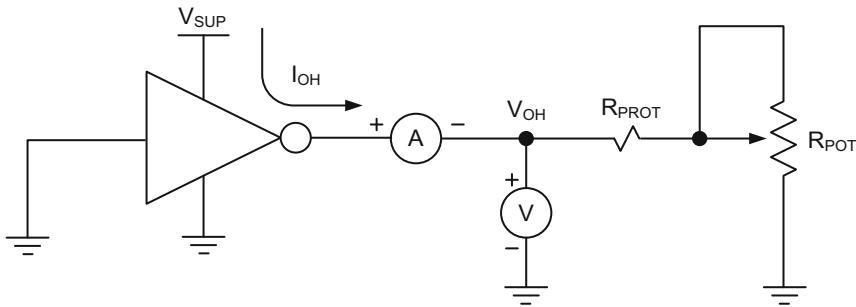
## 4.6 Output Current and Voltage Measurements of a Logic Gate

Current and voltage measurements can also be applied to the output terminals of a logic gate. Figure 4.21 shows the setup to measure the output current,  $I_{OL}$ , and the output voltage,  $V_{OL}$ , when the output voltage is low. Here,  $R_{PROT}$  limits the amount of  $I_{OL}$  while the potentiometer adjusts the value of  $V_{OL}$ . According to this figure, the minimum and maximum  $V_{OL}$  can be calculated as  $V_{SUP} - (R_{POT} + R_{PROT}) I_{OL}$  and  $V_{SUP} - R_{PROT} I_{OL}$ , respectively.



**Fig. 4.21** Output current ( $I_{OL}$ ) and voltage ( $V_{OL}$ ) measurements when the output voltage is low

The setup in Fig. 4.22 is used if the output terminal current,  $I_{OH}$ , and the output terminal voltage,  $V_{OH}$ , need to be measured when the output is high. Here,  $R_{PROT}$  limits the amount of  $I_{OH}$  while the potentiometer constantly adjusts the value of  $V_{OH}$ . According to this figure, the minimum and maximum values of  $V_{OH}$  can be calculated as  $R_{PROT} I_{OH}$  and  $(R_{POT} + R_{PROT}) I_{OH}$ , respectively.

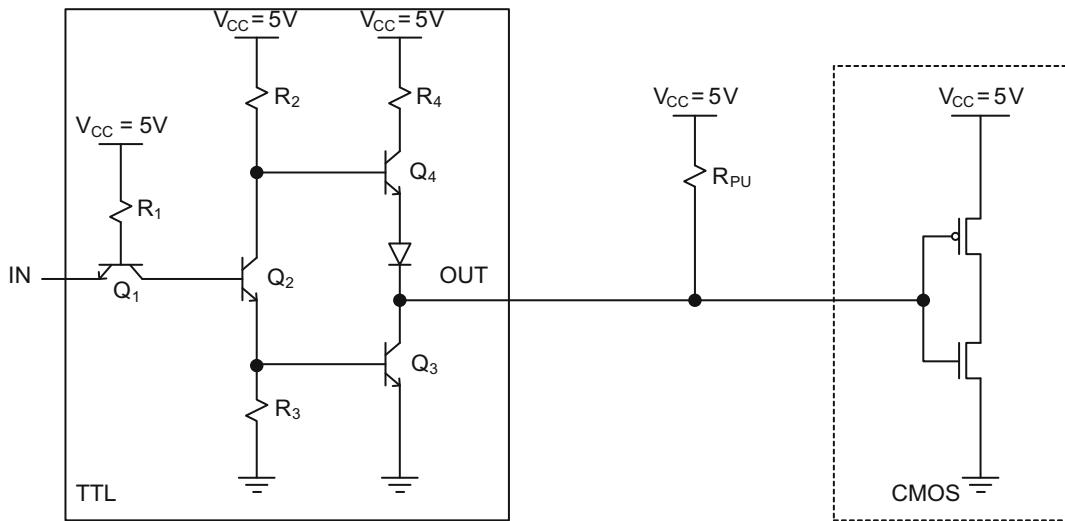


**Fig. 4.22** Output current ( $I_{OH}$ ) and voltage ( $V_{OH}$ ) measurements when the output voltage is high

It is very possible to interface an old TTL logic gate with a state-of-the art CMOS logic gate. Logic level adjustment is a critical when interfacing tasks take place between two different logic families. This is mainly because power supply voltages used in each logic family may be different or each logic family may require different input and output logic levels. When interfacing becomes unavoidable, TTL and CMOS gates can be connected either using a passive device such as a resistor or an active device such as a bipolar transistor.

#### 4.7 TTL-CMOS Interface with a Pull-up Resistor

When a TTL gate drives a CMOS gate, an external pull-up resistor,  $R_{PU}$ , must be added to the interface to be able to switch the CMOS gate.



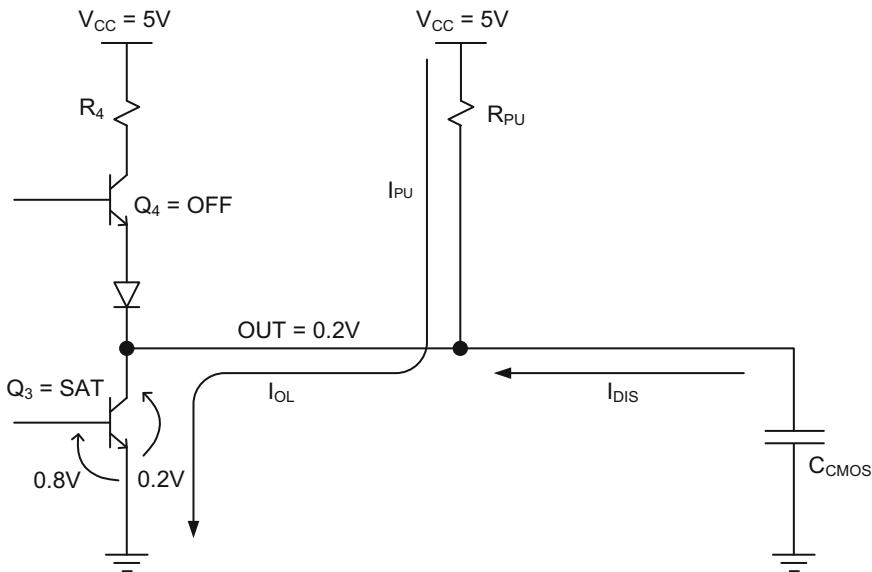
**Fig. 4.23** A common TTL gate driving a CMOS gate with a pull-up resistor,  $R_{PU}$

Figure 4.23 shows the interface where the TTL inverter input, IN, is at logic 1, and its output produces  $OUT = 0.2$  V. TTL does not have any problem to switch the state of the CMOS inverter at this voltage level. The combination of a transient discharge current from the input of the CMOS inverter and a static current through the pull-up resistor,  $I_{PU}$ , determines the maximum sink current value,  $I_{OL}$ , of the TTL inverter as shown in Fig. 4.24. The  $I_{OL}$  of the TTL inverter is fetched from the manufacturer's datasheet to calculate  $R_{PU}$ . Neglecting  $I_{DIS}$ ,  $I_{OL}$  becomes:

$$I_{OL} \approx I_{PU} = \frac{(5 - V_{OL})}{R_{PU}} = \frac{(5 - 0.2)}{R_{PU}} \quad (4.6)$$

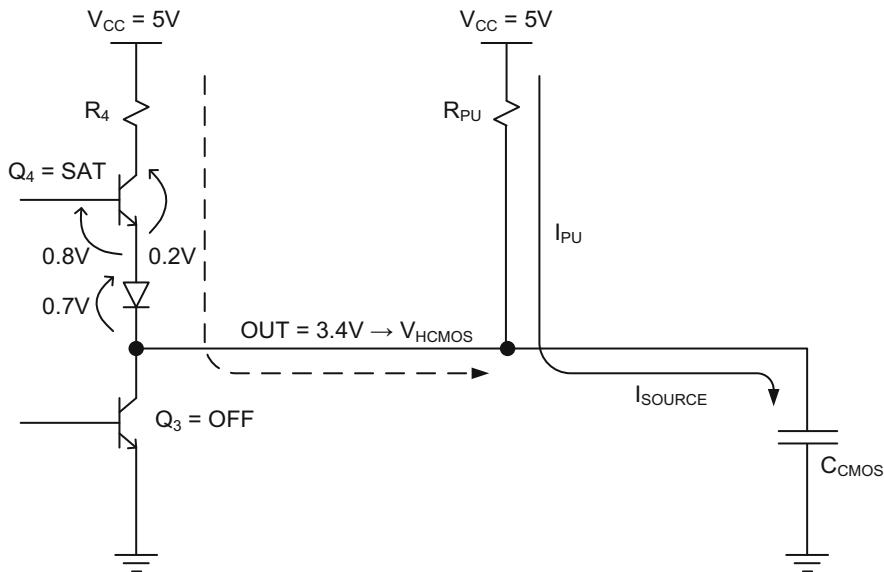
Thus,

$$R_{PU} \approx \frac{4.8}{I_{OL}} \quad (4.7)$$



**Fig. 4.24** A common TTL gate driving a CMOS gate with  $R_{PU}$  when TTL input is at logic 1

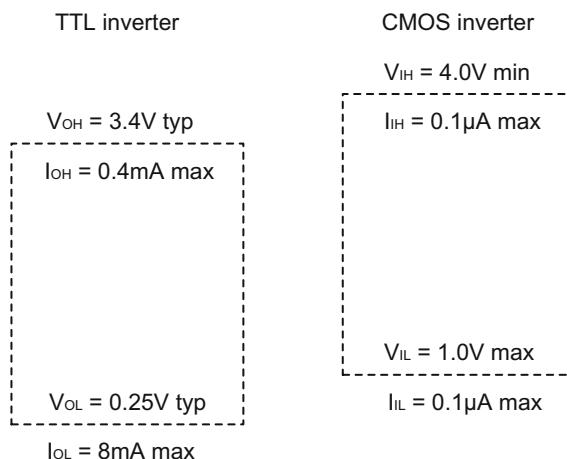
The CMOS switching becomes an issue when the TTL input is at logic 0 as shown in Fig. 4.25. In this case, the TTL output can produce a voltage in the neighborhood of 3.4–3.5 V. This voltage is more than enough to drive another common TTL gate. However, it may not be sufficient to drive a CMOS gate operating with a 5 V voltage supply. In general, CMOS gates require inputs well above their gate threshold voltage to be able to change the state of their output. Therefore, Q<sub>4</sub> will contribute delivering current to the input capacitance of the CMOS gate up until OUT reaches at 3.4 V. But, charging the input capacitance of the CMOS gate beyond 3.4 V is performed solely by the current through the pull-up resistor,  $I_{PU}$ , until the voltage across the capacitor eventually reaches 5 V.



**Fig. 4.25** A common TTL gate driving a CMOS gate with  $R_{PU}$  when TTL input is at logic 0

**Example 4.3** Assume that a TTL inverter operates with  $V_{CC} = 5\text{ V}$  and needs to drive several CMOS inverters, all operating with  $V_{DD} = 5\text{ V}$ . Design a passive interface so the TTL inverter has the capability to drive multiple CMOS inverters.

The DC specifications for both parts are given in Fig. 4.26. According to this table, the TTL inverter cannot drive CMOS inverter(s) because the typical  $V_{OHTTL} = 3.4\text{ V}$  is less than the minimum value of  $V_{IHCMOS} = 4\text{ V}$ . Therefore, a pull-up resistor must be included between the gates as a passive interface. The low level voltages are compatible since  $V_{OLTTL} = 0.25\text{ V}$  is below the maximum  $V_{ILCMOS} = 1\text{ V}$ .



**Fig. 4.26** DC parameter diagram for the TTL and CMOS inverters

When the output of the TTL inverter is at logic 0, one can calculate the minimum value of  $R_{PU}$  using Eq. 4.8.

$$R_{PU} = \frac{(5 - V_{OLTTL})}{I_{OLTTL}} = \frac{(5 - 0.25)}{8\text{mA}} = 593 \Omega \quad (4.8)$$

$R_{PU}$  can be increased as high as  $1 \text{ K}\Omega$ . However, this produces a static current of  $I_{OLTTL}$  of  $4.75 \text{ mA}$  while the remaining current,  $8 \text{ mA} - 4.75 \text{ mA} = 3.25 \text{ mA}$ , is reserved to accommodate transient discharge currents from CMOS inverter input capacitor(s).

When the TTL inverter output switches to logic 1 (only about  $3.4 \text{ V}$ ), the additional current to charge input capacitance of CMOS inverter(s) from  $3.4 \text{ V}$  to  $5 \text{ V}$  will be through  $R_{PU}$ . This passive charge path creates a time constant of  $\tau = R_{PU} C_{CMOS}$  where  $C_{CMOS}$  is the total input capacitance of CMOS inverter(s). Using smaller  $R_{PU}$  values decreases this transient period at the expense of more power consumption.

Another important consideration in the interface design is the output current capability of the TTL inverter. One can compute both the high and low fan-out values,  $N_H$  and  $N_L$ , as follows:

$$N_H = I_{OHTTL}/I_{IHCMOS} = 0.4 \text{ mA}/0.1 \mu\text{A} = 4000$$

and

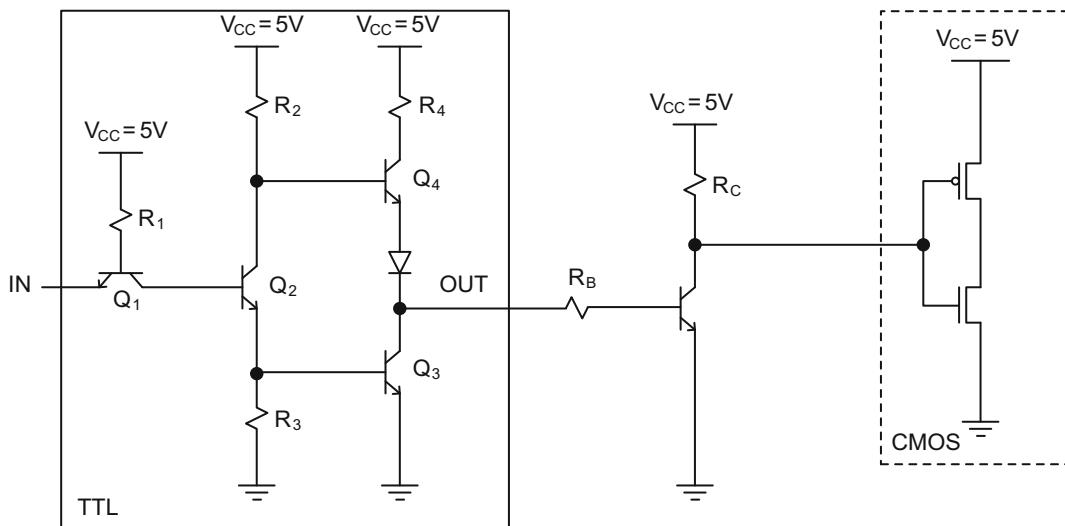
$$N_L = I_{OLTTL}/I_{ILCMOS} = 8 \text{ mA}/0.1 \mu\text{A} = 80,000$$

According to this calculation, the TTL inverter can support up to 4000 CMOS inverters at its output.

---

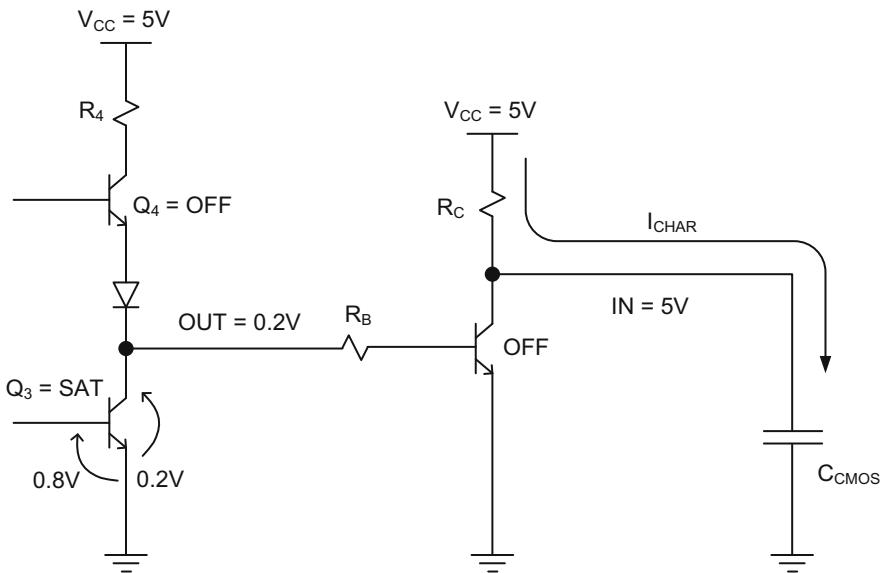
## 4.8 TTL-CMOS Interface with a Bipolar Transistor

When a TTL gate drives a CMOS gate, an external active device, such as an NPN bipolar transistor shown in Fig. 4.27, can be used to adjust the required input voltage levels for the CMOS gate. However, using an active device, such as an NPN transistor in the interface, also introduces an extra layer of inversion between the TTL and the CMOS gates, requiring design modifications in the original design for correct logic functionality.



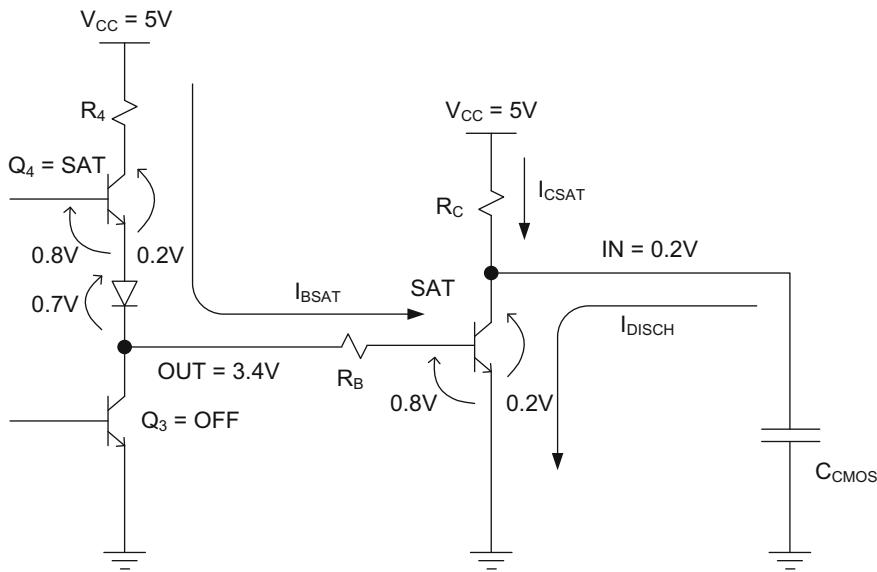
**Fig. 4.27** A common TTL gate driving a CMOS gate with an NPN bipolar transistor

When  $IN = 5\text{ V}$  is applied to the TTL input,  $OUT$  becomes  $0.2\text{ V}$ , which turns off the NPN transistor at the TTL-CMOS interface. The supply current,  $I_{CHAR}$ , through  $R_C$  charges the input capacitor of the CMOS gate to  $5\text{ V}$  as shown in Fig. 4.28.



**Fig. 4.28** A TTL gate driving a CMOS gate with an NPN when TTL input is at logic 1

Figure 4.29 examines the case when  $IN = 0\text{ V}$  at the TTL inverter's input. The TTL output becomes about  $3.4\text{ V}$ , which is more than enough to saturate the NPN transistor at the interface.



**Fig. 4.29** A TTL gate driving a CMOS gate with an NPN when TTL input is at logic 0

Therefore, we have:

$$I_{BSAT} = I_{OH} = \frac{(OUT - 0.8)}{R_B} \quad (4.9)$$

or,

$$R_B = \frac{(OUT - 0.8)}{I_{OH}} \quad (4.10)$$

Replacing OUT with V<sub>OH</sub> yields the value for R<sub>B</sub>. Thus,

$$R_B = \frac{(V_{OH} - 0.8)}{I_{OH}} \quad (4.11)$$

Also,

$$I_{CSAT} = \frac{(V_{CC} - 0.2)}{R_C} = \frac{(5 - 0.2)}{R_C} = \frac{4.8}{R_C} \quad (4.12)$$

or

$$R_C = \frac{4.8}{I_{CSAT}} \quad (4.13)$$

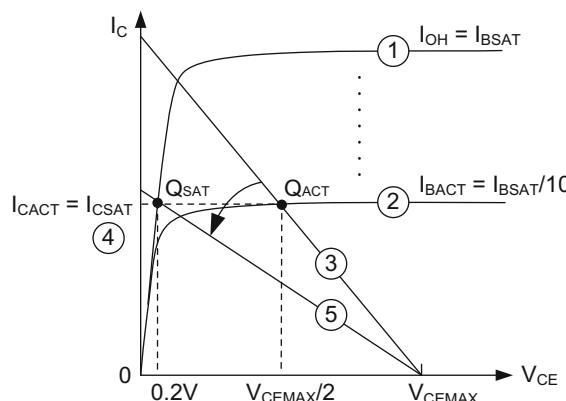
In order to find I<sub>CSAT</sub> in Eq. 4.13, we need to locate the I<sub>B</sub>-curve that corresponds to I<sub>OH</sub> = I<sub>BSAT</sub> in the I-V characteristics in Fig. 4.30 as the step no. 1.

However, we know that

$$I_{BACT} \approx \frac{I_{BSAT}}{10} \quad (4.14)$$

Therefore, in step no. 2 we locate the  $I_B$ -curve that corresponds to  $I_{BACT}$  in Fig. 4.30. In step no. 3, we draw a load line that intersects the  $I_{BACT}$ -curve at  $V_{CE} = V_{CEMAX}/2$ . This intersection is the quiescent point,  $Q_{ACT}$ , for the transistor operating in the middle of active region, and produces  $I_{CACT}$ . In step no. 4, we rotate the active-region load line counter-clockwise around  $V_{CEMAX}$  until the new load line intersects the  $I_{BSAT}$ -curve at  $I_{CSAT}$ . The new quiescent point,  $Q_{SAT}$ , switches transistor operation from active region to saturation region. The value of  $I_{CSAT}$  can now be used in Eq. 4.13.

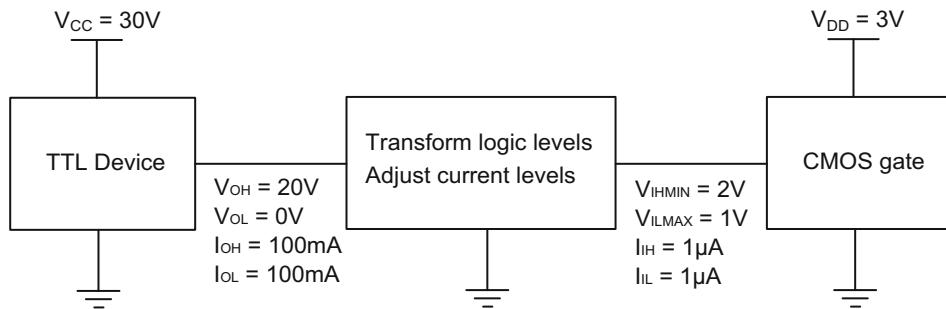
Note that the multiplier, 10, used to define the ratio between  $I_{BSAT}$  and  $I_{BACT}$  in Eq. 4.14 is a conservative number and creates high static power consumption when the NPN transistor is on at the interface. Multipliers in the order of 2 or 3 can be more realistic to achieve lower the power consumption.



**Fig. 4.30** Load line rotation of the bipolar circuit at the interface

**Example 4.4** Assume that a high-voltage TTL device operating with a power supply voltage of  $V_{CC} = 30$  V needs to drive a CMOS gate(s) with a supply voltage of  $V_{DD} = 3$  V as shown in Fig. 4.31.

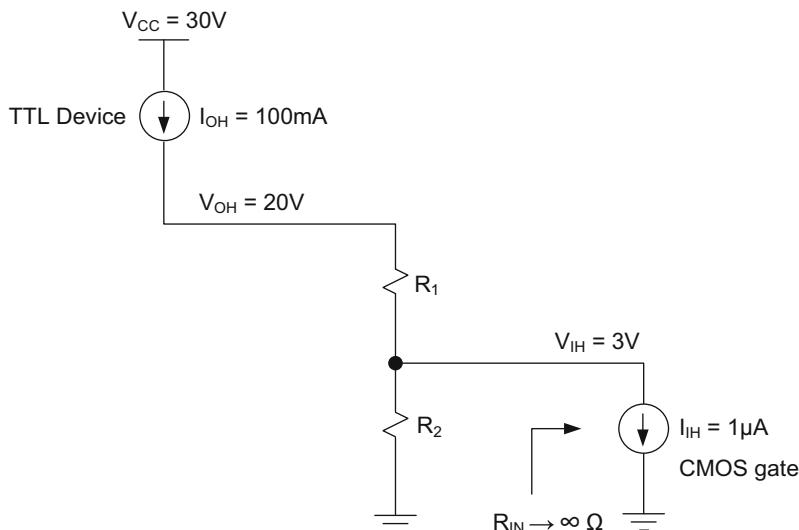
The nominal high and low output logic levels of the TTL device are  $V_{OH} = 20$  V and  $V_{OL} = 0$  V, respectively. The TTL device can generate up to  $I_{OH} = 100$  mA source and  $I_{OL} = 100$  mA sink currents for an external load. The CMOS gate, on the other hand, needs to have a minimum  $V_{IH} = 2$  V and a maximum of  $V_{IL}$  of 1 V to operate. However, the nominal CMOS input voltage levels are  $V_{IH} = 3$  V and  $V_{IL} = 0$  V at which the CMOS gate draws approximately  $I_{IH} = I_{IL} = 1$   $\mu$ A as shown in this figure.



**Fig. 4.31** Block diagram of a high voltage TTL device driving CMOS gate(s)

How can one design an interface between the TTL device and the CMOS gate using (a) a passive network and (b) an active NPN bipolar transistor that can transform the voltage levels?

The first task is to use passive elements, such as resistors, at the interface. When the TTL device produces  $V_{OH} = 20 V$ , the CMOS gate can only accommodate a maximum of  $V_{IH} = 3 V$ . If we use a passive pull-up resistor at the interface for logic translation, the TTL high output voltage appears directly at the input of the CMOS gate and burn the device. Therefore, the only other choice is to use a step-down resistor network composed of  $R_1$  and  $R_2$  as shown in Fig. 4.32.



**Fig. 4.32** Passive interface design when TTL device produces high output

Therefore,

$V_{OH} = (R_1 + R_2) I_{OH}$  since the input impedance of the CMOS gate approaches infinity. Substituting the values of  $V_{OH}$  and  $I_{OH}$  yields:

$$20 \text{ V} = (R_1 + R_2)100 \text{ mA}$$

or

$$R_1 + R_2 = 200 \Omega \quad (4.15)$$

However,

$$V_{IH} = R_2 I_{OH} = 3 \text{ V} \quad (4.16)$$

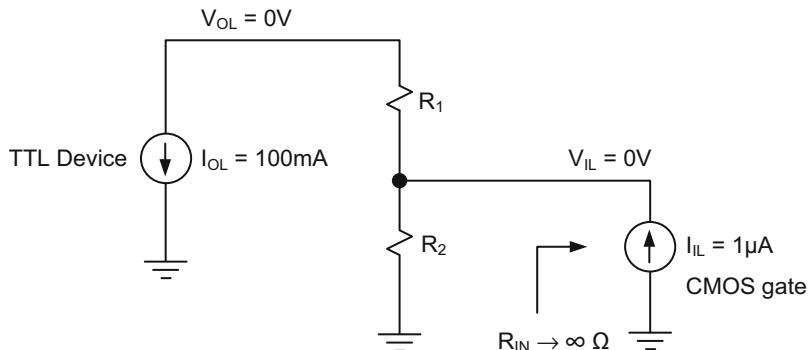
or

$$R_2 = 30 \Omega \quad (4.17)$$

Thus,

$$R_1 = 200 - 30 = 170 \Omega \quad (4.18)$$

Figure 4.33 shows the opposite case when the TTL output becomes  $V_{OL} = 0 \text{ V}$ , and sinks 100 mA current emanating from the interface. However, this case does not provide additional information in calculating anything else but identifying the transient properties of the interface composed of  $I_{OL}$  and the total input capacitance of CMOS gate(s), leading to the calculation of TTL-fanout.



**Fig. 4.33** Passive interface design when TTL device produces low output

The result of using resistive network creates a static power consumption of  $I_{OH} V_{OH} = 2 \text{ W}$ , which is quite high to be able to use resistors.

The next design suggests using a NPN bipolar transistor at the interface to adjust the current and voltage levels. However, the inclusion of this device also inverts the TTL output and requires redesigning the logic functionality.

When TTL output produces  $V_{OH} = 20 \text{ V}$  and  $I_{OH} = 100 \text{ mA}$  as shown in the equivalent circuit in Fig. 4.34, the bipolar transistor at the interface goes into saturation.  $I_{BSAT}$  becomes equal to  $I_{OH}$ .

Thus,

$$V_{OH} = R_B I_{BSAT} + V_{BESAT} = R_B I_{OH} + V_{BESAT} \quad (4.19)$$

$$20 - 0.8 = R_B \cdot 100 \text{ mA}$$

$$R_B = \frac{19.2}{100 \text{ mA}} = 192 \Omega \quad (4.20)$$

Similarly,

$$V_{SUP} = R_C I_A + V_{CESAT} \quad (4.21)$$

$$I_A \approx I_{CSAT} = \frac{(3 - 0.2)}{R_C} = \frac{2.8}{R_C} \quad (4.22)$$

However, to keep the bipolar transistor in saturation one needs:

$$I_{BSAT} \gg I_{BACT} \quad (4.23)$$

A conservative approach is to make  $I_{BSAT}$  10 times higher than  $I_{BACT}$ . However, the use of smaller multipliers, such as 2 or 3, is possible depending on the transistor output I-V characteristics as discussed earlier. Thus,

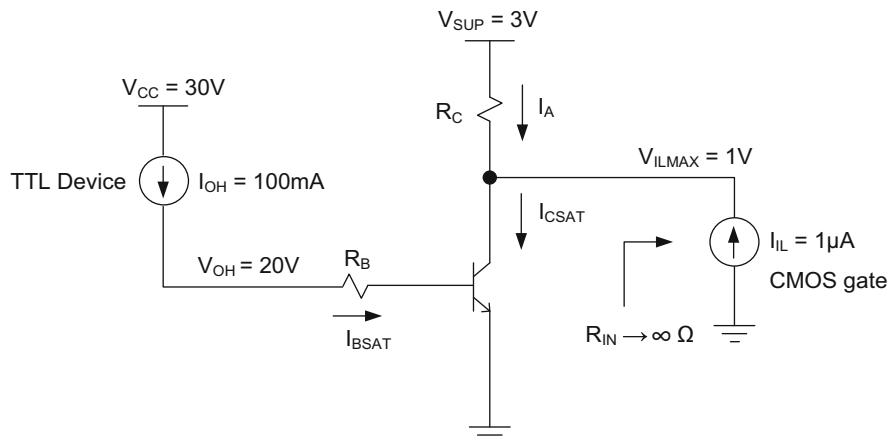
$$I_{BSAT} \approx 10 I_{BACT} = 10 \frac{I_{CACT}}{\beta} \quad (4.24)$$

Substituting the values for  $I_{BSAT} = 100 \text{ mA}$ , and  $I_{CACT} \approx I_{CSAT}$ , and Eq. 4.22 into Eq. 4.24 yields:

$$100 \text{ mA} \approx \frac{10}{\beta} \left( \frac{2.8}{R_C} \right) \quad (4.25)$$

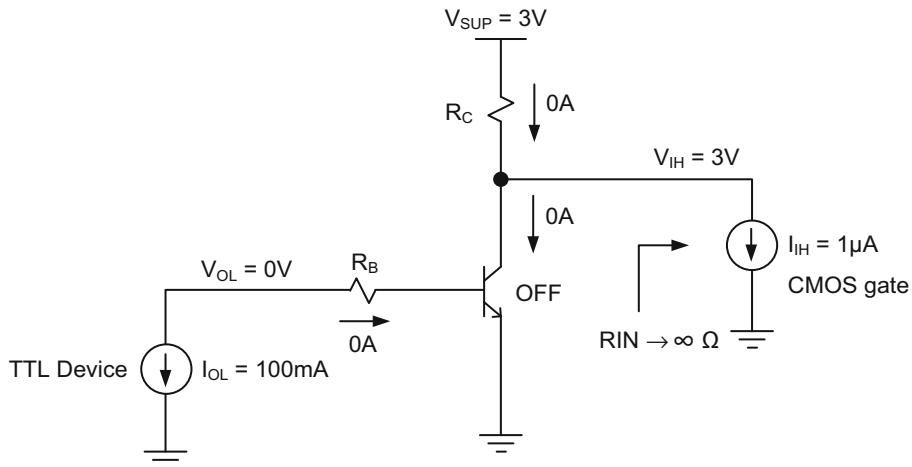
For  $\beta = 100$

$$R_C = \frac{10}{100} \left( \frac{2.8}{100 \text{ mA}} \right) = 2.8 \Omega \quad (4.26)$$



**Fig. 4.34** Active interface design when TTL device produces high output

Figure 4.35 illustrates the case where the TTL device output produces  $V_{OL} = 0 V$  and  $I_{OL} = 100 mA$ , which turns off the bipolar transistor. No further design data can be obtained from this case.

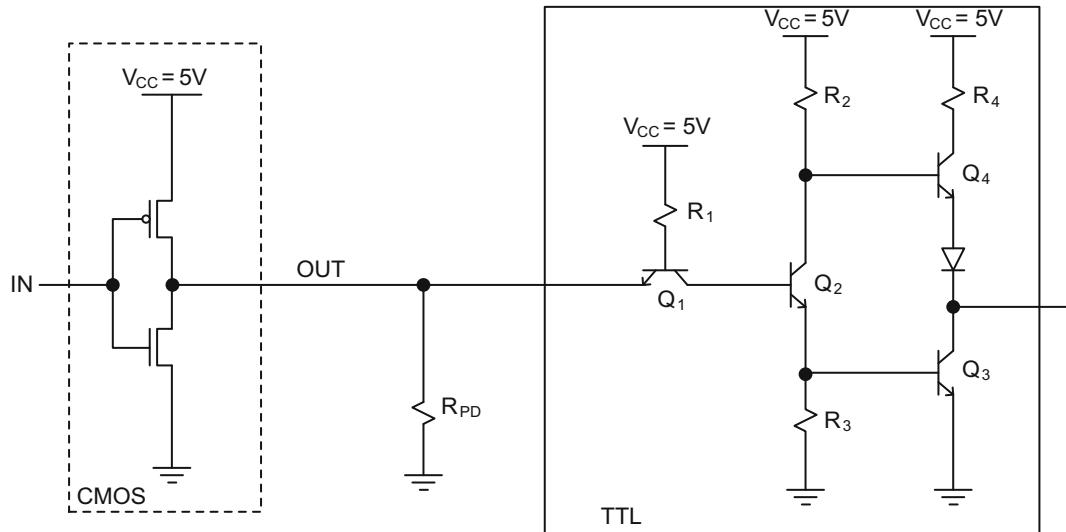


**Fig. 4.35** Active interface design when TTL device produces low output

#### 4.9 CMOS-TTL Interface with a Pull-Down Resistor

When a CMOS gate drives a TTL gate, the high logic output of the CMOS gate does not create an issue for the input circuitry of the TTL gate. The problem arises only when the output of the CMOS gate needs to lower the input of the TTL gate towards logic 0, and in

the process, it needs to drain a large input current from the TTL gate. An external pull-down resistor,  $R_{PD}$ , helps sinking the large TTL input current and eventually causes the TTL gate to switch its state as shown in Fig. 4.36.

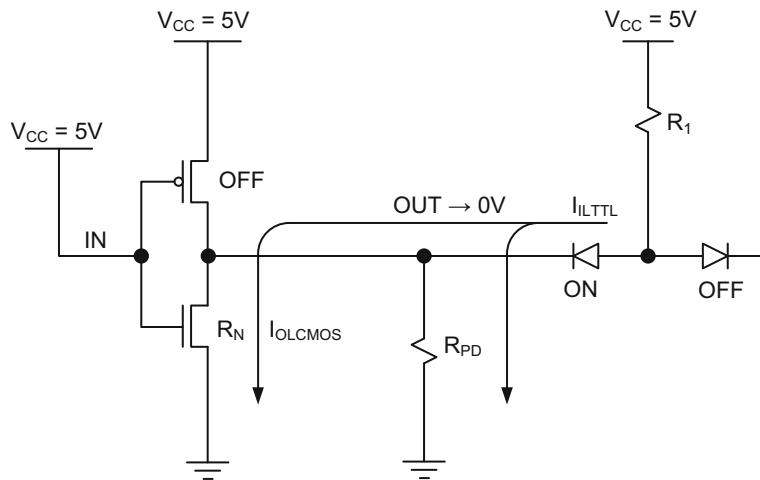


**Fig. 4.36** CMOS inverter driving a TTL gate with a passive pull-down resistor

When the CMOS input voltage is 5 V, the NMOS transistor turns on with an equivalent resistance of  $R_N$  as shown in Fig. 4.37. The diode corresponding to the emitter-base junction of  $Q_1$  turns on. In case the output current sinking capability of the CMOS gate is much smaller than the input current generated by the TTL gate, the pull-down resistor,  $R_{PD}$ , creates an extra drainage channel for the TTL input current as shown in Fig. 4.37.  $R_{PD}$  is determined as follows:

$$R_{PD} = \frac{V_{ILTTL}}{(I_{ILTTL} - I_{OLCMOS})} \quad (4.27)$$

Here,  $V_{ILTTL}$  and  $I_{ILTTL}$  are the low input voltage and the low input current of the TTL gate, respectively.  $I_{OLCMOS}$  is the low output current of the CMOS gate.

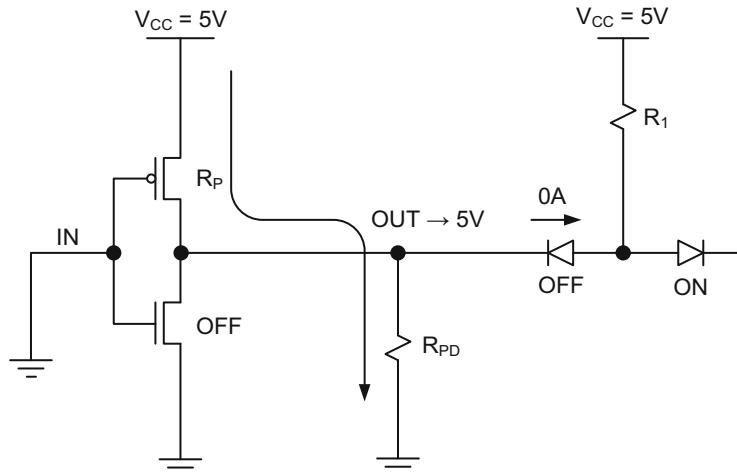


**Fig. 4.37** CMOS inverter driving a TTL gate with  $R_{PD}$  when CMOS input is at 5 V

When the CMOS input voltage is 0 V, the PMOS transistor of the CMOS gate turns on and exhibits an equivalent resistance,  $R_P$ , as shown in Fig. 4.38. Since the diode corresponding to the emitter-base junction of  $Q_1$  turns off, the output voltage becomes:

$$OUT = \frac{R_{PD}}{R_P + R_{PD}} V_{CC} \quad (4.28)$$

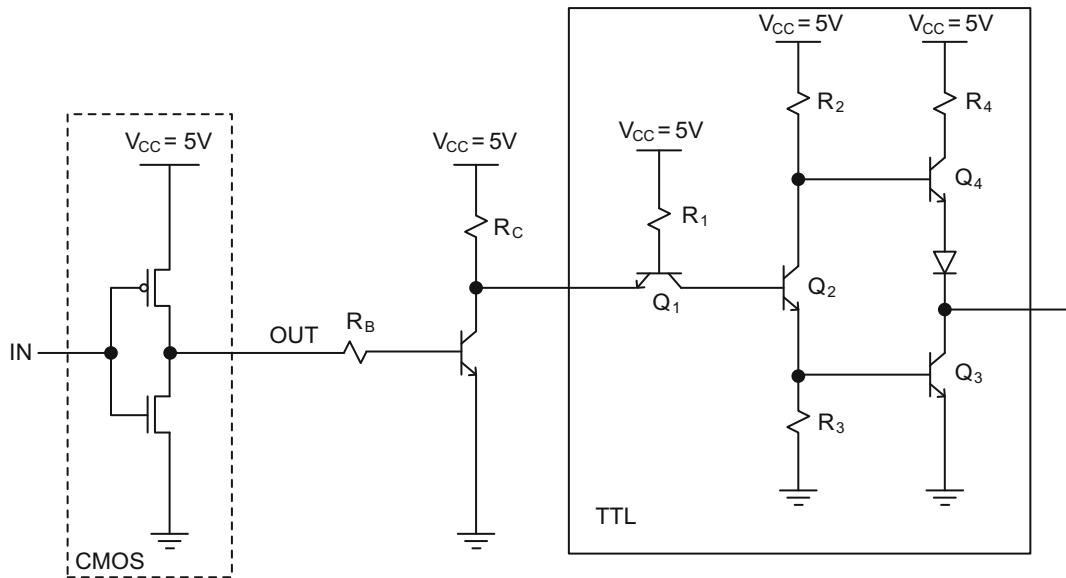
OUT approaches to  $V_{CC} = 5$  V as long as  $R_{PD} \gg R_P$ .



**Fig. 4.38** CMOS inverter driving a TTL gate with  $R_{PD}$  when CMOS input is at 0 V

#### 4.10 CMOS-TTL Interface with a Bipolar Transistor

When a CMOS gate drives a TTL gate, especially with a different supply voltage, it may be prudent to use an NPN transistor at the interface in order to translate the output voltage levels of the CMOS gate for the input voltage levels of the TTL gate as shown in Fig. 4.39. However, the bipolar circuit also creates a logic inversion as indicated previously.



**Fig. 4.39** CMOS inverter driving a TTL gate with an NPN bipolar transistor

Figure 4.40 shows when  $IN = 0$  V is applied to the input of the CMOS inverter. Since the output of the CMOS inverter reaches 5 V, the NPN transistor at the interface saturates. Therefore, one must satisfy  $I_{BSAT} \gg I_{BACT}$  for this transistor to enter into the saturation region.

Using a conservative multiplication factor of 10,  $I_{BSAT} \approx 10I_{BACT}$ .

Thus,

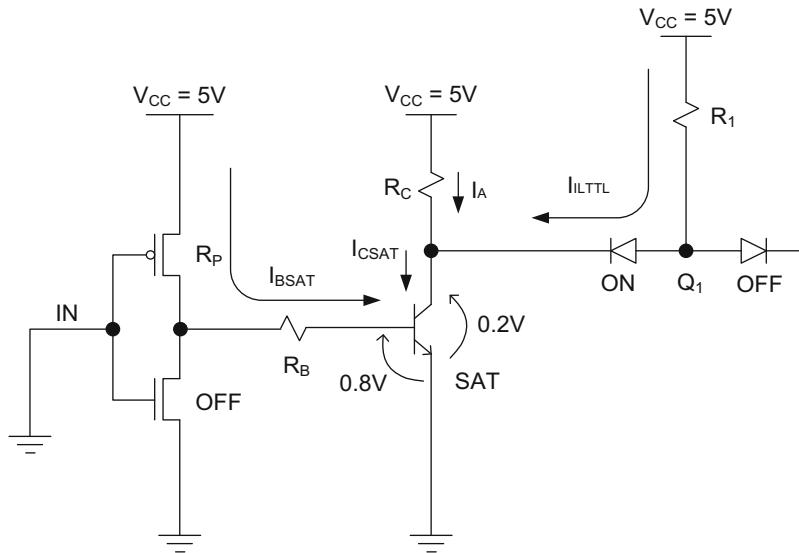
$$I_{BSAT} \approx 10 \frac{I_{CACT}}{\beta} \quad (4.29)$$

However,

$$I_{BSAT} = I_{OHCMS} \quad (4.30)$$

and

$$I_{CSAT} = I_A + I_{ILTTL} = \frac{(V_{CC} - V_{ILTTL})}{R_C} + I_{ILTTL} \approx I_{CACT} \quad (4.31)$$



**Fig. 4.40** CMOS inverter driving a TTL gate with a bipolar transistor when IN = 0 V

Here,  $I_{OHCMOS}$  is the high output current of the CMOS gate, and  $V_{ILTTL}$  and  $I_{ILTTL}$  are the low input voltage and input current of the TTL gate, respectively.

Substituting the values of  $I_{BSAT}$  in Eq. 4.30 and  $I_{CSAT} \approx I_{CACT}$  in Eq. 4.31 in Eq. 4.29 yields:

$$I_{OHCMOS} = \frac{10}{\beta} \left( \frac{V_{CC} - V_{ILTTL} + R_C I_{ILTTL}}{R_C} \right) \quad (4.32)$$

Reorganizing the terms in Eq. 4.32 yields:

$$R_C = \frac{10(V_{CC} - V_{ILTTL})}{(\beta I_{OHCMOS} - 10I_{ILTTL})} \quad (4.33)$$

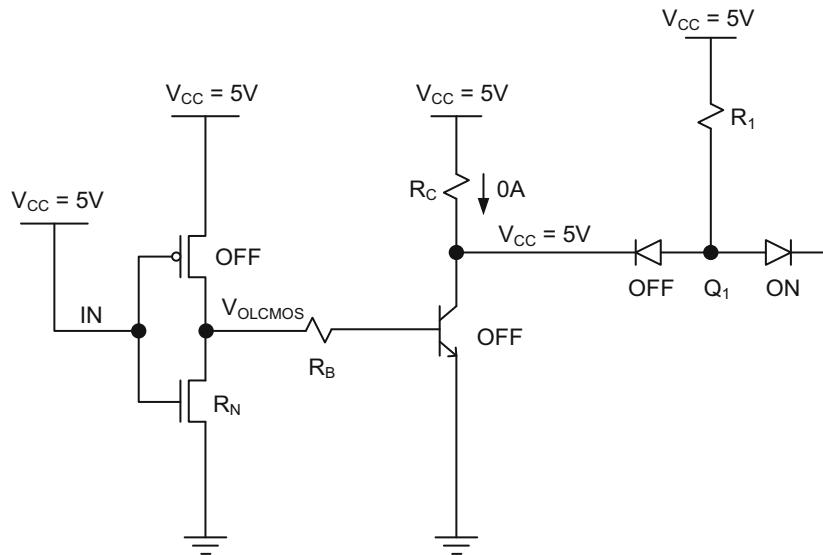
$R_B$  is selected as follows:

$$V_{OHCMOS} = R_B I_{BSAT} + 0.8 = R_B I_{OHCMOS} + 0.8 \quad (4.34)$$

Here,  $V_{OHCMOS}$  is the high output voltage of the CMOS gate. Then,

$$R_B = \frac{(V_{OHCMOS} - 0.8)}{I_{OHCMOS}} \quad (4.35)$$

When  $IN = 5 V$ , the output voltage of the CMOS gate becomes equal to  $V_{OLCMOS}$  as shown in Fig. 4.41. However, this voltage is not sufficient to turn on the interfacial bipolar transistor. Therefore, the NPN transistor's collector reaches 5 V and turns off the diode corresponding to the emitter-base junction of  $Q_1$  in the TTL gate.



**Fig. 4.41** CMOS inverter driving a TTL gate with a bipolar transistor when  $IN = 5 V$

## Review Questions

1. Determine  $R_1$ ,  $R_2$ ,  $R_3$  and  $R_4$  resistors in a TTL inverter with the following I/O characteristics. Use  $\beta = 100$  for all four NPN transistors. Assume  $V_{CC} = 5$  V.

$$V_{IL} = 0.2 \text{ V} \quad V_{OL} = 0.2 \text{ V}$$

$$V_{IH} = 3.5 \text{ V} \quad V_{OH} = 3.5 \text{ V}$$

$$I_{IL} = 1 \text{ mA} \quad I_{OL} = 10 \text{ mA}$$

$$I_{IH} = 0 \text{ mA} \quad I_{OH} = 10 \text{ mA}$$

2. Determine  $R_1$ ,  $R_2$  and  $R_3$  resistors in an open collector TTL inverter with the following I/O characteristics. Use  $\beta = 100$  for all four NPN transistors. Assume  $V_{CC} = 20$  V.

$$V_{IL} = 0.2 \text{ V} \quad V_{OL} = 0.2 \text{ V}$$

$$V_{IH} = 20 \text{ V} \quad V_{OH} = 20 \text{ V}$$

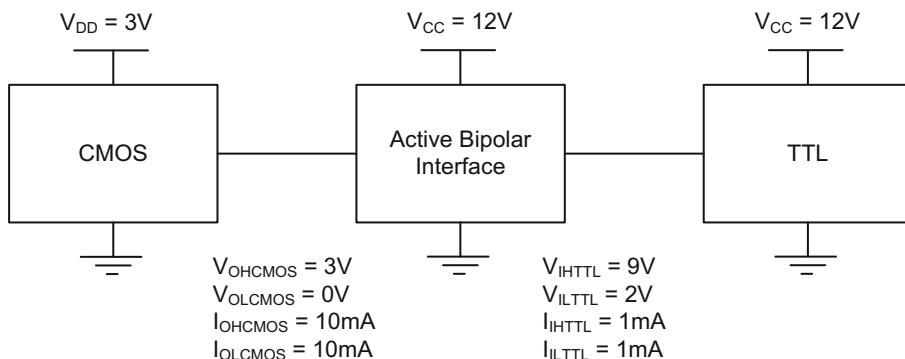
$$I_{IL} = 1 \text{ mA} \quad I_{OL} = 10 \text{ mA}$$

$$I_{IH} = 0 \text{ mA} \quad I_{OH} = 0 \text{ mA}$$

3. Implement  $out = A \oplus B$  in a TTL logic gate using discrete gates. Try to implement the same logic function in an integrated form using a single gate.

4. A CMOS inverter with 3 V power supply needs to be interfaced with a TTL inverter with 12 V power supply as shown below. The interface is an active interface composed of an NPN transistor and resistors operating with 12 V power supply. The NPN bipolar transistor has a current gain of  $\beta = 100$ , and it goes into saturation if  $I_{BSAT} \approx 10I_{BACT}$ . The output current and voltage ratings of the CMOS inverter and input current and voltage ratings of the TTL inverter are shown below.

Find the resistor values connected to the base and the collector of the NPN transistor in this active interface. Apply both high (3 V) and low logic values (0 V) at the input of the CMOS gate, and check the voltage levels at the interface.



5. A bipolar circuit with an emitter resistance,  $R_E$ , below is used to feed two CMOS inputs. TTL<sub>OUT</sub> port has the following characteristics:

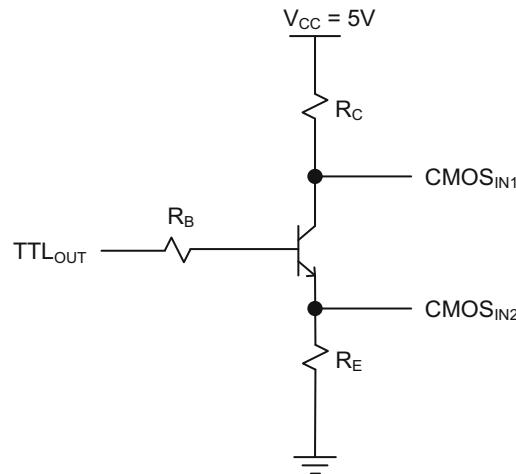
$$V_{OH} = 3.5 \text{ V}$$

$$V_{OL} = 0.2 \text{ V}$$

$$I_{OH} = 20 \text{ mA}$$

$$I_{OL} = 20 \text{ mA}$$

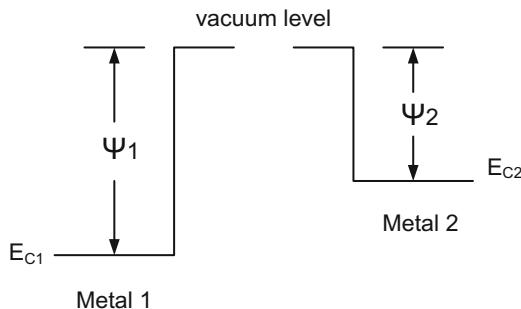
Determine the resistors  $R_B$ ,  $R_C$  and  $R_E$ . What are the current and voltage levels at CMOS<sub>IN1</sub> and CMOS<sub>IN2</sub> ports? According to your calculations, can CMOS<sub>IN2</sub> port be used to change the state of a CMOS gate?



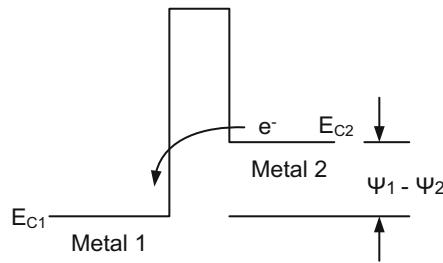
## 5.1 Thermocouple

The temperature sensing in embedded systems is achieved by a device called thermocouple. The basic thermocouple structure consists of a junction of two metals with different work functions,  $\Psi_1$  and  $\Psi_2$ , as shown in Fig. 5.1. Work function is an energy level between the conduction band of the metal and the vacuum level and changes with temperature. When two metals are welded together, electrons from the metal with lower work function transfer to the metal with the higher work function due to the difference in the conduction band energy levels as shown in Fig. 5.2. When the equilibrium is reached, temperature variations will change the amplitude of the work function difference,  $(\Psi_1 - \Psi_2)$ , and cause galvanometric voltage drop between the two metals as shown in Fig. 5.3.

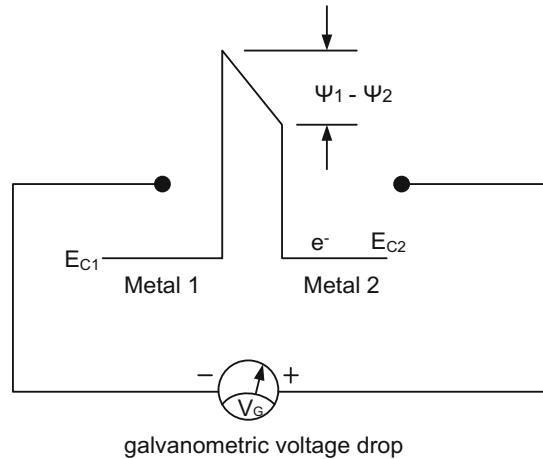
The work function difference is recorded as a function of temperature for the two metals used in the thermocouple. Therefore, when a certain voltage is read between the two thermocouple terminals, the corresponding reading identifies the ambient temperature.



**Fig. 5.1** Metal work functions to create a thermocouple

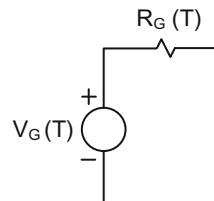


**Fig. 5.2** Transfer of electrons due to the conduction energy difference between metals



**Fig. 5.3** Galvanometric voltage drop after equilibrium is reached

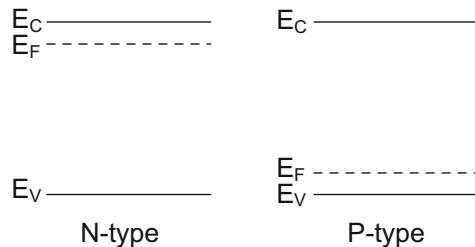
The equivalent circuit of the thermocouple is indicated in Fig. 5.4 where the equivalent resistance is in milliohms.  $V_G(T)$  corresponds to the difference in work functions as a function of temperature and it is usually in millivolts range.



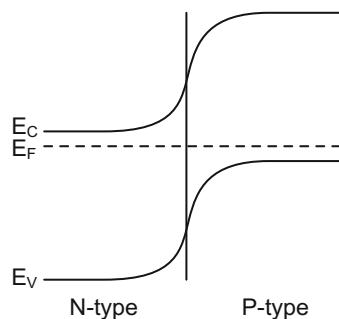
**Fig. 5.4** Equivalent circuit of the thermocouple

## 5.2 Photodiode

Photodiode is a two-terminal semiconductor device composed of N and P-type semiconductors much like a rectifying diode as shown in Fig. 5.5. Once a junction is formed, electrons flow from the N-type to the P-type semiconductor until the Fermi levels ( $E_F$ ) of both sides become equal. This makes the valance ( $E_V$ ) and conduction bands ( $E_C$ ) bend across the junction to form a basic diode as shown in Fig. 5.6.



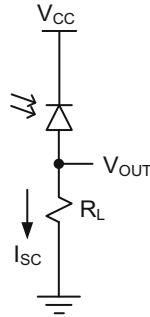
**Fig. 5.5** N and P-type semiconductors



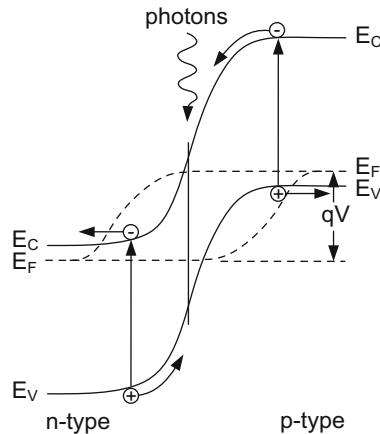
**Fig. 5.6** N and P-type semiconductors after forming the PN junction

This device is designed to operate with a reverse bias voltage applied across its terminals as shown in Fig. 5.7, and it produces a short circuit current,  $I_{SC}$ , composed of dark current,  $I_{DARK}$ , and photon-generated current,  $I_{PH}$ . The basic biasing and operation of the device are shown in Fig. 5.8. When the junction is reverse-biased with a voltage equal to  $V$ , the Fermi level on the P-side of the junction raises by an amount  $qV$ . This causes thermally generated minority carriers (holes in N-type semiconductor and electrons in P-type semiconductor) to be pushed towards the junction and combine there. The overall process forms a reverse saturation current across the junction. With the increase in temperature more electron-hole pairs are created, increasing the reverse saturation current. The same effect can occur by illuminating the junction with a light source. When light energy is exposed to the PN junction, more electrons in the valance band make the transition to the conduction band in

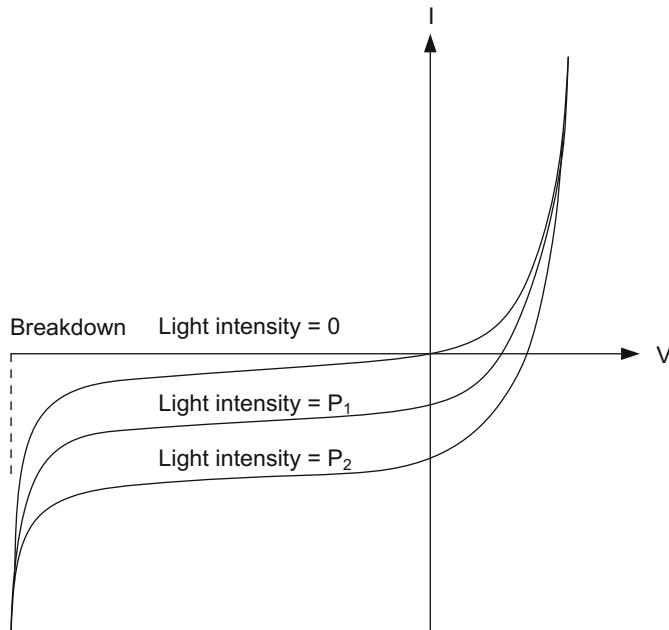
proportion to the light intensity, increasing the level of reverse saturation current as shown in Fig. 5.9. The dark current is essentially the reverse saturation current when no light is present, and equivalent to the minority carrier conduction when only thermal effects are present.



**Fig. 5.7** Photodiode biasing circuit



**Fig. 5.8** N and P-type semiconductors after reverse bias is applied to the PN junction

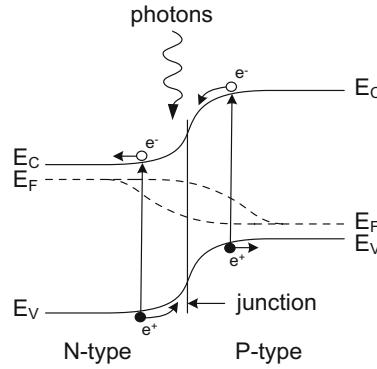


**Fig. 5.9** Photodiode I-V characteristics as a function of light intensity

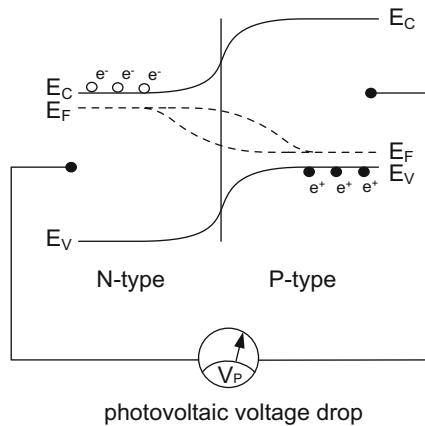
### 5.3 Solar Cell

Solar cell is not considered a sensor but it is another optical device that may be useful in an embedded system. The device requires no external voltage source and essentially a PN junction diode as shown in Fig. 5.6.

When the PN junction is illuminated with a light beam, electrons in the valance band transition to the conduction band and leave holes behind as shown in Fig. 5.10. The majority carriers (electrons in the N-type semiconductor and the holes in P-type semiconductor) have a very brief lifetime and recombine. The minority carriers, however, have a lot longer lifetime which enables them to move towards the junction because of the energy band bending in the band structure. When they arrive at the junction, they recombine. This process gives rise to a light-generated current and production of photovoltaic voltage as shown in Fig. 5.11.

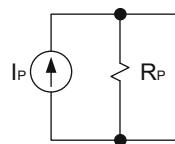


**Fig. 5.10** Light illumination at the PN junction and the resulting electron-hole pairs



**Fig. 5.11** Photovoltaic voltage drop between P and N regions as a result of light illumination

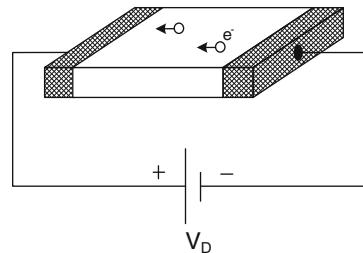
The resulting equivalent circuit of a photodiode exhibits constant current source characteristics,  $I_P$ , in proportion to the light intensity, and a large parallel resistance,  $R_P$ , as shown in Fig. 5.12.



**Fig. 5.12** Equivalent circuit of solar cell

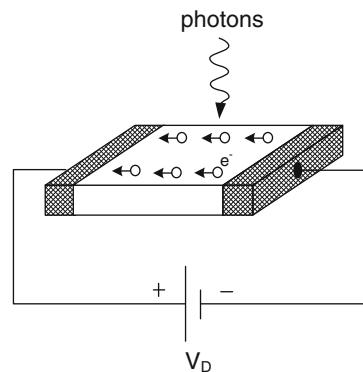
## 5.4 Photo-resistor

Detection of light can also be achieved using a photo-resistor as shown in Fig. 5.13. In normal conditions where light is not present, the photo-resistor slab shows low conductivity due to the lack of surface electrons.



**Fig. 5.13** Photo-detector made out of a photo-resistor

When a beam of light is exposed on a photo-resistive material shown in Fig. 5.14, photons are absorbed by the material, and electron-hole pairs are generated. Surface electrons (minority carriers) increase the conductivity of the material between the two terminals of the device, giving rise to a photonic drift current.

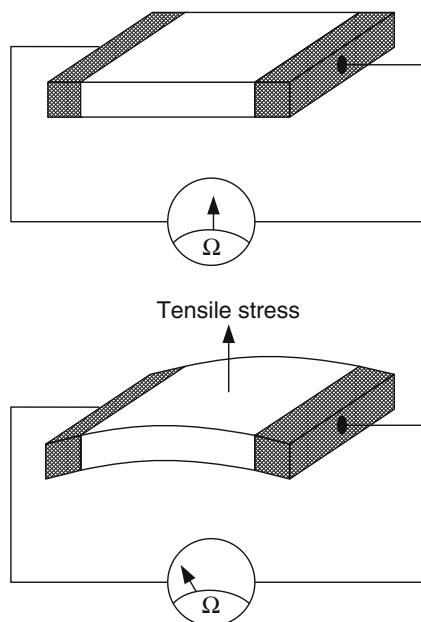


**Fig. 5.14** Photo-resistor under light produces more surface electrons

---

## 5.5 Piezoelectric Materials and Accelerometers

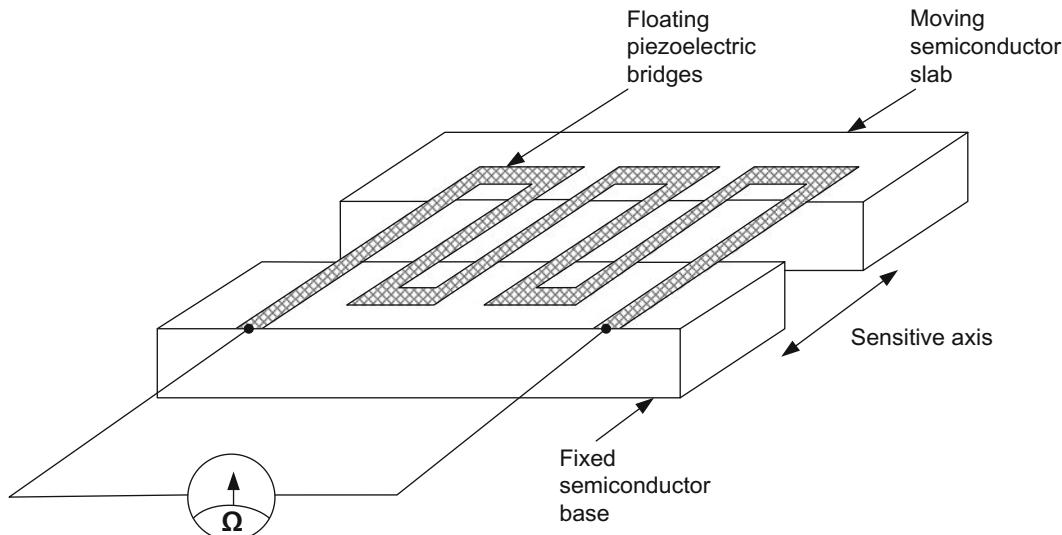
Accelerometers measure mass movement in specific directions. Figure 5.15 describes the use of piezoelectric resistor to measure the mass movement. If one end of the piezoelectric material is kept fixed while the other end is attached to a moving mass which moves freely, the tensile stress will bend the material and induce a change in conductivity. When the difference in conductivity is measured, the applied stress and the amount of mass movement can be correlated and quantified.



**Fig. 5.15** Piezoelectric-resistor under no tensile stress (*top*) and under stress (*bottom*)

Figure 5.16 shows the true structure of the piezoelectric resistor where the piezoelectric bridges in serpentine form are laid on a fixed and floating semiconductor slabs. As the sensitive axis moves because of the movement in semiconductor slab, the change in the piezoelectric bridge conductivity is recorded as shown in the figure. Since this measurement is directly proportional to the motion of the slab, it produces how much and how fast the slab moves along the sensitive axis.

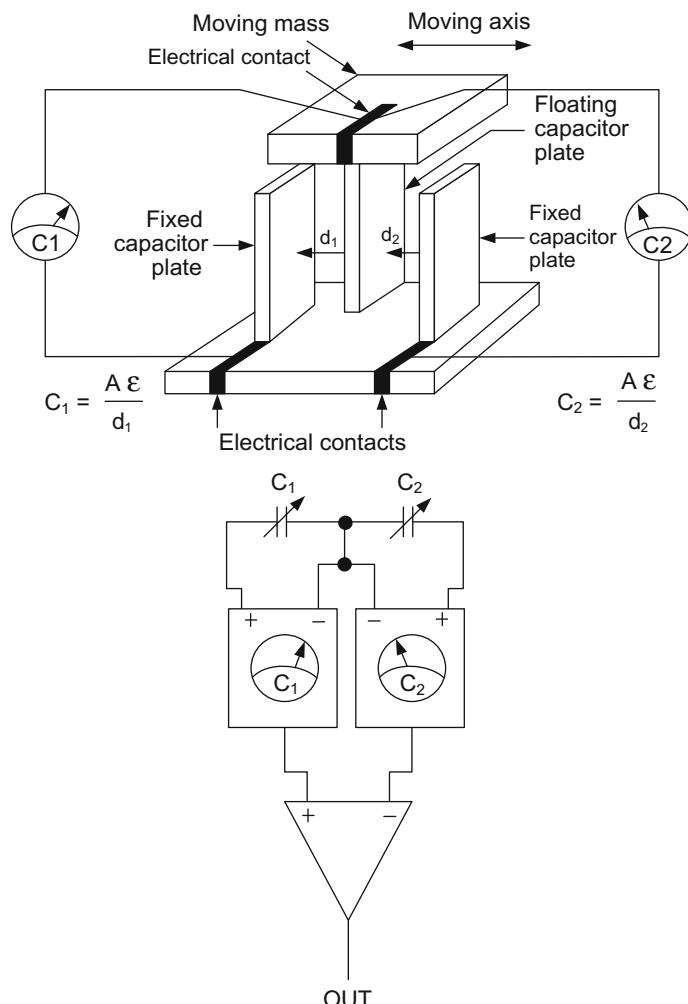
The same structure in Fig. 5.16 can be laid out to measure the movement in three different axes and provide valuable information for systems detecting the stability of a structure.



**Fig. 5.16** Piezoelectric-resistor on moving semiconductor slab under stress

Accelerometers can also be based on differential capacitance measurements. Such an accelerometer is shown by the top figure in Fig. 5.17. This time the moving mass floats between two fixed plates as shown in the figure, and allowed to move in one direction. As the slab movement takes place, the gap between the first pair of plates decreases in relation to the other. Since capacitance is inversely proportional to the distance between the plates, the change in capacitance produces how much and how fast the floating mass moves. Here,  $d_1$  and  $d_2$  correspond to the distance between the first and second parallel plates. A is the area of each plate, and  $\epsilon$  is the dielectric constant.

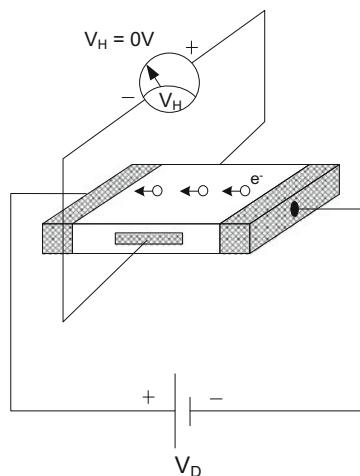
To increase the accuracy of the slab's movement further, the capacitance values,  $C_1$  and  $C_2$ , are measured and amplified simultaneously by a differential amplifier as shown by the bottom figure in Fig. 5.17.



**Fig. 5.17** Movement detection by differential capacitance measurement method

## 5.6 Hall-Effect Devices

Hall-Effect devices operate with the presence of a magnetic field and form a very useful device for embedded systems. The basic idea behind the Hall-effect device is the formation of Hall-voltage when the device is exposed to a magnetic field. When there is no magnetic field, electrons drift according to the direction of the electric field produced by  $V_D$  as shown in Fig. 5.18. The lateral voltage,  $V_H$ , measured between the two ends of the semiconductor will be 0 V.

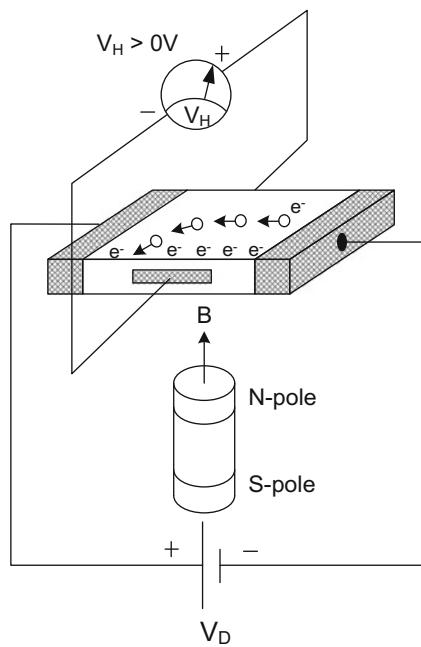


**Fig. 5.18** Hall-effect device with no magnetic field ( $V_H = 0$  V)

However, when a magnetic field,  $B$ , is applied perpendicular to the semiconductor surface as shown in Fig. 5.19, the conducting electrons are diverted from their original path by a lateral force field and accumulate on one side of the semiconductor slab according to the equation below.

$$F = -q(\vec{v} \times \vec{B})$$

Here,  $F$  is the magnetic force exerted on the electron,  $q$  is the electronic charge and  $v$  is the velocity of the electron. As a result of this accumulation,  $V_H$  forms between the two ends of the slab perpendicular to the electric field. This voltage is called Hall voltage and increases with the applied magnetic field.

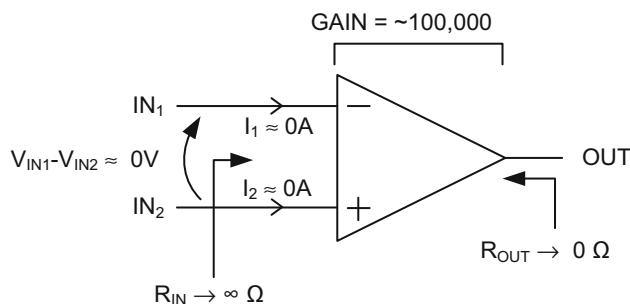


**Fig. 5.19** Hall-effect device with magnetic field ( $V_H > 0$  V)

## 6.1 Operational Amplifier Properties

Operational amplifiers are general purpose, dual input, single output devices that exhibit high input impedances, low output impedances and generate large gains within its frequency bandwidth.

Important characteristics of an operational amplifier are shown in Fig. 6.1. The input currents,  $I_1$  and  $I_2$ , of this amplifier are very close to 0 A because of the high impedance at negative and positive input terminals. A milli-volt range voltage difference between the input terminals, ( $V_{IN1} - V_{IN2}$ ), can saturate the output of the operational amplifier to either its positive or negative power supply voltage value depending on the polarity of ( $V_{IN1} - V_{IN2}$ ). However, we can approximate this potential difference to be 0 V for the purposes of simple circuit analysis.

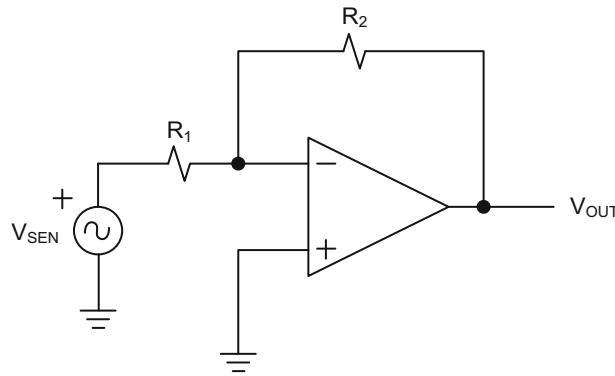


**Fig. 6.1** Characteristics of an operational amplifier

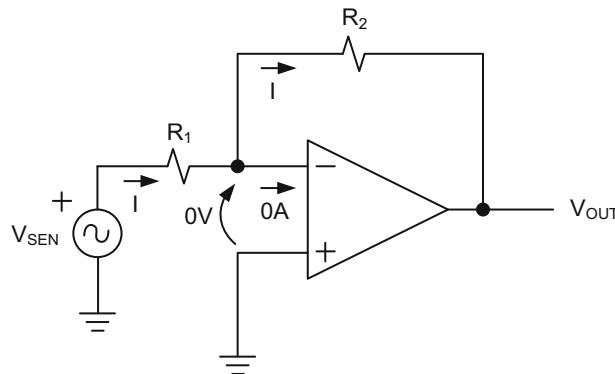
## 6.2 Voltage Amplifier Circuits for Sensors

### Inverting Voltage Sensor Amplifier

The following operational amplifier circuit in Fig. 6.2 produces voltage amplification, which results an inverted output, if one of the sensor terminals needs to be grounded. Figure 6.3 shows the current and voltage assignments of the circuit in Fig. 6.2 prior to circuit analysis.



**Fig. 6.2** Inverting voltage sensor amplification circuit if one sensor terminal needs to be grounded



**Fig. 6.3** Current and voltage assignments of the inverting sensor amplifier in Fig. 6.2

$$V_{SEN} = R_1 I \quad (6.1)$$

$$V_{SEN} = (R_1 + R_2) I + V_{OUT} \quad (6.2)$$

Substituting  $I = \frac{V_{SEN}}{R_1}$  into Eq. 6.2 yields:

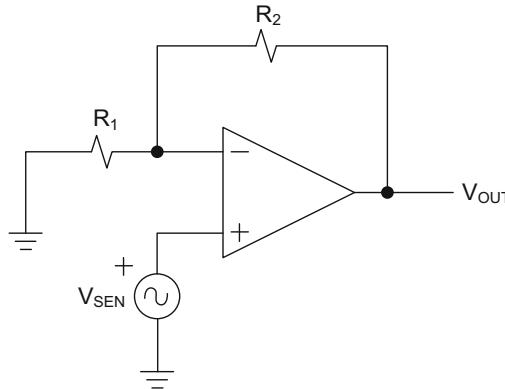
$$V_{SEN} = (R_1 + R_2) \frac{V_{SEN}}{R_1} + V_{OUT}$$

$$V_{SEN} = V_{SEN} + \frac{R_2}{R_1} V_{SEN} + V_{OUT}$$

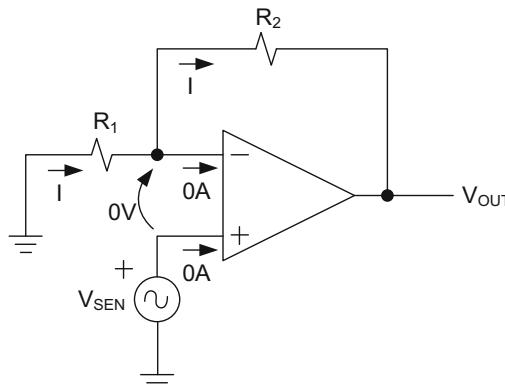
$$V_{OUT} = -\frac{R_2}{R_1} V_{SEN} \quad (6.3)$$

### Non-inverting Voltage Sensor Amplifier

The operational amplifier circuit in Fig. 6.4 produces voltage amplification with a non-inverted output if one sensor terminal needs to be grounded. Figure 6.5 shows the current and voltage assignments of the circuit in Fig. 6.4 prior to circuit analysis.



**Fig. 6.4** Non-inverting voltage sensor amplification circuit if one of the sensor terminals needs to be grounded



**Fig. 6.5** Current and voltage assignments of the inverting sensor amplifier in Fig. 6.4

Therefore,

$$0 = R_1 I + V_{SEN} \quad (6.4)$$

$$0 = (R_1 + R_2)I + V_{OUT} \quad (6.5)$$

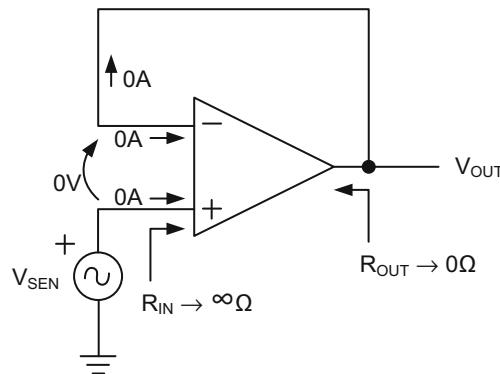
Substituting  $I = -\frac{V_{SEN}}{R_1}$  into Eq. 6.5 yields:

$$0 = -(R_1 + R_2) \frac{V_{SEN}}{R_1} + V_{OUT}$$

$$V_{OUT} = \frac{R_1 + R_2}{R_1} V_{SEN} \quad (6.6)$$

### Isolation Circuit with Unity Gain

Figure 6.6 represents the isolation circuit if the sensor requires very large input impedance before being interfaced to another circuitry. This circuit practically eliminates any output loading (isolation) on the sensor. One terminal of the sensor has to be grounded in order to use the isolation circuit.



**Fig. 6.6** Isolation circuit with unity gain

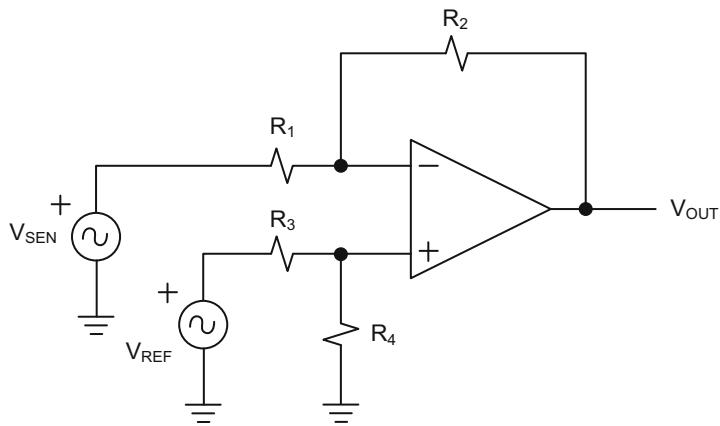
The only equation available for this circuit is:

$$V_{SEN} = 0 + V_{OUT} \quad (6.7)$$

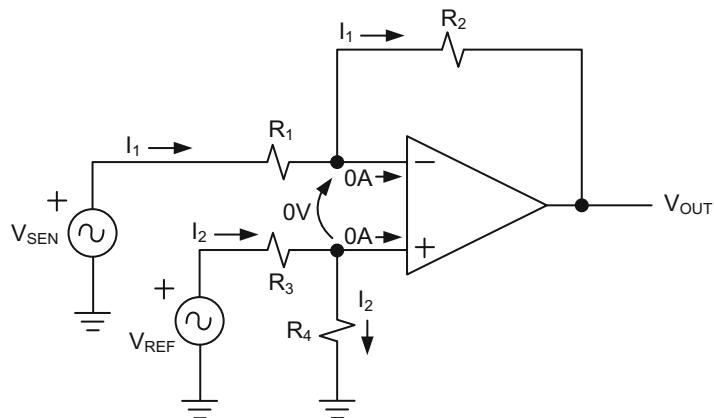
$$V_{OUT} = V_{SEN} \quad (6.8)$$

### Voltage Sensor Amplifier with Signal and Reference Inputs

If the sensor output needs to be compared against a reference voltage, a differential amplifier circuit in Fig. 6.7 is used. This circuit produces a differential amplification of the input signals,  $(V_{SEN} - V_{REF})$ , provided that one of the sensor (and the reference signal) terminals is connected to ground. Figure 6.8 shows the current and voltage assignments of the circuit in Fig. 6.7 prior to circuit analysis.



**Fig. 6.7** Voltage sensor amplification with a reference voltage if one sensor terminal is grounded



**Fig. 6.8** Current and voltage assignments of the sensor amplifier in Fig. 6.7

$$V_{SEN} = (R_1 + R_2)I_1 + V_{OUT} \quad (6.9)$$

$$V_{REF} = (R_3 + R_4) I_2 \quad (6.10)$$

$$R_4 I_2 = R_2 I_1 + V_{OUT} \quad (6.11)$$

From Eq. 6.11,

$$I_2 = \frac{R_2}{R_4} I_1 + \frac{V_{OUT}}{R_4} \quad (6.12)$$

Substituting Eq. 6.12 into Eq. 6.10 yields:

$$V_{REF} = (R_3 + R_4) \frac{R_2}{R_4} I_1 + (R_3 + R_4) \frac{V_{OUT}}{R_4} \quad (6.13)$$

or

$$I_1 = \frac{R_4 V_{REF} - (R_3 + R_4) V_{OUT}}{R_2(R_3 + R_4)} \quad (6.14)$$

Finally, substituting Eq. 6.14 into Eq. 6.9 yields:

$$V_{SEN} = (R_1 + R_2) \frac{R_4 V_{REF} - (R_3 + R_4) V_{OUT}}{R_2(R_3 + R_4)} + V_{OUT} \quad (6.15)$$

Reorganizing the terms in this equation produces:

$$V_{SEN} = \frac{R_4 (R_1 + R_2)}{R_2 (R_3 + R_4)} V_{REF} - \frac{R_1}{R_2} V_{OUT}$$

Thus,

$$V_{OUT} = -\frac{R_2}{R_1} V_{SEN} + \frac{R_4 (R_1 + R_2)}{R_1 (R_3 + R_4)} V_{REF} \quad (6.16)$$

If  $R_1 = R_3$  and  $R_2 = R_4$ , then Eq. 6.16 simplifies as:

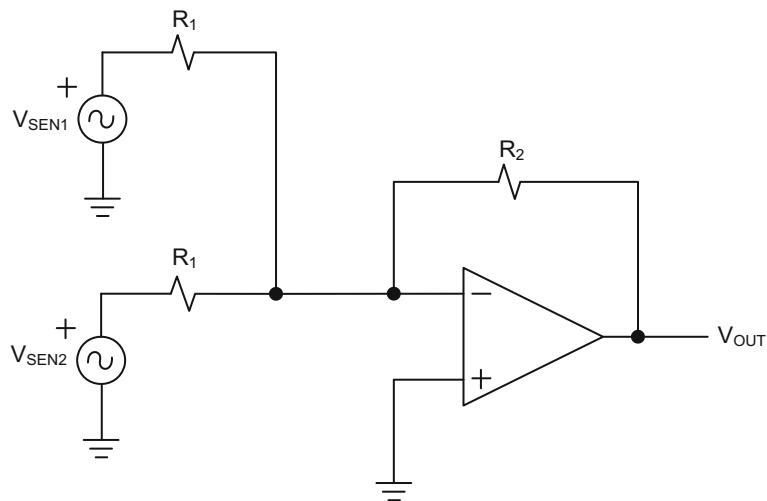
$$V_{OUT} = -\frac{R_2}{R_1} V_{SEN} + \frac{R_2 (R_1 + R_2)}{R_1 (R_1 + R_2)} V_{REF}$$

$$V_{OUT} = -\frac{R_2}{R_1} (V_{SEN} - V_{REF}) \quad (6.17)$$

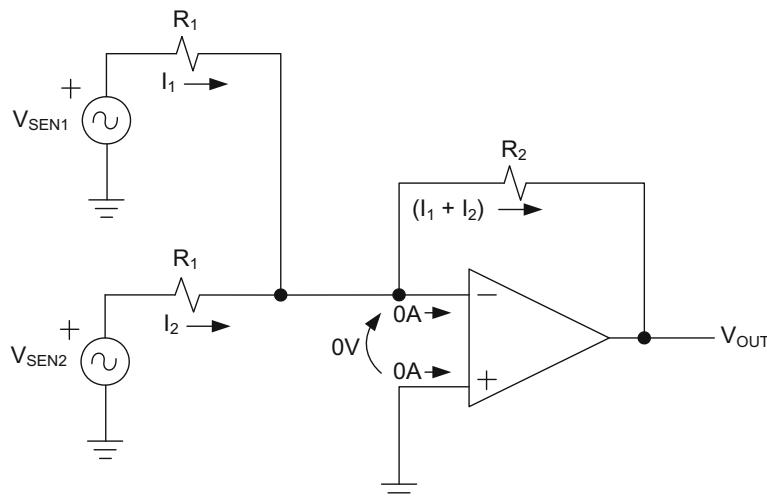
In Eq. 6.17, the amplification is adjusted by the ratio of  $\frac{R_2}{R_1}$ .

### Summation Voltage Sensor Amplifier

It is possible to add two sensor inputs according to Fig. 6.9 provided that both sensors have one of their terminals grounded. Figure 6.10 shows the current and voltage assignments of the circuit in Fig. 6.9 prior to circuit analysis.



**Fig. 6.9** Summation voltage sensor amplifier if one of the sensor terminals is grounded



**Fig. 6.10** Current and voltage assignments of the summation sensor amplifier in Fig. 6.9

$$V_{SEN1} = R_1 I_1 + R_2(I_1 + I_2) + V_{OUT} \quad (6.18)$$

$$V_{SEN2} = R_1 I_2 + R_2(I_1 + I_2) + V_{OUT} \quad (6.19)$$

$$R_2(I_1 + I_2) + V_{OUT} = 0 \quad (6.20)$$

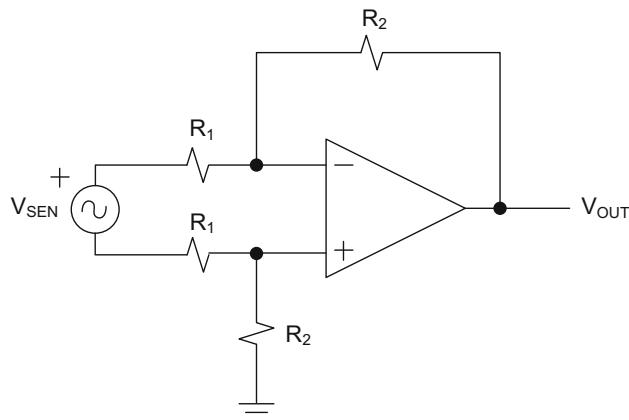
Eliminating  $I_1$  and  $I_2$  in Eqs. 6.18, 6.19 and 6.20 results in:

$$V_{\text{OUT}} = -\frac{R_2}{R_1}(V_{\text{SEN}1} + V_{\text{SEN}2}) \quad (6.21)$$

The output voltage in Eq. 6.21 forms the inverted sum of the sensor voltages,  $V_{\text{SEN}1}$  and  $V_{\text{SEN}2}$ , with a amplification factor of  $\frac{R_2}{R_1}$ .

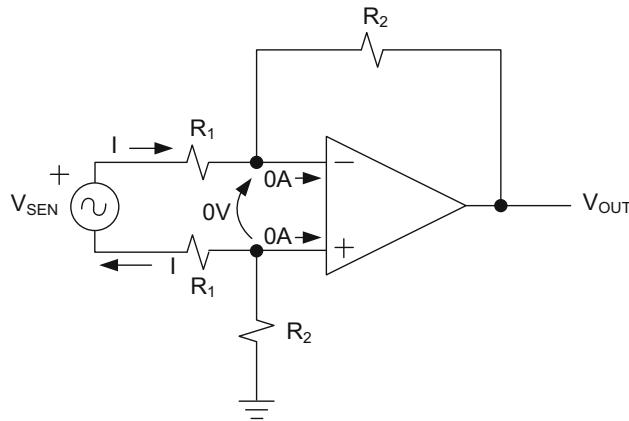
### **Voltage Sensor Amplifier with Ungrounded Sensor Terminals**

If neither of the sensor terminals can be grounded, then the circuit in Fig. 6.11 is used to obtain an amplified but an inverted sensor signal at the output. This schematic is similar to the differential amplifier schematic in Fig. 6.7; however, the sensor in this circuit is not grounded. The resistors,  $R_3$  and  $R_4$ , are also replaced with the resistors,  $R_1$  and  $R_2$ , respectively.



**Fig. 6.11** Voltage amplification of a sensor if no sensor terminal is grounded

Figure 6.12 shows the current and voltage assignments of the circuit in Fig. 6.11 prior to circuit analysis. In this circuit, the voltage drop at the input of the operational amplifier is assumed 0 V. Similarly, the currents going to the positive and negative input terminals of the operational amplifier are assumed to be very close to 0 A.



**Fig. 6.12** Current and voltage assignments of the sensor amplifier in Fig. 6.11

Therefore, from Fig. 6.12 we have:

$$I(R_1 + R_2) - V_{SEN} + I(R_1 + R_2) + V_{OUT} = 0$$

or

$$V_{SEN} - V_{OUT} = 2I(R_1 + R_2) \quad (6.22)$$

We also have:

$$V_{SEN} = I(R_1 + R_1) = 2I R_1$$

$$I = \frac{V_{SEN}}{2R_1} \quad (6.23)$$

Substituting Eq. 6.23 into Eq. 6.22 yields:

$$V_{SEN} - V_{OUT} = 2(R_1 + R_2) \frac{V_{SEN}}{2R_1}$$

or

$$V_{OUT} = -\frac{R_2}{R_1} V_{SEN} \quad (6.24)$$

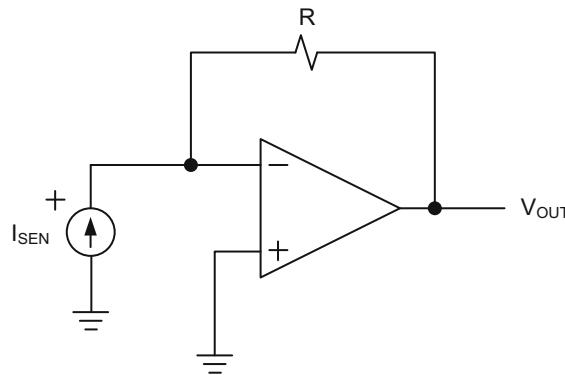
Equation 6.24 shows the sensor voltage at the input of Fig. 6.11 is inverted and amplified with a ratio of  $\frac{R_2}{R_1}$ .

### 6.3 Trans-resistance Amplifier Circuits for Sensors

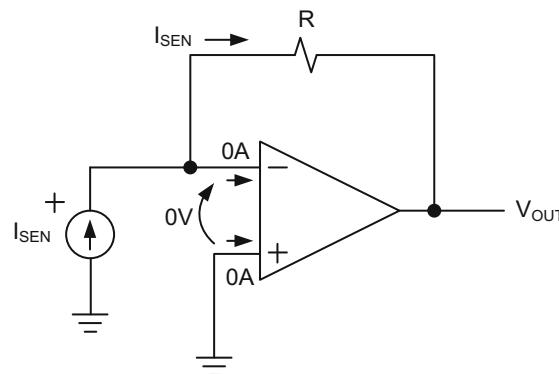
Trans-resistance amplifiers are for sensors that generate current instead of voltage. These sensors are usually opto-electronic devices such photodiodes, photo-detectors and solar cells. These specialty amplifiers are designed to convert microampere-range currents into voltages.

#### Inverting Trans-resistance Sensor Amplifier

The operational amplifier circuit in Fig. 6.13 produces inverted sensor amplification if one sensor terminal is grounded. Figure 6.14 shows the current and voltage assignments of the circuit in Fig. 6.13 prior to circuit analysis.



**Fig. 6.13** Inverting trans-resistance sensor amplifier if one sensor terminal is grounded



**Fig. 6.14** Current and voltage assignments of the trans-resistance amplifier in Fig. 6.13

The Kirchoff's voltage law for this circuit becomes:

$$0 = R I_{SEN} + V_{OUT} \quad (6.25)$$

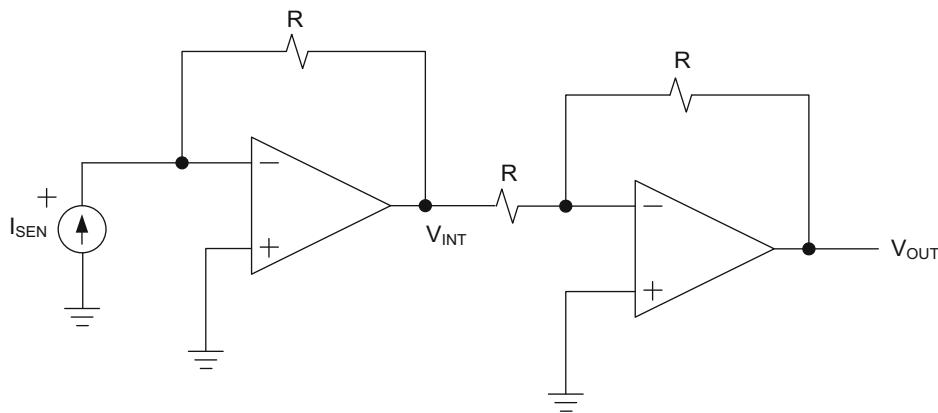
Therefore,

$$V_{\text{OUT}} = -R I_{\text{SEN}} \quad (6.26)$$

Here, the input sensor current,  $I_{\text{SEN}}$ , is inverted and amplified by the magnitude of  $R$  at the output.

### Non-inverting Trans-resistance Sensor Amplifier

The following operational amplifier circuit in Fig. 6.15 produces non-inverted sensor amplification if one sensor terminal is grounded. This circuit basically combines the inverting trans-resistance amplifier in Fig. 6.13 and the inverting voltage amplifier in Fig. 6.2 with unity gain.



**Fig. 6.15** Non-inverting trans-resistance sensor amplifier if one sensor terminal is grounded

Repeating Eq. 6.26 for the first stage yields:

$$V_{\text{INT}} = -R I_{\text{SEN}}$$

For the second stage, we have:

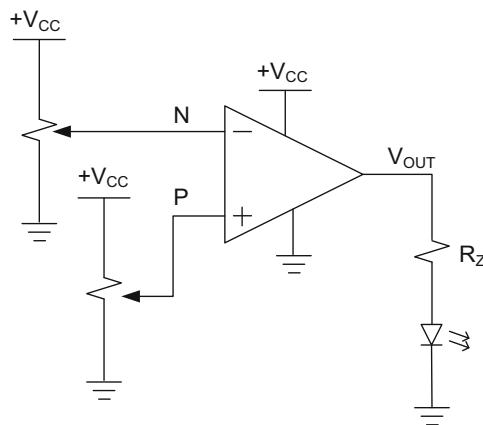
$$V_{\text{OUT}} = -\frac{R}{R} V_{\text{INT}} = -V_{\text{INT}} \quad (6.27)$$

Substituting  $V_{\text{INT}}$  in Eq. 6.27 yields:

$$V_{\text{OUT}} = -(-R I_{\text{SEN}}) = R I_{\text{SEN}} \quad (6.28)$$

## 6.4 Analog Voltage Comparator

Voltage comparators are used to compare two analog inputs to produce an output signal either equal to the positive voltage supply of the operational amplifier or to 0 V (the negative voltage supply can be 0 V or a negative value depending on the operational amplifier datasheet). Figure 6.16 shows the circuit diagram of a comparator. Here, voltage levels at the N and P inputs are compared with each other. If the voltage at P becomes larger than N,  $V_{OUT}$  transitions to  $V_{CC}$ , and the LED turns on. Otherwise,  $V_{OUT}$  stays at 0 V, and the LED turns off.



**Fig. 6.16** Basic comparator circuit

A good application of a voltage comparator is a voltage monitoring circuit. Suppose we want to monitor the changes in the 2.5 V CPU power supply voltage constantly, and generate a reset signal if the supply voltage drops below 2 V.

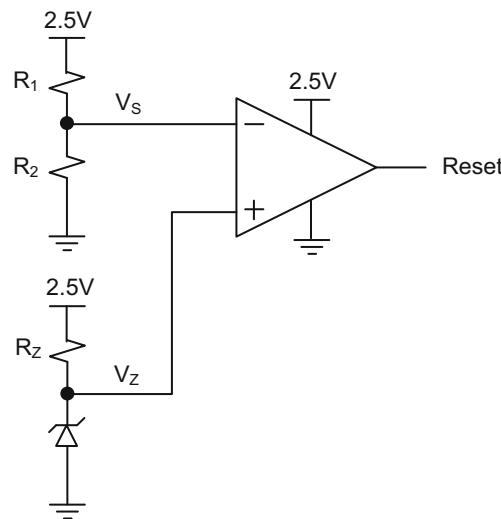
A circuit shown in Fig. 6.17 can be used as a voltage monitor. In this circuit, a Zener diode with a value of 1.25 V is connected in series with  $R_Z$ , which provides the biasing current for the diode. The voltage at the negative terminal of the operational amplifier must also be set at 1.25 V by  $R_1 = R_2$ . Thus,

$$V_S = \frac{R_2}{(R_1 + R_2)} V_{CC} = \frac{V_{CC}}{2} \quad (6.29)$$

Now, if  $V_{CC}$  drops below 2.5 V, such as 2 V,  $V_S$  proportionally changes:

$$V_S = \frac{V_{CC}}{2} = \frac{2}{2} = 1V$$

Since this voltage is less than  $V_Z = 1.25$  V, Reset output in Fig. 6.17 switches to 2.5 V and resets the CPU.



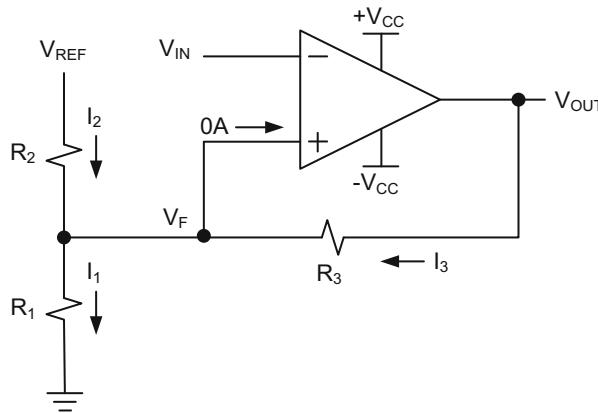
**Fig. 6.17** A voltage monitor circuit

## 6.5 Schmitt Trigger

Most TTL and CMOS logic gates require fast high and low logic transitions at their inputs. If the edges of the input signal are too slow to rise or fall, excessive current drainage, even uncontrolled oscillations at the output of a logic gate become possible. Schmitt triggers are used to remove slow transitions or noisy edges from signals and transform them into forms that will meet the input rise and fall time specifications of a logic gate.

A normal Schmitt trigger produces a low-to-high logic transition at a logic level much higher than the threshold voltage of a gate. Similarly, a high-to-low transition in the Schmitt trigger occurs at a logic level much lower than the threshold voltage of a gate. This creates hysteresis at the output of the Schmitt trigger despite noise-free and sharp logic transitions.

The basic Schmitt trigger circuit is shown in Fig. 6.18. In this figure, the output of the device transitions from one supply voltage ( $+V_{CC}$  or  $-V_{CC}$ ) to the next as soon as the input voltage exceeds a set voltage limit. Schmitt trigger operates under the same principle of the voltage comparator. However, it compares the input voltage against two set values instead of one, and its feedback loop ensures that a portion of the output value is always taken into account when the comparison is performed at the input.



**Fig. 6.18** Basic Schmitt trigger

We can adjust and set the voltage values in a Schmitt trigger by adjusting the values of  $R_1$ ,  $R_2$  and  $R_3$ .

If  $I_1$ ,  $I_2$  and  $I_3$  are the currents passing through  $R_1$ ,  $R_2$  and  $R_3$  in Fig. 6.18, then we can write the following:

$$I_1 = I_2 + I_3 \quad (6.30)$$

However,

$$I_1 = \frac{V_F}{R_1} \quad (6.31)$$

$$I_2 = \frac{(V_{REF} - V_F)}{R_2} \quad (6.32)$$

$$I_3 = \frac{(V_{OUT} - V_F)}{R_3} \quad (6.33)$$

Substituting Eqs. 6.31, 6.32 and Eq. 6.33 into Eq. 6.30 yields:

$$\frac{V_F}{R_1} = \frac{(V_{REF} - V_F)}{R_2} + \frac{(V_{OUT} - V_F)}{R_3}$$

$$V_F \left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} \right) = \frac{V_{REF}}{R_2} + \frac{V_{OUT}}{R_3}$$

If we define  $\frac{1}{R_{123}} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3}$ , then

$$\frac{V_F}{R_{123}} = \frac{V_{REF}}{R_2} + \frac{V_{OUT}}{R_3}$$

$$V_F = \frac{R_{123}}{R_2} V_{REF} + \frac{R_{123}}{R_3} V_{OUT} \quad (6.34)$$

If  $V_{OUT} = +V_{CC}$  then  $V_F$  becomes:

$$V_{FPOS} = \frac{R_{123}}{R_2} V_{REF} + \frac{R_{123}}{R_3} V_{CC} \quad (6.35)$$

Similarly,

If  $V_{OUT} = -V_{CC}$  then  $V_F$  becomes:

$$V_{FNNEG} = \frac{R_{123}}{R_2} V_{REF} - \frac{R_{123}}{R_3} V_{CC} \quad (6.36)$$

Here,  $V_{FPOS}$  and  $V_{FNNEG}$  are the two feedback voltages at the output terminal when  $V_{OUT} = +V_{CC}$  and  $V_{OUT} = -V_{CC}$ , respectively.

In this example, let us set the voltage level for the low-to-high logic transition to 3.3 V, and for the high-to-low logic transition to 0 V for an input voltage that swings between +5 V and -5 V. Therefore, we choose:

$$R_1 = R_2 = R_3 = 3K\Omega$$

$$V_{REF} = 5V, +V_{CC} = 5V \text{ and } -V_{CC} = -5V$$

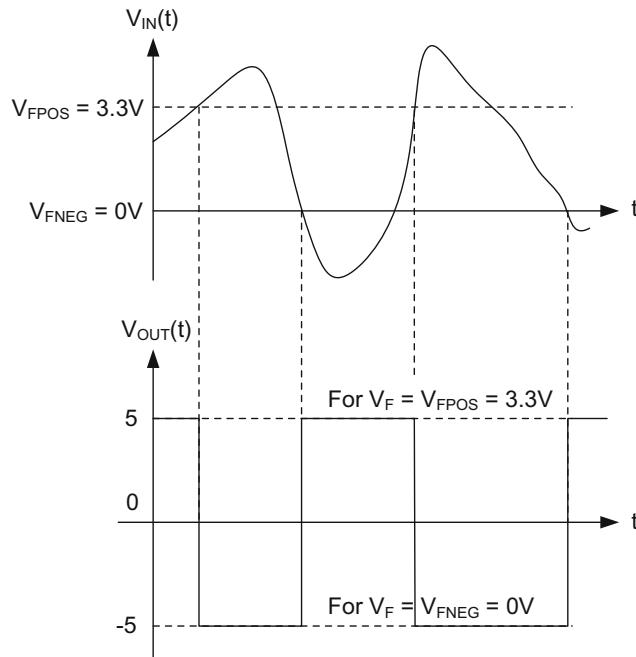
Then:

$$R_{123} = 1K\Omega$$

$$V_{FPOS} = \left( \frac{R_{123}}{R_2} \right) V_{REF} + \left( \frac{R_{123}}{R_3} \right) V_{CC} = \left( \frac{1}{3} \right) 5 + \left( \frac{1}{3} \right) 5 = \frac{10}{3} = 3.3V$$

$$V_{FNNEG} = \left( \frac{R_{123}}{R_2} \right) V_{REF} - \left( \frac{R_{123}}{R_3} \right) V_{CC} = \left( \frac{1}{3} \right) 5 - \left( \frac{1}{3} \right) 5 = 0V$$

The operation and the output waveform generation of the Schmitt trigger are explained in Fig. 6.19 according to these numerical values. If  $V_{IN}$  is less than the first set value,  $V_{FPOS} = 3.3$  V, then  $V_{OUT}$  transitions to  $+V_{CC} = 5$  V. For input voltages greater than 3.3 V,  $V_{OUT}$  transitions to  $-V_{CC} = -5$  V, and  $V_F$  becomes  $V_{FNNEG} = 0$  V. Here,  $V_{OUT}$  stays at -5 V until  $V_{IN}$  drops below the second set value, 0 V. As soon as  $V_{IN}$  is less than 0 V,



**Fig. 6.19** Schmitt trigger input and output waveforms

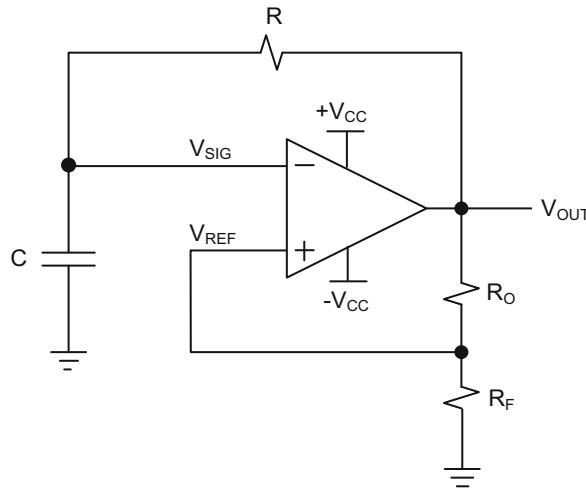
$V_{OUT}$  transitions back to  $+V_{CC} = 5$  V, and  $V_F$  becomes  $V_{FPOS} = 3.3$  V. The next output change to  $-5$  V does not take place until  $V_{IN}$  reaches above  $3.3$  V.

## 6.6 Square Waveform Generator

Operational amplifiers are excellent candidates to generate square waveforms with varying duty cycles. The circuit in Fig. 6.20 generates a square waveform at  $V_{OUT}$  based on an RC feedback circuit.

Suppose the operational amplifier has dual power supplies at  $+V_{CC}$  and  $-V_{CC}$ . Assume that the initial value of  $V_{OUT} = +V_{CC}$  and  $V_{SIG} = -V_{CC}$  since the output changes between  $+V_{CC}$  and  $-V_{CC}$ . When  $V_{OUT} = +V_{CC}$ ,  $V_{REF}$  becomes:

$$V_{REF} = V_{OUT} \frac{R_F}{R_F + R_O} = V_{CC} \frac{R_F}{R_F + R_O} \quad (6.37)$$



**Fig. 6.20** Square wave generator circuit

When  $V_{SIG}$  departs from  $-V_{CC}$  and approaches towards  $+V_{CC}$  with a time constant of  $RC$ , it produces the following expression:

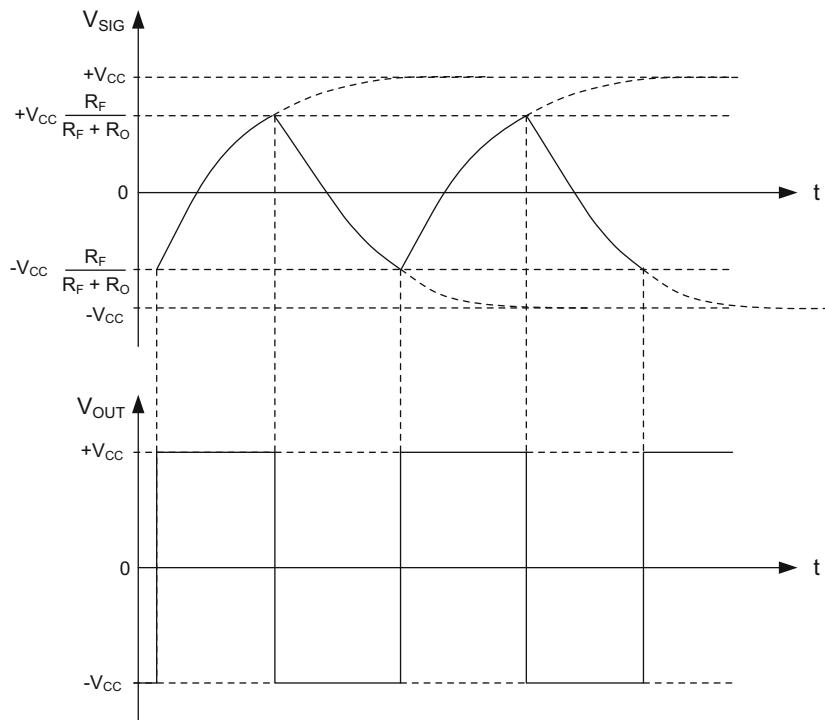
$$V_{SIG} = -V_{CC} + 2V_{CC} \left[ 1 - \exp\left(-\frac{t}{RC}\right) \right] \quad (6.38)$$

However, as soon as  $V_{SIG}$  climbs above  $V_{REF} = \frac{R_F}{R_F + R_O} V_{CC}$ ,  $V_{OUT}$  changes its polarity

and it becomes  $-V_{CC}$ . Now, the accumulated charge on the capacitor starts discharging through  $R$  and the operational amplifier's output impedance. As a result,  $V_{SIG}$  starts decreasing from  $\frac{R_F}{R_F + R_O} V_{CC}$  towards  $-V_{CC}$  with the same time constant,  $RC$ . Thus,

$$V_{SIG} = V_{CC} \left( 1 + \frac{R_F}{R_F + R_O} \right) \exp\left(-\frac{t}{RC}\right) - V_{CC} \quad (6.39)$$

This discharge cycle continues until  $V_{SIG}$  reaches a new  $V_{REF}$  value,  $-\frac{R_F}{R_F + R_O} V_{CC}$ , at which the output changes its polarity again to  $+V_{CC}$ . With  $V_{OUT} = +V_{CC}$ ,  $V_{REF}$  transitions back to  $\frac{R_F}{R_F + R_O} V_{CC}$ . Now,  $V_{SIG}$  starts increasing from  $-\frac{R_F}{R_F + R_O} V_{CC}$  towards  $+V_{CC}$ , but stops at  $\frac{R_F}{R_F + R_O} V_{CC}$  because the output transitions to  $-V_{CC}$  as mentioned previously. The waveforms at  $V_{SIG}$  and  $V_{OUT}$  are shown in Fig. 6.21.

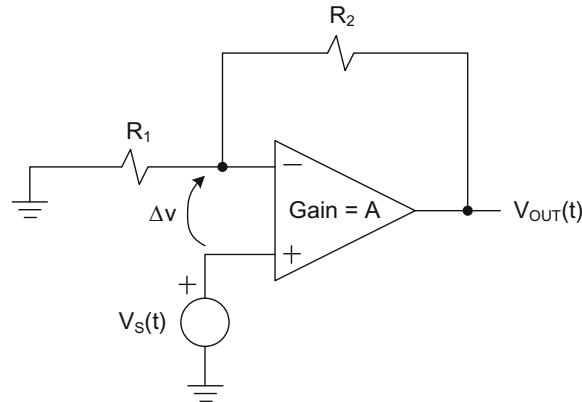


**Fig. 6.21**  $V_{SIG}$  and  $V_{OUT}$  waveforms of the square wave generator in Fig. 6.20

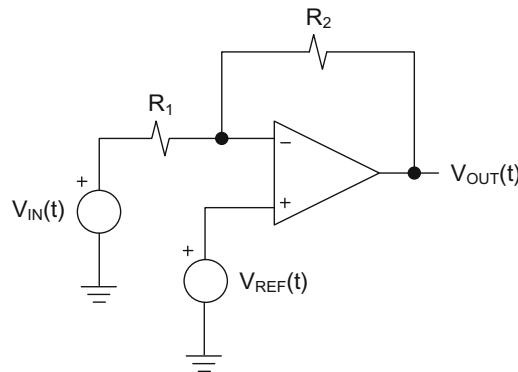
## Review Questions

1. Determine the amplification,  $V_{\text{OUT}}(t)/V_s(t)$ , in the circuit below.

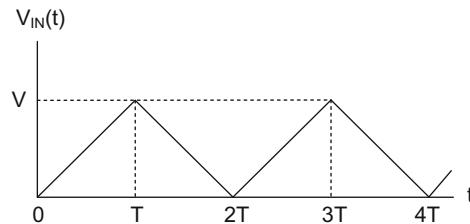
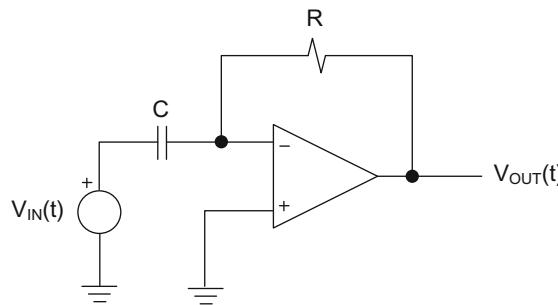
The operational amplifier is non-ideal, and there is a voltage drop between the negative and positive input terminals,  $\Delta v$ , as shown in the figure. The amplifier has also a limited gain,  $A$ , between its differential input,  $\Delta v$ , and its output,  $V_{\text{OUT}}(t)$ .



2. Assuming the operational amplifier is ideal in the circuit below, find the output voltage,  $V_{\text{OUT}}(t)$ , in terms of  $V_{\text{IN}}(t)$  and  $V_{\text{REF}}(t)$ .

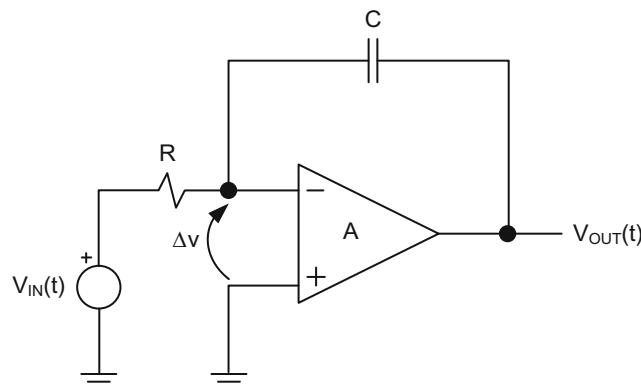


3. Assuming the operational amplifier in the circuit below is ideal, find the output voltage,  $V_{\text{OUT}}(t)$ , in terms of  $V_{\text{IN}}(t)$  using time-domain analysis. Plot the output voltage as a function of time if a square wave is applied at its input as shown below. The value of  $V$  in the input waveform is assumed to be smaller than the power supply voltage of the operational amplifier.

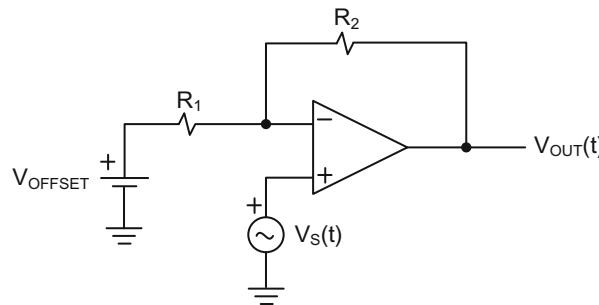


4. The following circuit shows a non-ideal operational amplifier where there is the voltage drop,  $\Delta v$ , between the positive and negative input terminals. Also, the amplifier exhibits a limited voltage gain,  $A$ , between its differential input and output. However, the input impedance,  $R_{IN}$ , at each input terminal is considerably high, approaching infinity. Thus, the current going into any input terminal is assumed to be 0 A.

- Find the overall voltage gain,  $V_{OUT}(t)/V_{IN}(t)$ , of the circuit.
- Now, assume the operational amplifier is ideal with  $\Delta v = 0$  V, and  $A$  approaching to infinity. Apply a small signal sinusoidal voltage to  $V_{IN}(t)$ . What value does  $V_{OUT}(t)/V_{IN}(t)$  approach as frequency increases? Compute the absolute value of  $V_{OUT}(t)/V_{IN}(t)$ .

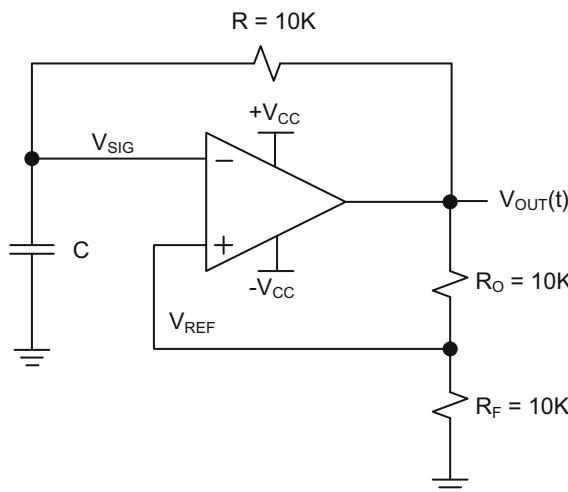


5. The following non-inverting operational amplifier circuit is designed to have a positive amplification of 10, and it is used to eliminate the DC component of the sensor signal. If  $V_O$  is the DC component and  $\Delta V_S$  is the time varying true component of the sensor signal,  $V_S = V_O + \Delta V_S$ , find the relative values of  $R_1$  and  $R_2$ , and compute the offset voltage,  $V_{OFFSET}$ , to eliminate  $V_O$ .



6. The following operational amplifier RC oscillator generates square waveform whose values change between  $+V_{CC}$  and  $-V_{CC}$ . Consider  $R_O = R_F = R = 10\text{ K}\Omega$  in this circuit.

- (a) Describe the operation of the circuit. First assume  $V_{OUT}$  is at  $+V_{CC}$ . Explain how the switching to  $-V_{CC}$  takes place at the output. Draw the waveform for  $V_{OUT}(t)$ ,  $V_{REF}(t)$  and  $V_{SIG}(t)$ .
- (b) Calculate the value of the capacitor, C, if the period of oscillation is 2  $\mu\text{s}$ . That means that  $V_{OUT}(t)$  is at  $+V_{CC}$  for a period of 1  $\mu\text{s}$  and at  $-V_{CC}$  for a period of 1  $\mu\text{s}$ .



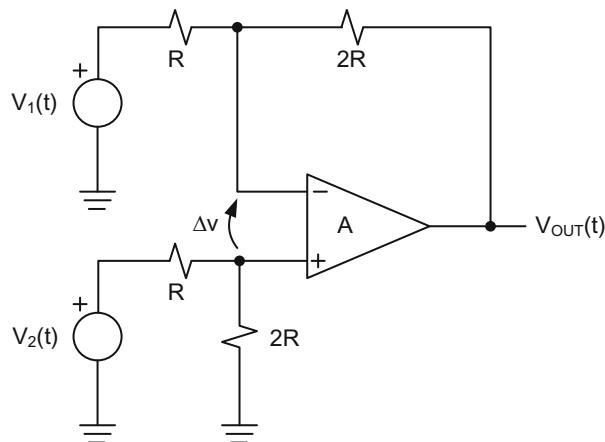
7. The following differential equation is given:

$$y(t) = A \frac{dx(t)}{dt} + Bx(t)$$

where A and B are constants,  $x(t)$  is a time varying voltage produced by a sensor, and  $y(t)$  is the analog output.

Generate  $y(t)$  using ideal operational amplifiers. Find the constants, A and B, in terms of the components used in the circuit.

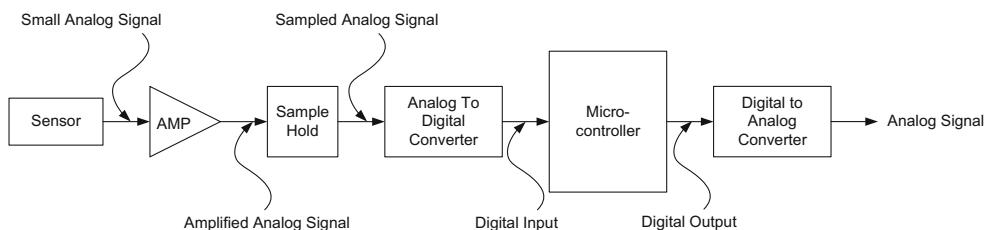
8. Assuming the operational amplifier in the circuit below is non-ideal, and there is a voltage drop,  $\Delta v$ , between the negative and positive inputs as shown in the figure. The amplification factor between the differential input and the output is A. Derive the equation for  $V_{OUT}(t)$  in terms of  $V_1(t) - V_2(t)$ .



# Data Converters

## 7.1 Analog-to-Digital Converter Principles

All analog domains interface with digital systems through Analog-to-Digital Converters (ADC). In Fig. 7.1, an analog signal from a sensor is amplified to a certain level before sampling takes place in a sample-and-hold circuit inside an ADC. The sampled analog signal is then converted into digital form and directed to the CPU for processing according to an embedded program.



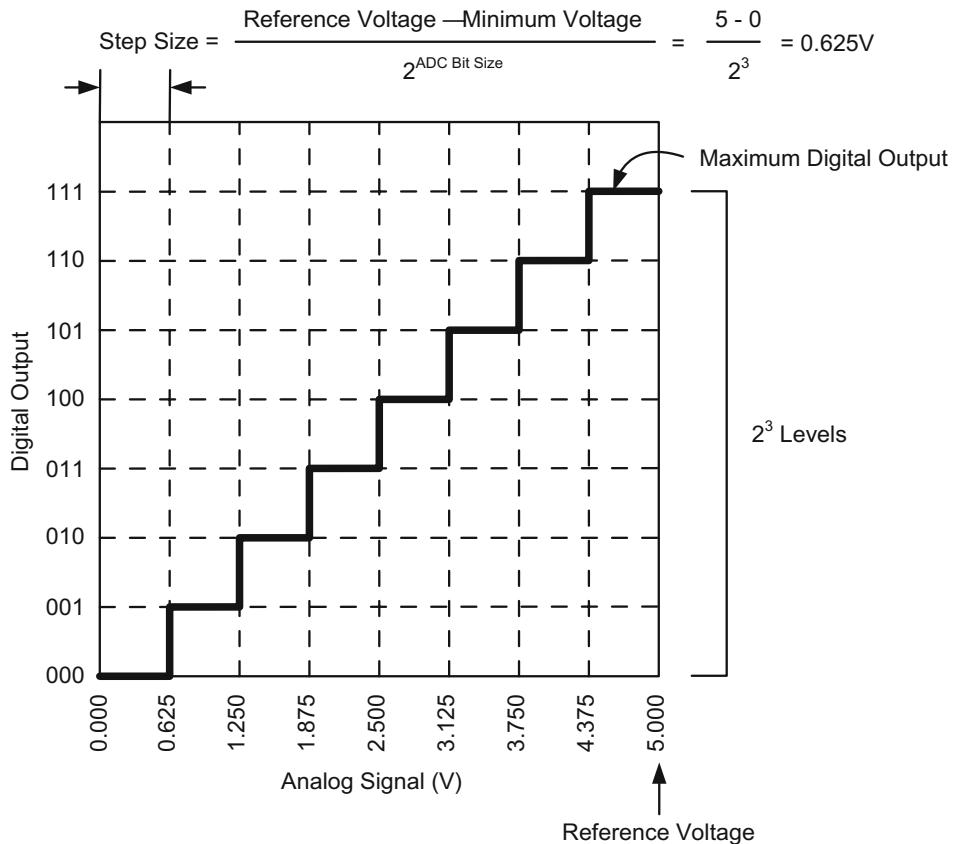
**Fig. 7.1** Typical Analog-to-Digital and Digital-to-Analog Converter data-paths

The signal resolution is an important factor to consider in an ADC design. It simply means dividing a sampled analog signal by  $2^N$  number of voltage levels to represent its value where N is the number of bits in digital domain. The second important consideration is the range of analog values an ADC can capture and process.

Figure 7.2 describes the ADC resolution in a numerical example where an analog signal changes between 0 and 5 V. The bit resolution is only three bits. Therefore, ADC divides the range of an analog signal into  $2^3 = 8$  levels between its maximum and minimum value, and identifies each analog level with three output bits. For example, an analog signal of 2.501 V is identified by a digital output of 100. If the analog signal increases to 3.124 V, the digital output that represents this voltage value still stays at 100. In other words, in a three-bit ADC there is no difference between 2.501 and 3.124 V in terms of their digital representation. The

0.625 V step size is the natural occurring error in a three-bit ADC, and it can be reduced only if the number of bits in the ADC is increased. In general, increasing the number of ADC bits by one halves the error. Therefore, designing a four-bit ADC instead of a three-bit ADC reduces the quantization error by 0.3125 V.

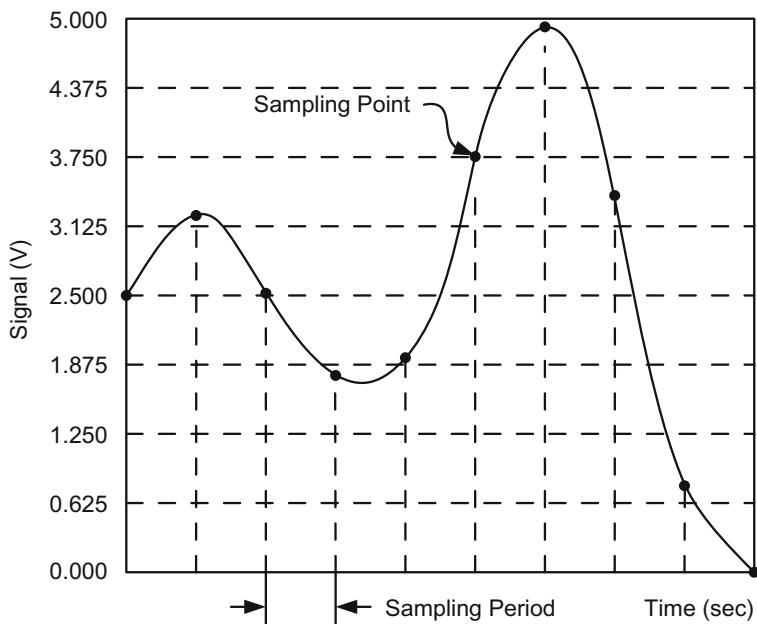
Reference voltage is generally determined by the maximum voltage level of the analog signal, and it is used to calculate the step size. In this three-bit ADC example in Fig. 7.2, the reference voltage is 5 V because the amplified analog voltage at the input of the ADC is limited not go beyond 5 V.



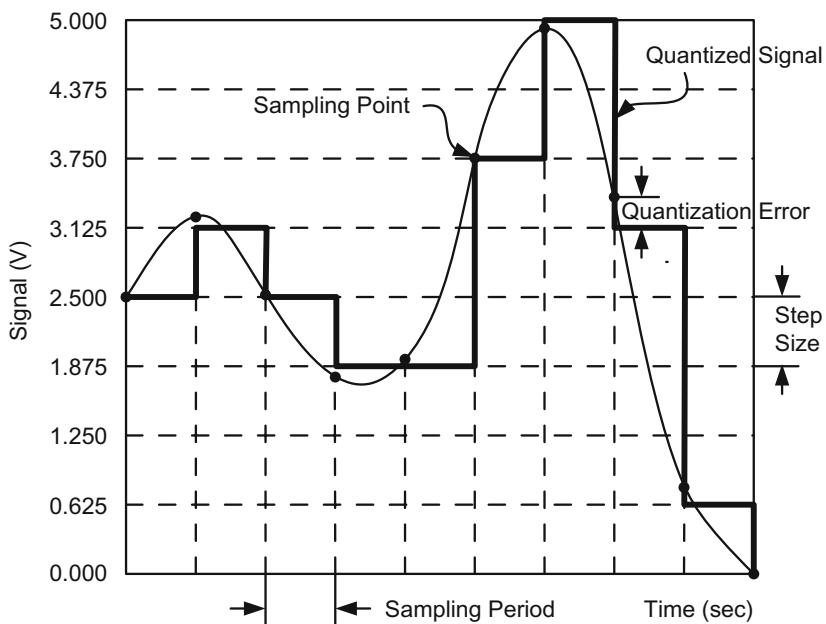
**Fig. 7.2** Input-output description of a three-bit ADC

ADC samples non-periodic analog signals in regular time intervals as described in Fig. 7.3. The time interval between sampling points is called the sampling period. The sampling period is adjusted according to the processing speed of the ADC in order to generate accurate digital outputs.

Once sampled, the analog voltage at the input of an ADC is held steady throughout the sampling period while the conversion takes place as shown in Fig. 7.4. The shape of the converted signal may be quite different from the original analog signal due to the ADC resolution and the time duration between samples. In a three-bit ADC, sampling takes place



**Fig. 7.3** Sampling a continuous analog signal



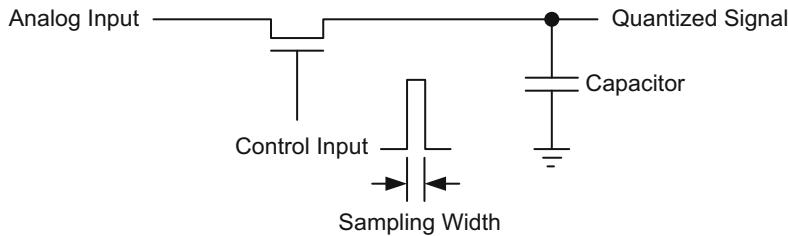
**Fig. 7.4** Sampling period, hold concept and regeneration of an analog signal

in 0.625 V increments. Therefore, each sampling point becomes subject to a dynamic quantization error which changes between 0 and 0.625 V. For example, a three-bit ADC samples 3.4 V according to its closest sampling level of 3.125 V, and produces a 0.275 V error. Arbitrary signals that change with a frequency faster than the sampling frequency are

subject to much larger dynamic errors. When converted back to their analog form, these signals show large deviations from their original shapes.

## 7.2 Sample and Hold Principle

A basic sample-and-hold circuit consists of an NMOS transistor and a capacitor as shown in Fig. 7.5. The control input simply turn on an N-channel MOSFET for a short period of time, called sampling width, during which the analog voltage level at the input is stored on the capacitor. When the transistor is turned off, this analog value is held constant until the next sampling point.

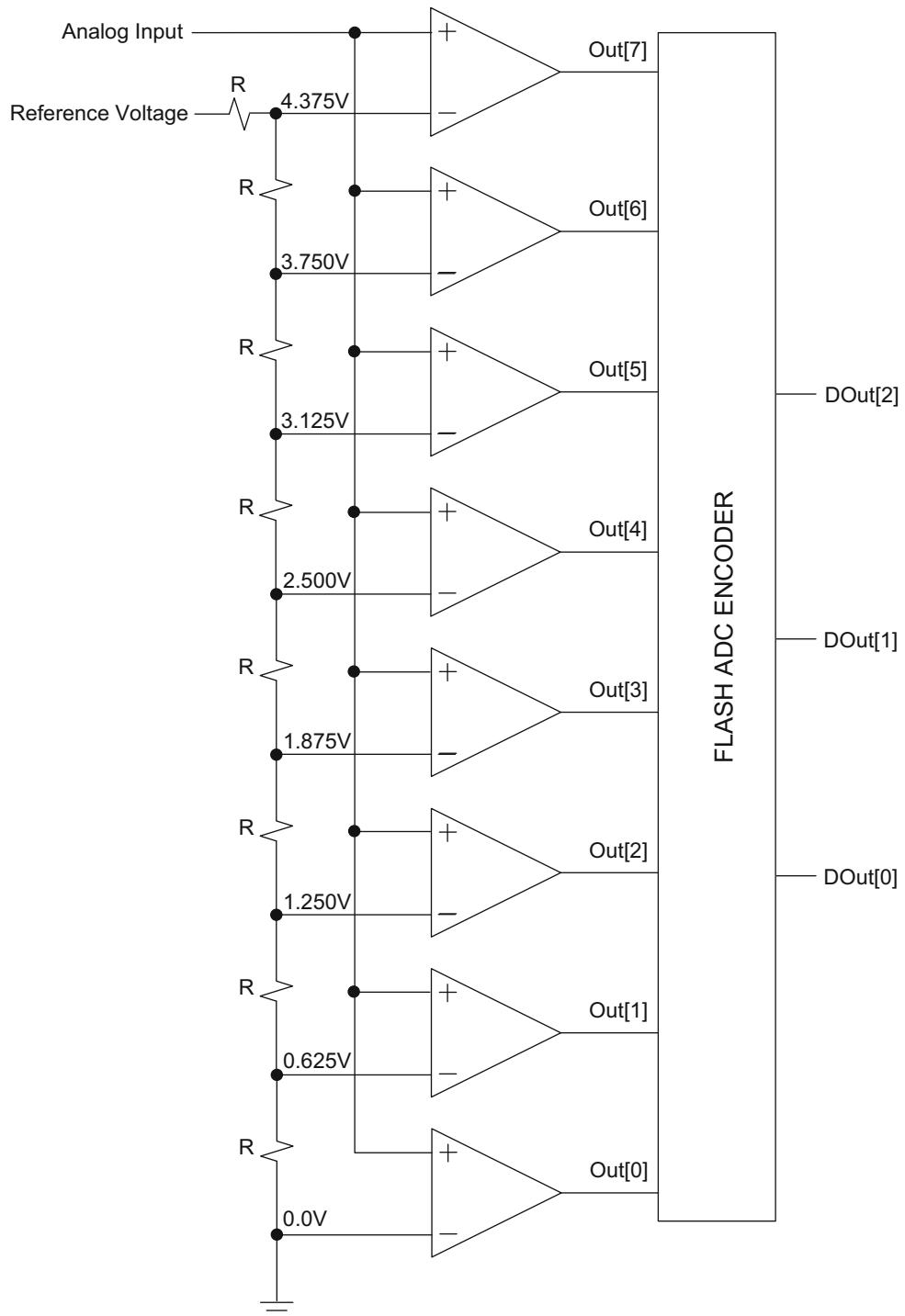


**Fig. 7.5** A typical sample-and-hold circuit

## 7.3 Flash Type Analog-to-Digital Converter

The simplest ADC is the flash-type as shown in Fig. 7.6. This three-bit ADC contains  $2^3 = 8$  operational amplifiers. The analog signal is applied to all eight positive input terminals. The reference voltage is distributed to each negative input terminal via a voltage divider circuit. Each operational amplifier acts as a differential amplifier and amplifies the difference between a continuously changing analog signal and the portion of the reference voltage.

Figure 7.7 describes the operation of the three-bit flash ADC and its encoder in a truth table. When the analog voltage is less than or equal to 0.625 V, only Out[0] becomes logic 1, all other outputs from Out[1] to Out[7] become logic 0. When the analog signal exceeds 0.625 V but less than 1.25 V, only Out[0] and Out[1] become logic 1, and again all others become logic 0. Higher analog voltages at the input successively produce more logic 1 levels as shown in Fig. 7.7. An encoder is placed at the output stage of all operational amplifiers to transform the voltage levels at Out[7:0] into a three-bit digital output, DOut[2:0]. The digital output is subject to an error of 0.625 V because only three bits are used for conversion.



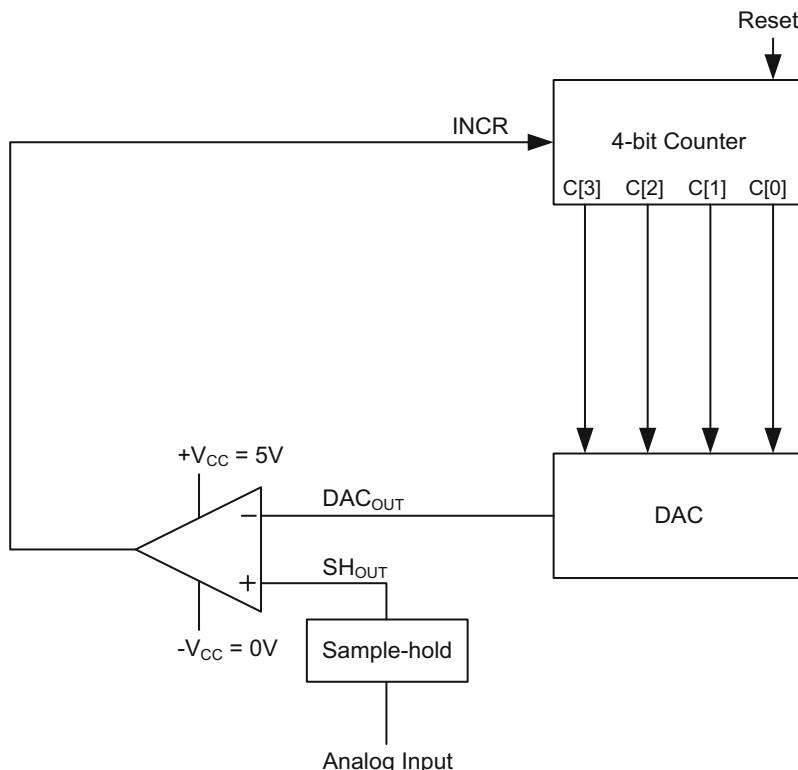
**Fig. 7.6** Typical three-bit flash ADC schematic

Analog Input	Out[7]	Out[6]	Out[5]	Out[4]	Out[3]	Out[2]	Out[1]	Out[0]	DOut[2]	DOut[1]	DOut[0]
$0.625 > V_{IN} > 0.000$	0	0	0	0	0	0	0	1	0	0	0
$1.250 > V_{IN} > 0.625$	0	0	0	0	0	0	1	1	0	0	1
$1.875 > V_{IN} > 1.250$	0	0	0	0	0	1	1	1	0	1	0
$2.500 > V_{IN} > 1.875$	0	0	0	0	1	1	1	1	0	1	1
$3.125 > V_{IN} > 2.500$	0	0	0	1	1	1	1	1	1	0	0
$3.750 > V_{IN} > 3.125$	0	0	1	1	1	1	1	1	1	0	1
$4.375 > V_{IN} > 3.750$	0	1	1	1	1	1	1	1	1	1	0
$5.000 > V_{IN} > 4.375$	1	1	1	1	1	1	1	1	1	1	1

**Fig. 7.7** Three-bit flash ADC truth table describing its operation

#### 7.4 Ramp Type Analog-to-Digital Converter

The ramp ADC uses only a single operation amplifier, but it employs an up-counter as well as a Digital-to-Analog Converter (DAC) in a loop structure shown in Fig. 7.8. The digital output is obtained from C[3:0] terminals, and it progressively forms within several clock periods as opposed to being almost instantaneous as in the flash ADC.



**Fig. 7.8** Typical four-bit ramp ADC schematic

The top portion of Fig. 7.9 describes the output voltage assignments of a four-bit ramp ADC using two different types of number-rounding schemes. The down-rounding scheme assigns a lower analog value for each digital output compared to the up-rounding scheme. For example, the down-rounding scheme produces a digital output of  $C[3:0] = 0100$  for analog voltages between 1.0937 and 1.4062 V applied to its input. If the up-rounding scheme is used, the same digital output becomes equivalent to an analog voltage anywhere between 1.4062 and 1.7187 V.

The middle table in Fig. 7.9 shows how the conversion takes place if the down-rounding mechanism is used in this four-bit ADC. Prior to its operation, the four-bit up-counter is reset and produces  $C[3:0] = 0000$ . Assuming an analog voltage of 2 V is applied to the input, which must be kept constant until the conversion is complete,  $C[3:0] = 0000$  forces the DAC output,  $DAC_{OUT}$ , to be 0 V according to the down-rounding scheme. Since this value is less than 2 V at the sample/hold circuit output,  $SH_{OUT}$ , the output of the differential amplifier, INCR, transitions to the positive supply potential of the operational amplifier,  $+V_{CC} = 5$  V, which prompts the four-bit counter to increment to  $C[3:0] = 0001$ . Consequently, the DAC generates  $DAC_{OUT} = 0.3125$  V according to the truth table in Fig. 7.9. However, this value is still less than  $SH_{OUT} = 2$  V. Therefore, the differential amplifier produces another  $INCR = 5$  V which prompts the counter to increment again to  $C[3:0] = 0010$ . Up-counting continues until  $C[3:0] = 0111$  or  $DAC_{OUT} = 2.1875$  V. Since this last voltage is greater than  $SH_{OUT} = 2$  V, the differential amplifier output switches back to its negative supply voltage,  $-V_{CC} = 0$  V, and stops the up-counter from incrementing further. The digital output stays steady at  $C[3:0] = 0111$  from this point forward, representing 2 V analog voltage with a dynamic error of 0.1875 V.

The table at the bottom part of Fig. 7.9 represents the conversion steps if the up-rounding mechanism is used in this ADC. External reset still produces  $C[3:0] = 0000$  initially. However, the DAC output starts the conversion with an increased amount of 0.3125 V instead of 0 V. The counter increments until  $C[3:0] = 0110$ , and produces 2.1875 V at the  $DAC_{OUT}$  node. At this value INCR becomes 0 V, and the up-counter stops incrementing further.  $C[3:0] = 0110$  becomes the ADC result for 2 V.

Step Size =  $5/2^4 = 0.3125V$

C[3]	C[2]	C[1]	C[0]	Down-Round(V)	Up-Round(V)
0	0	0	0	0.0000	0.3125
0	0	0	1	0.3125	0.6250
0	0	1	0	0.6250	0.9375
0	0	1	1	0.9375	1.2500
0	1	0	0	1.2500	1.5625
0	1	0	1	1.5625	1.8750
0	1	1	0	1.8750	2.1875
0	1	1	1	2.1875	2.5000
1	0	0	0	2.5000	2.8125
1	0	0	1	2.8125	3.1250
1	0	1	0	3.1250	3.4375
1	0	1	1	3.4375	3.7500
1	1	0	0	3.7500	4.0625
1	1	0	1	4.0625	4.3750
1	1	1	0	4.3750	4.6875
1	1	1	1	4.6875	5.0000

Analog Input = 2V with Down-Rounding Mechanism

Step	C[3]	C[2]	C[1]	C[0]	DAC <sub>OUT</sub> (V)	INCR(V)
1	0	0	0	0	0.0000	5.0
2	0	0	0	1	0.3125	5.0
3	0	0	1	0	0.6250	5.0
4	0	0	1	1	0.9375	5.0
5	0	1	0	0	1.2500	5.0
6	0	1	0	1	1.5625	5.0
7	0	1	1	0	1.8750	5.0
8	0	1	1	1	2.1875	0.0

Final output with quantization error of 0.3125V

Analog Input = 2V with Up-Rounding Mechanism

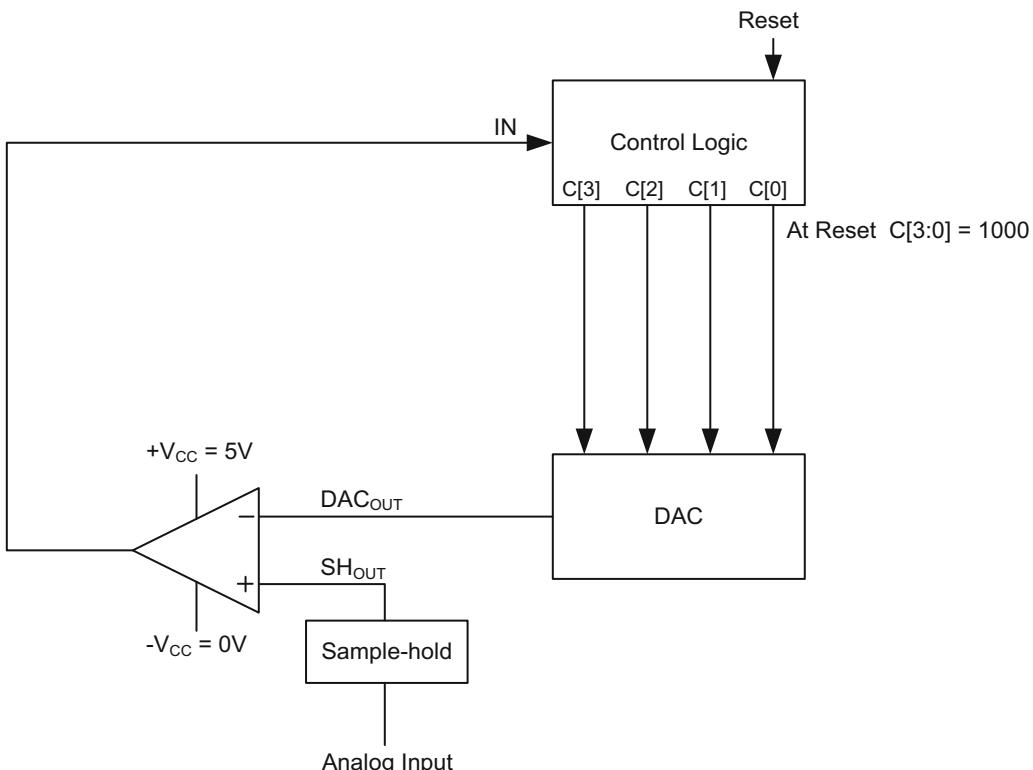
Step	C[3]	C[2]	C[1]	C[0]	DAC <sub>OUT</sub> (V)	INCR(V)
1	0	0	0	0	0.3125	5.0
2	0	0	0	1	0.6250	5.0
3	0	0	1	0	0.9375	5.0
4	0	0	1	1	1.2500	5.0
5	0	1	0	0	1.5625	5.0
6	0	1	0	1	1.8750	5.0
7	0	1	1	0	2.1875	0.0

Final output with quantization error of 0.3125V

**Fig. 7.9** Four-bit ramp ADC truth table describing its operation

## 7.5 Successive Approximation Type Analog-to-Digital Converter

The third type ADC is based on the successive approximation technique to estimate the value of the analog voltage. This type is a trade-off between the flash-type and the ramp-type in terms of speed and the number of components used in the circuit. As a numerical example, a typical four-bit successive approximation ADC schematic is shown in Fig. 7.10. In this figure, the up-counter in the ramp ADC is replaced by a control logic which successively converts an analog input into a digital form by a trial and error method. The output is obtained at the C[3:0] terminal.



**Fig. 7.10** Typical four-bit successive approximation ADC schematic

The top portion of Fig. 7.11 shows the truth table to operate a four-bit successive approximation ADC. Two numerical examples in this figure illustrate the down-rounding and up-rounding schemes used during conversion.

The first example in Fig. 7.11 illustrates the down rounding mechanism in a four-bit successive approximation ADC. In this example, an analog voltage of 3.5 V is applied to the analog input of the ADC. An external reset starts the converter at C[3:0] = 1000, which is considered a mid point between C[3:0] = 0000, representing the minimum analog input of 0 V, and C[3:0] = 1111, representing the maximum analog input of 5 V for this ADC.

For  $C[3:0] = 1000$ , the DAC generates an initial analog voltage of 2.5 V at the  $DAC_{OUT}$  node. Since this value is less than the sampled analog voltage of 3.5 V at the  $SH_{OUT}$  node, the operational amplifier produces  $IN = 5$  V, and prompts the control logic to try a slightly higher digital output. As a result, the control logic produces  $C[3:0] = 1100$  as its first trial, which is equivalent to a midway point between  $C[3:0] = 1000$  and  $1111$ .  $DAC_{OUT}$  becomes  $2.5 + (2.5/2) = 3.75$  V. But, this new voltage is larger than  $SH_{OUT} = 3.5$  V, and the operational amplifier produces  $IN = 0$  V in return. The drop at  $IN$  node is an indication to the control logic that its initial attempt of  $C[3:0] = 1100$  was too large, and it must lower its output. This time, the control logic tries  $C[3:0] = 1010$ , which is between  $C[3:0] = 1000$  and  $1100$ , and translates to  $DAC_{OUT} = 2.5 + (2.5/4) = 3.125$  V. This value, in turn, generates  $IN = 5$  V, and prompts the control logic to try a slightly higher output between  $C[3:0] = 1010$  and  $1100$ . In the third round, the control logic produces  $C[3:0] = 1011$ .  $DAC_{OUT}$  node becomes  $2.5 + (2.5/4) + (2.5/8) = 3.4375$  V and generates  $IN = 5$  V. This new input suggests the control logic to try even higher digital output, such as  $2.5 + (2.5/4) + (2.5/8) + (2.5/16) = 3.5937$  V, in the fourth round. However,  $2.5/8 = 0.3125$  V is the resolution limit for this four-bit ADC, and as a result, the controller stalls at  $C[3:0] = 1011$ , revealing  $DAC_{OUT} = 3.4375$  V. This voltage differs from  $SH_{OUT} = 3.5$  V by only 0.0625 V.

The second example in Fig. 7.11 explains the successive approximation technique if the up-rounding scheme is employed. The conversion again starts at  $C[3:0] = 1000$ , but with an incremented value of  $2.5 + 0.3125 = 2.8125$  V at the DAC output. Since this voltage is below  $SH_{OUT} = 3.5$  V,  $IN$  node becomes 5 V and prompts the control logic to produce a larger digital output. The control logic responds this with a digital output of  $C[3:0] = 1100$ , which corresponds to the analog voltage of  $2.8125 + (2.5/2) = 4.0625$  V at the  $DAC_{OUT}$  node. As a result,  $IN$  node becomes 0 V, and forces the control logic to lower its digital output. This time, the control logic tries  $C[3:0] = 1010$  which is equivalent to  $2.8125 + (2.5/4) = 3.4375$  at the  $DAC_{OUT}$  node. Due to the resolution limit of this four-bit ADC, this step also becomes the end of successive approximation.

Step Size =  $5/2^4 = 0.3125V$

C[3]	C[2]	C[1]	C[0]	Down-Round (V)	Up-Round (V)
0	0	0	0	0.0000	0.3125
0	0	0	1	0.3125	0.6250
0	0	1	0	0.6250	0.9375
0	0	1	1	0.9375	1.2500
0	1	0	0	1.2500	1.5625
0	1	0	1	1.5625	1.8750
0	1	1	0	1.8750	2.1875
0	1	1	1	2.1875	2.5000
1	0	0	0	2.5000	2.8125
1	0	0	1	2.8125	3.1250
1	0	1	0	3.1250	3.4375
1	0	1	1	3.4375	3.7500
1	1	0	0	3.7500	4.0625
1	1	0	1	4.0625	4.3750
1	1	1	0	4.3750	4.6875
1	1	1	1	4.6875	5.0000

Analog Input = 3.5V with Down-Rounding Mechanism   START with  $(5.0/2) = 2.5$

Step	C[3]	C[2]	C[1]	C[0]	DAC <sub>OUT</sub> (V)	Control Logic In
1	1	0	0	0	2.5000	$3.5 > 2.5000$ Thus try $2.5 + (2.5/2) = 3.75$
2	1	1	0	0	3.7500	$3.5 < 3.7500$ Thus try $2.5 + (2.5/4) = 3.125$
3	1	0	1	0	3.1250	$3.5 > 3.1250$ Thus try $2.5 + (2.5/4) + (2.5/8) = 3.4375$
4	1	0	1	1	3.4375	$3.5 > 3.4375$ Stop at 3.4375 since there is no resolution under $(2.5/8)$

Final output with quantization error of 0.3125V

Analog Input = 3.5V with Up-Rounding Mechanism   START with  $(5.0/2) = 2.8125$

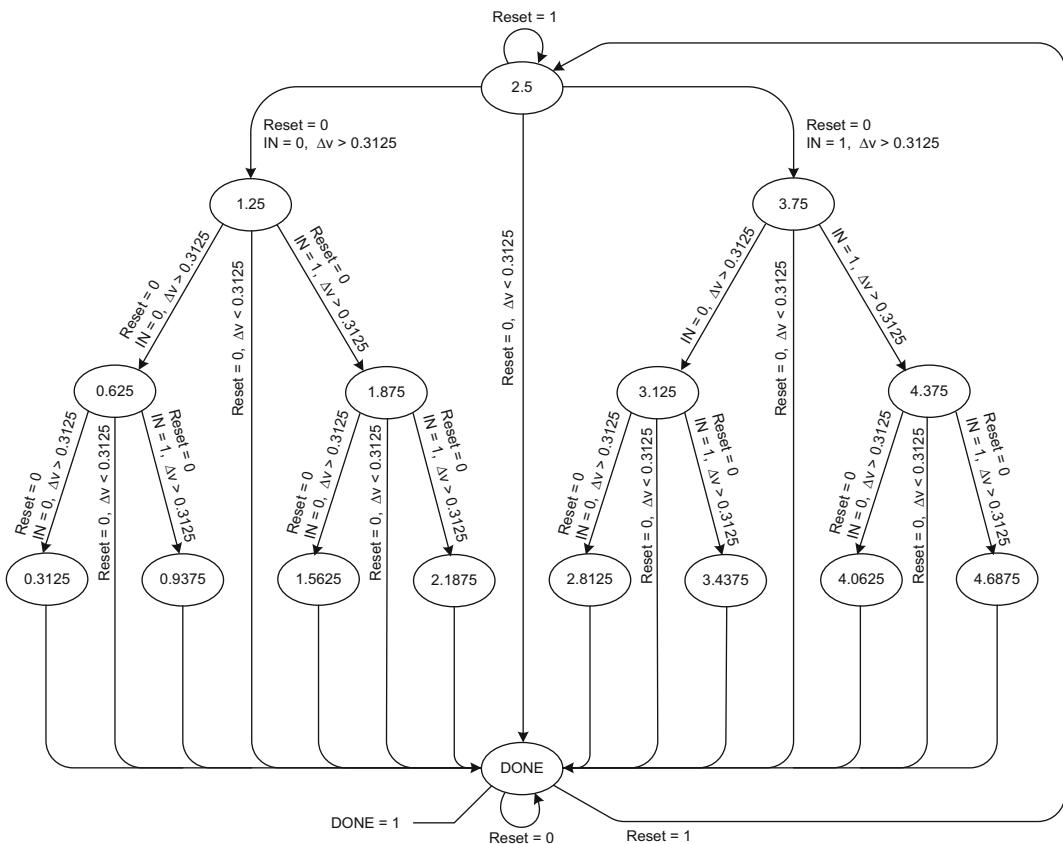
Step	C[3]	C[2]	C[1]	C[0]	DAC <sub>OUT</sub> (V)	Control Logic In
1	1	0	0	0	2.8125	$3.5 > 2.8125$ Thus try $2.8125 + (2.5/2) = 4.0625$
2	1	1	0	0	4.0625	$3.5 < 4.0625$ Thus try $2.8125 + (2.5/4) = 3.4375$
3	1	0	1	0	3.4375	$3.5 > 3.4375$ Stop at 3.4375 since there is no resolution under $(2.5/8)$

Final output with quantization error of 0.3125V

**Fig. 7.11** Four-bit successive approximation ADC truth table describing down-rounding and up-rounding approximation techniques

The control circuit of the four-bit ADC with down-rounding scheme is shown in Fig. 7.12. In this figure, the approximation process starts at the midpoint, C[3:0] = 1000, corresponding to DAC<sub>OUT</sub> = 2.5 V according to the table in Fig. 7.11. When the external reset is removed, and  $\Delta v$ , the difference between SH<sub>OUT</sub> and DAC<sub>OUT</sub>, is found to be greater

than the 0.3125 V step size, the control logic either goes to the state 1.25, and produces  $C[3:0] = 0100$  (equivalent to 1.25 V) or the state 3.75, and produces  $C[3:0] = 1100$  (equivalent to 3.75 V) in the first step of the successive approximation. This decision depends on the value of the control logic input, IN. If  $IN = 0$ , which translates to the analog input to be less than 2.5 V, the next state becomes the state 1.25. However, if  $IN = 1$ , the analog input is considered to be greater than 2.5 V, and the next state becomes the state 3.75. If  $\Delta V$  is less than 0.3125 V, on the other hand, the control logic cannot proceed further due to its resolution limit, and moves to the state DONE. In the second step of successive approximation, the state 1.25 either transitions to the state 0.625 or the state 1.875, depending on the value at IN node. Similar transitions take place from the state 3.75 to either the state 3.125 or the state 4.375, again depending on the value at IN. After this point, the state machine performs one last approximation to estimate the value of analog input voltage, and reaches the DONE state with an output value as shown in Fig. 7.12.



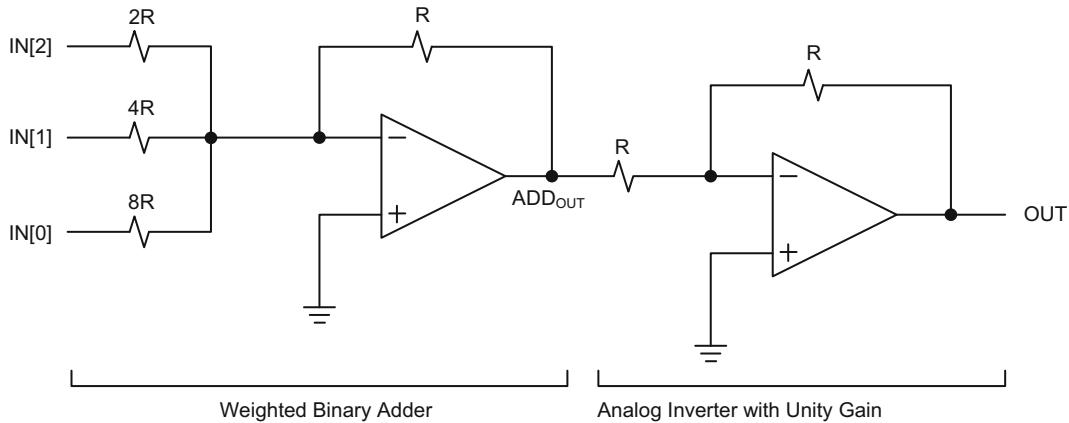
**Fig. 7.12** Four-bit successive approximation control circuit

## 7.6 Weighted Sum Type Digital-to-Analog Converter

The most common DAC utilizes the weighted summation method of digital inputs. A three-bit DAC with a weighted binary adder is shown in Fig. 7.13 as an example.

This circuit is composed of two parts. The first part adds all three binary input bits, IN[2] (the most significant bit), IN[1] and IN[0] (the least significant bit), and produces an output,  $\text{ADD}_{\text{OUT}} = -(0.5 \text{ IN}[2] + 0.25 \text{ IN}[1] + 0.125 \text{ IN}[0])$  according to the equation in Fig. 7.14. The second part is an analog inverter which forms  $\text{OUT} = -\text{ADD}_{\text{OUT}}$ .

Therefore, the circuit in Fig. 7.13 generates  $\text{OUT} = 0.5 \text{ IN}[2] + 0.25 \text{ IN}[1] + 0.125 \text{ IN}[0]$ , where each binary value at IN[2:0] input is multiplied by the coefficients,  $2^{-1}$ ,  $2^{-2}$  and  $2^{-3}$ , before they are added to produce an output. For example, the combination of  $\text{IN}[2] = 1$ ,  $\text{IN}[1] = 0$  and  $\text{IN}[0] = 1$ , with +5 and 0 V logic levels generates  $\text{OUT} = 2.5 + 0.625 = 3.125$  V. Similarly, all the other analog outputs in Fig. 7.14 can be generated using the equation at the top part of this figure with a maximum error of 0.625 V.



**Fig. 7.13** Three-bit DAC schematic with weighted binary adder

$$\begin{aligned} \text{ADD}_{\text{OUT}} &= -\frac{R}{2R} \text{IN}[2] - \frac{R}{4R} \text{IN}[1] - \frac{R}{8R} \text{IN}[0] \\ &= -0.5 \text{IN}[2] - 0.25 \text{IN}[1] - 0.125 \text{IN}[0] \end{aligned}$$

$$\text{OUT} = -\text{ADD}_{\text{OUT}} = 0.5 \text{IN}[2] + 0.25 \text{IN}[1] + 0.125 \text{IN}[0]$$

IN[2]	IN[1]	IN[0]	OUT(V)
0	0	0	0.000
0	0	1	0.625
0	1	0	1.250
0	1	1	1.875
1	0	0	2.500
1	0	1	3.125
1	1	0	3.750
1	1	1	4.375

Example:

IN[2] = 1, IN[1] = 0, IN[0] = 1 with +5V/0V logic levels

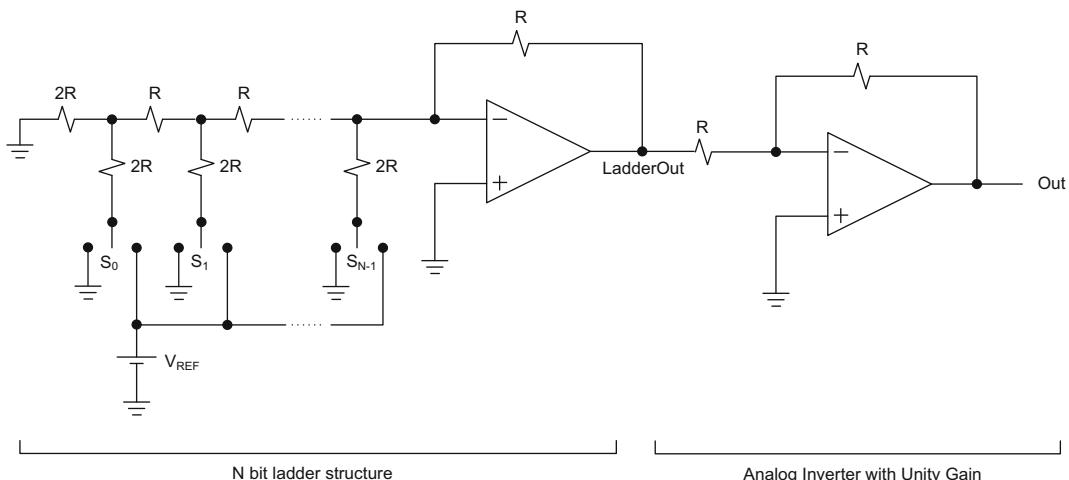
$$\text{ADD}_{\text{OUT}} = -2.5 - 0 - 0.625 = -3.125 \text{V}$$

$$\text{OUT} = +3.125 \text{V} \text{ with } 0.625 \text{V quantization error}$$

**Fig. 7.14** Three-bit DAC operation with weighted binary adder

## 7.7 Ladder Type Digital-to-Analog Converter

The second kind of DAC uses a ladder structure composed of resistors as shown in Fig. 7.15. Each ladder segment contains a switch,  $S_i$ , where  $i = 0, 1, \dots, (N-1)$ . The switch can be thrown to ground or  $V_{\text{REF}}$  depending on the value of the binary input.



**Fig. 7.15** N-bit DAC schematic with ladder configuration

The ladder structure produces an intermediate output, LadderOut, which is the weighted sum of all N binary inputs. The ratio,  $V_{REF}/2^N$ , determines the final value of LadderOut when it is multiplied with the weighted sum as shown in Fig. 7.16. The analog inverter generates Out =  $-LadderOut$  and creates a positive value. The example of a three-bit DAC with ladder configuration is shown at the bottom section of Fig. 7.16. With  $S_2$  and  $S_0$  switches thrown to  $V_{REF} = 5$  V and  $S_1$  to ground, this combination produces a weighted sum of  $(2^2 + 2^0) = 5$  V. When this value is multiplied with the ratio,  $V_{REF}/2^3 = 5/8 = 0.625$  V, the overall result becomes Out = 3.125 V for a digital input of  $S_2 = 1$ ,  $S_1 = 0$ ,  $S_0 = 1$ .

$$\text{LadderOut} = -\frac{V_{REF}}{2^N} (S_{N-1} 2^{N-1} + S_{N-2} 2^{N-2} + \dots + S_0 2^0)$$

$$\text{Out} = -\text{LadderOut} = \frac{V_{REF}}{2^N} (S_{N-1} 2^{N-1} + S_{N-2} 2^{N-2} + \dots + S_0 2^0)$$

Example:

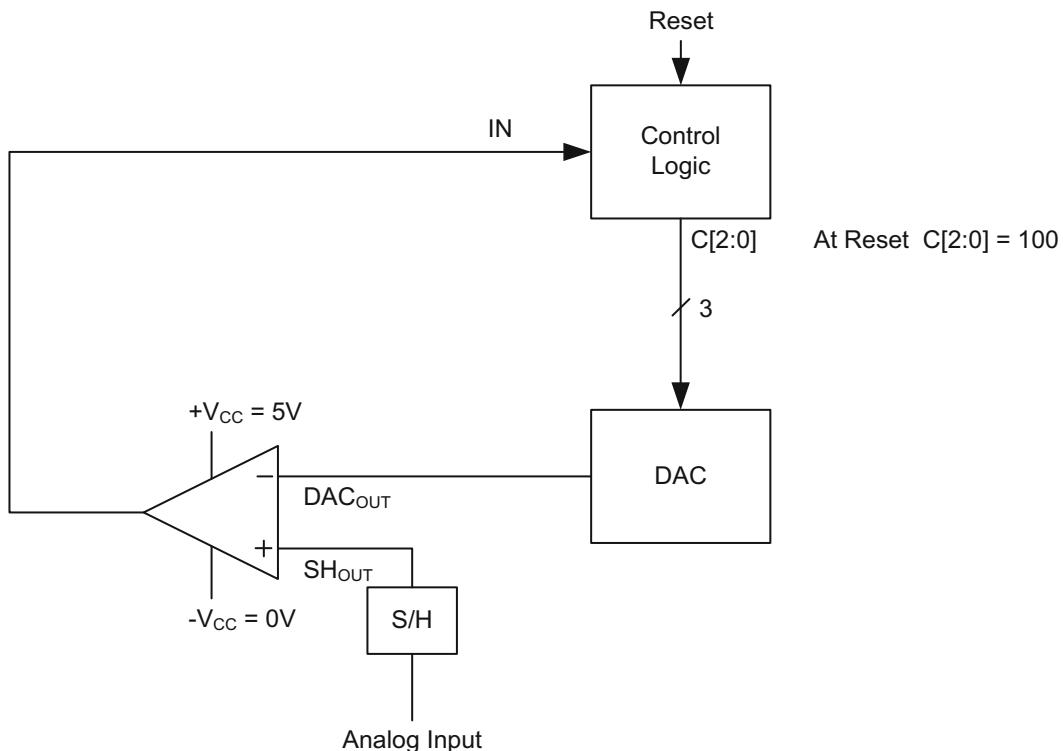
For a 3-bit DAC with  $S_2 = 1$ ,  $S_1 = 0$ ,  $S_0 = 1$  and  $V_{REF} = 5V$

$$\text{Out} = \frac{5}{2^3} (S_2 2^2 + S_1 2^1 + S_0 2^0) = \frac{5}{8} (4 + 0 + 1) = 3.125V$$

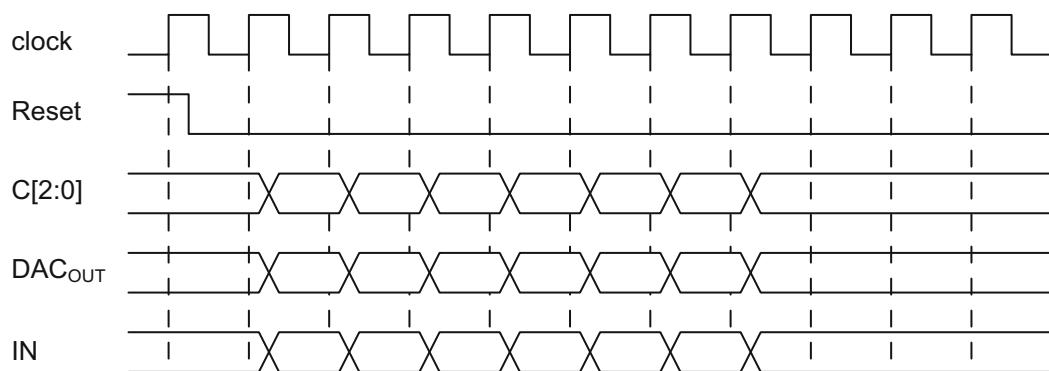
**Fig. 7.16** Three-bit DAC operation with ladder configuration

## Review Questions

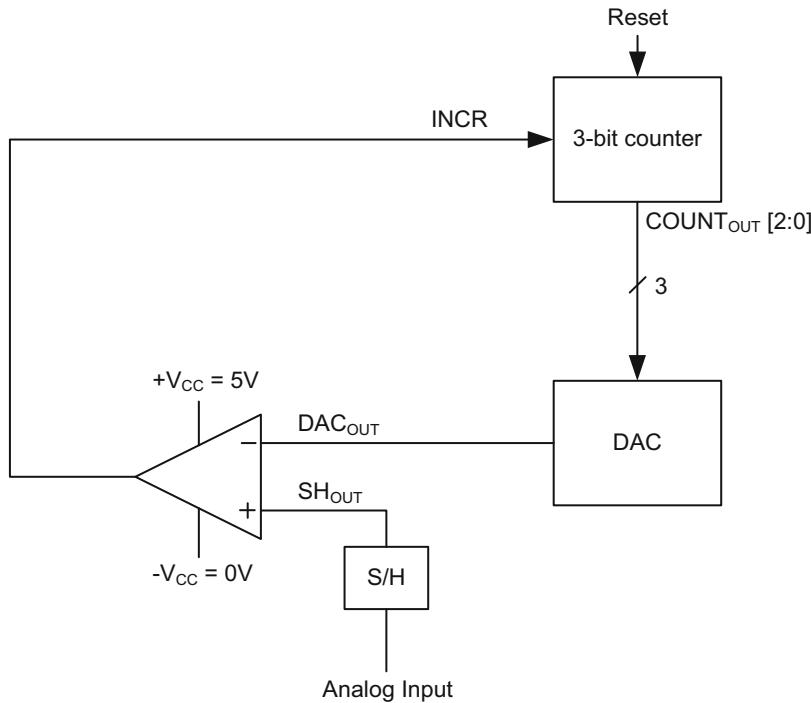
1. A three-bit successive approximation ADC is given below. A sample-Hold circuit (S/H) samples the Analog Input in periodic intervals and feeds the voltage level to the operational amplifier.



Use the timing diagram below, and fill the blanks with analog or digital data for an Analog Input = 0.5 V with down-rounding mechanism.



2. A three-bit ramp ADC below operates with  $+V_{CC} = 3\text{ V}$  and  $-V_{CC} = 0\text{ V}$ . Input to this ADC can take any value between 0 and 3 V.



- (a) Assume that DAC has an up-rounding scheme to generate analog outputs from digital inputs. Apply 1.2 V to the Analog Input port, and draw the timing diagram that contains the clock, Reset, counter output (COUNT<sub>OUT</sub>), DAC output (DAC<sub>OUT</sub>) and operational amplifier output (INCR). Show what happens to the timing diagram when the active-high Reset signal transitions to logic 1 after the ADC produces the desired digital output.
- (b) Now apply 2.9 V to the Analog Input. Draw the timing diagram with the same inputs and outputs listed above. Show what happens to the timing diagram when the active-high Reset signal transitions to logic 1 after the ADC produces the desired digital output.
- (c) Assume that the DAC rounding scheme is changed from up-rounding to down-rounding scheme. Apply 1.2 V to the Analog Input and generate the timing diagram with the input and output signals listed above. Show what happens to the timing diagram when the active-high Reset signal becomes logic 1 after the ADC produces the desired digital output. Do you see any issues with the operation of this circuit?

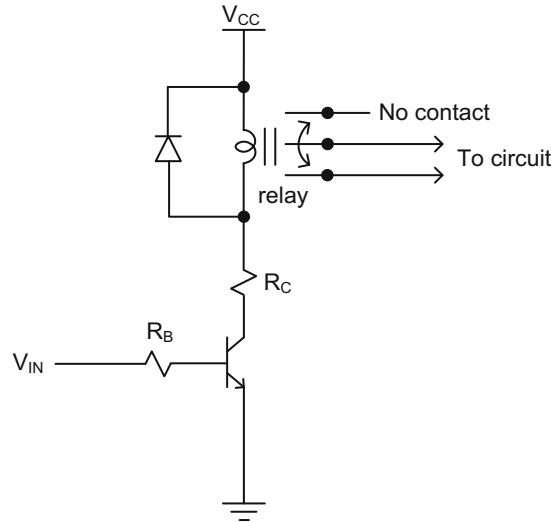
- (d) Now apply 2.9 V at the Analog Input port and generate the timing diagram with the input and output signals listed above. Do you see any issues in the operation of this circuit?
3. In a three-bit successive approximation ADC in Fig. 7.10, what kind of circuit can be built to detect the voltage difference between the Analog Input voltage at  $S_{HOUT}$  and the DAC output at  $DAC_{OUT}$  at each state to be able to make state transitions in Fig. 7.12?
4. Build an eight-bit DAC that uses weighted summation technique. Find the analog voltage levels for inputs, 00110011, 10000000, 00000001, 11111111 and 10101010. Compare the analog voltage values corresponding to 10000000, 00000001 and 11111111 inputs from this DAC with the analog output voltages corresponding to the 1000, 0001 and 1111 inputs from a four-bit DAC that uses weighted binary adder.
5. Build an eight bit DAC that uses ladder configuration. How can you implement the input switches used in the ladder configuration? Apply 00110011, 10000000, 00000001, 11111111 and 10101010 inputs to this DAC and show how the circuit works. Is there any difference between the output values between this type and the type that uses weighted summation?
6. Implement an eight-bit flash, ramp and successive approximation ADC. Compare each type in terms of speed, and the number of components used in the circuit.

This chapter is primarily dedicated to front-end analog electronics interfacing microcontrollers. It brings together most of the topics covered in the previous seven chapters. The chapter starts with basic electromechanical device control and sensor circuits, then moves on to implementing full scale electronic platforms that contain sensor(s), amplifier(s), logic blocks, ADC and DAC.

---

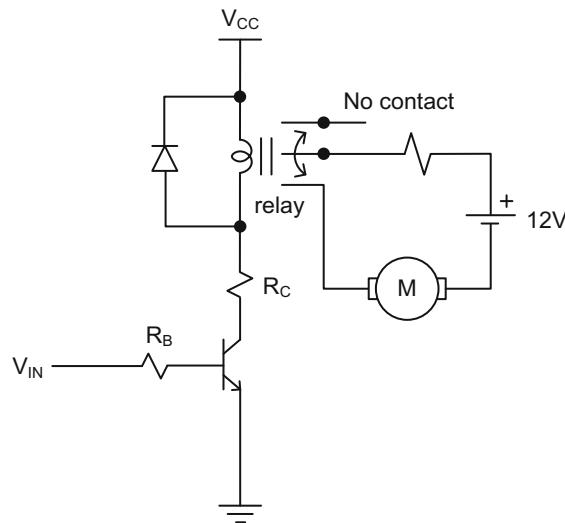
## 8.1 Electromechanical Device Control

Relay switches are excellent choices to isolate digital circuits from noisy electrical environments or from circuits that require high voltages or currents to operate electromechanical devices. Figure 8.1 shows the operation of a relay switch. The parallel diode placed across the relay prevents transient currents to form and protects rest of the circuit from current surges emanating from the relay. When a high logic level is applied to the  $V_{IN}$  terminal, the NPN transistor saturates. The resulting current transforms the inductor into an electromagnet, and closes the mechanical switch inside the relay enclosure. This action starts the current conduction in the secondary circuit to operate an electromechanical device. When there is no current present, the switch pulls back to its no contact position and stops the current in the secondary circuit.



**Fig. 8.1** Bipolar circuit controlling a relay

Figure 8.2 uses the relay to activate a DC motor. When closed, the 12 V supply runs the DC motor. In this circuit  $V_{CC}$  can safely be set to a voltage lot less than 12 V. Similarly, the supply voltage for the DC motor can be set much higher than 12 V without causing noise or harming the digital circuit.



**Fig. 8.2** A dc motor activation circuit using a relay for electrical isolation

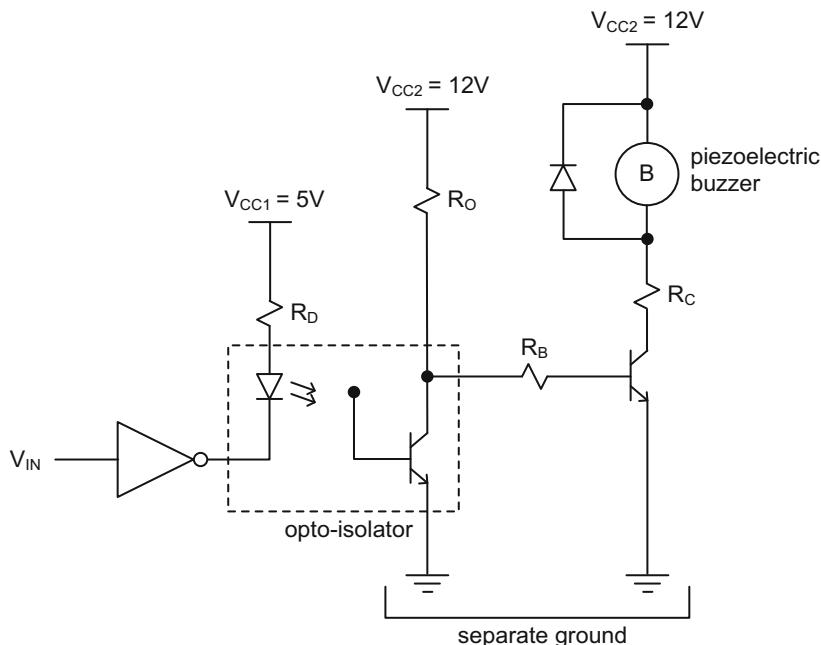
Opto-isolator is another device that uses optical means to isolate a digital circuit from an analog circuit that includes an electromechanical device and often requires high voltages or currents. The box defined by the dashed lines in Fig. 8.3 represents an opto-isolator. This device is composed of an infrared LED and a photo-transistor whose base is optically

coupled with the light emanating from the LED. When the LED emits infrared light, it produces high voltage at the base of the photo-transistor to saturate it. Since opto-isolator only comes with an open-collector configuration, the collector of the photo-transistor has to be externally connected to a supply voltage with a current limiting resistor to function.

Figure 8.3 shows an application of the opto-isolator to operate a piezo-electric buzzer. When  $V_{IN}$  terminal receives logic 1, the output of the inverter goes low. The supply current that flows through  $R_D$  turns on the “internal” LED. Infrared light that emanates from the LED saturates the photo-transistor provided that the current through  $R_O$  meets the manufacturer’s requirements for the transistor to saturate. As a result, the collector of the photo-transistor pulls low and turns off the output NPN transistor. Since there will be no current through the piezo-electric buzzer, it remains silent.

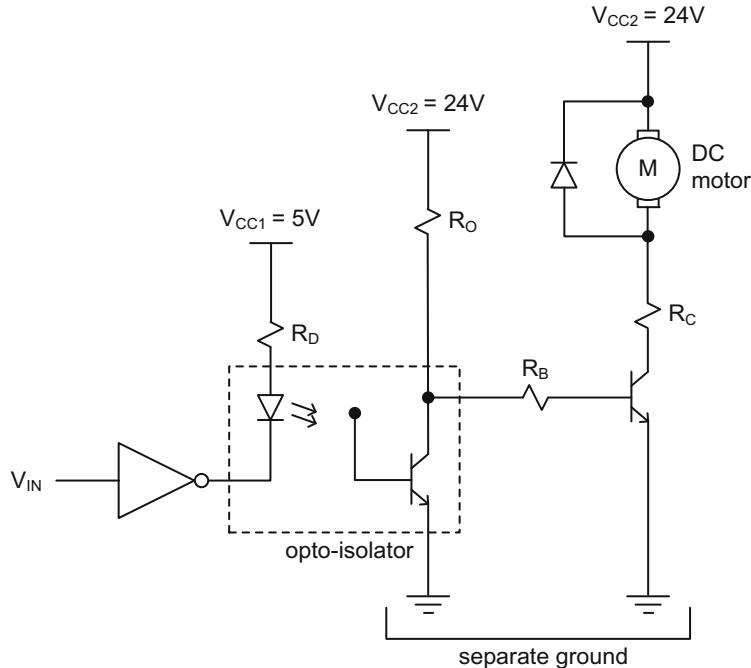
However, when  $V_{IN}$  is lowered to 0 V, the inverter output goes to logic 1, and this produces 0 V across the infrared LED. The LED does not emit light, and the photo-transistor stops operating. Consequently, the base terminal of the photo-transistor goes to 12 V, and saturates the output NPN transistor. The buzzer starts operating.

It is essential to separate the ground terminals of the 5 V digital supply from the 12 V analog supply to prevent transmitting any electrical noise through ground connections that may adversely affect the operation of the digital circuit.



**Fig. 8.3** Piezoelectric buzzer control using an opto-isolator

Figure 8.4 shows another application of the opto-isolator device if the DC motor requires even higher voltage than 12 V to operate. As discussed above with the piezo-electric buzzer, a low logic applied to  $V_{IN}$  disables the opto-isolator and operates the DC motor. Again, it is a vital importance to separate the ground connections of the 5 and 24 V supplies from each other to prevent electrical noise produced in the analog circuitry from propagating through the ground terminal.



**Fig. 8.4** DC motor control using an opto-isolator

## 8.2 Pulse Width Modulation Circuits

A simple Pulse Width Modulation (PWM) circuit is shown in Fig. 8.5. This circuit is an enhanced form of the square wave generator discussed earlier in Chapter 7.

In this circuit, there are two feedback loops. The primary feedback loop connects the output to the positive input of the operational amplifier with  $R_F$ , setting up the reference signal,  $V_{REF}$ , for the input. The secondary feedback loop connects the output to the negative input of the operational amplifier with the combination of a variable resistor, a diode and a capacitor. This RC circuit basically adjusts the pulse width at the output.

Assume  $V_{SIG} = -V_{CC}$  and  $V_{OUT} = +V_{CC}$  initially.  $V_{OUT} = +V_{CC}$  produces:

$$V_{REF} = V_{CC} \frac{R_F}{R_F + R_O} \quad (8.1)$$

The output node starts charging the capacitor through the resistor,  $\alpha R$  ( $\alpha$  represents the portion of the potentiometer), and the diode,  $D_1$ , as shown in the top portion of Fig. 8.6. This produces:

$$V_{SIG} = -V_{CC} + 2V_{CC} \left[ 1 - \exp\left(-\frac{t}{\alpha RC}\right) \right] \quad (8.2)$$

As soon as  $V_{SIG}$  reaches slightly above  $V_{REF} = V_{CC} \frac{R_F}{R_F + R_O}$ ,  $V_{OUT}$  changes its polarity,

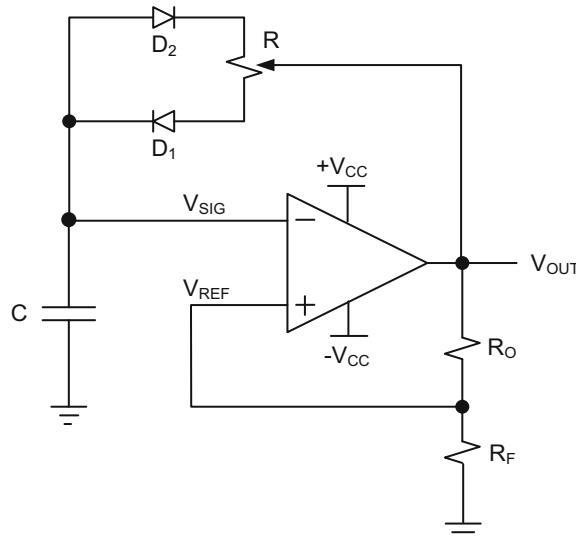
and becomes equal to  $-V_{CC}$ . As a result,  $V_{REF}$  changes its value and becomes:

$$V_{REF} = -V_{CC} \frac{R_F}{R_F + R_O} \quad (8.3)$$

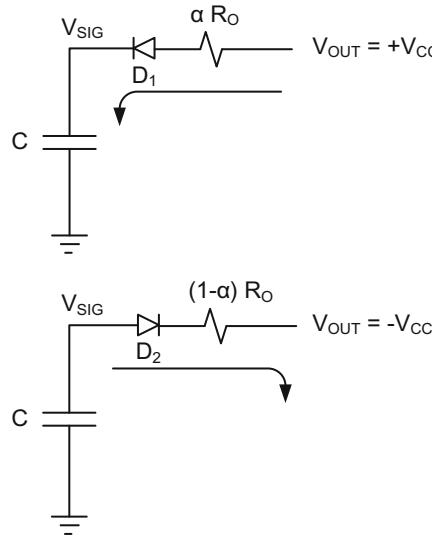
With  $V_{OUT} = -V_{CC}$ , this time the capacitor starts discharging through the diode,  $D_2$ , and the unused portion of  $R$ ,  $(1 - \alpha)R$ , from its initial value of  $\frac{R_F}{R_F + R_O} V_{CC}$  towards  $-V_{CC}$  as shown in the bottom portion of Fig. 8.6. Thus,

$$V_{SIG} = V_{CC} \left( 1 + \frac{R_F}{R_F + R_O} \right) \exp\left(-\frac{t}{(1 - \alpha)RC}\right) - V_{CC} \quad (8.4)$$

When  $V_{SIG}$  reaches  $-\frac{R_F}{R_F + R_O} V_{CC}$ , however, the output changes its polarity to  $+V_{CC}$  and forms a positive reference voltage,  $V_{REF} = V_{CC} \frac{R_F}{R_F + R_O}$ . As a result, the capacitor starts charging through  $\alpha R$  and  $D_1$  again, and  $V_{SIG}$  begins to increase from  $-\frac{R_F}{R_F + R_O} V_{CC}$  to  $\frac{R_F}{R_F + R_O} V_{CC}$ .

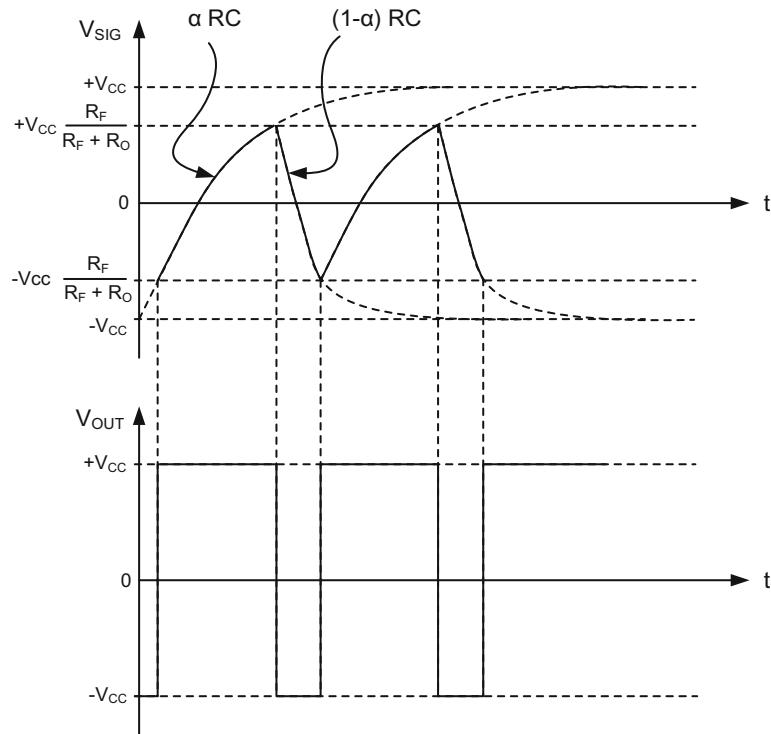


**Fig. 8.5** A simple pulse width modulation (PWM) circuit



**Fig. 8.6** Charge (*top*) and discharge (*bottom*) paths of the PWM circuit in Fig. 8.6

The waveforms of  $V_{SIG}$  and  $V_{OUT}$  are very similar to the ones produced by the square wave generator in Chapter 7, and shown here again in Fig. 8.7. In this figure, the time constants during the rise and the fall of  $V_{SIG}$  are different from each other due to  $\alpha$ . This property enables fine-tuning the pulse width, and therefore the duty cycle of the output waveform.

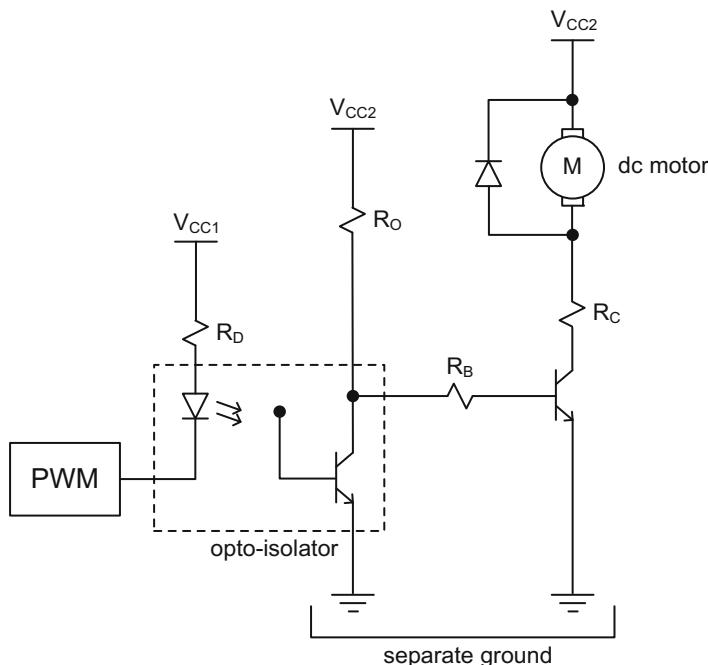


**Fig. 8.7** The waveforms at  $V_{SIG}$  and  $V_{OUT}$  of the PWM circuit in Fig. 8.6

### 8.3 DC Motor Control

Operating a DC motor in different speeds can be achieved by the combination of a PWM circuit cascaded with an opto-isolator circuit as shown in Fig. 8.8. The opto-isolator separates the ground terminals of the noise-receptive digital circuit from the circuit that operates the motor which usually requires high voltages.

The PWM can generate voltages between 0 V and  $+V_{CC}$  to turn on and off the optical transistor in high speeds not too suitable for a relay.



**Fig. 8.8** DC motor control

The pulse duration of the PWM output at  $+V_{CC}$  lasts only micro seconds, during which the output bipolar transistor supplies current to the DC motor. The only way to observe this is through measuring the motor's rpm. Motor's speed can be increased by increasing the time constant of the secondary feedback circuit or  $\alpha$ RC. The motor can practically be turned off if the value of the variable resistance is reduced to  $0 \Omega$ .

---

### 8.4 Servo Control

It is possible to design a pulse generator to control a servo motor. The signal input to an analog servo calls for a pulse with a period of 20 ms as shown in Fig. 8.9.

If the pulse width is maintained at  $T = 1.5$  ms, the servo arm stays at neutral position. Reducing  $T$  to 1.25 ms turns the servo arm to  $-90^\circ$  from the neutral axis. Similarly

increasing T to 1.75 ms rotates the servo arm to  $+90^\circ$ . However, some servo motor manufacturers specify T = 1 ms and T = 2 ms pulse widths for  $-90$  and  $+90^\circ$  arm rotations in their datasheets, respectively, and this requires calibrating servo arm movement for each servo manufacturer.

$$\begin{aligned} T = 1 - 1.25\text{ms} & \text{ servo} = -90^\circ \\ T = 1.5\text{ms} & \text{ servo} = \text{neutral} \\ T = 1.75 - 2\text{ms} & \text{ servo} = +90^\circ \end{aligned}$$

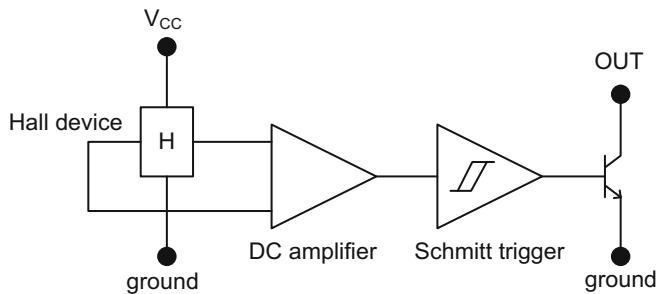


**Fig. 8.9** Servo control waveforms

## 8.5 Hall-Effect Sensor Control

The Hall-effect device studied in Chapter 5 operates in the presence of a magnetic field, and forms a very useful device for systems that require controlling mechanical rotation.

The magnetic properties of Hall-effect devices have enabled manufacturers to devise a simple device resistant to heat, cold, pressure and many other external effects. The basic usage of this device is car industry. However, many other industries that employ magnetic field in the operation of their systems also use this device. Figure 8.10 shows a unipolar Hall-effect device manufactured by Allegro Microsystems.



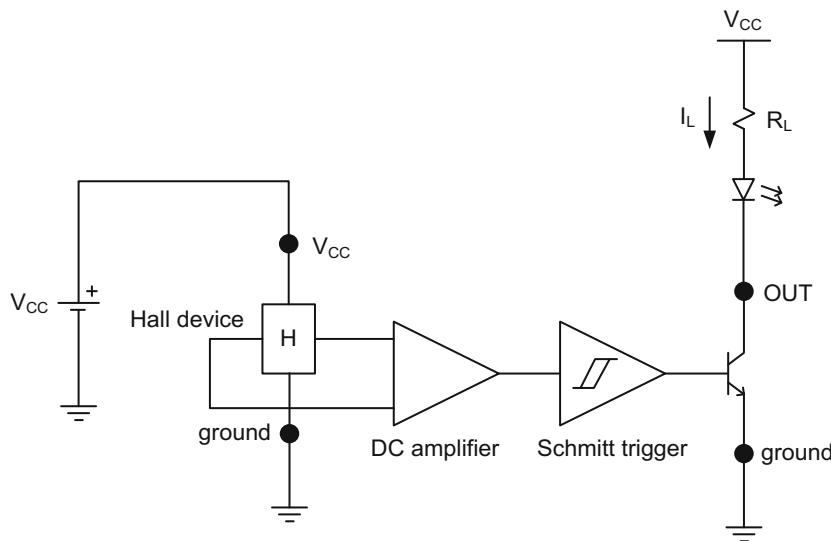
**Fig. 8.10** Unipolar Hall-effect device manufactured by Allegro Microsystems

The package contains a Hall-effect device that produces milli-volt range Hall voltage when exposed to a magnetic field, a DC amplifier that amplifies the Hall voltage to full logic levels at its output, and finally an NPN bipolar transistor with an open collector.

When this device is connected to activate an LED as shown in Fig. 8.11, the exposure to magnetic field enables the Schmitt trigger to produce a high logic output to saturate the NPN transistor. The resulting current,  $I_L$ , turns on the LED and becomes:

$$I_L = \frac{(V_{CC} - V_{LED} - V_{CESAT})}{R_L} \quad (8.5)$$

Here,  $V_{LED}$  is the voltage drop across the LED.  $I_L$  in this equation must be adjusted to be able to saturate the bipolar transistor and turn on the LED.



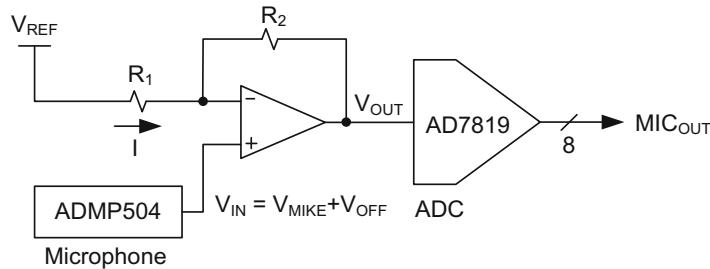
**Fig. 8.11** Usage of the unipolar Hall-effect device to activate LED

The remainder of this chapter is dedicated to numerous design projects that prepare the sensor signal to interface with the microcontroller. Every project uses a different type of sensor and follows a unique data-path to condition the analog signal prior to analog-to-digital conversion. The last two projects integrate digital logic blocks with analog circuitry, therefore they require logic design skills outlined in Chapter 9.

## 8.6 Design Project 1: Designing Front-End Electronics for an Analog Microphone

The first design project simply prepares the analog signal from a microphone for the ADC. The microphone used in Fig. 8.12 is analog microphone ADMP504 produced by Analog Devices. From the manufacturer's datasheet, the microphone produces a maximum voltage

of 0.25 V, which needs to be amplified to a maximum value of 5 V for the analog-to-digital converter, AD7819, also produced by Analog Devices. The microphone produces an output DC offset voltage of 0.8 V that needs to be cleaned from the analog signal before the signal is amplified. Therefore,  $V_{REF}$  in Fig. 8.12 should be adjusted to a certain value that offsets the 0.8 V DC offset in the microphone output signal.



**Fig. 8.12** Microphone front end circuit

According to the datasheet we have:

$$V_{IN} = V_{MIC} + V_{OFF} \quad (8.6)$$

where,  $V_{MIC}$  is the analog voltage produced by the microphone without DC offset, and  $V_{OFF} = 0.8$  V. We also have:

$$V_{REF} = (R_1 + R_2)I + V_{OUT} \quad (8.7)$$

$$V_{REF} = R_1 I + V_{IN} \quad (8.8)$$

From Eq. 8.8,

$$I = \frac{V_{REF} - V_{IN}}{R_1} \quad (8.9)$$

Substituting Eq. 8.9 into Eq. 8.7 yields:

$$V_{REF} = (R_1 + R_2) \frac{V_{REF} - V_{IN}}{R_1} + V_{OUT} \quad (8.10)$$

Rearranging the terms in Eq. 8.10 yields:

$$V_{OUT} = \frac{(R_1 + R_2)}{R_1} V_{IN} - \frac{R_2}{R_1} V_{REF} \quad (8.11)$$

Substituting Eq. 8.6 into Eq. 8.11 yields:

$$\begin{aligned} V_{\text{OUT}} &= \frac{(R_1 + R_2)}{R_1} (V_{\text{MIKE}} + V_{\text{OFF}}) - \frac{R_2}{R_1} V_{\text{REF}} \\ &= \frac{(R_1 + R_2)}{R_1} V_{\text{MIKE}} + \left[ \frac{(R_1 + R_2)}{R_1} V_{\text{OFF}} - \frac{R_2}{R_1} V_{\text{REF}} \right] \end{aligned} \quad (8.12)$$

In order to isolate the term with  $V_{\text{MIKE}}$ , we need to eliminate the terms with  $V_{\text{OFF}}$  and  $V_{\text{REF}}$ . Thus,

$$\frac{(R_1 + R_2)}{R_1} V_{\text{OFF}} = \frac{R_2}{R_1} V_{\text{REF}}$$

or

$$V_{\text{REF}} = \frac{(R_1 + R_2)}{R_2} V_{\text{OFF}} \quad (8.13)$$

Then,

$$V_{\text{OUT}} = \frac{(R_1 + R_2)}{R_1} V_{\text{MIKE}} \quad (8.14)$$

From the datasheets, the maximum value produced by the microphone is  $V_{\text{MIKE}} = 0.25$  V and the maximum value at the input of the ADC is  $V_{\text{OUT}} = 5$  V. Therefore, from Eq. 8.14:

$$\frac{V_{\text{OUT}}}{V_{\text{MIKE}}} = \frac{5}{0.25} = 20 = \frac{(R_1 + R_2)}{R_1}$$

$$R_2 = 19 R_1 \quad (8.15)$$

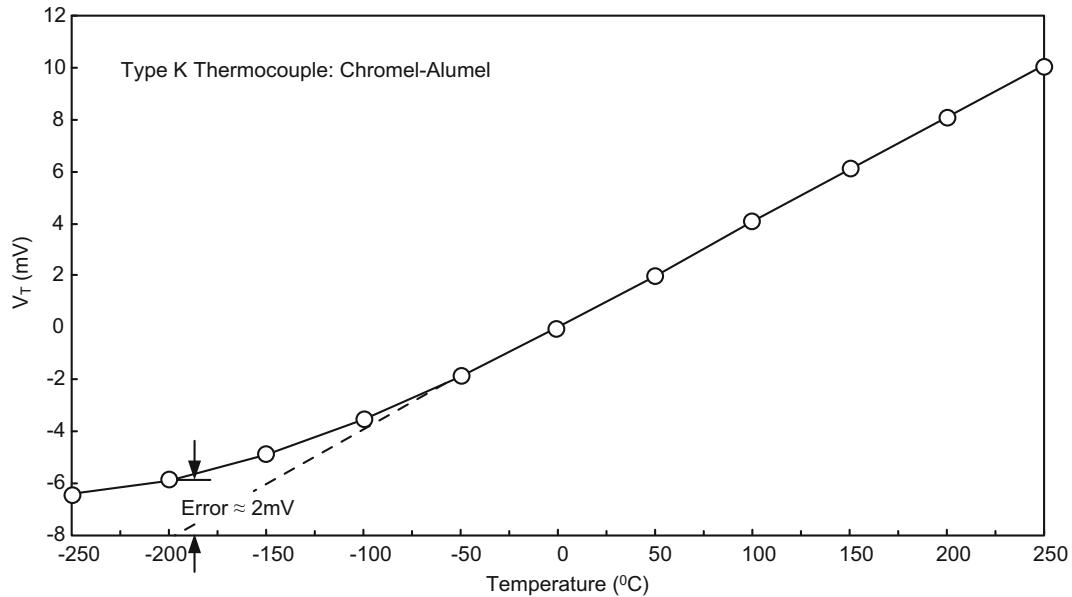
We use  $R_1 = 1 \text{ K}\Omega$  and  $R_2 = 19 \text{ K}\Omega$

Using Eqs. 8.15 in 8.13 yields  $V_{\text{REF}} = 1.05 \times 0.8 = 0.85$  V.

Therefore, we must find a Zener diode at 0.85 V or a potentiometer that can produce 0.85 V. However, the latter choice is prone to temperature fluctuations.

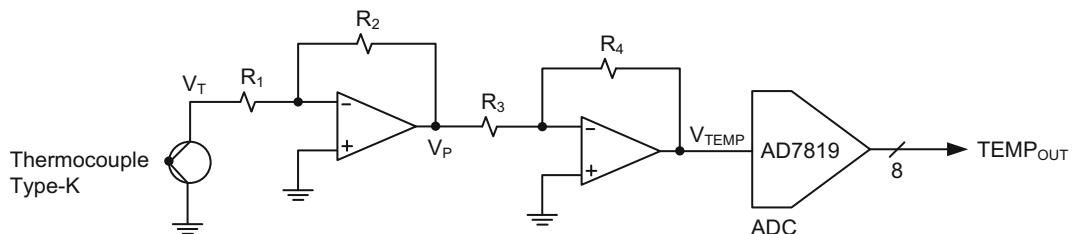
## 8.7 Design Project 2: Designing Front-End Electronics for a Temperature Measurement System

The second design project amplifies the signal from a type-K thermocouple, and prepares it for the ADC. The thermocouple signal changes linearly with temperature as shown in Fig. 8.13. From this figure, the thermocouple signal can be approximated as  $V_T \approx 0.0407 T$  in mV. However, at low temperatures the deviation from linearity introduces error as large as 2 mV.



**Fig. 8.13** Voltage-temperature characteristics of type-K thermocouple

In the first attempt, let us build a circuit that measures ambient temperatures between 0 and 200 °C. This is a very straightforward design that requires only signal amplification as shown in Fig. 8.14.



**Fig. 8.14** A circuit that measures temperatures between 0 and 200 °C

The amplification is performed by two cascaded inverting stages. In the first stage, we have:

$$\frac{V_P}{V_T} = -\frac{R_2}{R_1} \quad (8.16)$$

In the second stage,

$$\frac{V_{TEMP}}{V_P} = -\frac{R_4}{R_3} \quad (8.17)$$

We know the maximum voltages at  $V_T = 8.14 \text{ mV}$  (at  $T = 200 \text{ }^\circ\text{C}$ ) and  $V_{\text{TEMP}} = 5 \text{ V}$ . This requires an overall gain of approximately 614. If the first stage amplifies the thermocouple signal by 20 times, then the second stage will need an amplification of 30.7. Thus,

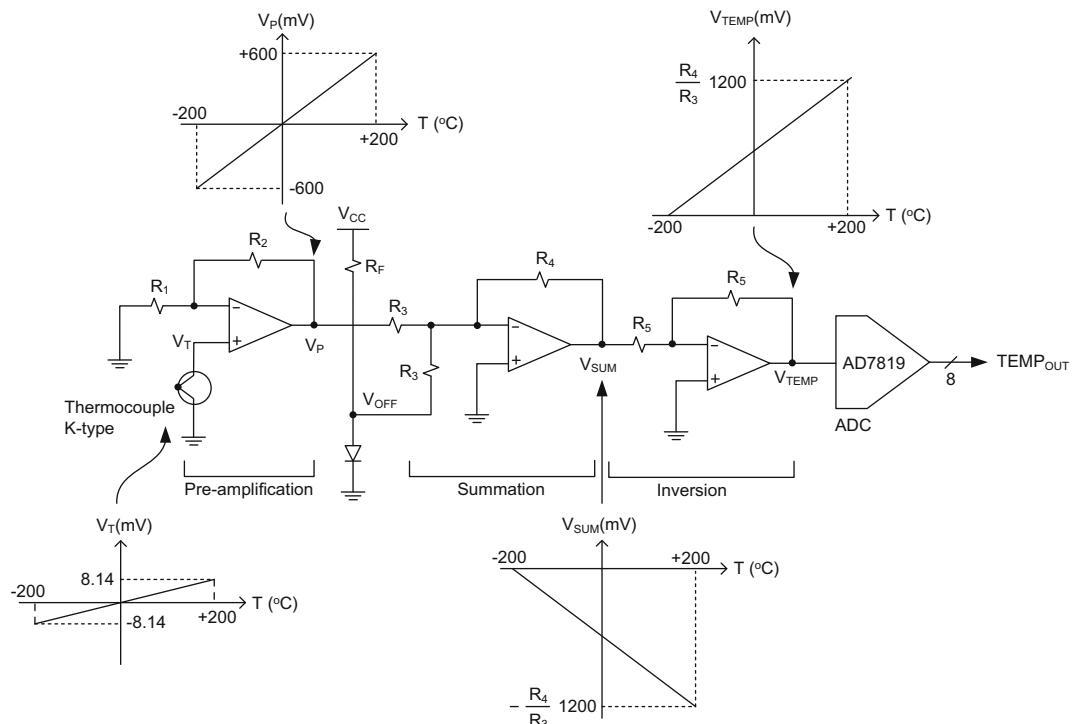
$$R_2 = 20 R_1 \quad (8.18)$$

Similarly,

$$R_4 = 30.7 R_3 \quad (8.19)$$

Selecting  $R_1 = 1 \text{ k}\Omega$ ,  $R_2 = 20 \text{ k}\Omega$ ,  $R_3 = 2 \text{ k}\Omega$ , and  $R_4 \approx 62 \text{ k}\Omega$  according to Eqs. 8.18 and 8.19 will proportionally change the sensor readings at  $V_{\text{TEMP}}$  between 0 and 5 V.

The second circuit in Fig. 8.15 conditions the temperature readings at  $V_T$  between  $-200$  and  $200 \text{ }^\circ\text{C}$ .



**Fig. 8.15** A circuit that measures temperatures between  $-200$  and  $200 \text{ }^\circ\text{C}$

Since the thermocouple voltage readings become negative for temperatures less than  $0 \text{ }^\circ\text{C}$ , the best method is to add a DC offset to the actual thermocouple signal to eliminate the negative values of  $V_T$ . The maximum value of the “extrapolated”  $V_T$  at  $-200 \text{ }^\circ\text{C}$  is  $-8.14 \text{ mV}$  according to Fig. 8.13. This is a very small value to offset. Therefore, a better option is to amplify the thermocouple signal without any DC offset in the first stage, and then

employ an offset voltage for the amplified signal in the next stage. The amplification in the first stage produces the result in Eq. 8.20.

$$\frac{V_P}{V_T} = \frac{R_1 + R_2}{R_1} \quad (8.20)$$

In the second stage, a steady DC offset voltage can be generated by a rectifying diode. Assuming the diode has  $V_F = 600$  mV, the amplification at the first stage must yield:

$$\frac{V_P}{V_T} = \frac{R_1 + R_2}{R_1} = \frac{600 \text{ mV}}{8.14 \text{ mV}} \approx 73.7 \quad (8.21)$$

In this equation, if  $R_1 = 1 \text{ K}\Omega$  then  $R_2 \approx 72 \text{ K}\Omega$ .

This amplification factor makes the value of  $V_P$  to be approximately  $-600$  mV at  $-200^\circ\text{C}$ , and  $+600$  mV at  $+200^\circ\text{C}$ .

For the second stage, when  $V_{OFF} = 600$  mV, then:

$$V_{SUM} = -\frac{R_4}{R_3}(V_P + V_{OFF}) = -\frac{R_4}{R_3}(V_P + 600 \text{ mV}) \quad (8.22)$$

According to Eq. 8.22, when temperature drops to  $-200^\circ\text{C}$  and  $V_P$  becomes  $-600$  mV, then:

$$V_{SUM} = -\frac{R_4}{R_3}(-600 \text{ mV} + 600 \text{ mV}) = 0 \text{ V} \quad (8.23)$$

Similarly, when temperature rises to  $200^\circ\text{C}$  and  $V_P$  becomes  $+600$  mV, then:

$$V_{SUM} = -\frac{R_4}{R_3}(600 \text{ mV} + 600 \text{ mV}) = -\frac{R_4}{R_3}1200 \text{ mV} \quad (8.24)$$

A Zener diode can also be used to create a DC offset in the second stage. The smallest reference voltage from a Zener diode is approximately 1.8 V, and using this value in Eq. 8.21 requires a gain over 200 in the first stage. However, producing a high voltage gain from a single stage amplifier should be avoided because small deviations in resistor values introduce large errors in amplifier gain.

To calculate the biasing resistor value,  $R_F$ , in the second stage requires determining the forward current,  $I_F$ , of the rectifying diode. Assuming that  $V_{CC} = 5$  V, and this particular diode calls for  $I_F = 1$  mA in order to produce  $V_F = 600$  mV across its terminals,  $R_F$  becomes approximately equal to  $4.4 \text{ K}\Omega$  to set up the correct DC offset,  $V_{OFF}$ .

The third stage simply provides a voltage inversion. Thus:

$$V_{TEMP} = -\frac{R_5}{R_5}V_{SUM} = -V_{SUM} = \frac{R_4}{R_3}(V_P + V_{OFF}) \quad (8.25)$$

The maximum value of  $V_{TEMP}$  should be around 5 V before the analog-to-digital converter stage. Thus:

$$V_{TEMP} = 5000 \text{ mV} = \frac{R_4}{R_3}(V_P + V_{OFF}) = \frac{R_4}{R_3} 1200 \text{ mV} \quad (8.26)$$

$$\frac{R_4}{R_3} = \frac{5000 \text{ mV}}{1200 \text{ mV}} \approx 4.2 \quad (8.27)$$

According to Eq. 8.27, we can use  $R_3 = 1.1 \text{ K}\Omega$  and  $R_4 = 4.7 \text{ K}\Omega$ . In Eq. 8.25,  $R_5$  can be taken as  $10 \text{ K}\Omega$ .

All three cascaded stages produce the following output voltage at  $V_{TEMP}$ :

$$V_{TEMP} = \frac{R_4}{R_3}(V_P + V_{OFF}) = \frac{R_4}{R_3} \left( \frac{R_1 + R_2}{R_1} V_T + V_{OFF} \right) \quad (8.28)$$

In this equation,  $V_{TEMP}$  becomes 0 V at  $T = -200 \text{ }^{\circ}\text{C}$  (and  $V_T = -8.14 \text{ mV}$ ). Similarly, at  $T = 200 \text{ }^{\circ}\text{C}$  (and  $V_T = 8.14 \text{ mV}$ )  $V_{TEMP}$  becomes approximately 5 V.

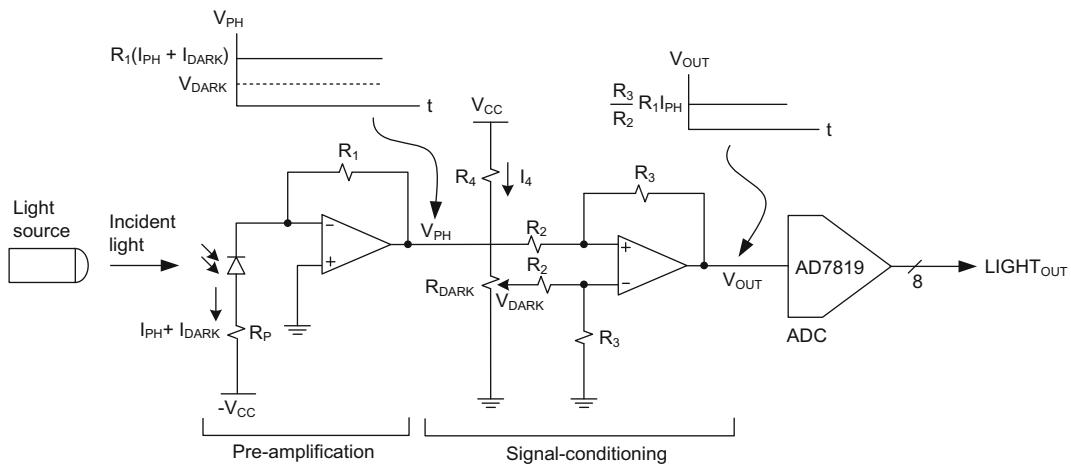
## 8.8 Project 3: Designing Front-End Electronics for a Light Level Measurement System

Another interesting project with operational amplifiers is to measure the intensity of a light source with a photodiode. Once reverse biased, photodiodes produce high levels of reverse saturation current when exposed to light. An OSRAM SFH-213 is a popular photodiode with a spectral range of 400–1100 nm, peaking at 900 nm. It has a relatively small dark (thermal) current,  $I_{DARK}$ , of 1 nA and area of  $1 \text{ mm}^2$ . Its light sensitivity of 550 mA/W makes this diode generate  $5.5 \mu\text{A}$  of photo current (short circuit),  $I_{PH}$ , when exposed to  $10 \text{ W/m}^2$  intensity light source within its spectral range.

The reverse breakdown voltage of SFH-213 is 50 V per manufacturer's datasheet. Therefore, reverse biasing this diode with  $-V_{CC} = -15 \text{ V}$  as shown in Fig. 8.16 is safe although this voltage level may also increase the dark current characteristics beyond what is specified in the datasheet. The pre-amplification stage in Fig. 8.16 produces a gain proportional to the resistor  $R_1$  as shown in Eq. 8.29.

$$V_{PH} = R_1(I_{PH} + I_{DARK}) \quad (8.29)$$

The resistor,  $R_P$ , limits the current through the photodiode and also provides a reverse biased voltage less than  $V_{CC}$  across its terminals in case the power supply voltage is greater than the breakdown voltage.



**Fig. 8.16** A circuit that measures light level ( $-V_{CC}$  is used to bias the photodiode)

The second stage in Fig. 8.16 is primarily used for signal conditioning such as elimination of the dark current component of the photodiode and possibly noise cancellation with some voltage amplification. This stage produces the following voltage gain:

$$V_{OUT} = \frac{R_3}{R_2} (V_{PH} - V_{DARK}) \quad (8.30)$$

Substituting Eq. 8.29 into Eq. 8.30 yields:

$$V_{OUT} = \frac{R_3}{R_2} [R_1(I_{PH} + I_{DARK}) - V_{DARK}] \quad (8.31)$$

In Eq. 8.31, if we tune the potentiometer such that:

$$V_{DARK} = R_1 I_{DARK} \quad (8.32)$$

Then Eq. 8.31 can be reformed as:

$$V_{OUT} = \frac{R_3}{R_2} R_1 I_{PH} \quad (8.33)$$

Using  $I_{PH} = 5.5 \mu\text{A}$  and the maximum value of  $V_{OUT} = 5 \text{ V}$ , Eq. 8.33 can be rewritten as:

$$\frac{R_3}{R_2} R_1 = \frac{5\text{V}}{5.5 \times 10^{-6}\text{A}} \approx 909 \text{ K}\Omega \quad (8.34)$$

According to Eq. 8.34, selecting  $R_2 = 1 \text{ K}\Omega$ ,  $R_1 = 27 \text{ K}\Omega$  produces  $R_3 = 33 \text{ K}\Omega$ .

From Eq. 8.32, the dark voltage becomes:

$$V_{DARK} = 27 \times 10^3 \times 1 \times 10^{-9} = 27 \mu\text{V} \quad (8.35)$$

Also from Eq. 8.29,  $V_{PH}$  becomes approximately equal to 0.15 V.

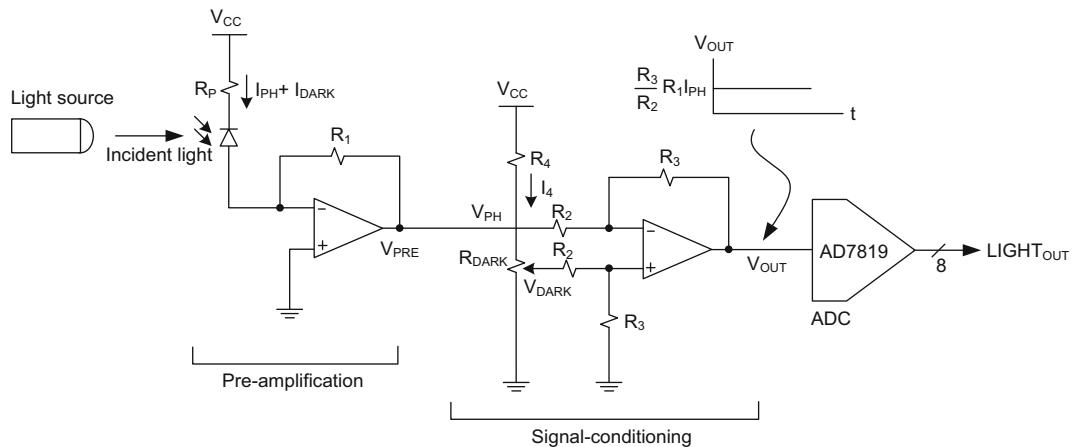
If we allow  $I_4 = 1 \text{ mA}$  at the input of the second stage,  $R_4$  can be approximately equal to:

$$R_4 = \frac{15V}{1 \times 10^{-3}A} \approx 15 K\Omega \quad (8.36)$$

The potentiometer,  $R_{\text{DARK}}$ , should be a small value compared to  $R_4$  such as  $100 \Omega$ . This way, values up to  $100 \mu\text{V}$  can be produced at the  $V_{\text{DARK}}$  input.

The second schematic in Fig. 8.17 is used if the reverse bias voltage for the photodiode is  $+V_{\text{CC}} = 15 \text{ V}$  for the same project. This configuration requires both amplification stages in Fig. 8.16 to be inverting type.

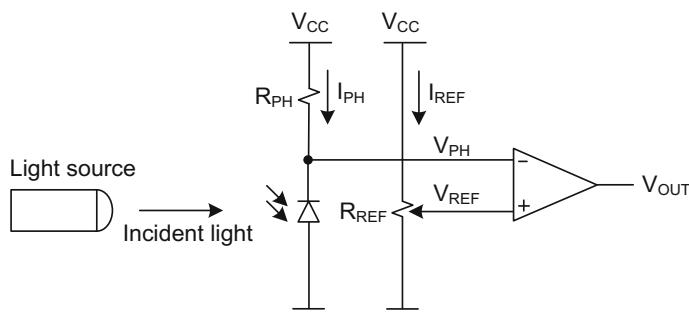
Both circuits use Analog Devices eight-bit ADC, AD7819.



**Fig. 8.17** A circuit that measures light level ( $+V_{\text{CC}}$  is used to bias the photodiode)

## 8.9 Project 4: Designing Photo Detector Circuits

The circuits in Figs. 8.18 and 8.19 are simple on/off circuits sensitive to light, and do not require any ADC to interface with the microcontroller. The sole purpose of these circuits is to create a hardware interrupt for the microcontroller or engage an electro-mechanical device based on light detection.



**Fig. 8.18** Photo detector circuit with an operational amplifier

When the light is exposed to the photodiode in Fig. 8.18, the voltage level at  $V_{PH}$  drops with respect to  $V_{REF}$  because the photo current through the diode,  $I_{PH}$ , becomes much larger than the thermally-generated reverse saturation current,  $I_{DARK}$ . Therefore,

$$V_{PH} = V_{CC} - R_{PH} I_{PH} \quad (8.37)$$

Suppose  $V_{CC} = 5$  V. If we use the same photodiode, SFH-213, from the earlier design with  $I_{PH} = 5.5 \mu A$ , and assume  $V_{PH} = 3$  V (although values less than 3 V or closer to 5 V are also acceptable design entries for  $V_{PH}$ ) when the light is on, then  $R_{PH}$  becomes:

$$R_{PH} = \frac{(V_{CC} - V_{PH})}{I_{PH}} = \frac{5 - 3}{5.5 \times 10^{-6}} \approx 363 K\Omega \quad (8.38)$$

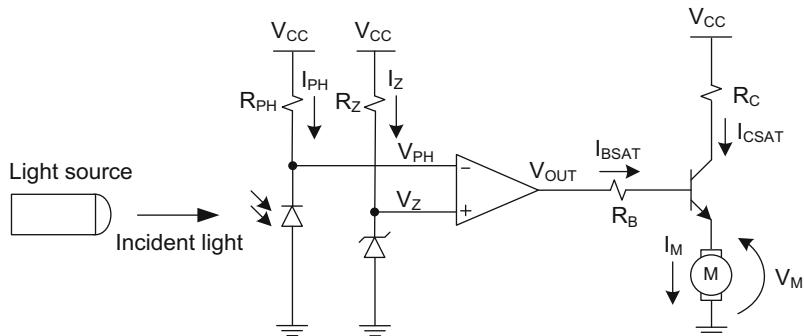
When the light is off,  $I_{PH} \approx 0$  A and  $V_{PH} = V_{CC} = 5$  V.

This means that  $V_{REF}$  should be somewhere between  $V_{PH} = 3$  and 5 V. Suppose  $V_{REF} = 4$  V and  $I_{REF} = 1$  mA. Then the maximum potentiometer value,  $R_{REF}$ , becomes:

$$R_{REF} = \frac{V_{CC}}{I_{REF}} = \frac{5}{1 \times 10^{-3}} = 5 K\Omega \quad (8.39)$$

If we adjust the tap on the potentiometer,  $R_{REF}$ , such that  $V_{REF}$  becomes 4 V, then  $\Delta v = V_{PH} - V_{REF}$  becomes  $\Delta v = 3 - 4 = -1$  V when the light turns on and  $V_{OUT} = 5$  V. When the light turns off,  $V_{REF}$  is still at 4 V, but  $V_{PH}$  becomes 5 V. This produces  $\Delta v = 5 - 4 = +1$  V, and  $V_{OUT}$  becomes 0 V.

The circuit in Fig. 8.19 is an extension of the photo detector circuit in Fig. 8.18, and is used to operate a small DC motor. In this figure, the DC motor spins as long as the light is on, but it turns off when the light is switched off. A Zener diode is used to set a reference voltage for the difference amplifier.



**Fig. 8.19** Photo detector circuit operating a small DC motor

Assume that the power supply voltages are  $+V_{CC} = 15$  V and  $-V_{CC} = 0$  V.  $I_{PH} = 5 \mu A$  when the light turns on, and the DC motor operates at a voltage of  $V_M = 5$  V at  $I_M = 100$  mA.

When the light is on, assume  $V_{PH} = 13$  V. This produces:

$$R_{PH} = \frac{(V_{CC} - V_{PH})}{I_{PH}} = \frac{15 - 13}{5 \times 10^{-6}} = 400 \text{ K}\Omega \quad (8.40)$$

When the light is off, the photodiode only generates negligible reverse biased current ( $I_{DARK}$ ), and  $V_{PH}$  becomes  $V_{CC} = 15$  V.

Therefore,  $V_Z$  must be selected somewhere between  $V_{PH} = 13$  and 15 V although voltages much lower than  $V_{PH} = 13$  V are also acceptable design entries. Thus, assume  $V_Z = 14$  V with  $I_Z = 1$  mA from the Zener diode datasheet.

When the light is on,  $\Delta v = V_{PH} - V_Z$ , becomes  $\Delta v = 13 - 14 = -1$  V, and the output of the operational amplifier,  $V_{OUT}$ , becomes 15 V. This voltage should drive the NPN transistor into saturation. Thus:

$$V_{OUT} = R_B I_{BSAT} + V_{BESAT} + V_M \quad (8.41)$$

$$V_{CC} = R_C I_{CSAT} + V_{CESAT} + V_M \quad (8.42)$$

Substituting  $V_{OUT} = 15$  V,  $V_{CC} = 15$  V, and the typical values of  $V_{BESAT}$  and  $V_{CESAT}$  into Eqs. 8.41 and 8.42 yields:

$$R_B I_{BSAT} = 15 - 0.8 - 5 = 9.2 \text{ V} \quad (8.43)$$

$$R_C I_{CSAT} = 15 - 0.2 - 5 = 9.8 \text{ V} \quad (8.44)$$

If 2N3904 is used in the circuit, then we can ignore the value of  $I_{BSAT}$  compared to  $I_{CSAT}$ . This produces  $I_M = 100$  mA  $\approx I_{CSAT}$ . Then from Eq. 8.44:

$$R_C = \frac{9.8 \text{ V}}{100 \text{ mA}} \approx 100 \Omega \quad (8.45)$$

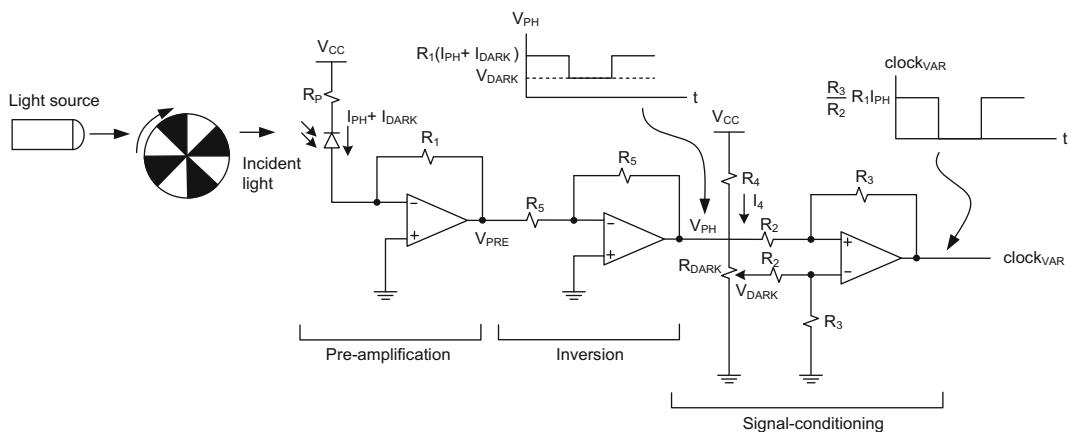
From the I-V characteristics of 2N3904, the NPN transistor goes into saturation at  $I_{CSAT} = 100$  mA when  $I_{BSAT} \geq 100 \mu\text{A}$ . Thus:

$$R_B = \frac{9.2 \text{ V}}{100 \mu\text{A}} \approx 92 \text{ K}\Omega \quad (8.46)$$

When the light turns off,  $\Delta v = V_{PH} - V_Z = 15 - 14 = +1$  V, and  $V_{OUT}$  becomes 0 V. This voltage turns off the NPN transistor and therefore the DC motor.

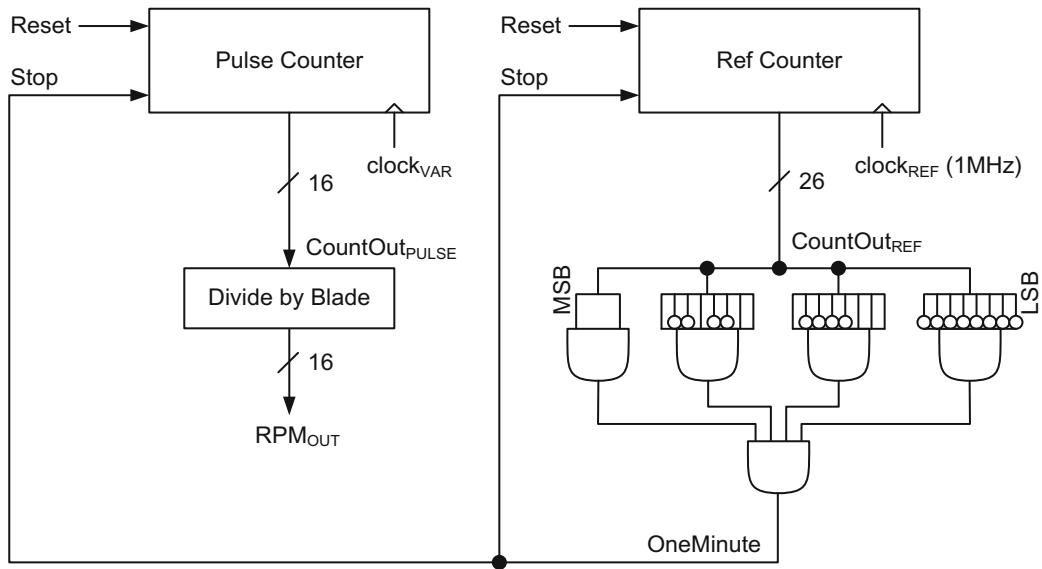
## 8.10 Project 5: Designing Front-End Electronics for an Optoelectronic Tachometer

Among all the other projects the optoelectronic tachometer design is the most demanding because it contains both analog and digital building blocks to complete the design. The analog part shown in Fig. 8.20 is similar to Fig. 8.17 except the incident light beam to the photodiode is chopped by a four-bladed propeller. The data path in Fig. 8.20 amplifies and cleans the pulsed photo current,  $I_{PH}$ , generated by the photodiode. The output of this circuit,  $\text{clock}_{\text{VAR}}$ , indicates a variable clock signal, and is used to compute the prop rpm which varies in time. This variable but low frequency clock signal is tracked by a known reference clock,  $\text{clock}_{\text{REF}}$ , oscillating at 1 MHz, in order to compute the unknown frequency at  $\text{clock}_{\text{VAR}}$ .



**Fig. 8.20** Clock generation circuit for optoelectronic tachometer

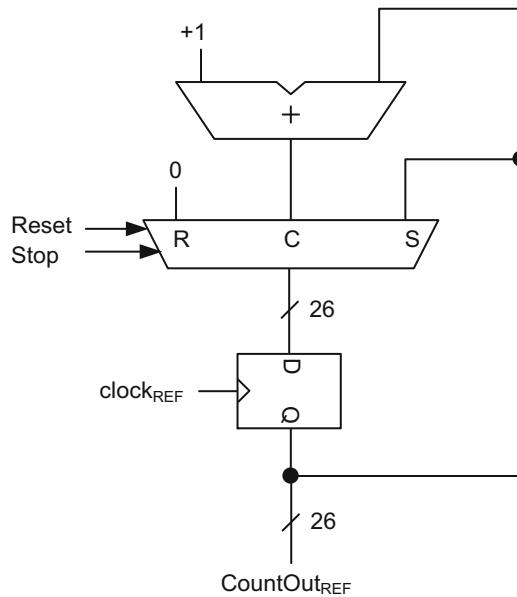
The circuit in Fig. 8.21 shows two counters. The first counter, Pulse Counter, is a 16-bit wide counter and operates with the unknown clock frequency at  $\text{clock}_{\text{VAR}}$ . The second reference counter, Ref Counter, operates with 1 MHz clock at  $\text{clock}_{\text{REF}}$ , and it is 26 bits wide. The output of Ref Counter,  $\text{CountOut}_{\text{REF}}$ , is connected to a decoder whose sole purpose is to detect the end of a minute long time period. This translates to generating logic 1 at OneMinute node for a binary value of 11-1001-0011-1000-0111-0000-0000 at  $\text{CountOut}_{\text{REF}}$  or a waiting for a period of  $6 \times 10^7 \mu\text{s}$  before generating a Stop signal with the 1 MHz reference clock. When the end detection occurs, the pulse generated at OneMinute port stops both the Pulse and Ref Counters. The total number of counts at the output of Pulse Counter,  $\text{CountOut}_{\text{PULSE}}$ , signifies the number of optical pulses received within 1 min time interval. However, depending on the number of blades on the propeller, this number still needs to be divided by the number of blades on the prop. Therefore, a four bladed prop requires  $\text{CountOut}_{\text{PULSE}}$  to be shifted to the right by two. The output of the shifter,  $\text{RPM}_{\text{OUT}}$ , thus reads the number of revolutions of the four-bladed prop per minute.



**Fig. 8.21** An optoelectronic tachometer architecture

Figure 8.22 shows the circuit schematic of the counters. After an external reset, the pulse and reference counter outputs become  $\text{CountOut}_{\text{PULSE}} = 0$  and  $\text{CountOut}_{\text{REF}} = 0$ , respectively, and both counters immediately start counting upwards using different clocks until they reach the 1 min mark. At the 1 min mark, the pulse received from the Stop input switches the port configuration of the 3-1 MUX from C-port to S-port, halting the count mechanism and putting both counters in idle mode. The counters stay in this state until they simultaneously receive an external reset.

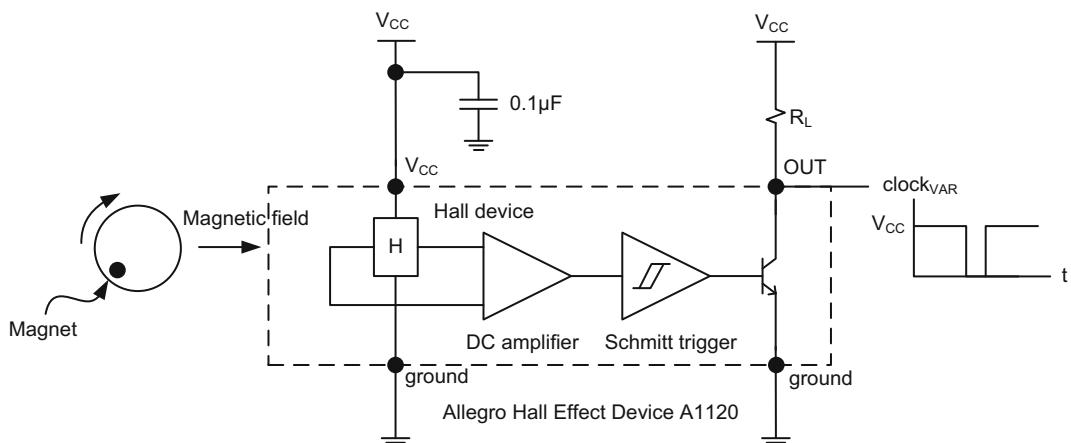
The digital circuits in Fig. 8.21 can be implemented with discrete ICs from Texas instruments. However, it is best to consider a Field-Programmable-Gate-Array (FPGA) platform for implementations that require higher clock frequencies, especially when building timing-critical units such as a reference counter in a similar design.



**Fig. 8.22** Pulse and reference counter circuit for the optoelectronic tachometer

### 8.11 Project 6: Designing Front-End Electronics for a Hall-Effect-Based Tachometer

The circuit in Fig. 8.23 operates similar to the optoelectronic tachometer in the previous project but with a Hall-effect device in place. This project uses Allegro A1120 Hall-effect sensor shown in Fig. 8.10. Every time the sensor is exposed to magnetic field, as small as 35 Gauss, on a rotating disc in Fig. 8.23, the NPN bipolar transistor in the device turns on and pulls down the output node, clock<sub>V<sub>AR</sub></sub>, to 0 V.



**Fig. 8.23** The clock generation circuit for a Hall-effect-based tachometer

When the magnetic field is removed from the sensor or drops below 25 Gauss, on the other hand, the NPN transistor turns off, and the resistor,  $R_L$ , pulls up the  $\text{clock}_{\text{VAR}}$  node to  $V_{\text{CC}}$ . This produces a clock waveform with a very brief duration at 0 V. A capacitor is added between  $V$  and ground to eliminate noise components in the power supply.

Applying this clock signal to the Pulse Counter, and a 1 MHz reference clock to the Ref Counter in Fig. 8.21 produces the same scenario described in the previous project. When 1 min elapses,  $\text{CountOut}_{\text{REF}}$  produces a pulse at OneMinute port, and disables the counting mechanism in both the Pulse and Ref counters. The number of pulses at  $\text{CountOut}_{\text{PULSE}}$  determines the rpm of the rotating disc. There is no need for a Divide by Blade unit, representing the number of blades in a prop, unless there is more than one permanent magnet on the rotating disc.

## Review Questions

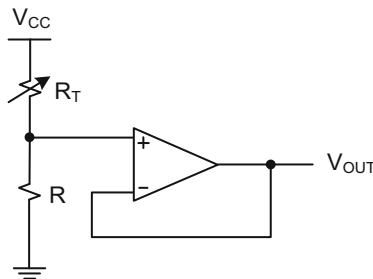
1. Design an analog servo controller that can rotate a servo arm to the left and right by producing pulse widths between 1 and 2 ms in a 20 ms period. How can this servo controller be designed using only digital blocks from Chapter 9?
2. Design a controller that generates pulses for a brushless motor. The design should include fine tuning the pulse width as well as the period of the pulse. Implement the same design using only digital blocks.
3. Design an analog interface to a microcontroller for reading the changing capacitance values of an accelerometer. Refer to Chapter 5 to be acquainted with the physical principles of accelerometers. Use digital logic design concepts from Chapter 9 to support the front-end electronics.
4. Wenner four pin method measures the resistivity of soil. A constant current is applied to the outer pins while the inner pins measure the voltage. The resistivity is calculated based on  $\rho = 2\pi LR$ , where L is the distance between the pins that measures voltage, and R is the resistance obtained from dividing the measured voltage by the constant current. When the resistance value rises above  $R = 30 \Omega$ , irrigation becomes necessary.  $R = 3 \Omega$ , on the other hand, indicates well-moist soil. The irrigation system for soil turns on with an active-high pulse when  $R = 30 \Omega$  at 6 am every morning and stays on as long as the pulse. A 10 min irrigation drops the resistance back to  $3 \Omega$ . Design a system composed of analog and digital components that operate the irrigation system. Refer to Chapter 9 to design the logic blocks required for this system.
5. Design a traffic light module composed of red, yellow and green lights, which controls the flow of traffic on a major street. An optoelectronic device, such as a photodiode, monitors the rate of traffic (cars per minute) on this street. When the rate is 10 cars/min or less, the controller stops the incoming traffic for 1 min and turn on the green light for the cross traffic. The yellow light stays on for 1 s before the module transitions from green to red. Use the digital design principles from Chapter 9 to implement this system.
6. Design the front end electronics of a wind meter. Assume that wind rotates a four bladed propeller connected to the wind meter. Draw the circuit schematic between the prop and the microcontroller. Indicate the appropriate sensor(s) to be used in the system. Refer to Chapter 9 in order to design the digital portion of this system.
7. Distance between two points needs to be measured. A pulsed laser beam is placed at the point of origin; a reflective target is located at the destination. When the laser beam hits the target and reflected back to the unit, the unit effectively measures the time difference between the rising edge of the original pulsed laser beam and the rising edge of the

reflected beam to calculate the distance. Design the front-end electronics between the laser beam and the microcontroller. Use the digital logic design concepts given in Chapter 9 to support the analog electronics.

8. A resistive transducer measures the applied pressure in grams. When there is no force, the sensor shows in excess of  $1 \text{ M}\Omega$  resistance. However, when pressure is applied, the resistance decreases with increasing pressure. Some of the resistance readings are given below:

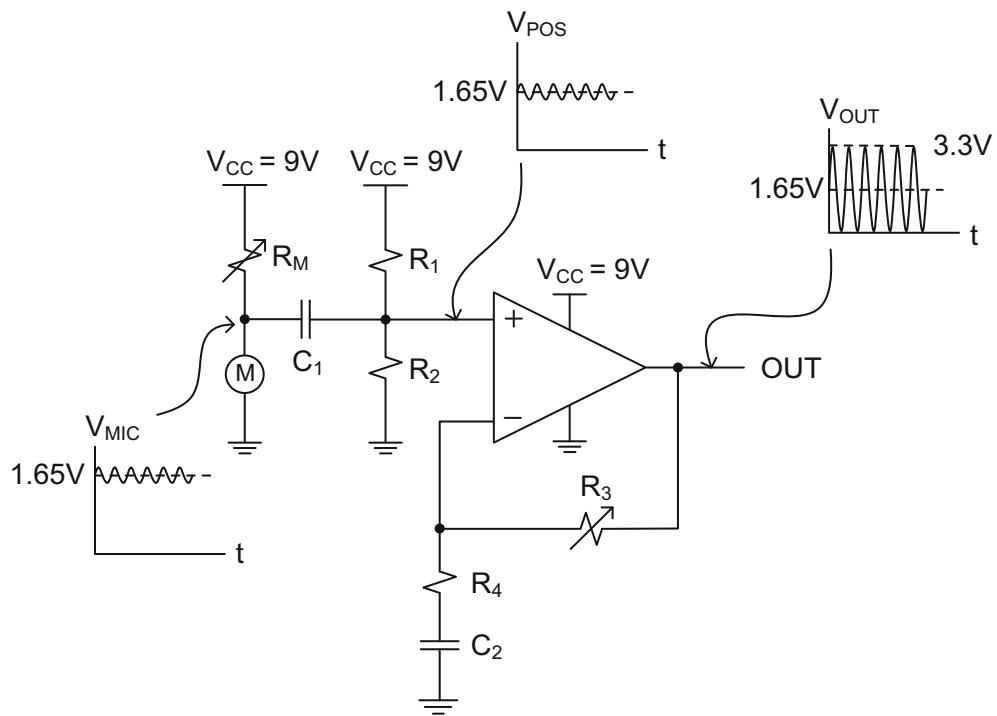
Force = 10 grams	Resistance = $100 \text{ K}\Omega$
Force = 100 grams	Resistance = $10 \text{ K}\Omega$
Force = 1000 grams	Resistance = $1 \text{ K}\Omega$
Force = 10000 grams	Resistance = $0.1 \text{ K}\Omega$

An operational amplifier circuit given below can be used as the initial stage for this system with  $R_T$ , corresponding to the changing resistance of the transducer.



If it becomes necessary, use the digital logic design principles given in Chapter 9 to implement the system.

9. A triple light optoelectronic detection system needs to be designed such that it can distinguish red, yellow and green lights from each other. Conceptualize the type of sensor array to implement such a system, and design its analog front-end electronics for the microcontroller. Refer to Chapter 9 in order to use the necessary digital logic blocks to complete this system.
10. Use the operational amplifier circuit below to amplify human voice with an analog microphone. Use  $V_{CC} = 9 \text{ V}$ ,  $C_1 = C_2 = 10 \mu\text{F}$  and the potentiometers,  $R_M = R_3 = 100 \text{ K}\Omega$ . Here,  $C_1$  and  $C_2$  become short circuits when an Alternating Current (AC) signal (i.e. human voice) is applied to the circuit. However, these capacitors become open circuits in Direct Current (DC) when there is no AC signal. Obtain the values of  $R_1$ ,  $R_2$  and  $R_4$ , in order to obtain the waveforms at  $V_{MIC}$ ,  $V_{POS}$  and  $V_{OUT}$  as shown.



---

## Review of Combinational and Sequential Logic Circuits and Design

This chapter is primarily dedicated to reviewing combinational and sequential logic circuits. Understanding and designing logic blocks are part of the integration effort to build the front-end electronics for microcontrollers.

The first part of this chapter starts with defining logic gates and the concept of truth table which then leads to the implementation of basic logic circuits. Later in the chapter, the concept of Karnaugh maps is introduced in order to minimize gate count, thereby completing the basic requirements of combinational circuit design. Following the minimization techniques, various fundamental logic blocks such as multiplexers, encoders, decoders and one-bit adders are introduced so that they can be used to construct larger scale combinational logic circuits. These include different types of adders such as ripple-carry adder, carry-look-ahead adder, carry-select adder, and the combination of all three types depending on gate count, circuit speed and power consumption. Subtractors, linear and barrel shifters, and array multipliers are also considered combinational mega cells and they will be examined in this chapter.

The definition of clock and system timing are the integral elements for sequential logic circuits. Data in a digital system moves from one storage device to the next by the virtue of a system clock. During its travel, data is routed in and out of different combinational logic blocks, and becomes modified to satisfy a specified functionality.

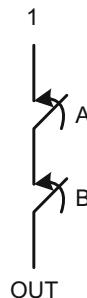
Therefore, in the second part of this chapter we will review the memory devices that store data, and explain the process of designing sequential circuits that require clock input. We will begin with the introduction of two basic memory elements, the latch and the flip-flop. We will explain how data travels between memory elements using timing diagrams, and analyze timing violations as a result of unexpected combinational logic delays on the data path or the clock line. Later in the chapter, we will design the basic sequential building blocks, such as registers, shift registers and counters, and show the Moore-type and the Mealy-type state machines that control data movement. We will study the advantages and disadvantages of these conventional controllers against counter-decoder type controllers in various design environments. We will introduce the concept of block memory and how it is used in a digital system at the end of this chapter. We will conclude the chapter with a comprehensive example of transferring data from one memory block to the next, the use of

timing diagrams in the development of the design, and show how to incrementally build a data-path and a controller using timing diagrams.

## 9.1 Logic Gates

### AND Gate

Assume that the output, OUT, in Fig. 9.1 is at logic 0 when both switches, A and B, are open. Unless both A and B close, the output stays at logic 0.



**Fig. 9.1** Switch representation of a two-input AND gate

A two-input AND gate functions similarly to the circuit in Fig. 9.1. If any two inputs, A and B, of the AND gate in Fig. 9.2 are at logic 0, the gate produces an output, OUT, at logic 0. Both inputs of the gate must be equal to logic 1 in order to produce an output at logic 1. This behavior is tabulated in Table 9.1, which is called a “truth table”.



**Fig. 9.2** Two-input AND gate symbol

**Table 9.1** Two-input AND gate truth table

A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

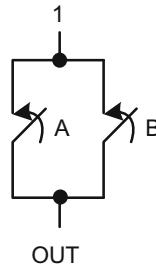
The functional representation of the two-input AND gate is:

$$\text{OUT} = A \cdot B$$

Here, the symbol “.” between inputs A and B represents the AND-function.

### OR Gate

Now, assume a parallel connectivity between switches A and B as shown in Fig. 9.3. OUT becomes to logic 1 if any of the switches close; otherwise the output will stay at logic 0.



**Fig. 9.3** Switch representation of two-input OR gate

A two-input OR gate shown in Fig. 9.4 also functions similarly to the circuit in Fig. 9.3. If any two inputs are at logic 1, the gate produces an output, OUT, at logic 1. Both inputs of the gate must be equal to logic 0 in order to produce an output at logic 0. This behavior is tabulated in the truth table, Table 9.2.



**Fig. 9.4** Two-input OR gate symbol

**Table 9.2** Two-input OR gate truth table

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

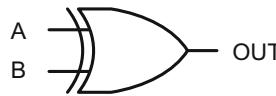
The functional representation of the two-input OR gate is:

$$\text{OUT} = A + B$$

where, the symbol “+” between inputs A and B signifies the OR-function.

### Exclusive OR Gate

A two-input Exclusive OR gate, XOR gate, is shown in Fig. 9.5. The XOR gate produces a logic 0 output if both inputs are equal. Therefore, in many logic applications this gate is used to compare the input logic levels to see if they are equal. The functional behavior of the gate is tabulated in Table 9.3.



**Fig. 9.5** Two-input XOR gate symbol

**Table 9.3** Two-input XOR gate truth table

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

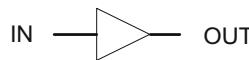
The functional representation of the two-input XOR gate is:

$$\text{OUT} = A \oplus B$$

where, the symbol “ $\oplus$ ” between inputs A and B signifies the XOR function.

### Buffer

A buffer is a single input device whose output is logically equal to its input. The only use of this gate is to be able to supply current to the cumulative capacitive loading of logic gates at the output node. The logical representation of this gate is shown in Fig. 9.6.



**Fig. 9.6** Buffer symbol

## Complementary Logic Gates

All basic logic gates need to have complemented forms. If a single input needs to be complemented, an inverter shown in Fig. 9.7 is used. The inverter truth table is shown in Table 9.4.



**Fig. 9.7** Inverter symbol

**Table 9.4** Inverter truth table

IN	OUT
0	1
1	0

The functional representation of the inverter is:

$$\text{OUT} = \overline{\text{IN}}$$

where, the symbol “—” on top of the input, IN, represents the complement-function.

The complemented form of two-input AND gate is called two-input NAND gate, where “N” signifies negation. The logic representation is shown in Fig. 9.8, where a circle at the output of the gate means complemented output. The truth table of this gate is shown in Table 9.5. Note that all output values in this table are exact opposites of the values given in Table 9.1.



**Fig. 9.8** Two-input NAND gate symbol

**Table 9.5** Two-input NAND gate truth table

A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

The functional representation of the two-input NAND gate is:

$$\text{OUT} = \overline{A \cdot B}$$

Similar to the NAND gate, two-input OR and XOR gates have complemented configurations, called two-input NOR and XNOR gates, respectively.

The symbolic representation and truth table of a two-input NOR gate is shown in Fig. 9.9 and Table 9.6, respectively. Again, all the outputs in Table 9.6 are exact complements of Table 9.2.



**Fig. 9.9** Two-input NOR gate symbol

**Table 9.6** Two-input NOR gate truth table

A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

The functional representation of the two-input NOR gate is:

$$\text{OUT} = \overline{A + B}$$

The symbolic representation and truth table of a two-input XNOR gate is shown in Fig. 9.10 and Table 9.7, respectively. This gate, like its counterpart two-input XOR gate, is often used to detect if input logic levels are equal.



**Fig. 9.10** Two-input XNOR gate symbol

**Table 9.7** Two-input XNOR gate truth table

A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	1

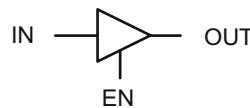
The functional representation of the two-input XNOR gate is:

$$\text{OUT} = \overline{A \oplus B}$$

### Tri-State Buffer and Inverter

It is often necessary to create an open circuit between the input and the output of a logic gate if the gate is not enabled. This need creates two more basic logic gates, the tri-state buffer and tri-state inverter.

The tri-state buffer is shown in Fig. 9.11. Its truth table in Table 9.8 indicates continuity between the input and the output terminals if the control input, EN, is at logic 1; when EN is lowered to logic 0, an open circuit exists between IN and OUT, which is defined as a high impedance, HiZ, condition at the output terminal.

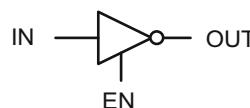


**Fig. 9.11** Tri-state buffer symbol

**Table 9.8** Tri-state buffer truth table

EN	IN	OUT
0	0	HiZ
0	1	HiZ
1	0	0
1	1	1

The tri-state inverter is shown in Fig. 9.12 along with its truth table in Table 9.9. This gate behaves like an inverter when EN input is at logic 1; however, if EN is lowered to logic 0, its output disconnects from its input.



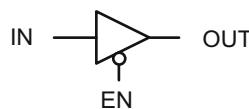
**Fig. 9.12** Tri-state inverter symbol

**Table 9.9** Tri-state inverter truth table

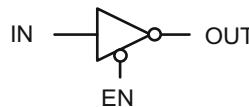
EN	IN	OUT
0	0	HiZ
0	1	HiZ
1	0	1
1	1	0

The control input, EN, to tri-state buffer and inverter can also be complemented in order to produce an active-low enabling scheme.

The tri-state buffer with the active-low enable input in Fig. 9.13 creates continuity when EN = 0.

**Fig. 9.13** Tri-state buffer symbol with complemented enable input

The tri-state inverter with the active-low input in Fig. 9.14 also functions like an inverter when EN is at logic 0, but its output becomes HiZ when EN is changed to logic 1.

**Fig. 9.14** Tri-state inverter symbol with complemented enable input

## 9.2 Boolean Algebra

It is essential to be able to reconfigure logic gates to suit our design goals. Logical reconfigurations may be as simple as re-grouping the inputs of a single gate or complementing the inputs of several gates to reach a design objective.

Identity, commutative, associative, distributive laws and DeMorgan's negation rules are used to perform logical manipulations. Table 9.10 tabulates these laws.

**Table 9.10** Identity, commutative, associative, distributive and DeMorgan's rules

$A \cdot 1 = A$	
$A \cdot 0 = 0$	
$A \cdot A = A$	
$A \cdot \overline{A} = 0$	
$A + 1 = 1$	Identity
$A + 0 = A$	
$A + A = A$	
$A + \overline{A} = 1$	
$\overline{\overline{A}} = A$	
$A \cdot B = B \cdot A$	Commutative
$A + B = B + A$	
$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	Associative
$A + (B + C) = (A + B) + C$	
$A \cdot (B + C) = A \cdot B + A \cdot C$	Distributive
$A + B \cdot C = (A + B) \cdot (A + C)$	
$\overline{A \cdot B} = \overline{A} + \overline{B}$	DeMorgan's
$\overline{A + B} = \overline{A} \cdot \overline{B}$	

**Example 9.1** Reduce  $OUT = A \cdot \overline{B} \cdot C + A \cdot B \cdot C + A \cdot \overline{B}$  using algebraic rules.

$$\begin{aligned} OUT &= A \cdot \overline{B} \cdot C + A \cdot B \cdot C + A \cdot \overline{B} \\ &= A \cdot C \cdot (\overline{B} + B) + A \cdot \overline{B} \\ &= A \cdot (C + \overline{B}) \end{aligned}$$

**Example 9.2** Reduce  $OUT = A + \overline{A} \cdot B$  using algebraic rules.

$$\begin{aligned} OUT &= A + \overline{A} \cdot B \\ &= (A + \overline{A}) \cdot (A + B) \\ &= A + B \end{aligned}$$

**Example 9.3** Reduce  $OUT = A \cdot B + \overline{A} \cdot C + B \cdot C$  using algebraic rules.

$$\begin{aligned} OUT &= A \cdot B + \overline{A} \cdot C + B \cdot C \\ &= A \cdot B + \overline{A} \cdot C + B \cdot C \cdot (A \cdot \overline{A}) \\ &= A \cdot B + \overline{A} \cdot C + A \cdot B \cdot C + \overline{A} \cdot B \cdot C \\ &= A \cdot B \cdot (1 + C) + \overline{A} \cdot C \cdot (1 + B) \\ &= A \cdot B + \overline{A} \cdot C \end{aligned}$$

**Example 9.4** Reduce  $OUT = (A + B) \cdot (\overline{A} + C)$  using algebraic rules.

$$\begin{aligned} OUT &= (A + B) \cdot (\overline{A} + C) \\ &= A \cdot \overline{A} + A \cdot C + \overline{A} \cdot B + B \cdot C \\ &= A \cdot C + \overline{A} \cdot B + B \cdot C \\ &= A \cdot C + \overline{A} \cdot B + B \cdot C \cdot (A + \overline{A}) \\ &= A \cdot C + \overline{A} \cdot B + A \cdot B \cdot C + \overline{A} \cdot B \cdot C \\ &= A \cdot C \cdot (1 + B) + \overline{A} \cdot B \cdot (1 + C) \\ &= A \cdot C + \overline{A} \cdot B \end{aligned}$$

**Example 9.5** Convert  $OUT = (A + B) \cdot \overline{C \cdot D}$  into an OR-combination of two-input AND gates using algebraic laws and DeMorgan's theorem.

$$\begin{aligned} OUT &= (A + B) \cdot \overline{C \cdot D} \\ &= (A + B) \cdot (\overline{C} + \overline{D}) \\ &= A \cdot \overline{C} + A \cdot \overline{D} + B \cdot \overline{C} + B \cdot \overline{D} \end{aligned}$$

**Example 9.6** Convert  $OUT = A \cdot B + C \cdot D$  into an AND-combination of two-input OR gates using algebraic laws and DeMorgan's theorem.

$$\begin{aligned} OUT &= A \cdot B + C \cdot D \\ &= \overline{\overline{A \cdot B} + \overline{C \cdot D}} \\ &= \left( \overline{\overline{A} + \overline{B}} \cdot \overline{\overline{C} + \overline{D}} \right) \end{aligned}$$

### 9.3 Designing Combinational Circuits Using Truth Tables

A combinational circuit is cascaded form of basic logic gates without any feedback from the output to any input. The logic function is obtained from a truth table that specifies the complete functionality of the digital circuit.

**Example 9.7** Using the truth table given in Table 9.11 determine the output function of the digital circuit.

**Table 9.11** An arbitrary truth table with four inputs

A	B	C	D	OUT
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

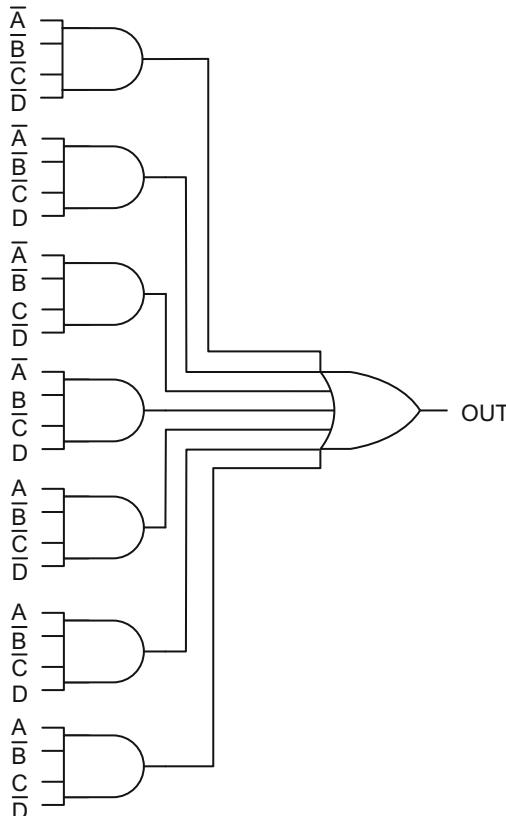
The output function can be expressed either as the OR combination of AND gates or the AND combination of OR gates.

If the output is expressed in terms of AND gates, all output entries that are equal to one in the truth table must be grouped together as a single OR gate.

$$\begin{aligned} \text{OUT} = & \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D \\ & + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot C \cdot \overline{D} \end{aligned}$$

This expression is called the Sum Of Products (SOP), and it contains seven terms each of which is called a “minterm”. In the first minterm,  $\bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \bar{D}$ , each A, B, C and D input is complemented to produce OUT = 1 for the A = B = C = D = 0 entry of the truth table. Each of the remaining six minterms also complies with producing OUT = 1 for their respective input entries.

The resulting combinational circuit is shown in Fig. 9.15.



**Fig. 9.15** AND-OR logic representation of the truth table in Table 9.11

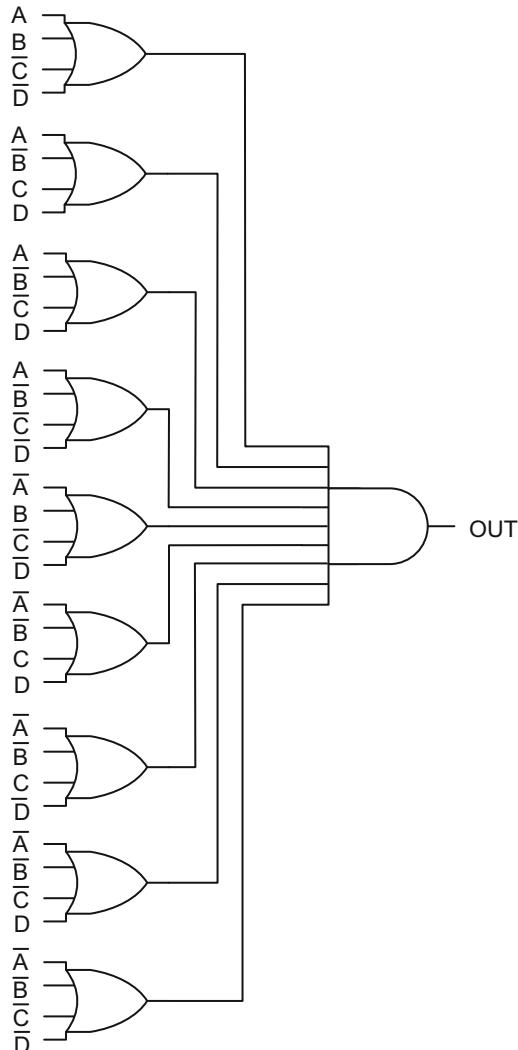
If the output function needs to be expressed in terms of OR gates, all the output entries that are equal to zero in the truth table must be grouped as a single AND gate.

$$\begin{aligned} \text{OUT} = & (A + B + \bar{C} + \bar{D}) \cdot (A + \bar{B} + C + D) \cdot (A + \bar{B} + \bar{C} + D) \\ & \cdot (A + \bar{B} + \bar{C} + \bar{D}) \cdot (\bar{A} + B + \bar{C} + \bar{D}) \cdot (\bar{A} + \bar{B} + C + D) \\ & \cdot (\bar{A} + \bar{B} + C + \bar{D}) \cdot (\bar{A} + \bar{B} + \bar{C} + D) \cdot (\bar{A} + \bar{B} + \bar{C} + \bar{D}) \end{aligned}$$

This expression is called the Product Of Sums (POS), and it contains nine terms each of which is called a “maxterm”. The first maxterm,  $A + B + \bar{C} + \bar{D}$ , produces OUT = 0 for the

$ABCD = 0011$  entry of the truth table. Since the output is formed with a nine-input AND gate, the values of the other maxterms do not matter to produce  $OUT = 0$ . Each of the remaining eight maxterms generates  $OUT = 0$  for their corresponding truth table input entries.

The resulting combinational circuit is shown in Fig. 9.16.



**Fig. 9.16** OR-AND logic representation of the truth table in Table 9.11

## 9.4 Combinational Logic Minimization—Karnaugh Maps

One of the most useful tools in logic design is the use of Karnaugh maps (K-map) to minimize combinational logic functions.

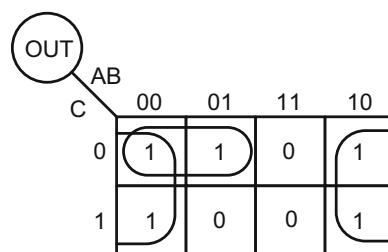
Minimization can be performed in two ways. To obtain SOP form of a minimized logic function, the entries with logic 1 in the truth table must be grouped together in the K-map. To obtain POS form of a minimized logic function, the entries with logic 0 must be grouped together in the K-map.

**Example 9.8** Using the truth table in Table 9.12, determine the minimized SOP and POS output functions. Prove them to be identical.

**Table 9.12** An arbitrary truth table with three inputs

A	B	C	OUT
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

The K-map formed according to the truth table groups 1s to obtain the minimized output function, OUT, in SOP form in Fig. 9.17.



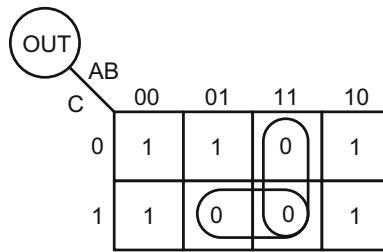
**Fig. 9.17** K-map of the truth table in Table 9.12 to determine SOP

Grouping 1s takes place among neighboring boxes in the K-map where only one variable is allowed to change at a time. For instance, the first grouping of 1s combines ABC = 000 and ABC = 010 as they are in neighboring boxes. Only B changes from logic 0 to logic 1 while A and C stay constant at logic 0. To obtain OUT = 1, both A and C need to be complemented; this produces the first term,  $\overline{A} \cdot \overline{C}$ , for the output function. Similarly, the second grouping of 1s combines the neighboring boxes, ABC = 000, 001, 100 and 101, where both A and C change while B stays constant at logic 0. To obtain OUT = 1, B needs to be complemented; this generates the second term,  $\overline{B}$ , for the output function.

This means that either the term  $\overline{A} \cdot \overline{C}$  or  $\overline{B}$  makes OUT equal to logic 1. Therefore, the minimized output function, OUT, in the SOP form is:

$$\text{OUT} = \overline{B} + \overline{A} \cdot \overline{C}$$

Grouping 0s produces the minimized POS output function as shown in Fig. 9.18.



**Fig. 9.18** K-map of the truth table in Table 9.12 to determine POS

This time, the first grouping of 0s combines the boxes,  $ABC = 011$  and  $111$ , where A changes from logic 0 to logic 1 while B and C stay constant at logic 1. This grouping targets  $\text{OUT} = 0$ , which requires both B and C to be complemented. As a result, the first term of the output function,  $\overline{B} + \overline{C}$ , is generated. The second grouping combines  $ABC = 110$  and  $111$  where C changes value while A and B are equal to logic 1. To obtain  $\text{OUT} = 0$ , both A and B need to be complemented. Consequently, the second term,  $\overline{A} + \overline{B}$ , is generated.

Either of the terms  $\overline{B} + \overline{C}$  or  $\overline{A} + \overline{B}$  makes  $\text{OUT} = 0$ . Therefore, the minimized output function in the POS form becomes:

$$\text{OUT} = (\overline{B} + \overline{C}) \cdot (\overline{A} + \overline{B})$$

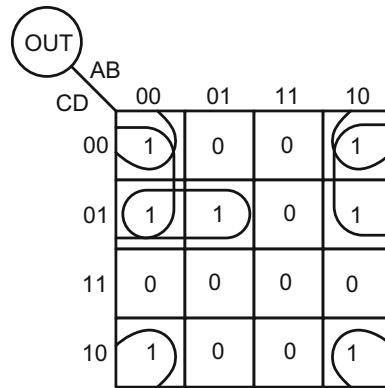
To find out if the SOP and POS forms are identical, one can manipulate the POS using the algebraic rules given earlier.

$$\begin{aligned} \text{OUT} &= (\overline{B} + \overline{C}) \cdot (\overline{A} + \overline{B}) \\ &= \overline{A} \cdot \overline{B} + \overline{B} \cdot \overline{B} + \overline{A} \cdot \overline{C} + \overline{B} \cdot \overline{C} \\ &= \overline{A} \cdot \overline{B} + \overline{B} + \overline{A} \cdot \overline{C} + \overline{B} \cdot \overline{C} \\ &= \overline{B} \cdot (\overline{A} + 1 + \overline{C}) + \overline{A} \cdot \overline{C} \\ &= \overline{B} + \overline{A} \cdot \overline{C} \end{aligned}$$

However, this is the SOP form of the output function derived above.

**Example 9.9** Using the truth table in Example 9.7 determine the minimized SOP and POS output functions.

To obtain an output function in SOP form, 1s in the K-map in Fig. 9.19 is grouped together as shown below.

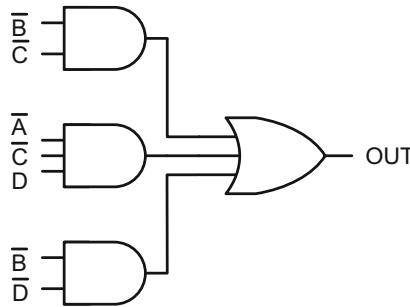


**Fig. 9.19** K-map of the truth table in Table 9.11 to determine SOP

The minimized output function contains only three minterms compared to seven minterms in Example 9.7. Also, the minterms are reduced to groups of two or three inputs instead of four.

$$\text{OUT} = \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{C} \cdot D + \overline{B} \cdot \overline{D}$$

The resultant combinational circuit is shown in Fig. 9.20.

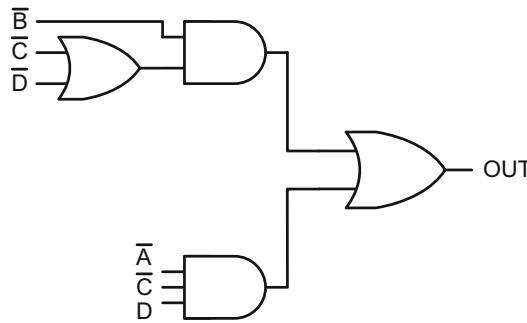


**Fig. 9.20** Minimized logic circuit in SOP form from the K-map in Fig. 9.19

Further minimization can be achieved algebraically, which then reduces the number of terms from three to two.

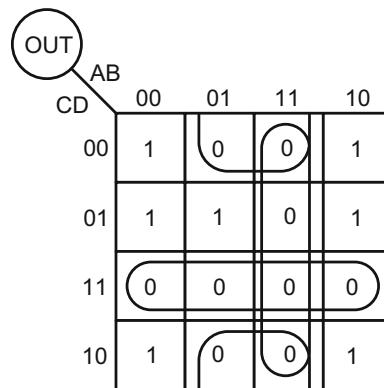
$$\text{OUT} = \overline{B} \cdot (\overline{C} + \overline{D}) + \overline{A} \cdot \overline{C} \cdot D$$

The corresponding combinational circuit is shown in Fig. 9.21.



**Fig. 9.21** Logic circuit in Fig. 9.20 after algebraic minimizations are applied

To obtain a POS output function, 0s are grouped together as shown in Fig. 9.22.

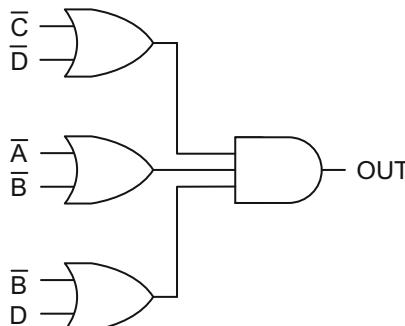


**Fig. 9.22** K-map of the truth table in Table 9.11 to determine POS

The minimized output function contains only three maxterms compared to nine in Example 9.7. Also, the maxterms are reduced to groups of two inputs instead of four.

$$\text{OUT} = (\overline{C} + \overline{D}) \cdot (\overline{A} + \overline{B}) \cdot (\overline{B} + D)$$

The resultant combinational circuit is shown in Fig. 9.23.



**Fig. 9.23** Minimized logic circuit in POS form from the K-map in Fig. 9.22

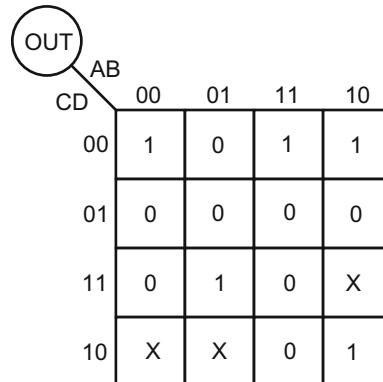
**Example 9.10** Determine if the minimized SOP and POS output functions in Example 9.9 are identical to each other.

Rewriting the POS form of OUT from Example 9.9 and using the algebraic laws shown earlier, this expression can be re-written as:

$$\begin{aligned} \text{OUT} &= (\overline{C} + \overline{D}) \cdot (\overline{A} + \overline{B}) \cdot (\overline{B} + D) \\ &= (\overline{A} \cdot \overline{C} + \overline{A} \cdot \overline{D} + \overline{B} \cdot \overline{C} + \overline{B} \cdot \overline{D}) \cdot (\overline{B} + D) \\ &= \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot \overline{D} + \overline{B} \cdot \overline{C} + \overline{B} \cdot \overline{D} \\ &\quad + \overline{A} \cdot \overline{C} \cdot D + \overline{B} \cdot \overline{C} \cdot D \\ &= \overline{B} \cdot \overline{C} + \overline{B} \cdot \overline{D} + \overline{A} \cdot \overline{C} \cdot D \end{aligned}$$

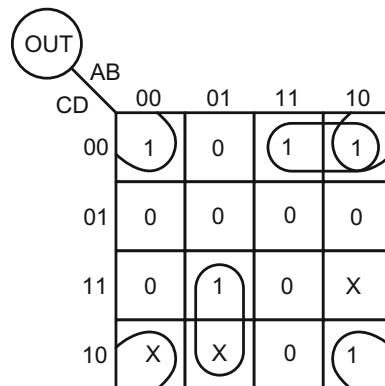
The result is identical to the SOP expression given in Example 9.9.

**Example 9.11** Determine the minimal SOP and POS forms of the output function, OUT, from the K-map in Fig. 9.24. Note that the “X” sign corresponds to a “don’t care” condition that represents either logic 0 or logic 1.



**Fig. 9.24** An arbitrary K-map with “don’t care” entries

For SOP, we group 1s in the K-map in Fig. 9.25. Boxes with “don’t care” are used as 1s to achieve a minimal SOP expression.

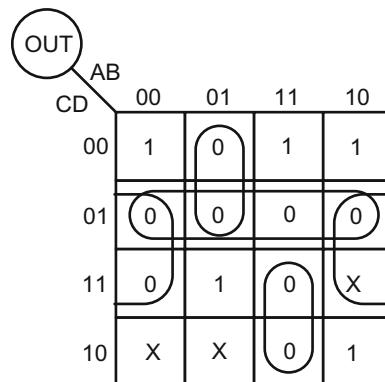


**Fig. 9.25** Grouping to determine SOP form for the K-map in Fig. 9.24

The SOP functional expression for OUT is:

$$\text{OUT} = A \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot B \cdot C + \overline{B} \cdot \overline{D}$$

For POS, we group 0s in the K-map in Fig. 9.26. Boxes with “don’t care” symbols are used as 0s to achieve a minimal POS expression.



**Fig. 9.26** Grouping to determine POS form for the K-map in Fig. 9.24

The POS functional expression for OUT is:

$$\text{OUT} = (C + \overline{D}) \cdot (B + \overline{D}) \cdot (A + \overline{B} + C) \cdot (\overline{A} + \overline{B} + \overline{C})$$

To show that the SOP and POS expressions are identical, we start with the POS expression using the algebraic manipulations described earlier in Table 9.10.

$$\begin{aligned}
 \text{OUT} &= (\text{C} + \overline{\text{D}}) \cdot (\text{B} + \overline{\text{D}}) \cdot (\text{A} + \overline{\text{B}} + \text{C}) \cdot (\overline{\text{A}} + \overline{\text{B}} + \overline{\text{C}}) \\
 &= (\text{B} \cdot \text{C} + \text{C} \cdot \overline{\text{D}} + \text{B} \cdot \overline{\text{D}} + \overline{\text{D}}) \cdot (\text{A} \cdot \overline{\text{B}} + \text{A} \cdot \overline{\text{C}} + \overline{\text{A}} \cdot \overline{\text{B}} + \overline{\text{B}} \cdot \overline{\text{C}} + \overline{\text{A}} \cdot \text{C} + \overline{\text{B}} \cdot \text{C}) \\
 &= (\text{B} \cdot \text{C} + \overline{\text{D}}) \cdot (\text{A} \cdot \overline{\text{C}} + \overline{\text{A}} \cdot \text{C} + \overline{\text{B}}) \\
 &= \overline{\text{A}} \cdot \text{B} \cdot \text{C} + \text{A} \cdot \overline{\text{C}} \cdot \overline{\text{D}} + \overline{\text{A}} \cdot \text{C} \cdot \overline{\text{D}} + \overline{\text{B}} \cdot \overline{\text{D}} \\
 &= \overline{\text{A}} \cdot \text{B} \cdot \text{C} + \text{A} \cdot \overline{\text{C}} \cdot \overline{\text{D}} + \overline{\text{B}} \cdot \overline{\text{D}} + \overline{\text{A}} \cdot \text{C} \cdot \overline{\text{D}} \\
 &= \overline{\text{A}} \cdot \text{B} \cdot \text{C} + \text{A} \cdot \overline{\text{C}} \cdot \overline{\text{D}} + \overline{\text{B}} \cdot \overline{\text{D}} + \overline{\text{A}} \cdot \text{B} \cdot \text{C} \cdot \overline{\text{D}} + \overline{\text{A}} \cdot \overline{\text{B}} \cdot \text{C} \cdot \overline{\text{D}} \\
 &= \overline{\text{A}} \cdot \text{B} \cdot \text{C} \cdot (1 + \overline{\text{D}}) + \text{A} \cdot \overline{\text{C}} \cdot \overline{\text{D}} + \overline{\text{B}} \cdot \overline{\text{D}} \cdot (1 + \overline{\text{A}} \cdot \text{C}) \\
 &= \overline{\text{A}} \cdot \text{B} \cdot \text{C} + \text{A} \cdot \overline{\text{C}} \cdot \overline{\text{D}} + \overline{\text{B}} \cdot \overline{\text{D}}
 \end{aligned}$$

The result is identical to the minimal SOP expression for OUT shown above.

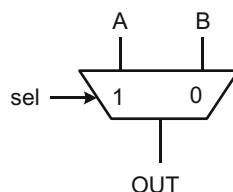
## 9.5 Basic Logic Blocks

### 2-1 Multiplexer

A 2-1 multiplexer (MUX) is one of the most versatile logic elements in logic design. It is defined as follows:

$$\text{OUT} = \begin{cases} \text{A} & \text{if } \text{sel} = 1 \\ \text{B} & \text{else} \end{cases}$$

A functional diagram of the 2-1 MUX is given in Fig. 9.27. According to the functional description of this device, when  $\text{sel} = 1$  input A is passed through the device to become its output. When  $\text{sel} = 0$  input B is passed through the device to become its output.



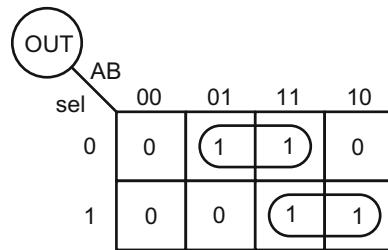
**Fig. 9.27** 2-1 MUX symbol

According to this definition, the truth table in Table 9.13 can be formed:

**Table 9.13** 2-1 MUX truth table

sel	A	B	OUT
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

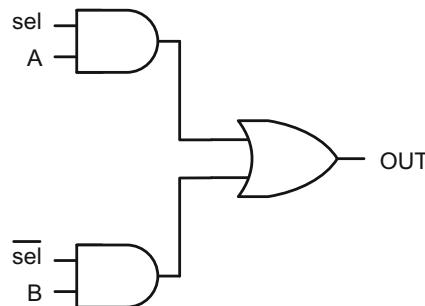
Now, let us transfer the output values from the truth table to the K-map in Fig. 9.28.

**Fig. 9.28** 2-1 MUX K-map

Grouping 1s in the K-map reveals the minimal output function of the 2-1 MUX in SOP form:

$$\text{OUT} = \text{sel} \cdot A + \overline{\text{sel}} \cdot B$$

The corresponding combinational circuit is shown in Fig. 9.29.

**Fig. 9.29** 2-1 MUX logic circuit

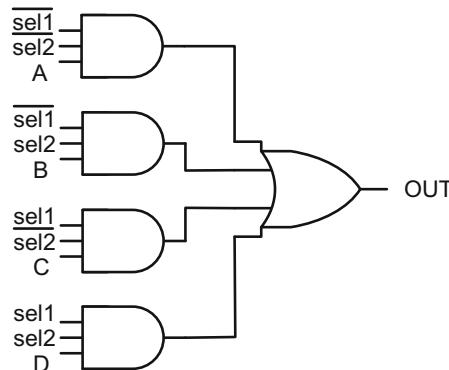
## 4-1 Multiplexer

Considering A, B, C and D are the inputs, the functional description of a 4-1 MUX becomes as follows:

$$\text{OUT} = \begin{cases} A & \text{if } \text{sel1} = 0 \text{ and } \text{sel2} = 0 \\ B & \text{if } \text{sel1} = 0 \text{ and } \text{sel2} = 1 \\ C & \text{if } \text{sel1} = 1 \text{ and } \text{sel2} = 0 \\ D & \text{else} \end{cases}$$

According to this description, we can form a truth table and obtain the minimal SOP or POS expression for OUT. However, it is quite easy to decipher the SOP expression for OUT from the description above. The AND-combination of A, complemented sel1 and complemented sel2 inputs constitute the first minterm of our SOP. The second minterm should contain B, complemented sel1 and uncomplemented sel2 according to the description above. Similarly, the third minterm contains C, uncomplemented sel1 and complemented sel2. Finally, the last minterm contains D, uncomplemented sel1 and uncomplemented sel2 control inputs. Therefore, the SOP expression for the 4-1 MUX becomes equal to the logic expression below and is implemented in Fig. 9.30.

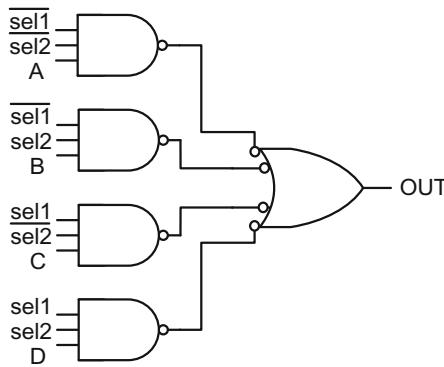
$$\text{OUT} = \overline{\text{sel1}} \cdot \overline{\text{sel2}} \cdot A + \overline{\text{sel1}} \cdot \text{sel2} \cdot B + \text{sel1} \cdot \overline{\text{sel2}} \cdot C + \text{sel1} \cdot \text{sel2} \cdot D$$



**Fig. 9.30** 4-1 MUX logic circuit in SOP form

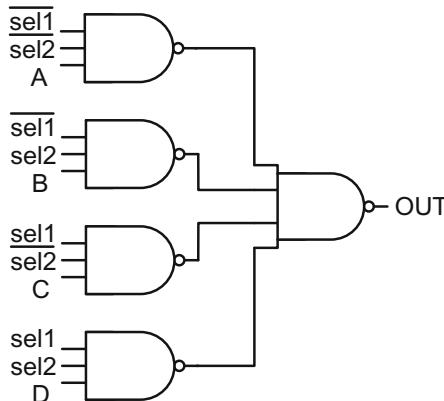
However, implementing 4-1 MUX this way is not advantageous due to the amount of gate delays; a three-input AND gate is a serial combination of a three-input NAND and an inverter, and similarly a four-input OR connects a four-input NOR to an inverter. Therefore, we obtain a minimum of four gate delays instead of two according to this circuit.

Logic translations are possible to reduce the gate delay. The first stage of this process is to complement the outputs of all four three-input AND gates. This necessitates complementing the inputs of the four-input OR gate, and results in a circuit in Fig. 9.31.



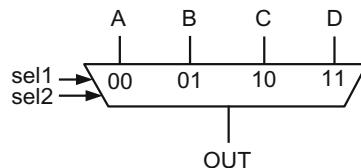
**Fig. 9.31** Logic conversion of 4-1 MUX in Fig. 9.30

However, an OR gate with complemented inputs is equivalent to a NAND gate. Therefore, the circuit in Fig. 9.32 becomes optimal for implementation purposes because the total MUX delay is only the sum of a three-input NAND and a four-input NAND gate delays instead of the earlier four gate delays.



**Fig. 9.32** 4-1 MUX logic circuit in NAND-NAND form

The symbolic diagram of the 4-1 MUX is shown in Fig. 9.33.



**Fig. 9.33** 4-1 MUX symbol

## Encoders

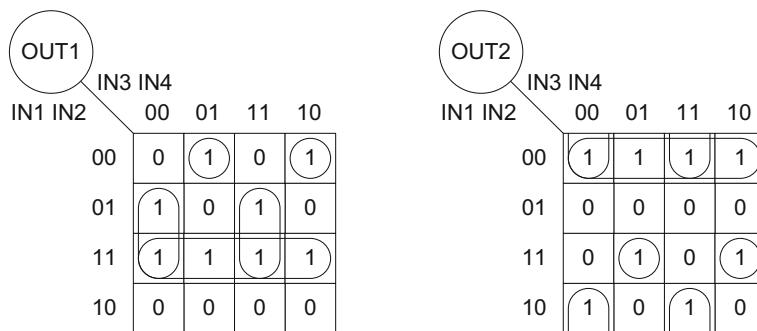
Encoders are combinational logic blocks that receive  $2^N$  number of inputs and produce N number of encoded outputs.

**Example 9.12** Generate an encoding logic from the truth table given in Table 9.14.

**Table 9.14** An arbitrary encoder truth table with four inputs

IN1	IN2	IN3	IN4	OUT1	OUT2
0	0	0	0	0	1
0	0	0	1	1	1
0	0	1	0	1	1
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	0
0	1	1	0	0	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	0	1	0	0
1	0	1	0	0	0
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	0

The K-maps shown in Fig. 9.34 groups 1s and produces the SOP expressions for OUT1 and OUT2.



**Fig. 9.34** K-maps of the truth table in Table 9.14

$$\begin{aligned}
 \text{OUT1} &= \text{IN1} \cdot \text{IN2} + \overline{\text{IN1}} \cdot \overline{\text{IN2}} \cdot \text{IN3} \cdot \overline{\text{IN4}} + \overline{\text{IN1}} \cdot \overline{\text{IN2}} \cdot \overline{\text{IN3}} \cdot \text{IN4} \\
 &\quad + \text{IN2} \cdot \overline{\text{IN3}} \cdot \overline{\text{IN4}} + \text{IN2} \cdot \text{IN3} \cdot \text{IN4} \\
 &= \text{IN1} \cdot \text{IN2} + \overline{\text{IN1}} \cdot \overline{\text{IN2}} \cdot (\text{IN3} \oplus \text{IN4}) + \text{IN2} \cdot (\overline{\text{IN3}} \oplus \overline{\text{IN4}})
 \end{aligned}$$

$$\begin{aligned}
 \text{OUT2} &= \overline{\text{IN1}} \cdot \overline{\text{IN2}} + \overline{\text{IN2}} \cdot \overline{\text{IN3}} \cdot \overline{\text{IN4}} + \overline{\text{IN2}} \cdot \text{IN3} \cdot \text{IN4} \\
 &\quad + \text{IN1} \cdot \text{IN2} \cdot \overline{\text{IN3}} \cdot \text{IN4} + \text{IN1} \cdot \text{IN2} \cdot \text{IN3} \cdot \overline{\text{IN4}} \\
 &= \overline{\text{IN1}} \cdot \overline{\text{IN2}} + \overline{\text{IN2}} \cdot (\overline{\text{IN3}} \oplus \overline{\text{IN4}}) + \text{IN1} \cdot \text{IN2} \cdot (\text{IN3} \oplus \text{IN4})
 \end{aligned}$$

## Decoders

Decoders are combinational logic blocks used to decode encoded inputs. An ordinary decoder takes N inputs and produces  $2^N$  outputs.

**Example 9.13** Design a line decoder in which an active high enable signal activates only one of eight independent outputs according to truth table in Table 9.15. When the enable signal is lowered to logic 0, all eight outputs are disabled and stay at logic 0.

**Table 9.15** Truth table of a line decoder with three inputs with enable

EN	IN[2]	IN[1]	IN[0]	OUT[7]	OUT[6]	OUT[5]	OUT[4]	OUT[3]	OUT[2]	OUT[1]	OUT[0]
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

In this table, all outputs become logic 0 when the enable signal, EN, is at logic 0. However, when EN = 1, activation of the output starts. For each three-bit input entry, there is always one output at logic 1. For example, when IN[2] = IN[1] = IN[0] = 0, OUT[0] becomes active and equals to logic 1 while all other outputs stay at logic 0. IN[2] = IN[1] = IN[0] = 1 activates OUT[7] and disables all other outputs..

We can produce each output expression from OUT[7] to OUT[0] simply by reading the input values from the truth table. The accompanying circuit is composed of eight AND gates, each with four inputs as shown in Fig. 9.35.

$$\text{OUT}[7] = \text{EN} \cdot \text{IN}[2] \cdot \text{IN}[1] \cdot \text{IN}[0]$$

$$\text{OUT}[6] = \text{EN} \cdot \text{IN}[2] \cdot \text{IN}[1] \cdot \overline{\text{IN}[0]}$$

$$\text{OUT}[5] = \text{EN} \cdot \text{IN}[2] \cdot \overline{\text{IN}[1]} \cdot \text{IN}[0]$$

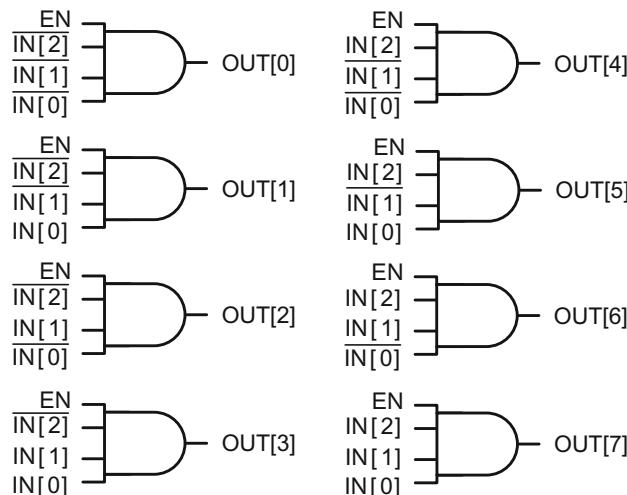
$$\text{OUT}[4] = \text{EN} \cdot \text{IN}[2] \cdot \overline{\text{IN}[1]} \cdot \overline{\text{IN}[0]}$$

$$\text{OUT}[3] = \text{EN} \cdot \overline{\text{IN}[2]} \cdot \text{IN}[1] \cdot \text{IN}[0]$$

$$\text{OUT}[2] = \text{EN} \cdot \overline{\text{IN}[2]} \cdot \text{IN}[1] \cdot \overline{\text{IN}[0]}$$

$$\text{OUT}[1] = \text{EN} \cdot \overline{\text{IN}[2]} \cdot \overline{\text{IN}[1]} \cdot \text{IN}[0]$$

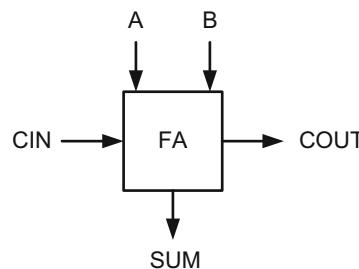
$$\text{OUT}[0] = \text{EN} \cdot \overline{\text{IN}[2]} \cdot \overline{\text{IN}[1]} \cdot \overline{\text{IN}[0]}$$



**Fig. 9.35** Logic circuit of a line decoder in Table 9.15

### One-Bit Full Adder

A one-bit full adder has three inputs: A, B, and carry-in (CIN), and two outputs: sum (SUM) and carry-out (COUT). The symbolic representation of a full adder is shown in Fig. 9.36.



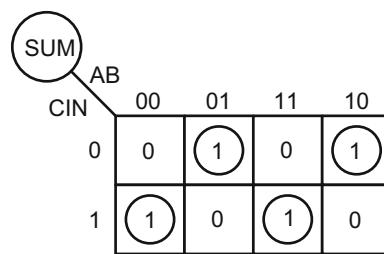
**Fig. 9.36** One-bit full adder symbol

A one-bit full adder simply adds the contents of its two inputs, A and B, to the contents of CIN, and forms the truth table given in Table 9.16.

**Table 9.16** One-bit full adder truth table

CIN	A	B	SUM	COUT
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

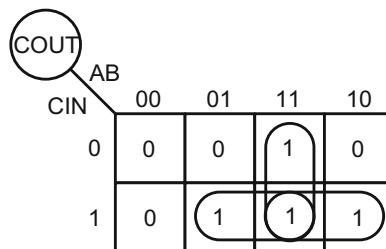
We can obtain the minimized SOP expressions for SUM and COUT from the K-maps in Figs. 9.37 and 9.38.



**Fig. 9.37** SUM output of a one-bit full adder Consequently

Consequently,

$$\begin{aligned}
 \text{SUM} &= \overline{A} \cdot \overline{B} \cdot \text{CIN} + \overline{A} \cdot B \cdot \overline{\text{CIN}} + A \cdot B \cdot \text{CIN} + A \cdot \overline{B} \cdot \overline{\text{CIN}} \\
 &= \text{CIN} \cdot (\overline{A} \cdot \overline{B} + A \cdot B) + \overline{\text{CIN}} \cdot (\overline{A} \cdot B + A \cdot \overline{B}) \\
 &= \text{CIN} \cdot (\overline{A} \oplus B) + \overline{\text{CIN}} \cdot (A \oplus B) \\
 &= A \oplus B \oplus \text{CIN}
 \end{aligned}$$

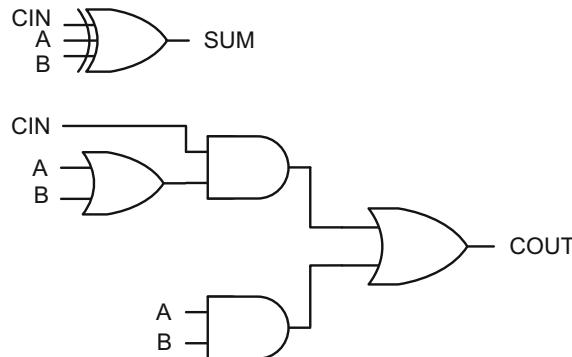


**Fig. 9.38** COUT output of a one-bit full adder

Thus,

$$\begin{aligned} \text{COUT} &= \text{CIN} \cdot \text{B} + \text{A} \cdot \text{B} + \text{A} \cdot \text{CIN} \\ &= \text{CIN} \cdot (\text{A} + \text{B}) + \text{A} \cdot \text{B} \end{aligned}$$

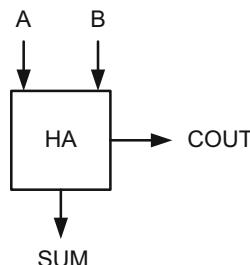
The resultant logic circuits for SUM and COUT are shown in Fig. 9.39.



**Fig. 9.39** One-bit full adder logic circuit

### One-Bit Half Adder

A one-bit half adder has only two inputs, A and B with no CIN. A and B inputs are added to generate SUM and COUT outputs. The symbolic representation of a half-adder is shown in Fig. 9.40.



**Fig. 9.40** One-bit half-adder symbol

The truth table given in Table 9.17 describes the functionality of the half adder.

**Table 9.17** One-bit half-adder truth table

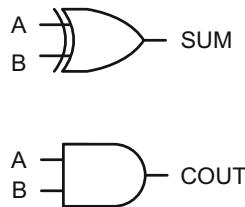
A	B	SUM	COUT
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

From the truth table, the POS expressions for SUM and COUT can be written as:

$$\text{SUM} = A \oplus B$$

$$\text{COUT} = A \cdot B$$

Therefore, we can produce SUM and COUT circuits as shown in Fig. 9.41.



**Fig. 9.41** One-bit half adder logic circuit

## 9.6 Adders

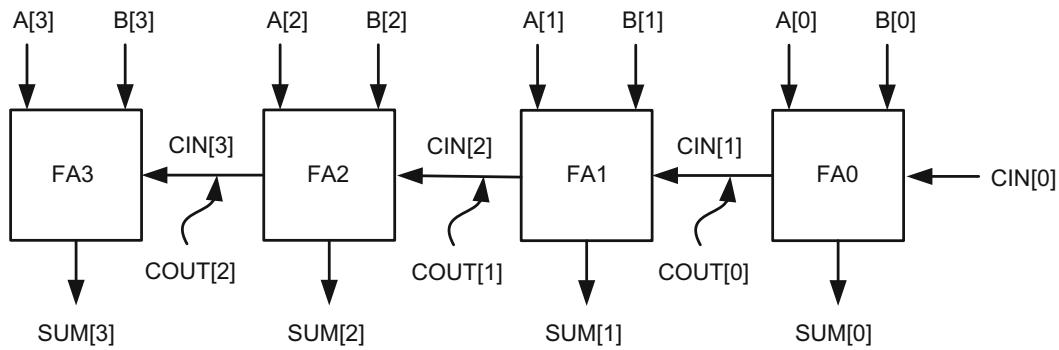
One-bit full-adders can be cascaded serially to produce multiple-bit adder configurations. There are three basic adder types:

- Ripple-Carry Adder
- Carry-Look-Ahead (CLA) Adder
- Carry-Select Adder

However, different hybrid adder topologies can be designed by combining these configurations. For the sake of simplicity, we will limit the number of bits to four and explain each topology in detail.

### Ripple-Carry Adder

The ripple-carry adder is a cascaded configuration of multiple one-bit full adders. The circuit topology of a four-bit ripple carry adder is shown in Fig. 9.42. In this figure, the carry-out output of a one-bit full adder is connected to the carry-in input of the next full adder to propagate carry from CIN[0] to higher bits.



**Fig. 9.42** Four-bit ripple-carry adder

For the 0-th bit of this adder, we have:

$$\text{SUM}[0] = A[0] \oplus B[0] \oplus \text{CIN}[0]$$

$$\text{COUT}[0] = \text{CIN}[1] = A[0] \cdot B[0] + \text{CIN}[0] \cdot (A[0] + B[0]) = G[0] + P[0] \cdot \text{CIN}[0]$$

where,

$G[0] = A[0] \cdot B[0]$  as the zeroth order generation term

$P[0] = A[0] + B[0]$  as the zeroth order propagation term

For the first bit:

$$\text{SUM}[1] = A[1] \oplus B[1] \oplus \text{CIN}[1] = A[1] \oplus B[1] \oplus (G[0] + P[0] \cdot \text{CIN}[0])$$

$$\text{COUT}[1] = \text{CIN}[2] = G[1] + P[1] \cdot \text{CIN}[1]$$

$$= G[1] + P[1] \cdot (G[0] + P[0] \cdot \text{CIN}[0]) = G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot \text{CIN}[0]$$

where,

$G[1] = A[1] \cdot B[1]$  as the first order generation term

$P[1] = A[1] + B[1]$  as the first order propagation term

For the second bit:

$$\begin{aligned} \text{SUM}[2] &= A[2] \oplus B[2] \oplus \text{CIN}[2] = A[2] \oplus B[2] \oplus \{G[1] + P[1] \cdot (G[0] + P[0] \cdot \text{CIN}[0])\} \\ &= A[2] \oplus B[2] \oplus (G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot \text{CIN}[0]) \end{aligned}$$

$$\text{COUT}[2] = \text{CIN}[3] = G[2] + P[2] \cdot \text{CIN}[2]$$

$$= G[2] + P[2] \cdot \{G[1] + P[1] \cdot (G[0] + P[0] \cdot \text{CIN}[0])\}$$

$$= G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$$

where,

$$G[2] = A[2] \cdot B[2] \text{ as the second order generation term}$$

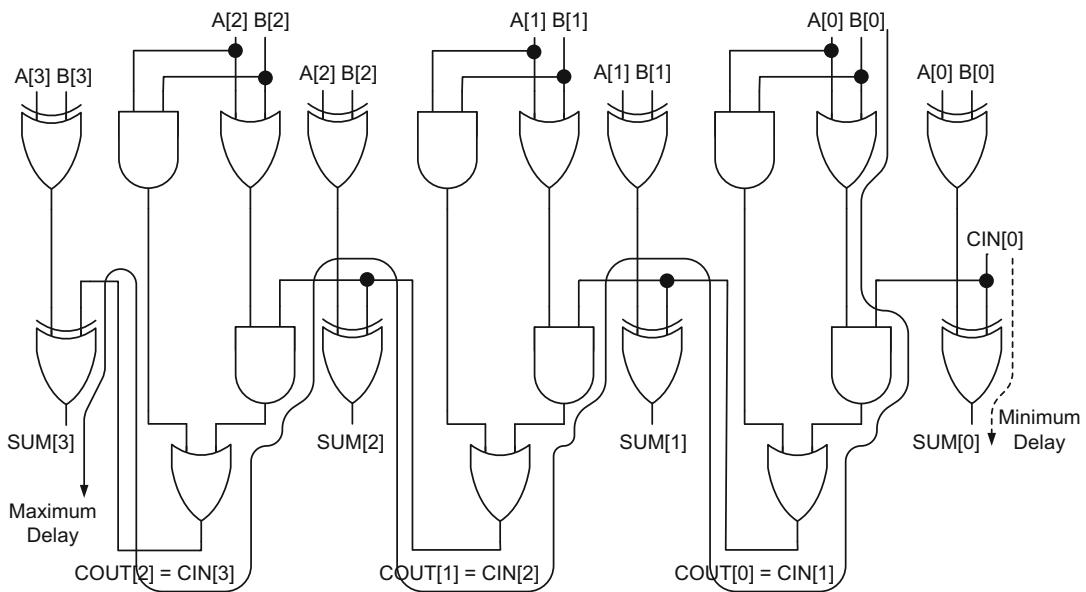
$$P[2] = A[2] + B[2] \text{ as the second order propagation term}$$

And for the third bit:

$$\begin{aligned} \text{SUM}[3] &= A[3] \oplus B[3] \oplus \text{CIN}[3] = A[3] \oplus B[3] \oplus \{G[2] + P[2] \cdot \{G[1] \\ &\quad + P[1] \cdot (G[0] + P[0] \cdot \text{CIN}[0])\}\} \\ &= A[3] \oplus B[3] \oplus (G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]) \end{aligned}$$

These functional expressions of SUM and COUT also serve to estimate the maximum gate delays for each bit of the adder.

The circuit diagram in Fig. 9.43 explains the maximum delay path through each bit.



**Fig. 9.43** Logic circuit of the four-bit adder with the maximum and minimum delays

The maximum gate delay from A[0] or B[0] inputs to SUM[0] is  $2T_{XOR2}$ , where  $T_{XOR2}$  is a single two-input XOR gate delay.

The maximum gate delay from A[0] or B[0] to COUT[0] is  $2T_{OR2} + T_{AND2}$  where  $T_{OR2}$  and  $T_{AND2}$  are two-input OR gate and two-input AND gate delays, respectively.

The gate delay from A[1] or B[1] to SUM[1] is still  $2T_{XOR2}$ ; however, the delay from A[0] or B[0] to SUM[1] is  $2T_{OR2} + T_{AND2} + T_{XOR2}$ , which is more than  $2T_{XOR2}$  and must be considered the maximum gate delay for this particular bit position and for more significant bits.

The maximum gate delay from A[0] or B[0] to COUT[1] is  $3T_{OR2} + 2T_{AND2}$ . It may make more sense to expand the expression for COUT[1] as  $COUT[1] = G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot CIN[0]$ , and figure out if the overall gate delay,  $T_{OR2} + T_{AND3} + T_{OR3}$ , is smaller compared to  $3T_{OR2} + 2T_{AND2}$ . Here,  $T_{AND3}$  and  $T_{OR3}$  are single three-input AND and OR gate delays, respectively.

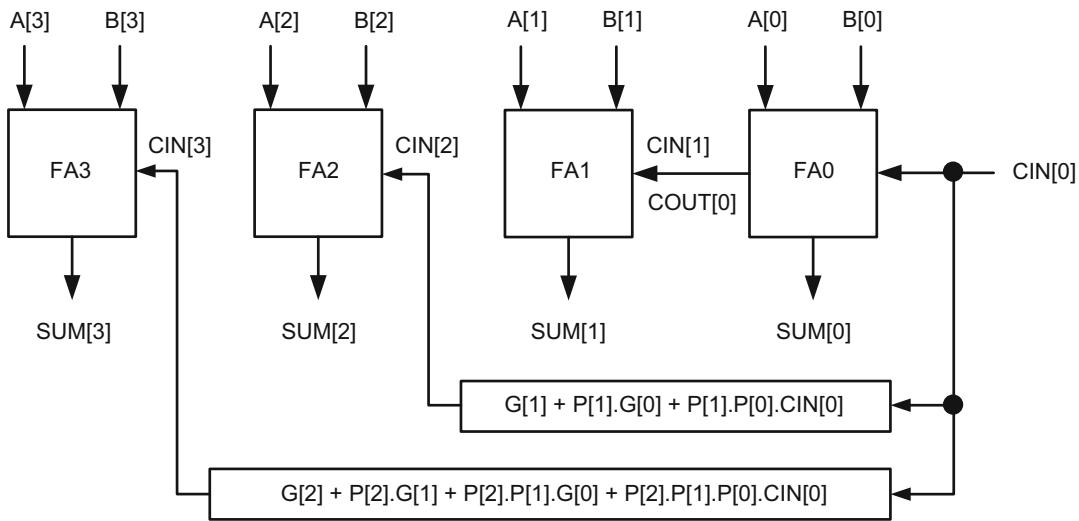
The maximum gate delay from A[0] or B[0] to SUM[2] is  $3T_{OR2} + 2T_{AND2} + T_{XOR2}$ . When the expression for SUM[2] is expanded as  $SUM[2] = A[2] \oplus B[2] \oplus (G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot CIN[0])$ , we see that this delay becomes  $T_{OR2} + T_{AND3} + T_{OR3} + T_{XOR2}$ , and may be smaller than the original delay if  $T_{AND3} < 2T_{AND2}$  and  $T_{OR3} < 2T_{OR2}$ .

The maximum gate delay from A[0] or B[0] to COUT[2] is  $4T_{OR2} + 3T_{AND2}$ . When COUT[2] is expanded as  $COUT[2] = G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot CIN[0]$ , the maximum delay becomes  $T_{OR2} + T_{AND4} + T_{OR4}$ , and may be smaller than the original delay if  $T_{AND4} < 3T_{AND2}$  and  $T_{OR4} < 3T_{OR2}$ . Here,  $T_{AND4}$  and  $T_{OR4}$  are single four-input AND and OR gate delays, respectively.

Finally, the maximum delay from A[0] or B[0] to SUM[3] is  $4T_{OR2} + 3T_{AND2} + T_{XOR2}$ , which is also the maximum propagation delay for this adder. When the functional expression for SUM[3] is expanded as  $SUM[3] = A[3] \oplus B[3] \oplus (G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot CIN[0])$ , the total propagation delay becomes  $T_{OR2} + T_{AND4} + T_{OR4} + T_{XOR2}$ , and again it may be smaller compared to the original delay if  $T_{AND4} < 3T_{AND2}$  and  $T_{OR4} < 3T_{OR2}$ .

### Carry-Look-Ahead Adder

The idea behind carry-look-ahead (CLA) adders is to create a topology where carry-in bits to all one-bit full adders are available simultaneously. A four-bit CLA circuit topology is shown in Fig. 9.44. In this figure, the SUM output of a more significant bit does not have to wait until the carry bit ripples from the least significant bit position, but it gets computed after some logic delay. In reality, all carry-in signals are generated by complex combinational logic blocks called Carry-Look-Ahead (CLA) hook-ups, as shown in Fig. 9.44. Each CLA block adds a certain propagation delay on top of the two-input XOR gate delay to produce a SUM output.



**Fig. 9.44** A four-bit carry-look-ahead adder

The earlier SUM and CIN expressions derived for the ripple carry adder can be applied to the CLA adder to generate its functional equations.

Therefore,

$$\text{SUM}[0] = A[0] \oplus B[0] \oplus \text{CIN}[0]$$

$$\text{SUM}[1] = A[1] \oplus B[1] \oplus \text{CIN}[1]$$

$$\text{SUM}[2] = A[2] \oplus B[2] \oplus \text{CIN}[2]$$

$$\text{SUM}[3] = A[3] \oplus B[3] \oplus \text{CIN}[3]$$

where,

$$\text{CIN}[1] = G[0] + P[0] \cdot \text{CIN}[0]$$

$$\text{CIN}[2] = G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot \text{CIN}[0]$$

$$\text{CIN}[3] = G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$$

Therefore, CIN[1] is generated by the COUT[0] function within FA0. However, CIN[2] and CIN[3] have to be produced by separate logic blocks in order to provide CIN signals for FA2 and FA3.

According to Fig. 9.44, once a valid CIN[0] becomes available, it takes successively longer times to generate valid signals for higher order SUM outputs due to the increasing complexity in CLA hook-ups.

Assume that  $T_{SUM0}$ ,  $T_{SUM1}$ ,  $T_{SUM2}$  and  $T_{SUM3}$  are the propagation delays of bits 0, 1, 2 and 3 with respect to the CIN[0] signal. We can approximate  $T_{SUM0} = T_{XOR2}$ . To compute  $T_{SUM1}$ , we need to examine the expression for CIN[1]. In this expression,  $P[0] \cdot CIN[0]$  produces a two-input AND gate delay, and  $G[0] + (P[0] \cdot CIN[0])$  produces a two-input OR gate delay to be added on top of  $T_{XOR2}$ . Therefore,  $T_{SUM1}$  becomes  $T_{SUM1} = T_{AND2} + T_{OR2} + T_{XOR2}$ . Similarly, the expressions for CIN[2] and CIN[3] produce  $T_{SUM2} = T_{AND3} + T_{OR3} + T_{XOR2}$  and  $T_{SUM3} = T_{AND4} + T_{OR4} + T_{XOR2}$ , respectively.

The maximum propagation delay for this adder is, therefore,  $T_{SUM3} = T_{AND4} + T_{OR4} + T_{XOR2}$ .

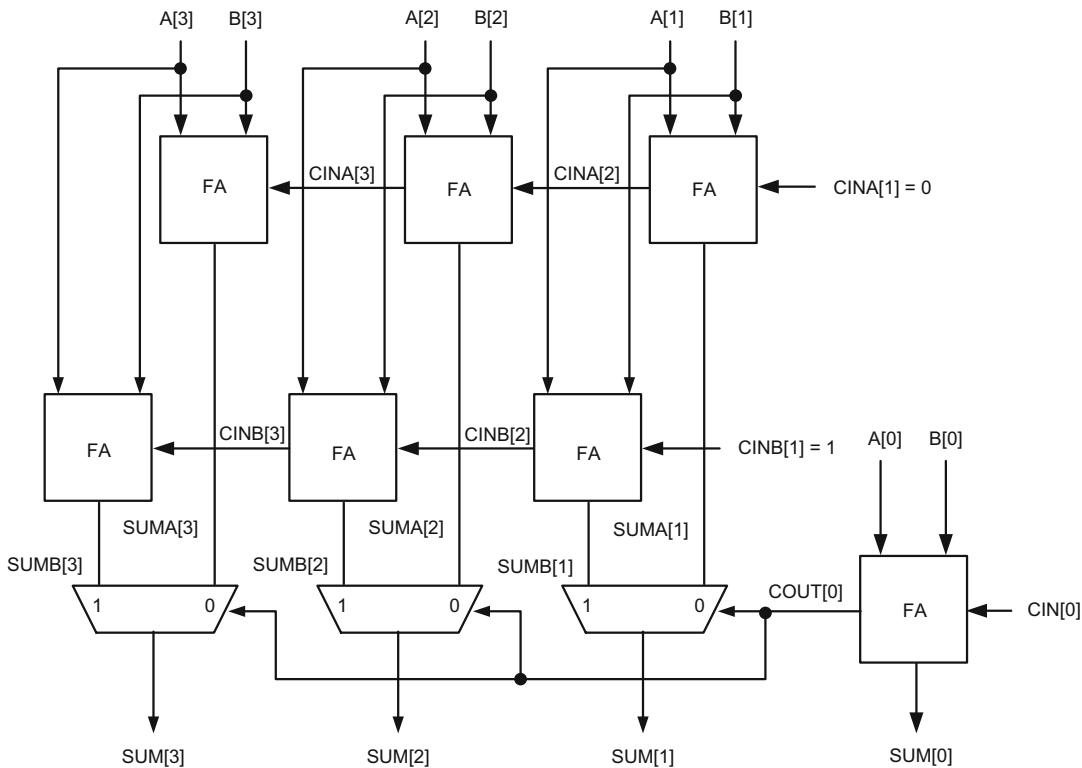
Despite the CLA adder's advantage of being faster than the ripple-carry adder, in most cases the extra CLA logic blocks make this adder topology occupy a larger chip area if the number of adder bits is above eight.

### **Carry-Select Adder**

Carry-Select Adders require two rows of identical adders. The identical adders can be as simple as two rows of identical ripple-carry adders or CLA adders depending on the design requirements. Figure 9.45 shows the circuit topology of a four-bit carry-select adder composed of two rows of ripple carry adders.

In this figure, the full adder at the least significant bit position operates normally and generates a value for COUT[0]. As this value is generated, the two one-bit full adders, one with CINA[1] = 0 and the other with CINB[1] = 1, simultaneously generate SUMA[1] and SUMB[1]. If COUT[0] becomes equal to one, SUMB[1] gets selected and becomes SUM[1]; otherwise, SUMA[1] becomes the SUM[1] output. Whichever value ends up being SUM[1], it is produced after a 2-1 MUX propagation delay.

However, we cannot say the same in generating SUM[2] and SUM[3] outputs in this figure. After producing SUM[1], carry ripples through both adders normally to generate SUM[2] and SUM[3]; hence, the speed advantage of having two rows of adders becomes negligible. Therefore, we must be careful when employing a carry-select scheme before designing an adder, as this method practically doubles the chip area.



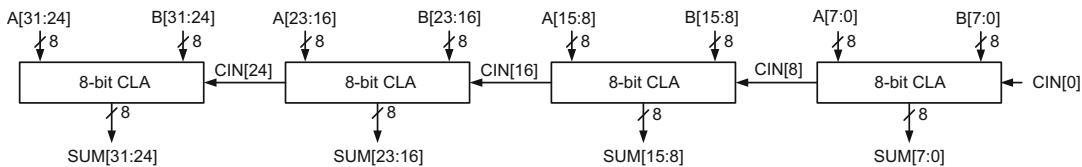
**Fig. 9.45** A four-bit carry-select adder

Even though carry-select topology is ineffective in speeding up this particular four-bit adder, it may be advantageous if employed to an adder with greater number of bits in conjunction with another adder topology such as the CLA.

**Example 9.14** Design a 32-bit carry-look-ahead adder. Compute the worst-case propagation delay in the circuit.

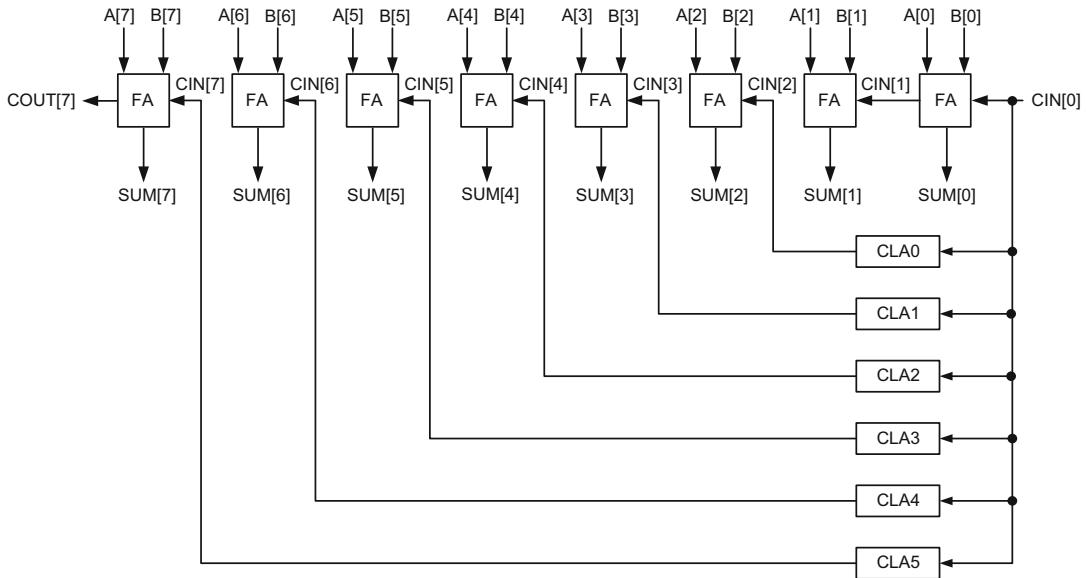
We need to be careful in dealing with the CLA hook-ups when generating higher order terms because the complexity of these logic blocks can “grow” exponentially in size while they may only provide marginal speed gain when compared to ripple-carry scheme.

Therefore, the first step of the design process is to separate the adder into eight-bit segments with full CLA hook-ups. The proposed topology is shown in Fig. 9.46.



**Fig. 9.46** A 32-bit carry-look-ahead topology

Each eight-bit CLA segment contains six CLA hook-ups from CLA0 to CLA5 as shown in Fig. 9.47.



**Fig. 9.47** An eight-bit segment of the carry-look ahead adder in Fig. 9.46

CIN and SUM expressions from bit 0 through bit 7 are given below.

$$\text{SUM}[0] = A[0] \oplus B[0] \oplus \text{CIN}[0]$$

$$\text{SUM}[1] = A[1] \oplus B[1] \oplus \text{CIN}[1]$$

where,  $\text{CIN}[1] = G[0] + P[0] \cdot \text{CIN}[0]$

$$\text{SUM}[2] = A[2] \oplus B[2] \oplus \text{CIN}[2]$$

where,  $\text{CIN}[2] = G[1] + P[1] \cdot G[0] + P[1] \cdot P[0] \cdot \text{CIN}[0]$

$$\text{SUM}[3] = A[3] \oplus B[3] \oplus \text{CIN}[3]$$

where,  $\text{CIN}[3] = G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$

$$\text{SUM}[4] = A[4] \oplus B[4] \oplus \text{CIN}[4]$$

where,  $\text{CIN}[4] = G[3] + P[3] \cdot (G[2] + P[2]) \cdot G[1] + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$

$$\text{SUM}[5] = A[5] \oplus B[5] \oplus \text{CIN}[5]$$

$$\begin{aligned} \text{where, } \text{CIN}[5] &= G[4] + P[4] \cdot \{G[3] + P[3] \cdot (G[2] + P[2] \cdot G[1] \\ &\quad + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0])\} \\ &= G[4] + P[4] \cdot G[3] + P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] \\ &\quad + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]) \end{aligned}$$

$$\text{SUM}[6] = A[6] \oplus B[6] \oplus \text{CIN}[6]$$

$$\begin{aligned} \text{where, } \text{CIN}[6] &= G[5] + P[5] \cdot \{G[4] + P[4] \cdot G[3] + P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] \\ &\quad + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0])\} \\ &= G[5] + P[5] \cdot G[4] + P[5] \cdot P[4] \cdot G[3] + P[5] \cdot P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] \\ &\quad + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]) \end{aligned}$$

$$\text{SUM}[7] = A[7] \oplus B[7] \oplus \text{CIN}[7]$$

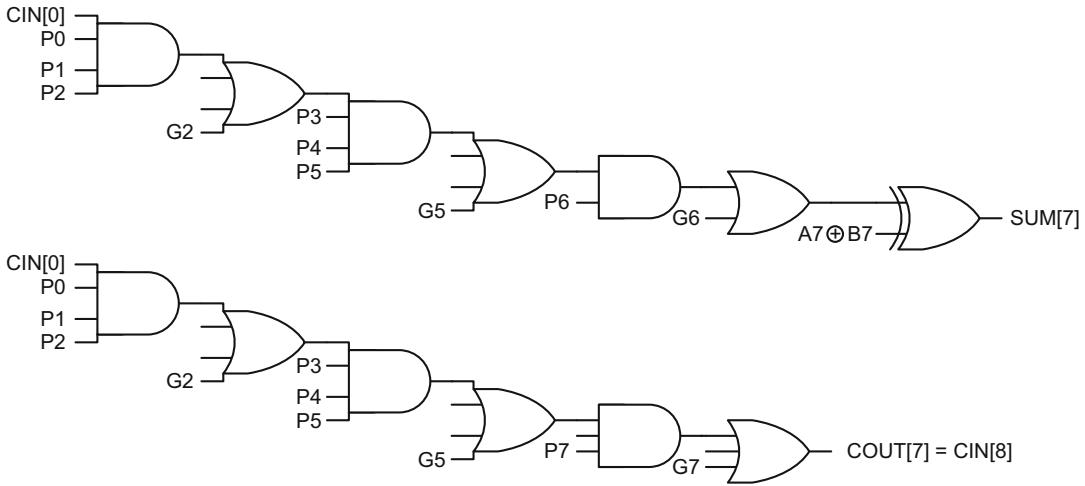
$$\begin{aligned} \text{where, } \text{CIN}[7] &= G[6] + P[6] \cdot \{G[5] + P[5] \cdot G[4] + P[5] \cdot P[4] \cdot G[3] \\ &\quad + P[5] \cdot P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] \\ &\quad + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0])\} \end{aligned}$$

And finally,

$$\begin{aligned} \text{COUT}[7] = \text{CIN}[8] &= G[7] + P[7] \cdot \{G[6] + P[6] \cdot \{G[5] + P[5] \cdot G[4] + P[5] \cdot P[4] \cdot G[3] \\ &\quad + P[5] \cdot P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] + P[2] \cdot P[1] \cdot G[0] \\ &\quad + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0])\}\} \\ &= G[7] + P[7] \cdot G[6] + P[7] \cdot P[6] \cdot \{G[5] + P[5] \cdot G[4] \\ &\quad + P[5] \cdot P[4] \cdot G[3] + P[5] \cdot P[4] \cdot P[3] \cdot (G[2] + P[2] \cdot G[1] \\ &\quad + P[2] \cdot P[1] \cdot G[0] + P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0])\} \end{aligned}$$

In these derivations, particular attention was paid to limit the number of inputs to four in all AND and OR gates since larger gate inputs are counterproductive in reducing the overall propagation delay.

From these functional expressions, maximum propagation delays for  $\text{SUM}[7]$  and  $\text{COUT}[7]$  are Estimated using the longest logic strings in Fig. 9.48.



**Fig. 9.48** Propagation delay estimation of the eight-bit carry-look-ahead adder in Fig. 9.47

For  $\text{SUM}[7]$ , the minterm,  $P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$ , generates the first four-input AND gate. This is followed by a four-input OR-gate whose minterms are  $G[2], P[2] \cdot G[1], P[2] \cdot P[1] \cdot G[0]$ , and  $P[2] \cdot P[1] \cdot P[0] \cdot \text{CIN}[0]$ . This gate is followed by a four-input AND, four-input OR, two-input AND, two-input OR and two-input XOR-gates in successive order. The entire string creates a propagation delay of  $T_{\text{SUM}7} = 2(T_{\text{AND}4} + T_{\text{OR}4}) + T_{\text{AND}2} + T_{\text{OR}2} + T_{\text{XOR}2}$  from  $\text{CIN}[0]$  to  $\text{SUM}[7]$ .

For  $\text{COUT}[7]$ , the longest propagation delay is between  $\text{CIN}[0]$  to  $\text{COUT}[7]$  as shown in Fig. 9.48, and it is equal to  $T_{\text{COUT}7} = 2(T_{\text{AND}4} + T_{\text{OR}4}) + T_{\text{AND}3} + T_{\text{OR}3}$ . The delays for the rest of the circuit in Fig. 9.46 become easy to determine since the longest propagation delays have already been evaluated.

The delay from  $\text{CIN}[0]$  to  $\text{SUM}[15]$ ,  $T_{\text{SUM}15}$ , simply becomes equal to the sum of  $T_{\text{COUT}7}$  and  $T_{\text{SUM}7}$ . In other words,  $T_{\text{SUM}15} = 4(T_{\text{AND}4} + T_{\text{OR}4}) + T_{\text{AND}3} + T_{\text{OR}3} + T_{\text{AND}2} + T_{\text{OR}2} + T_{\text{XOR}2}$ . Similarly, the delay from  $\text{CIN}[0]$  to  $\text{COUT}[15]$ ,  $T_{\text{COUT}15}$ , is equal to  $T_{\text{COUT}15} = 4(T_{\text{AND}4} + T_{\text{OR}4}) + 2(T_{\text{AND}3} + T_{\text{OR}3})$ .

The remaining delays are evaluated in the same way and lead to the longest propagation delay in this circuit, which is from  $\text{CIN}[0]$  to  $\text{SUM}[31]$ ,  $T_{\text{SUM}31}$ .

Thus,

$$T_{\text{SUM}31} = 3[2(T_{\text{AND}4} + T_{\text{OR}4}) + T_{\text{AND}3} + T_{\text{OR}3}] + 2(T_{\text{AND}4} + T_{\text{OR}4}) + T_{\text{AND}2} + T_{\text{OR}2} + T_{\text{XOR}2}$$

or

$$T_{\text{SUM}31} = 8(T_{\text{AND}4} + T_{\text{OR}4}) + 3(T_{\text{AND}3} + T_{\text{OR}3}) + T_{\text{AND}2} + T_{\text{OR}2} + T_{\text{XOR}2}$$

**Example 9.15** Design a 32-bit hybrid carry-select/carry-look-ahead adder. Compute the worst-case propagation delay in the circuit.

Large adders are where the carry-select scheme shines! This is a classical example in which the maximum propagation delay is reduced considerably compared to the CLA scheme examined in Example 9.14.

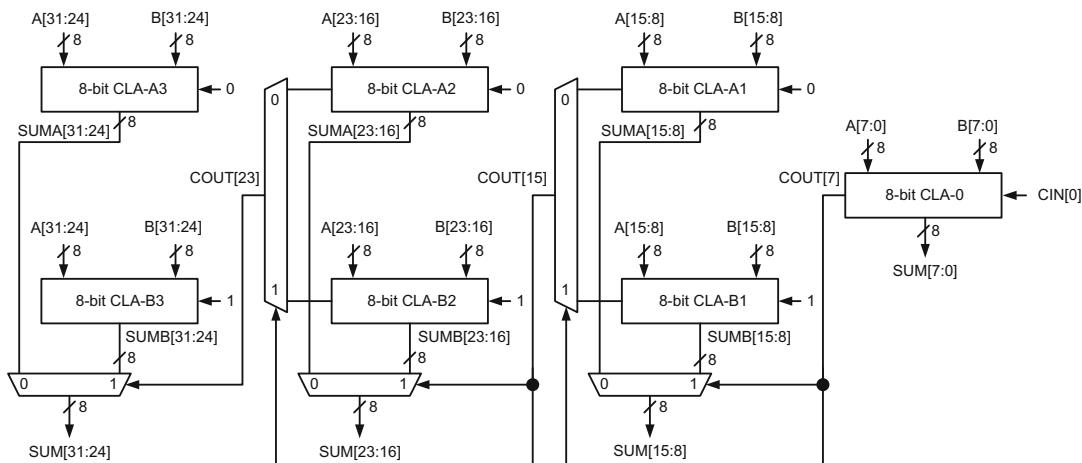
As mentioned earlier, a twin set of an adder configuration is required by the carry-select scheme. The adders can be ripple-carry, carry-look-ahead or the combination of the two.

In this example, the 32-bit adder is again divided in eight-bit segments where each segment consists of a full CLA adder as shown in Fig. 9.49. The first segment, CLA-0, is a single unit which produces COUT[7] with full CLA hook-ups. The rest of the eight-bit segments are mirror images of each other and they are either named A-segments (A1, A2 and A3) or B-segments (B1, B2 and B3).

As the CLA-0 generates a valid COUT[7], the CLA-A1 and CLA-B1 simultaneously generate COUTA[15] and COUTB[15]. When COUT[7] finally forms, it selects either COUTA[15] or COUTB[15] depending on the value of CIN[0]. This segment produces COUT[15] and SUM[15:8].

COUT[15], on the other hand, is used to select between COUTA[23] and COUTB[23], both of which have already been formed when COUT[15] arrives at the 2-1 MUX as a control input. COUT[15] also selects between SUMA[23:16] and SUMB[23:16] to determine the correct SUM[23:16].

Similarly, COUT[23] is used as a control input to select between SUMA[31:24] and SUMB[31:24]. If there is a need for COUT[31], COUT[23] can serve to determine the value of COUT[31].

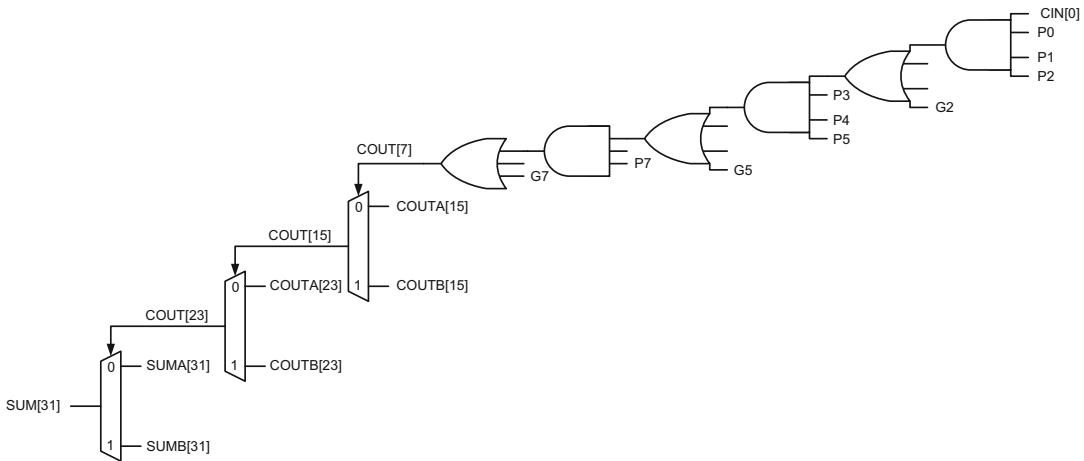


**Fig. 9.49** A 32-bit carry-look-ahead/carry-select adder

The maximum propagation delay for the 32-bit carry-select/CLA adder can be found using the logic string in Fig. 9.50. The first section of this string from CIN[0] to COUT[7] is identical to the eight-bit CLA carry delay in Fig. 9.48. There are three cascaded MUX stages which correspond to the selection of COUT[15], COUT[23], and SUM[31].

Considering that a 2-1 MUX propagation delay consists of a two-input AND gate followed by a two-input OR gate, we obtain the following maximum delay for this 32-bit adder:

$$T_{\text{SUM31}} = 2(T_{\text{AND4}} + T_{\text{OR4}}) + T_{\text{AND3}} + T_{\text{OR3}} + 3(T_{\text{AND2}} + T_{\text{OR2}})$$



**Fig. 9.50** Maximum delay propagation of the 32-bit adder in Fig. 9.49

Considering the maximum propagation delay in Example 9.14, this delay is shorter by at least  $6(T_{AND4} + T_{OR4})$ . Larger carry-select/carry-look-ahead adder schemes provide greater speed benefits at the cost of approximately doubling the adder area.

## 9.7 Subtractors

Subtraction is performed by a technique called Twos (2s) complement addition. Twos complement addition requires complementing one of the adder inputs (1s complement), and then adding one to the least significant bit.

**Example 9.16** Form  $-4$  using 2s complement addition using four bits.

A negative number is created by first inverting every bit of  $+4$  (1s complement representation) and then adding 1 to it.  $+4$  is equal to 0100 in four-bit binary form.

Its 1s complement is 1011. Its 2s complement is  $1011 + 0001 = 1100$ .

Therefore, logic 0 signifies a positive sign, and logic 1 signifies a negative sign at the most significant bit position of a signed number.

**Example 9.17** Add  $+4$  to  $-4$  using 2s complement addition

$$+4 = 0100 \text{ in binary form}$$

$$-4 = 1100 \text{ in 2s complement form of } +4.$$

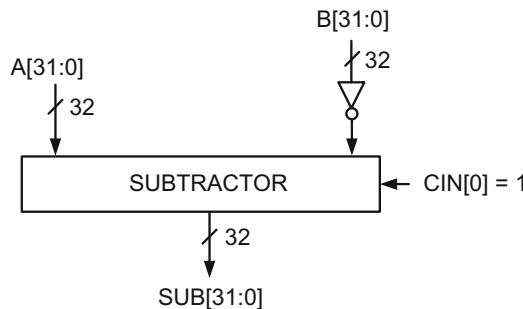
$$\text{Perform } +4 - 4 = 0100 + 1100 = 10000$$

where, logic 1 at the overflow bit position is neglected in a four-bit binary format.

Therefore, we obtain 0000 = 0 as expected.

Subtractors function according to 2s complement addition. We need to form 1s complement of the adder input to be subtracted and use  $CIN[0] = 1$  at the adder's least significant bit position to perform subtraction.

Figure 9.51 illustrates the topology of a 32-bit subtractor where input B is complemented and  $CIN[0]$  is tied to logic 1 to satisfy 2s complement addition and produce  $A - B$ .



**Fig. 9.51** A symbolic representation of a 32-bit subtractor

## 9.8 Shifters

There are two commonly used shifters in logic design:

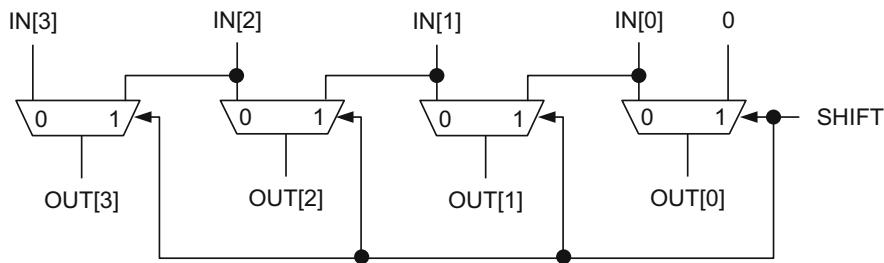
- Linear shifters
- Barrel shifters

### Linear Shifters

A linear shifter shifts its inputs by a number of bits to the right or left and routes the result to its output.

**Example 9.18** Design a four-bit linear shifter that shifts its inputs to the left by one bit and produces logic 0 at the least significant output bit when SHIFT = 1. When SHIFT = 0, the shifter routes each input directly to the corresponding output.

The logic diagram for this shifter is given in Fig. 9.52. In this figure, each input is connected to the port 0 terminal of the 2-1 MUX as well as the port 1 terminal of the next MUX at the higher bit position. Therefore, when SHIFT = 1, logic 0, IN[0], IN[1], and IN[2] are routed through port 1 terminal of each 2-1 MUX and become OUT[0], OUT[1], OUT[2], and OUT[3], respectively. When SHIFT = 0, each input goes through port 0 terminal of the corresponding 2-1 MUX and becomes the shifter output.



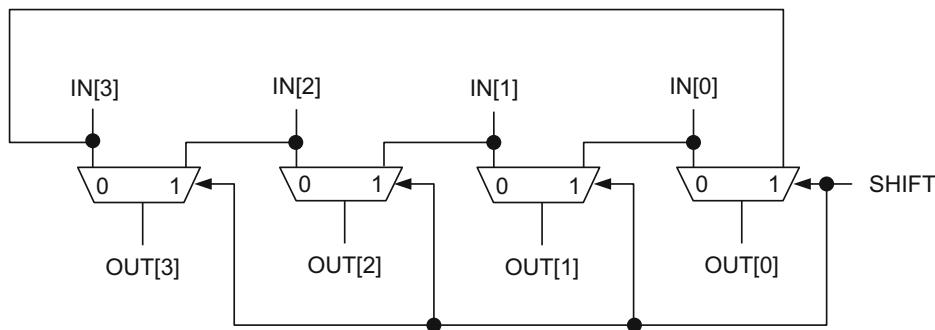
**Fig. 9.52** Four-bit linear shifter

### Barrel Shifters

Barrel shifters rotate their inputs either in clockwise or counterclockwise direction by a number of bits but preserve all their inputs when generating an output.

**Example 9.19** Design a four-bit barrel shifter that rotates its inputs in a clockwise direction by one bit when SHIFT = 1. When SHIFT = 0, the shifter routes each one of its four inputs to its corresponding output.

The logic diagram for this shifter is given in Fig. 9.53. The only difference between this circuit and the linear shifter in Fig. 9.52 is the removal of logic 0 from the least significant bit, and connecting this input to the IN[3] pin instead. Consequently, this leads to OUT[0] = IN[3], OUT[1] = IN[0], OUT[2] = IN[1] and OUT[3] = IN[2] when SHIFT = 1, and OUT[0] = IN[0], OUT[1] = IN[1], OUT[2] = IN[2] and OUT[3] = IN[3] when SHIFT = 0.



**Fig. 9.53** Four-bit barrel shifter

**Example 9.20** Design a four-bit barrel shifter that rotates its inputs clockwise by one or two bits.

First, there must be three control inputs specifying “no shift”, “shift 1 bit” and “shift 2 bits”. This requires a two control-bit input, SHIFT[1:0], as shown in Table 9.18. All

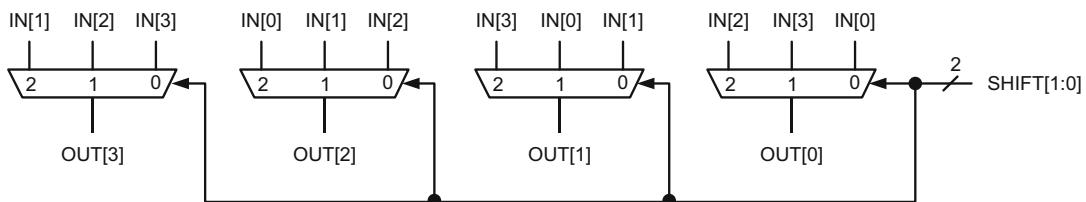
assignments in this table are done arbitrarily. However, it makes sense to assign a “No shift” to SHIFT[1:0] = 0, “Shift 1 bit” to SHIFT[1:0] = 1 and “Shift 2 bits” to SHIFT[1:0] = 2 for actual rotation amount.

**Table 9.18** A four-bit barrel shifter truth table

SHIFT[1]	SHIFT[0]	OPERATION	OUT[3]	OUT[2]	OUT[1]	OUT[0]
0	0	No shift	IN[3]	IN[2]	IN[1]	IN[0]
0	1	Shift 1 bit	IN[2]	IN[1]	IN[0]	IN[3]
1	0	Shift 2 bits	IN[1]	IN[0]	IN[3]	IN[2]
1	1	No shift	IN[3]	IN[2]	IN[1]	IN[0]

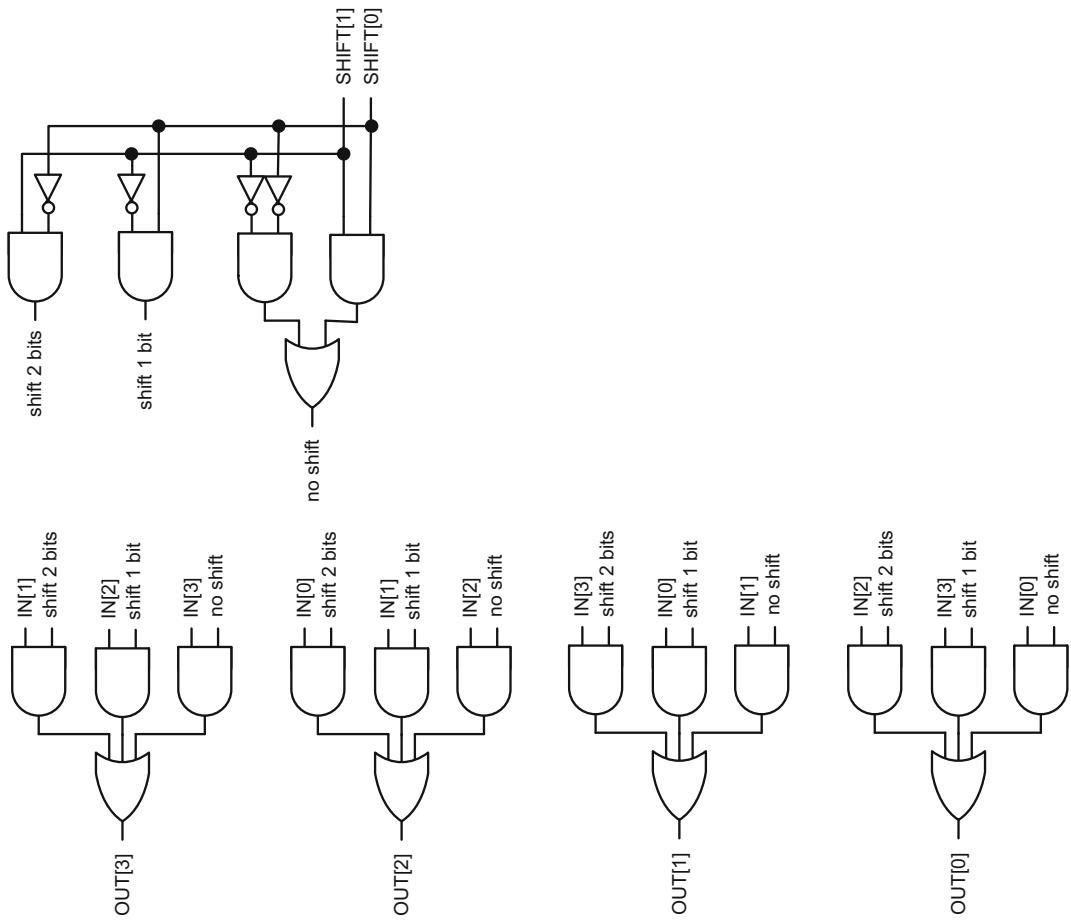
According to this table, if there is no shift, each input bit is simply routed to its own output. If “shift 1 bit” input is active, then each input is routed to the neighboring output at the next significant bit position. In other words, IN[3] rotates clockwise and becomes OUT[0]; IN[0], IN[1] and IN[2] shift 1 bit to the left and become OUT[1], OUT[2] and OUT[3], respectively. If “shift 2 bits” becomes active, then each input is routed to the output of the neighboring bit which is two significant bits higher. This gives the impression that all input bits have been rotated twice before being routed to the output. Thus, OUT[0] = IN[2], OUT[1] = IN[3], OUT[2] = IN[0] and OUT[3] = IN[1].

Therefore, using Table 9.18, we can conclude the logic diagram in Fig. 9.54.



**Fig. 9.54** Logic diagram of the barrel shifter in Table 9.18

A more detailed view of Fig. 9.54 is given in Fig. 9.55.



**Fig. 9.55** Logic circuit of the barrel shifter in Fig. 9.54

## 9.9 Multipliers

Similar to our everyday hand multiplication method, an array multiplier generates all partial products before summing each column in the partial product tree to obtain a result. This scheme is explained in Fig. 9.56 for a four-bit array multiplier.

	A[3]	A[2]	A[1]	A[0]	MULTIPLICAND			
	B[3]	B[2]	B[1]	B[0]	MULTIPLIER			
$\times$								
	B[0].A[3]	B[0].A[2]	B[0].A[1]	B[0].A[0]	0 <sup>th</sup> PARTIAL PRODUCT			
	B[1].A[3]	B[1].A[2]	B[1].A[1]	B[1].A[0]	1 <sup>st</sup> PARTIAL PRODUCT			
	B[2].A[3]	B[2].A[2]	B[2].A[1]	B[2].A[0]	2 <sup>nd</sup> PARTIAL PRODUCT			
	B[3].A[3]	B[3].A[2]	B[3].A[1]	B[3].A[0]	3 <sup>rd</sup> PARTIAL PRODUCT			
+								
SUM[7]	SUM[6]	SUM[5]	SUM[4]	SUM[3]	SUM[2]	SUM[1]	SUM[0]	SUM OUTPUT

**Fig. 9.56**  $4 \times 4$  array multiplier algorithm

The rules of partial product generation are as follows:

1. The zeroth partial product aligns with multiplicand and multiplier bit columns.
2. Each partial product is shifted one bit to the left with respect to the previous one once it is created.
3. Each partial product is the exact replica of the multiplicand if the multiplier bit is one. Otherwise, it is deleted.

**Example 9.21** Multiply 1101 and 1001 according to the rules of array multiplication.

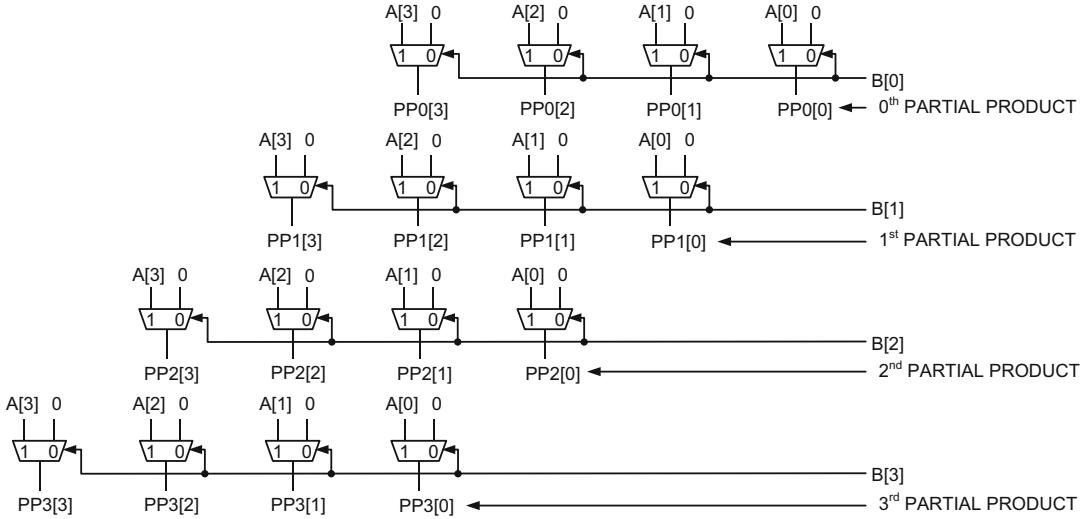
Suppose 1101 is a multiplicand and 1001 is a multiplier. Then, for a four-bit multiplier four partial products are formed. The bits in each column of the partial product are added successively. Carry bits are propagated to more significant bit positions. This process is illustrated in Fig. 9.57.

	1	1	0	1	MULTIPLICAND
	1	0	0	1	MULTIPLIER
$\times$					
	1	1	0	1	0 <sup>th</sup> PARTIAL PRODUCT
	0	0	0	0	1 <sup>st</sup> PARTIAL PRODUCT
	0	0	0	0	2 <sup>nd</sup> PARTIAL PRODUCT
	1	1	0	1	3 <sup>rd</sup> PARTIAL PRODUCT
+					
	1	1	1	0	SUM OUTPUT

**Fig. 9.57**  $4 \times 4$  array multiplier algorithm example

**Example 9.22** Design the partial product tree for a four-bit array multiplier.

Following the convention in Fig. 9.56 and the rules of partial product generation for an array multiplier, we can implement the partial product tree as shown in Fig. 9.58.

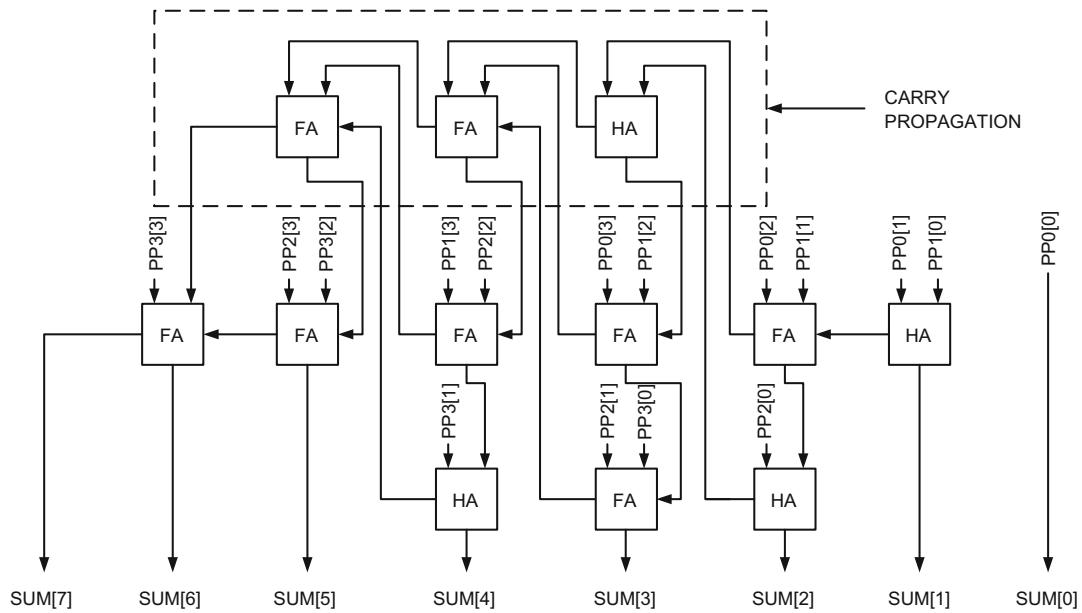


**Fig. 9.58**  $4 \times 4$  array multiplier bit selector tree

In this figure, partial product elements of the zeroth partial product,  $B[0].A[3]$ ,  $B[0].A[2]$ ,  $B[0].A[1]$  and  $B[0].A[0]$ , are replaced by  $PP0[3:0]$  for purposes of better illustration. Similarly,  $PP1[3:0]$ ,  $PP2[3:0]$  and  $PP3[3:0]$  are the new partial product outputs corresponding to the rows one, two and three.

**Example 9.23** Design a full adder tree responsible for adding every partial product in the partial product tree for a four-bit array multiplier.

After generating partial products, the next design step is to add partial product elements column by column to generate SUM outputs,  $SUM[7:0]$ , while propagating carry bits for higher order columns. Following the naming convention in Fig. 9.58, all 16 partial product elements are fed to the full adder tree in Fig. 9.59. The box outlined by dashed lines shows how the carry propagation takes place from one column to the next.

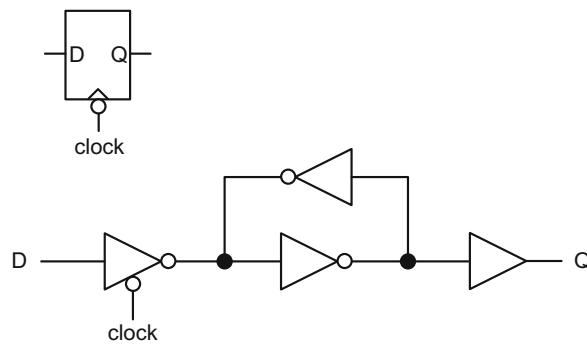


**Fig. 9.59** An eight-bit propagate adder for the bit selector tree in Fig. 9.58

## 9.10 D Latch

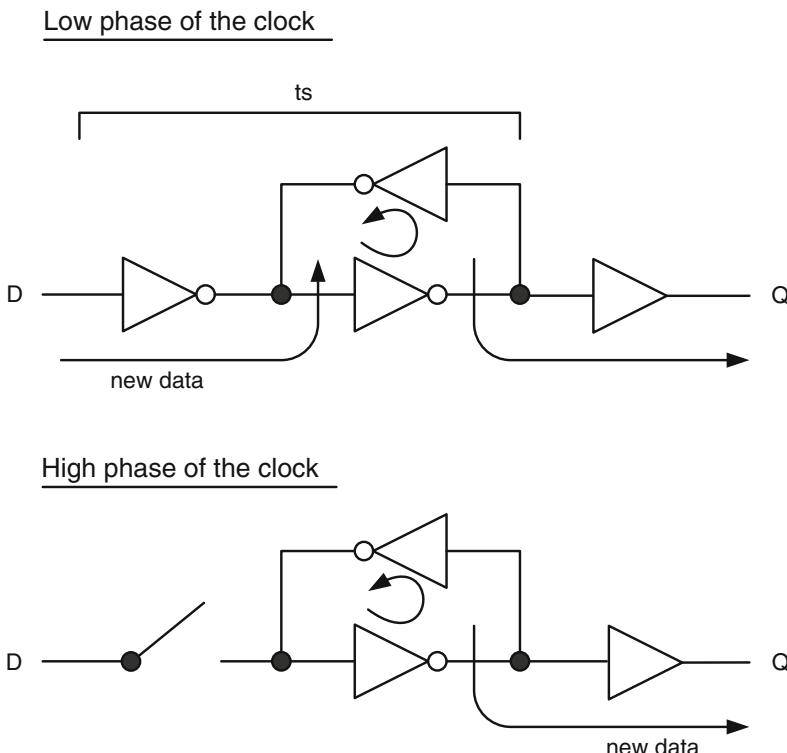
The D Latch is the most basic memory element in logic design. It has a data input, D, a clock input and a data output, Q, as shown at the top portion of Fig. 9.60. It contains a tri-state inverter at its input stage followed by two back-to-back inverters connected in a loop configuration, which serves to store data.

The clock signal connected to the enable input of the tri-state inverter can be set either to active-high or active-low. In Fig. 9.60, the changes at the input transmit through the memory element and appear at the output during the low phase of the clock. In contrast, changes at the input are blocked during the high phase of the clock, and no data transmits to the output. Once the data is stored in the back-to-back inverter loop, it becomes stable and does not change until different data is introduced at the input. The buffer at the output stage of the latch is used to drive multiple gate inputs.



**Fig. 9.60** Logic and circuit diagrams of a D latch

The operation of the D latch is shown in Fig. 9.61. During the low phase of the clock, the tri-state inverter is enabled. The new data transmits through the tri-state inverter, overwrites the old data in the back-to-back inverter stage, and reaches the output. When the clock switches to its high phase, the input-output data transmission stops because the tri-state buffer is disabled and blocks any new data transfer. Therefore, if certain data needs to be retained in the latch, it needs to be stored in the latch some time before the rising edge of the clock. This time interval is called the set-up time,  $t_S$ , and it is approximately equal to the sum of delays through the tri-state inverter and the inverter in the memory element. At the high phase of the clock, the data stored in the latch can no longer change as shown in Fig. 9.61.



**Fig. 9.61** Operation of D latch

## 9.11 Timing Methodology Using D Latches

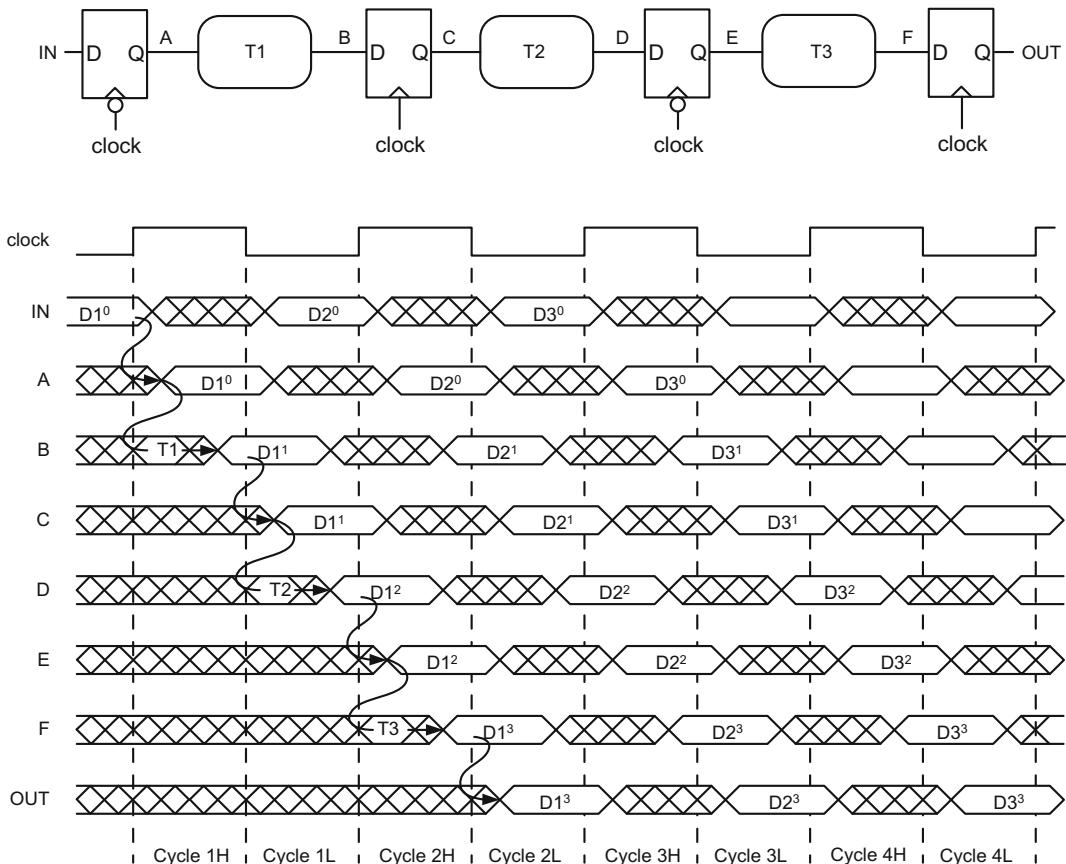
Timing in logic systems is maintained by pipeline structures. A pipeline consists of combinational logic blocks bounded by memory elements as shown at the top portion of Fig. 9.62. The main purpose of pipelines is to process several data packets within the same clock cycle and maximize the data throughput.

To illustrate the concept of pipeline, latches are used as memory elements in the pipeline shown in Fig. 9.62. In every latch boundary, data propagates from one combinational logic stage to the next at the high and low phases of clock.

The bottom figure in Fig. 9.62 shows the timing diagram of a data transfer for a set of data packets ranging from D1 to D3 at the IN terminal. The first data packet,  $D1^0$ , retains its original value during the high phase of the clock (Cycle 1H) at the node A.  $D1^0$  then propagates through the T1 stage and settles at the node B in a modified form,  $D1^1$ , sometime before the falling edge of the clock. Similarly,  $D1^1$  at the node C retains its value during the low phase of the clock while its processed form,  $D1^2$ , propagates through the T2 stage and arrives at the node D before the rising edge of the clock. This data is processed further in the T3 stage and transforms into  $D1^3$  before it becomes available at the OUT terminal at the falling edge of the clock in Cycle 2L.

Similarly, the next two data packets,  $D2^0$  and  $D3^0$ , are also fed into the pipeline at the subsequent positive clock edges. Both of these data propagate through the combinational logic stages, T1, T2 and T3, and become available at the OUT terminal at the falling edge of Cycle 3L and Cycle 4L, respectively.

The total execution time for all three data packets takes four clock cycles according to the timing diagram in Fig. 9.62. If we were to remove all the latch boundaries between nodes A and F and wait until all three data packets, D1, D2 and D3, were processed through the sum of the three combinational logic stages, T1, T2 and T3, the total execution time would have been  $3 \times 3 = 9$  clock cycles as each combinational logic stage requires one clock cycle to process data. Therefore, pipelining can be used advantageously to process data in a shorter amount of time and increase data throughput.



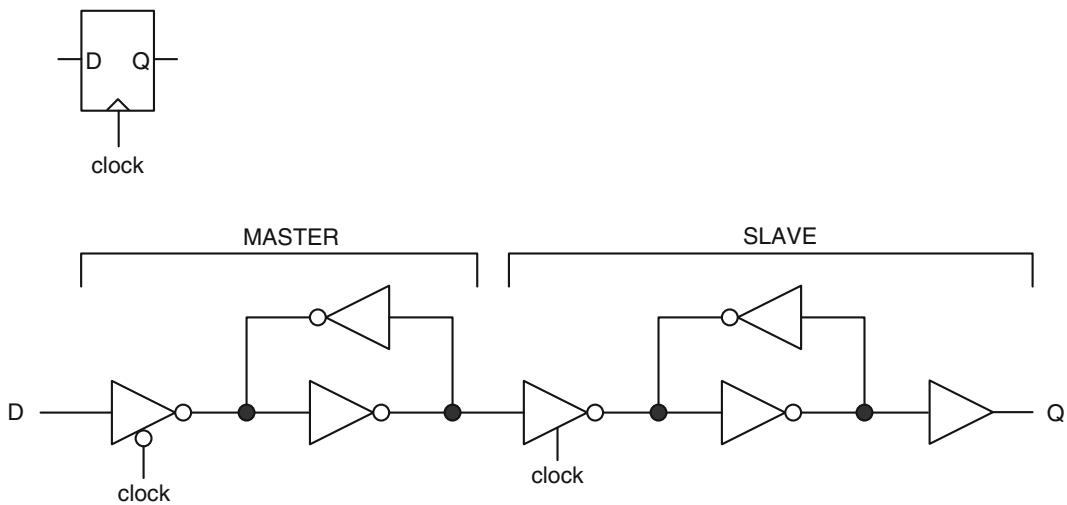
**Fig. 9.62** Timing methodology using D latches

## 9.12 D Flip-Flop

D flip-flop is another important timing element in logic design to maintain timing while transferring data from one combinational logic block to the next.

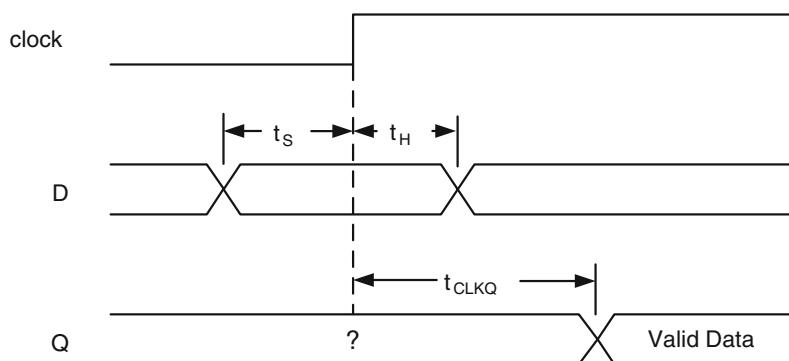
Similar to a latch, a flip-flop also has a data input, D, a clock input and a data output, Q, as shown at the top portion of Fig. 9.63.

The bottom portion of Fig. 9.63 shows the circuit schematic of a typical flip-flop which contains two latches in series. The first latch has an active-low clock input, and it is called the master. The second latch has an active-high clock input, and it is called the slave. The master accepts new data during the low phase of the clock, and transfers this data to the slave during the high phase of the clock.



**Fig. 9.63** Logic and circuit diagrams of a D flip-flop

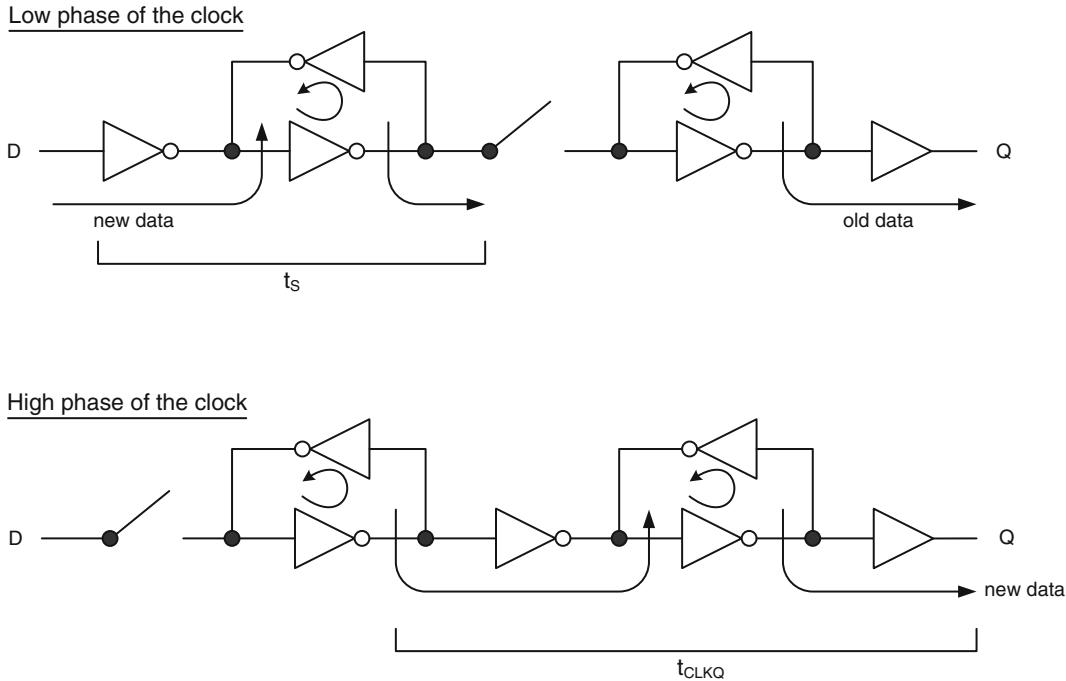
Figure 9.64 shows the timing attributes of a flip-flop. The set-up time,  $t_S$ , is the time interval for the valid data to arrive and settle in the master latch before the rising edge of the clock. The hold time,  $t_H$ , is the time interval after the positive edge of the clock when the valid data needs to be kept steady and unchanged. The data stored in the master latch propagates through the slave latch and becomes the flip-flop output some time after the rising edge of the clock. This is called clock-to-q delay or  $t_{CLKQ}$ .



**Fig. 9.64** Timing attributes of a D flip-flop

The operation of the flip-flop in two different phases of clock is shown in Fig. 9.65. During the low phase of the clock, new data enters the master latch, and it is stored. This data cannot propagate beyond the master latch because the tri-state inverter in the slave latch acts as an open circuit during the low phase of the clock. The flip-flop output reveals only the old data stored in the slave latch. When the clock goes high, however, the new data stored in the

master latch transmits through the slave and arrives at the output. One can calculate approximate values of  $t_s$  and  $t_{CLKQ}$  using the existing gate delays in the flip-flop.



**Fig. 9.65** Operation of D flip-flop

### 9.13 Timing Methodology Using D Flip-Flops

Data propagation through a pipeline with D flip-flops is shown in Fig. 9.66. The bottom figure in Fig. 9.66 shows the timing diagram of a data transfer for a set of data packets ranging from D1 to D3 at the IN terminal.

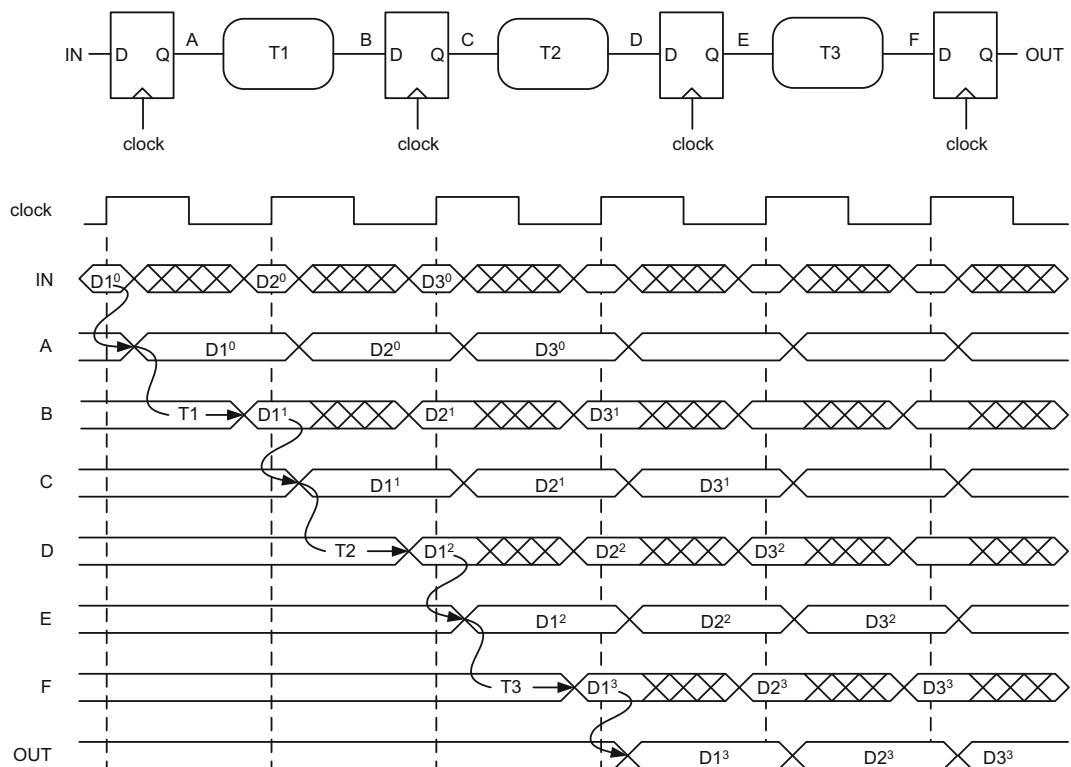
The first data packet,  $D1^0$ , at the IN terminal has to be steady and valid during the set-up and hold periods of the flip-flop, but it is free to change during the remaining part of the clock period as shown by oscillatory lines. Once the clock goes high, the valid  $D1^0$  starts to propagate through the combinational logic block of T1 and reaches the second flip-flop boundary. The processed data,  $D1^1$ , has to arrive at the second flip-flop input, B, no later than the set-up time of the flip-flop. Otherwise, the correct data cannot be latched.  $D1^1$  propagates through the second (T2) and third (T3) combinational logic stages, and becomes  $D1^2$  and  $D1^3$ , respectively, before exiting at the OUT terminal as shown in the timing diagram in Fig. 9.66.

The subsequent data packets,  $D2^0$  and  $D3^0$  are similarly fed into the pipeline stage at consecutive positive clock edges following  $D1^0$ . They are processed and modified by the T1, T2 and T3 combinational logic stages as they propagate through the pipeline, and emerge at the OUT terminal.

The total execution time for three input data packets,  $D1^0$ ,  $D2^0$  and  $D3^0$ , takes six clock cycles, including the initial three clock cycle build-up period before  $D1^3$  emerges at the OUT terminal. If we were to remove all the flip-flop boundaries between the nodes A and F, and wait for these three data packets to be processed without any pipeline structure, the total execution time would have been  $3 \times 3 = 9$  clock cycles, assuming each T1, T2 or T3 logic stage imposes one clock cycle delay.

Once again, the pipelining technique considerably reduces the overall processing time and data throughput whether the timing methodology is latch-based or flip-flop-based.

The advantage of using latches as opposed to flip-flops is to be able to borrow time from neighboring stages. For example, the propagation delay in T1 stage can be extended at the expense of shortening the propagation delay in T2. This flexibility does not exist in a flip-flop based design in Fig. 9.66.



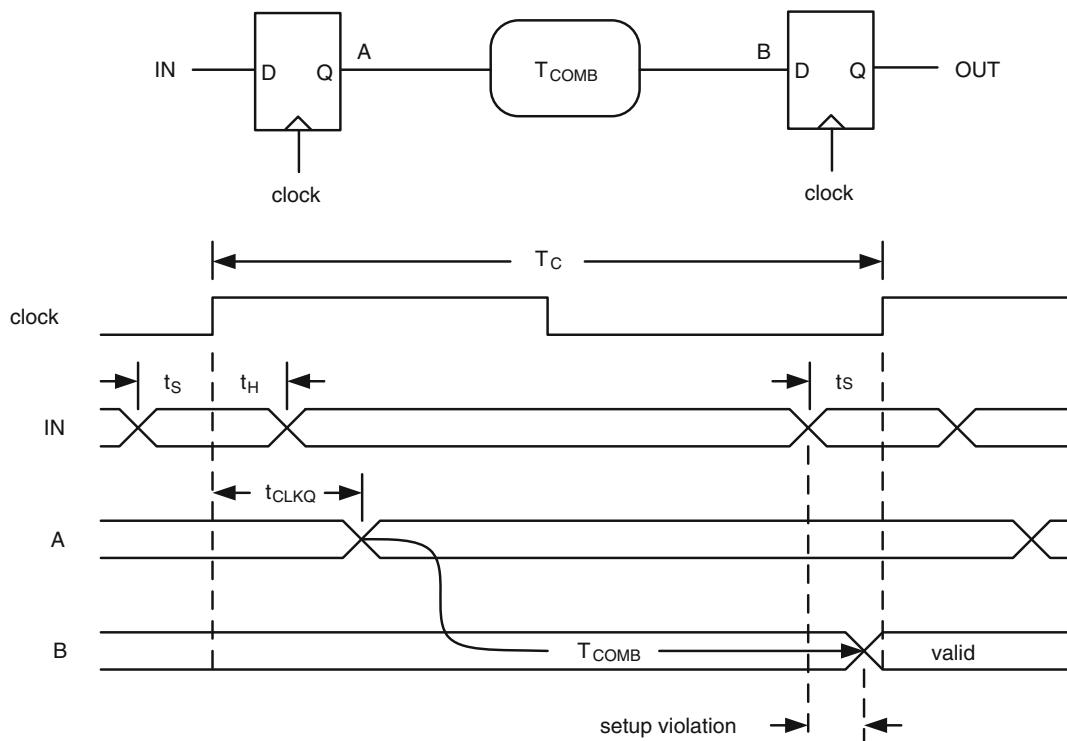
**Fig. 9.66** Timing methodology using D flip-flops

## 9.14 Timing Violations

Although pipelining scheme helps to reduce the overall data processing time, we still need to watch out for possible timing violations because occasionally data does not comply with the requirements of flip-flop (or latch) set-up and hold times at pipeline boundaries.

This section examines the set-up and hold timing violations in a pipeline controlled by flip-flops. Figure 9.67 shows a section of a pipeline where a combinational logic block with a propagation delay of  $T_{COMB}$  is sandwiched between two flip-flops. At the rising edge of the clock, the valid and steady data that meets the set-up and hold times is fed into the pipeline from the IN terminal. After  $t_{CLKQ}$  delay at the node A, the data propagates through the combinational logic block and emerges at the node B as shown in the timing diagram. However, the data arrives at the node B past the allocated set-up time of the flip-flop and creates a set-up violation. The amount of violation is dependent on the clock period and is calculated as follows:

$$\text{Set-up violation} = t_s - [T_C - (t_{CLKQ} + T_{COMB})] \quad (9.1)$$



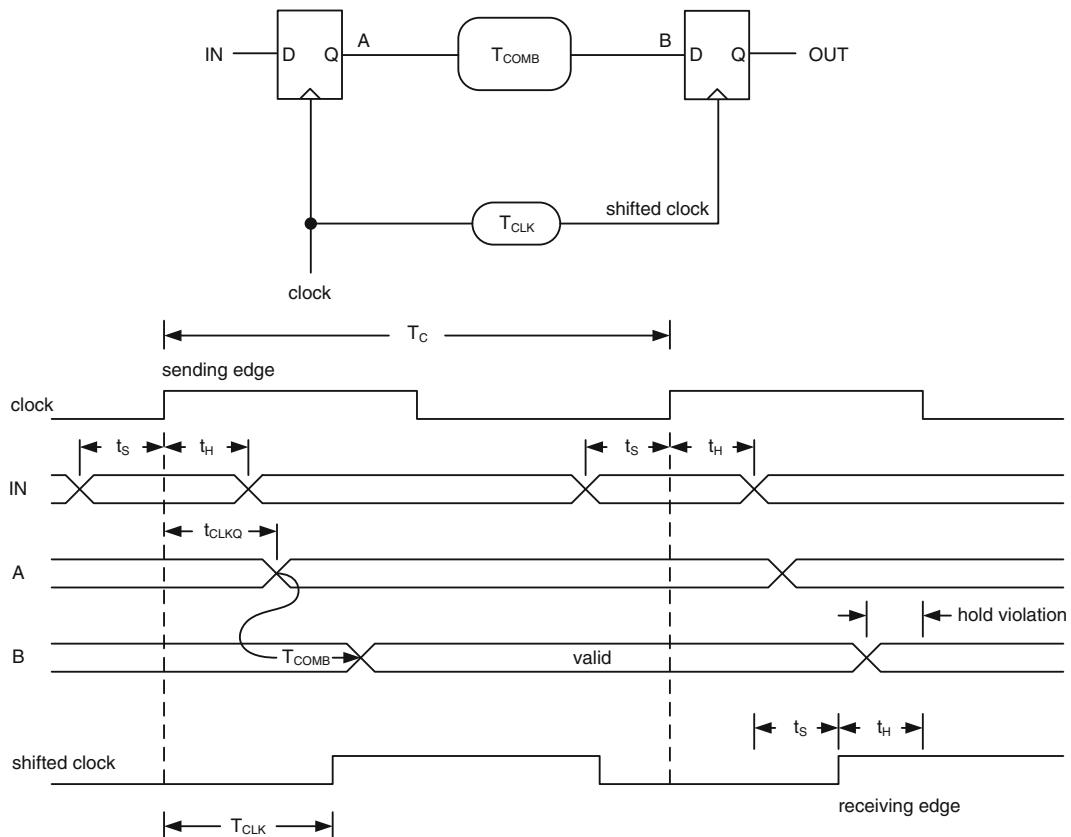
**Fig. 9.67** Setup violation

Figure 9.68 describes the hold time violation where the clock shifts by  $T_{CLK}$  due to an unexpected delay in the clock line. In the timing diagram, the valid data is fed into the pipeline from the IN terminal, and arrives at the node B after  $t_{CLKQ}$  and  $T_{COMB}$  delays. Due to the shifted clock, the data arrives at the node B very early. This creates a substantial set-up time slack equal to  $(T_C + T_{CLK} - t_s - t_{CLKQ} - T_{COMB})$  but produces a hold time violation at the delayed clock edge. The amount of violation is dependent on the clock delay and is calculated as follows:

$$\text{Hold violation} = (T_{CLK} + t_H) - (t_{CLKQ} + T_{COMB}) \quad (9.2)$$

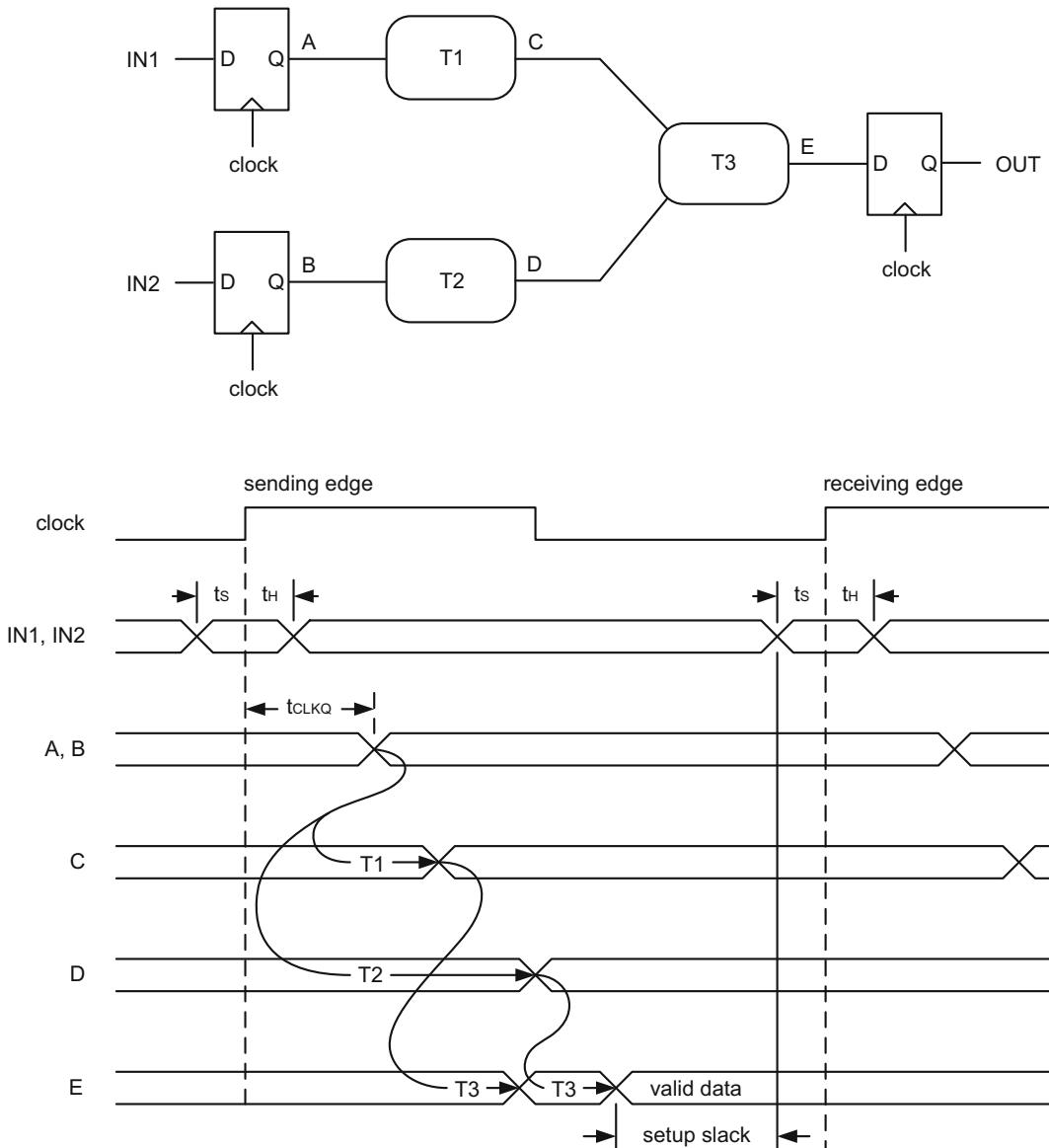
Set-up violations can be recovered simply by increasing the clock period,  $T_C$ . However, there is no easy way to fix hold violations. We need to search for hold time violation at each flip-flop input, and add additional delay to the combinational logic block terminating at the particular flip-flop in order to avoid the violation.

The schematic in Fig. 9.69 examines the timing ramifications of two combinational logic blocks with different propagation delays merging into one block in a pipeline stage. The data



**Fig. 9.68** Hold violation

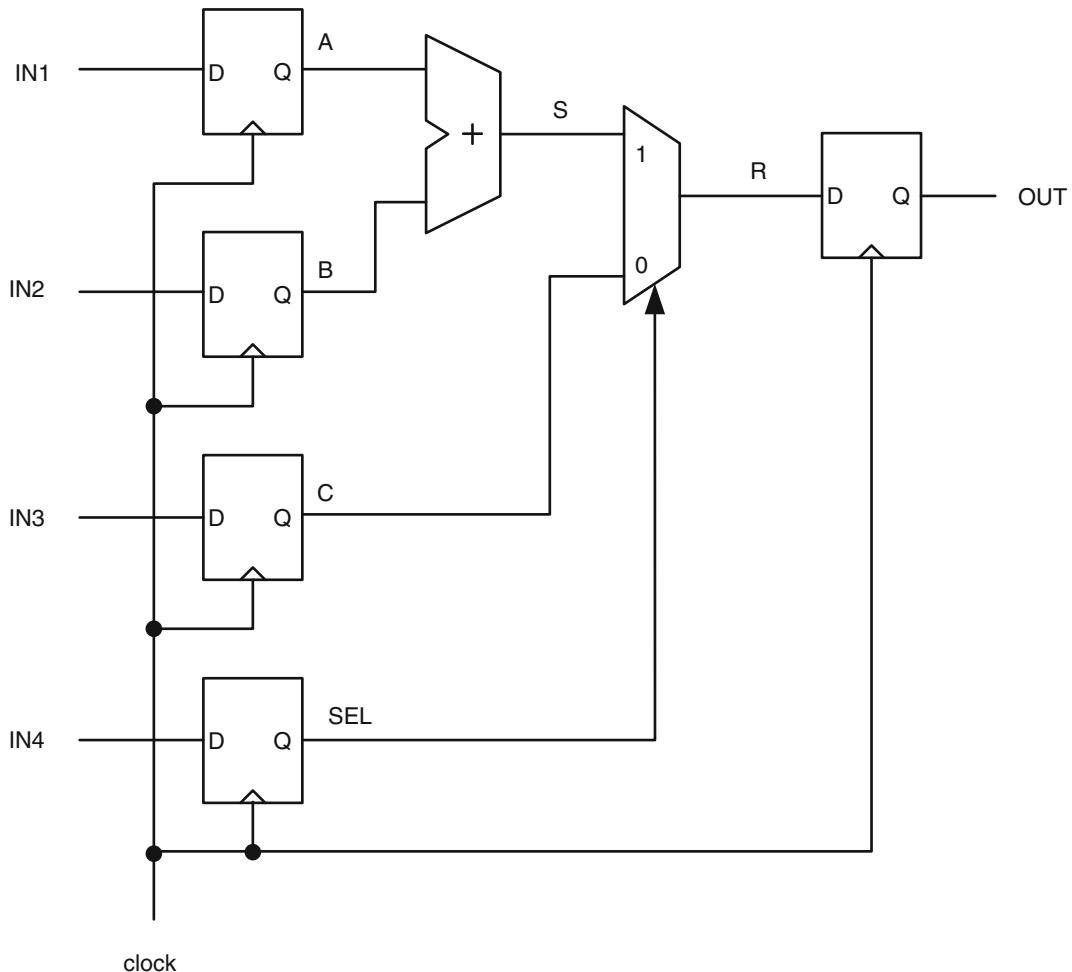
at the node A arrives at the node C much earlier than the data arriving at the node D as shown in the timing diagram. The data at the nodes C and D propagate through the last combinational logic block and arrive at the node E. This scenario creates minimum and maximum paths at the node E. We need to focus on the maximum path, ( $T_2 + T_3$ ), when examining the possibility of a set-up violation and the minimum path, ( $T_1 + T_3$ ), when examining the possibility of a hold violation at the next clock edge.



**Fig. 9.69** A timing example combining two independent data-paths

To further illustrate the timing issues through multiple combinational logic blocks, an example with logic gates is given in Fig. 9.70. In this example, the inputs of a one-bit adder are connected to the nodes A and B. The adder is bypassed with the inclusion of a 2-1 MUX which selects either the output of the adder or the bypass path by a selector input, SEL.

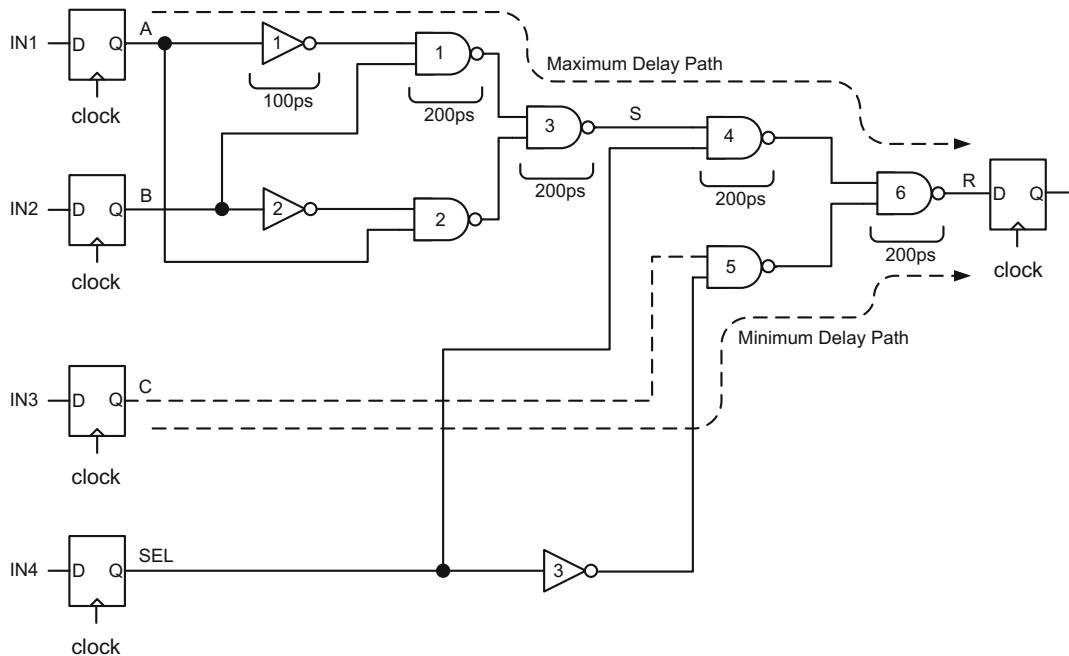
The propagation delays of the inverter,  $T_{INV}$ , and the two-input NAND gate,  $T_{NAND2}$ , are given as 100 ps and 200 ps, respectively. The set-up, hold and clock-to-q delays are also given as 100, 0 and 300 ps, respectively.



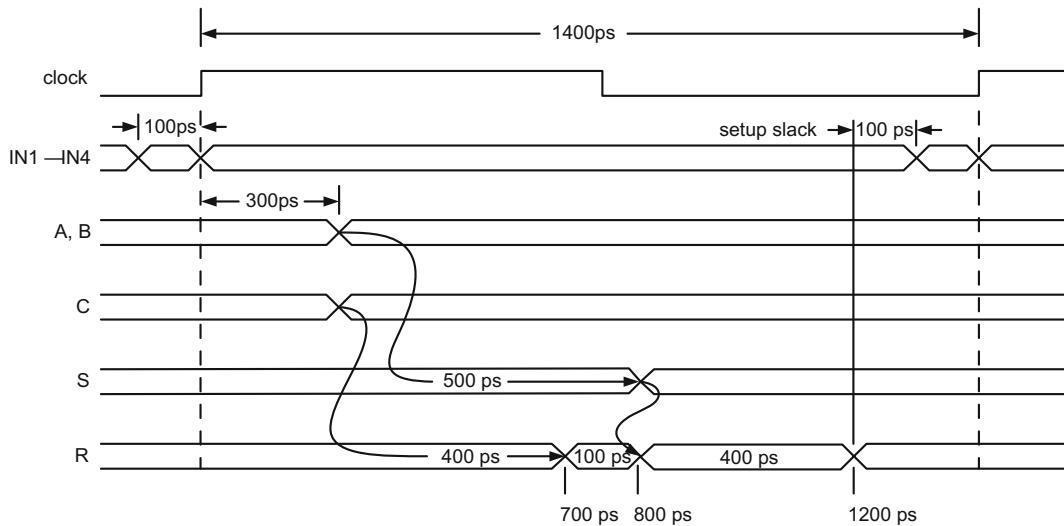
**Fig. 9.70** An example with multiple propagation paths

One-bit full adder and the 2-1 MUX are shown in terms of inverters and two-input NAND gates in Fig. 9.71. We obtain a total of seven propagation paths all merging at the node R. However, we only need to search for the maximum and the minimum paths to locate possible set-up and hold violations.

The maximum delay path consists of the inverter 1, and the series combination of four two-input NAND gates numbered as 1, 3, 4 and 6 shown in the schematic. This path results in a total delay of 900 ps. The minimum delay path, on the other hand, contains two two-input NAND gates numbered as 5 and 6, and it produces a delay of 400 ps. Placing these propagation delays in a timing diagram in Fig. 9.72 ultimately yields a set-up slack of 100 ps at the node R when a clock period of 1400 ps is used. There is no need to investigate for hold violations because there is no shift in the clock edge. The data emerges at the nodes A, B, C and SEL after  $t_{CLKQ} = 300$  ps where  $t_H = 0$  ps.



**Fig. 9.71** Logic circuit of Fig. 9.70 showing maximum and minimum paths



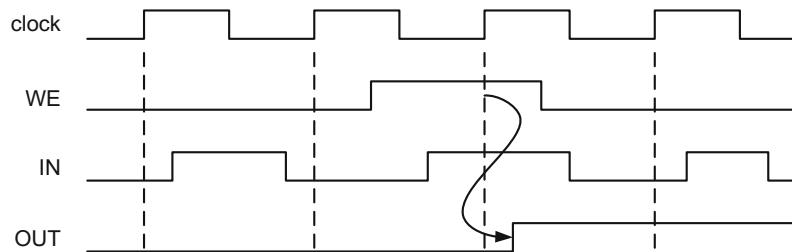
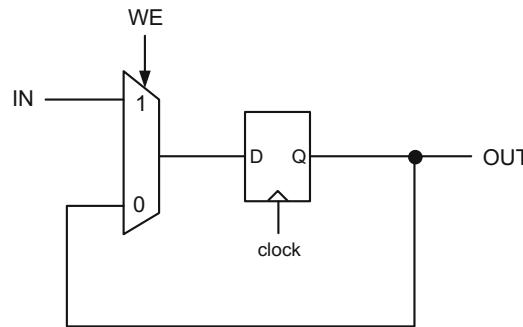
**Fig. 9.72** Timing diagram of the circuit in Fig. 9.71

## 9.15 Register

While a flip-flop can hold data only for one clock cycle until a new data arrives at the next clock edge, a register can hold the same data perpetually until the power is turned off.

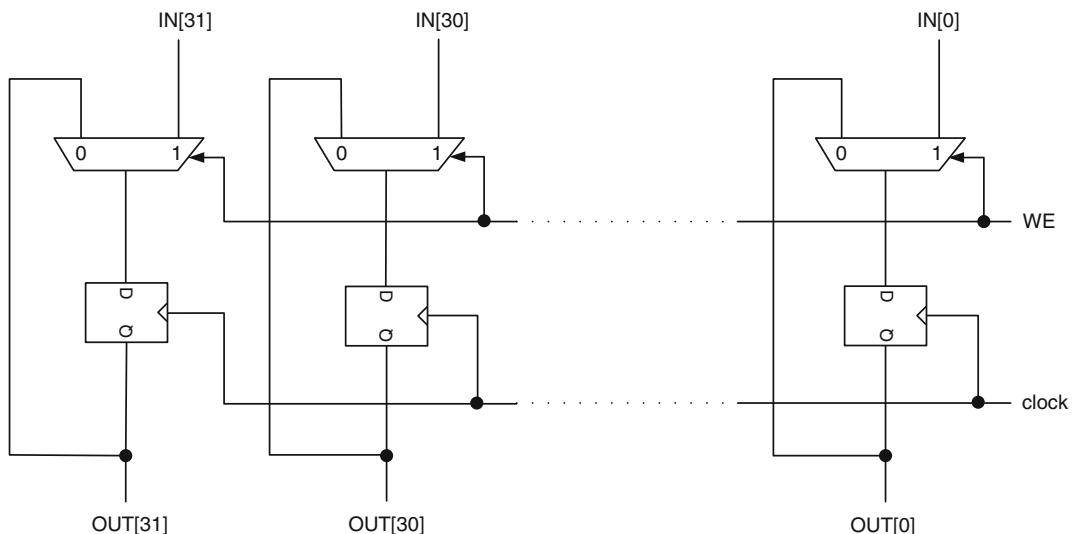
Figure 9.73 shows the circuit diagram of a one-bit register composed of a flip-flop and a 2-1 MUX. The Write Enable pin, WE, is a selector input to the 2-1 MUX and transfers new data from the IN terminal to the flip-flop input when WE = 1. If the WE input is at logic 0, any attempt to write a new data to the register is blocked; the old data stored in the flip-flop simply circulates around the feedback loop from one clock cycle to the next.

The timing diagram at the bottom of Fig. 9.73 describes the operation of the one-bit register. The data at the IN terminal is blocked until the WE input becomes logic 1 in the middle of the second clock cycle. At this point, the new data is allowed to pass through the 2-1 MUX and updates the contents of the register at the rising edge of the third clock cycle. Since the WE input is pulled to logic 0 before the end of the third clock cycle, the register output, OUT, stays at logic 1 during the fourth clock cycle.



**Fig. 9.73** One-bit register and a sample timing diagram

A 32-bit register shown in Fig. 9.74 is composed of 32 one-bit registers. All 32 registers have a common clock and WE input. Therefore, any new 32-bit data introduced at the register input changes the contents of the register at the rising edge of the clock if the WE input is kept at logic 1.



**Fig. 9.74** 32-bit register

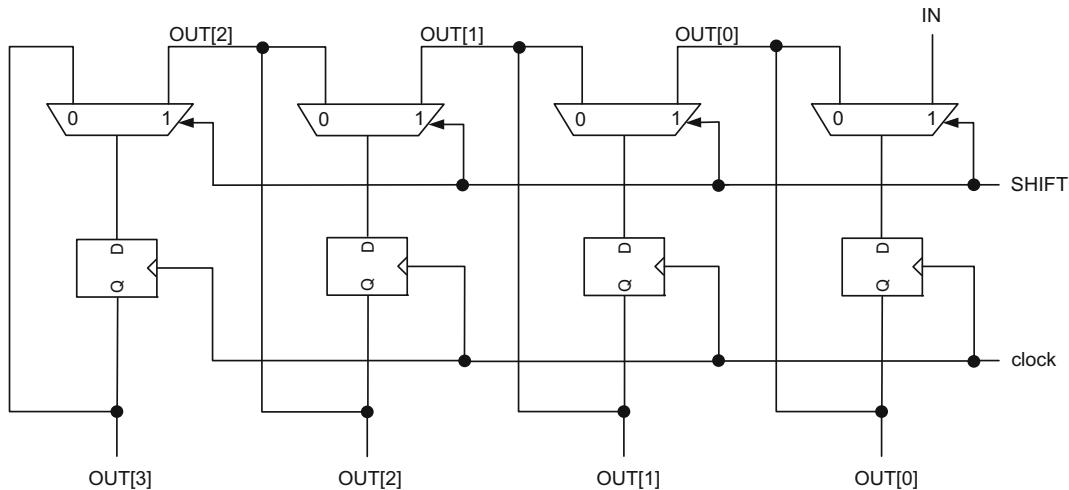
## 9.16 Shift Register

The shift register is a particular version of an ordinary register and specializes in shifting data to the right or left according to design needs.

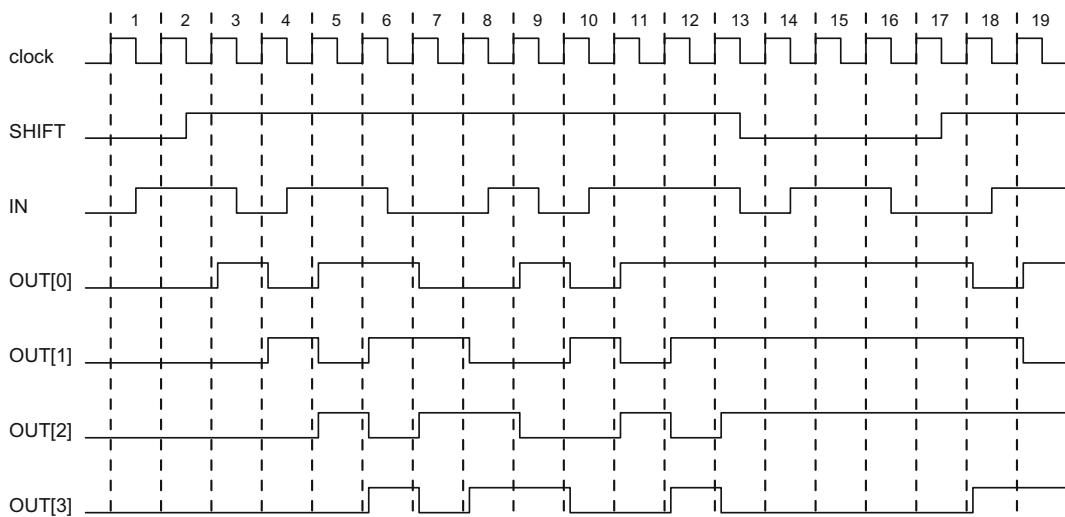
Figure 9.75 shows the circuit schematic of a four-bit shift register that shifts serial data at the IN terminal to the left at every positive clock edge.

The operation of this shift register is explained in the timing diagram in Fig. 9.76. In cycle 1, SHIFT = 0. Therefore, the change at the IN terminal during this cycle does not affect the register outputs. However, when the SHIFT input transitions to logic 1 in the middle of cycle 2, it allows IN = 1 to pass to the least significant output bit, OUT[0], at the beginning of the third clock cycle. From the middle of cycle 2 to cycle 13, SHIFT is kept at logic 1. Therefore, any change at the IN node directly transmits to the OUT[0] node at the positive edge of every clock cycle. The other outputs, OUT[1], OUT[2] and OUT[3], produce the delayed version of the OUT[0] one clock cycle apart from each other because the output of a lesser significant bit is connected to the input of a greater significant bit in the shift register.

When the SHIFT input transitions to logic 0 from the middle of cycle 13 to cycle 17, the shift register becomes impervious to any new data entry at the IN terminal, and retains the old values from the beginning of cycle 13 to cycle 18 as shown in Fig. 9.76. From the middle of cycle 17, the SHIFT input becomes logic 1 again, and the shift register distributes new data entries at the IN terminal to its outputs.



**Fig. 9.75** Four-bit shift register



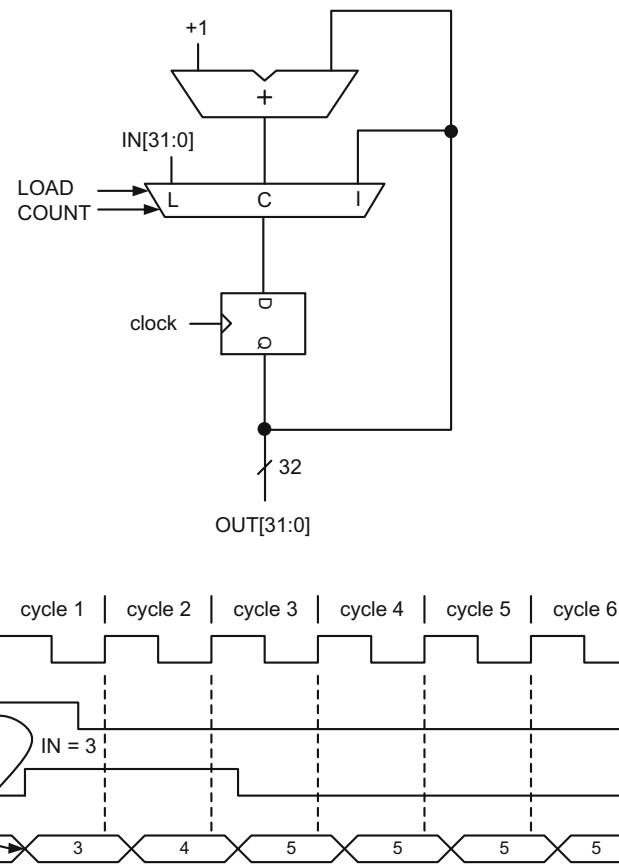
**Fig. 9.76** A sample timing diagram of the four-bit shift register in Fig. 9.75

## 9.17 Counter

The counter is a special form of a register which is designed to count up (or down) at each rising edge of the clock.

The counter in Fig. 9.77 shows a typical 32-bit up-counter with two control inputs, COUNT and LOAD. COUNT = 1 enables the counter to count upwards at the rising edge of each clock cycle. LOAD = 1 loads new data at the IN terminal to the counter. Once loaded, the counter output, OUT[31:0], increments by one at every positive clock edge until all of its outputs become logic 1. The next increment automatically resets all the counter outputs to logic 0. When LOAD = COUNT = 0, the counter neither loads new data nor is able to count upwards; it stalls and repeats its old output value.

The sample timing diagram at the bottom of Fig. 9.77 illustrates its operation. Prior to the first clock edge, the LOAD input is at logic 1 which allows an input value, IN[31:0] = 3, to be stored in the counter. This results in OUT[31:0] = 3 at the positive edge of the first clock cycle. During the first clock cycle, LOAD = 0 and COUNT = 1 start the up-count process. The contents of the output, OUT[31:0] = 3, subsequently increments by one. The result,  $3 + 1 = 4$ , passes through the C-port of the 3-1 MUX and arrives at the flip-flop inputs. At the positive edge of the second clock cycle, this new value overwrites the old registered value, and the OUT[31:0] node becomes equal to 4. In the next cycle, the counter goes through the same process and increments by one. However, in the same cycle, the LOAD input also transitions to logic 0, and turns on the I-port of the 3-1 MUX. This new port prevents any new data from entering the up-counter, but keeps the old data in the following clock cycles. As a result, the counter output stops incrementing and stalls at the value of OUT[31:0] = 5.



**Fig. 9.77** A 32-bit counter and a sample timing diagram

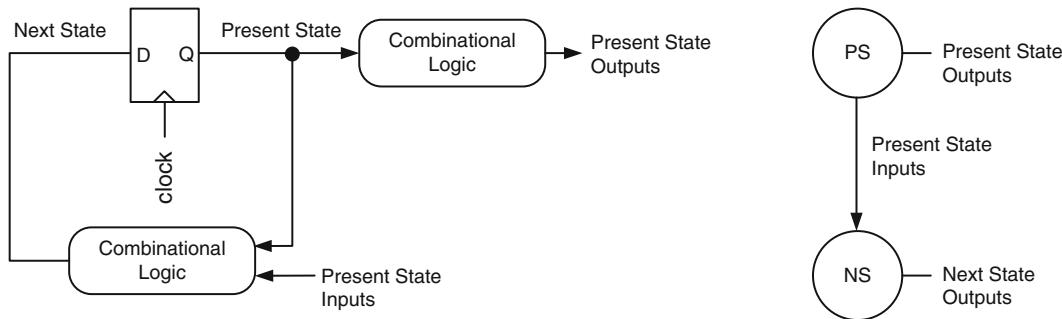
## 9.18 Moore-Type State Machine

State machines are mainly used in digital designs to control the proper data-flow. Their topology is mainly composed of one or multiple flip-flops and feedback loop(s) that connect flip-flop outputs to flip-flop inputs. There are two types of state machines: Moore-type and Mealy-type.

Figure 9.78 shows the Moore-type state machine topology consisting of a flip-flop and a feedback loop. In this configuration, the feedback loop includes a combinational logic block that accepts both the flip-flop output and external inputs. If there are multiple flip-flops, the combination of all flip-flop outputs constitutes the “present” state of the machine. The combination of all flip-flop inputs is called the “next” state because at the positive edge of the clock these inputs become the flip-flop outputs, and form the “next” present state. Flip-flop outputs may be processed further by an additional combinational logic block to form present state outputs.

The basic state diagram of a Moore machine, therefore, includes the present state (PS) and the next state (NS) as shown on the right hand side of Fig. 9.78. The machine can transition

from the PS to the NS if the required present state inputs are supplied. The outputs of the Moore machine are solely generated by the present state. Therefore, machine outputs emerge only from the present and the next states as shown in the basic state diagram.



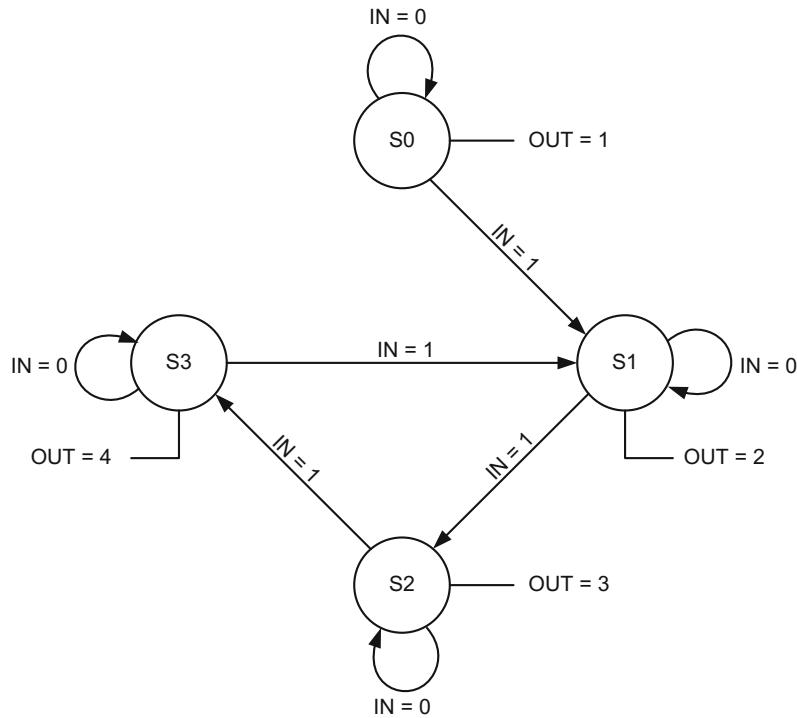
**Fig. 9.78** Block diagram and state representation of Moore machine

The state diagram in Fig. 9.79 shows an example of a Moore-type machine with four states. Note that every state-to-state transition in the state diagram requires a valid present state input entry, and every node generates one present state output.

The state 0, S0, produces a present state output, OUT = 1, regardless of the value of the present state input, IN. When IN = 1, the state S0 transitions to the next state S1; otherwise, it maps onto itself. The state S1 produces OUT = 2. Its next state becomes the state S1 if IN = 0; otherwise, it transitions to a new state S2. The state S2 also produces a present state output, OUT = 3, and transitions to the state S3 if IN = 1. The state S2 remains unchanged if IN = 0. In the fourth and final state S3, the present state output, OUT = 4, is produced. The machine stays in this state if IN stays at 0; otherwise, it goes back to the state S1.

The present state inputs and outputs of this Moore machine and its states can be tabulated in a table called the “state table” given in Fig. 9.80. In this table, the first column under PS lists all the possible present states of the state diagram in Fig. 9.79. The middle two columns contain the next state entries for IN = 0 and IN = 1. The last column lists the present state outputs, one for each present state.

The binary state assignment is performed in Fig. 9.81 where only one bit is changed between adjacent states.

**Fig. 9.79** State diagram of a Moore machine with four states

PS	NS		
	IN = 0	IN = 1	OUT
S0	S0	S1	1
S1	S1	S2	2
S2	S2	S3	3
S3	S3	S1	4

**Fig. 9.80** State table of the Moore machine in Fig. 9.79

States	NS1	NS0
S0	0	0
S1	0	1
S2	1	1
S3	1	0

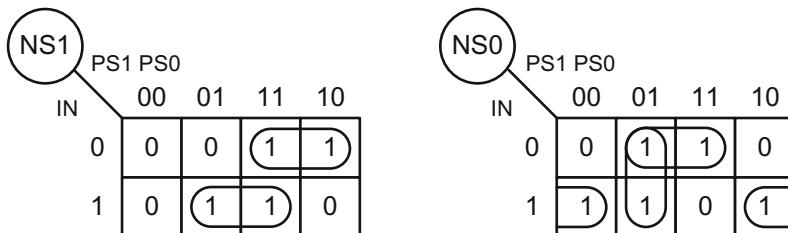
**Fig. 9.81** Bit representations of states S0, S1, S2 and S3

The binary form of the state table in Fig. 9.80 is reconstructed in Fig. 9.82 according to the state assignment in Fig. 9.81. This table, called the “transition table”, includes the binary representation of the next state and the present state outputs.

PS1	PS0	IN = 0		IN = 1		OUT2	OUT1	OUT0
		NS1	NS0	NS1	NS0			
0	0	0	0	0	1	0	0	1
0	1	0	1	1	1	0	1	0
1	1	1	1	1	0	0	1	1
1	0	1	0	0	1	1	0	0

**Fig. 9.82** Transition table of the Moore machine in Fig. 9.79

Forming this machine’s K-maps for the NS0, NS1, OUT0, OUT1 and OUT2 requires grouping all the input terms, PS1, PS0 and IN, according to the table in Fig. 9.82. The K-maps and their corresponding SOP representations are shown in Fig. 9.83.



$$NS1 = PS0 \cdot IN + PS1 \cdot \overline{IN}$$

$$NS0 = PS0 \cdot \overline{IN} + \overline{PS1} \cdot PS0 + \overline{PS0} \cdot IN \\ = (PS0 \oplus IN) + \overline{PS1} \cdot PS0$$

$$OUT2 = PS1 \cdot \overline{PS0}$$

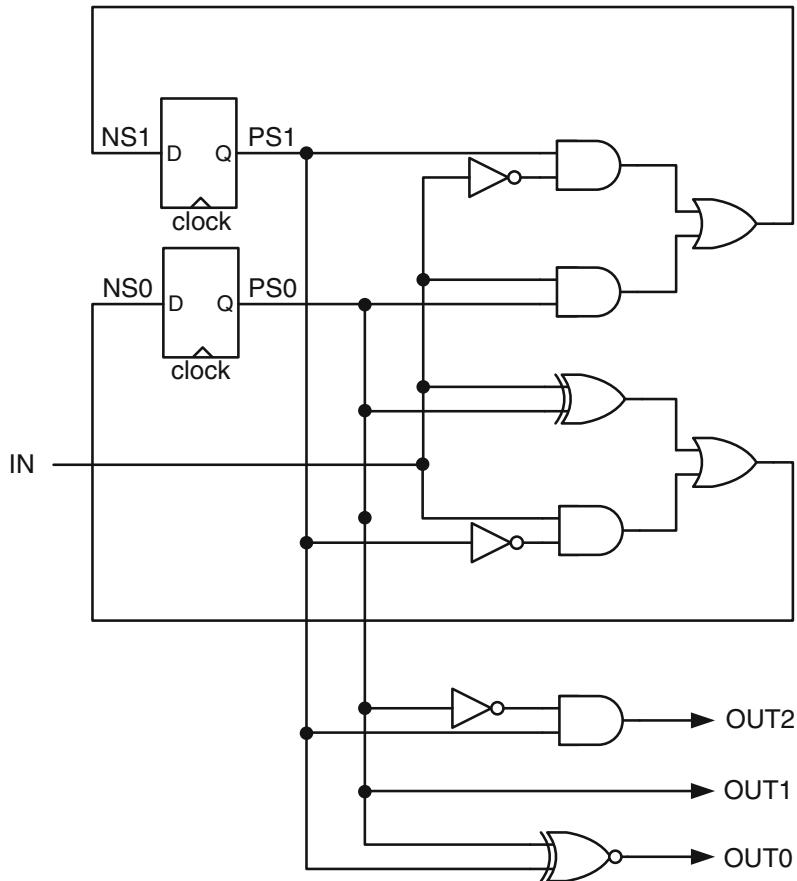
$$OUT1 = \overline{PS1} \cdot PS0 + PS1 \cdot PS0 = PS0$$

$$OUT0 = \overline{PS1} \cdot \overline{PS0} + PS1 \cdot PS0 = PS0 \oplus PS0$$

**Fig. 9.83** K-maps and SOP expressions for the Moore machine in Fig. 9.79

The next step is to generate the circuit diagram that produces all five outputs of the Moore machine according to the SOP expressions in Fig. 9.83. This circuit diagram is given in Fig. 9.84.

In order to generate this circuit, the individual combinational logic blocks for the NS0 and NS1 must be constructed first in terms of PS0, PS1 and IN. Then each NS0 and NS1 node is connected to the corresponding flip-flop input to form the feedback loops of the state machine. The logic blocks for the OUT0, OUT1 and OUT2 are generated directly from PS0 and PS1.

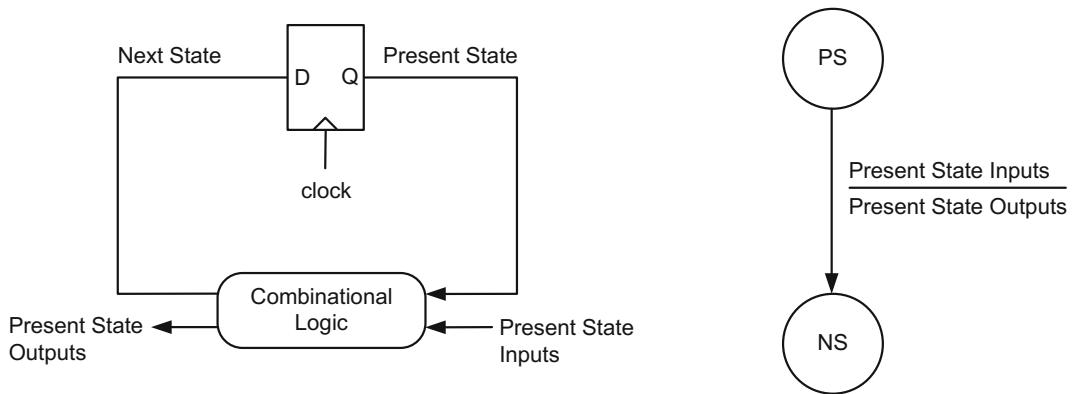


**Fig. 9.84** Logic circuit of the Moore machine in Fig. 9.79

### 9.19 Mealy-Type State Machine

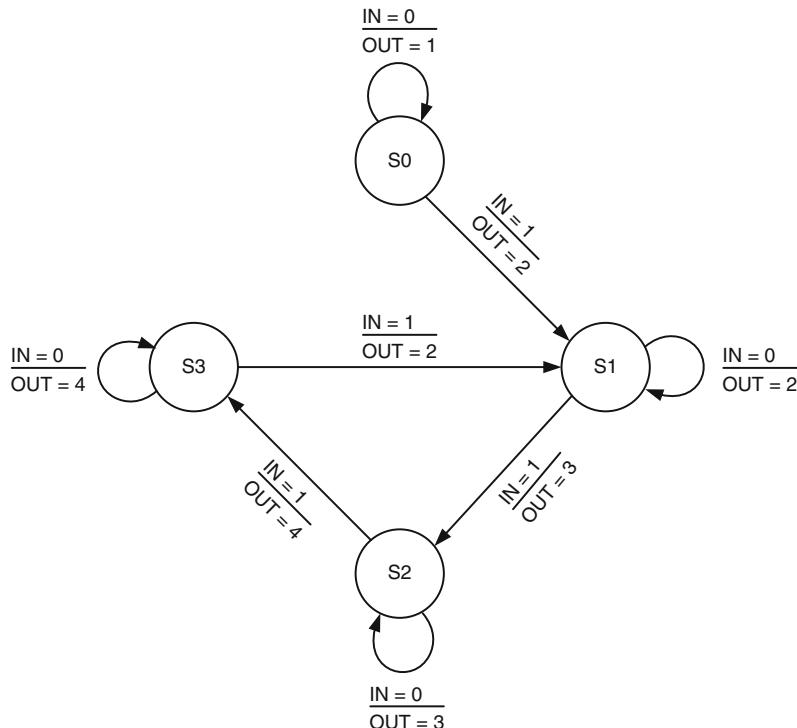
The Mealy-type state machine shares the same circuit topology as the Moore-type machine. The machine contains one or more flip-flops and feedback loop(s) as shown in Fig. 9.85. However, present state outputs are generated from the combinational logic block in the feedback loop rather than from present states in the Moore-type machines.

As a result of this topology, the basic state diagram of a Mealy machine includes the present state, the next state and the input condition that transition the present state to the next state as shown on the right hand side of Fig. 9.85. The present state outputs do not emerge from each present state; instead, they are functions of the present state inputs and the present state.



**Fig. 9.85** Block diagram and state representation of Mealy-type machine

The Mealy state diagram in Fig. 9.86 exhibits similar characteristics compared to the Moore state diagram in Fig. 9.79, and all the state names and the state-to-state transitions in this diagram are kept the same for comparison purposes. However, each arrow connecting one state to the next now carries the value of the present state output as a function of the present state input as indicated in Fig. 9.85. As a result, the Mealy state table in Fig. 9.87 contains two separate columns that tabulate the values of NS and OUT for IN = 0 and IN = 1. The binary state assignment is the same as in Fig. 9.81, which results in a transition table in Fig. 9.88.



**Fig. 9.86** State diagram of a Mealy-type machine with four states

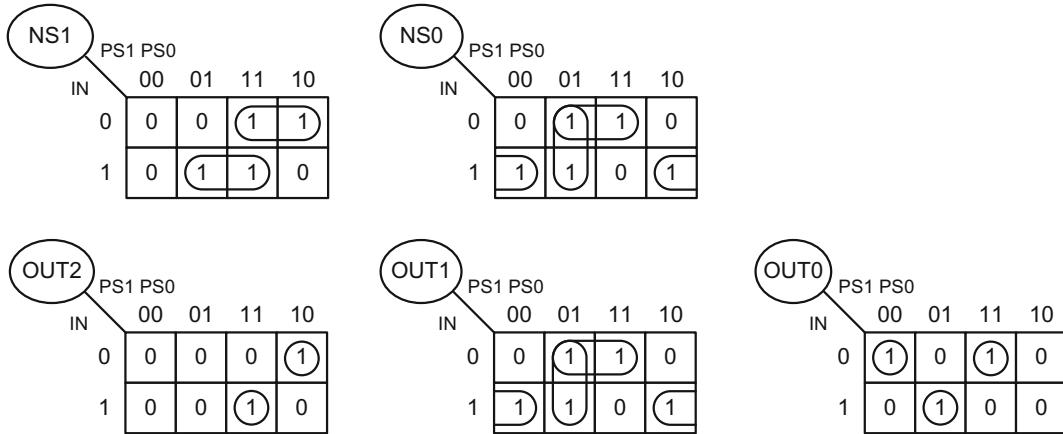
PS	NS		OUT	
	IN = 0	IN = 1	IN = 0	IN = 1
S0	S0	S1	1	2
S1	S1	S2	2	3
S2	S2	S3	3	4
S3	S3	S1	4	2

**Fig. 9.87** State table of the Mealy-type machine in Fig. 9.86

PS1	PS0	IN = 0		IN = 1		IN = 0			IN = 1		
		NS1	NS0	NS1	NS0	OUT2	OUT1	OUT0	OUT2	OUT1	OUT0
0	0	0	0	0	1	0	0	1	0	1	0
0	1	0	1	1	1	0	1	0	0	1	1
1	1	1	1	1	0	0	1	1	1	0	0
1	0	1	0	0	1	1	0	0	0	1	0

**Fig. 9.88** Transition table of the Mealy-type machine in Fig. 9.86

The K-maps for the NS0, NS1, OUT0, OUT1 and OUT2 are formed according to the table in Fig. 9.88 and are shown in Fig. 9.89 with the corresponding SOP expressions. Figure 9.90 shows the circuit diagram of this machine according to the expressions in Fig. 9.89. The methodology used to construct this circuit diagram is identical to the methodology used in the circuit diagram for the Moore machine in Fig. 9.84.



$$NS1 = PS0.IN + PS1.\overline{IN}$$

$$NS0 = PS0.\overline{IN} + \overline{PS1}.PS0 + \overline{PS0}.IN$$

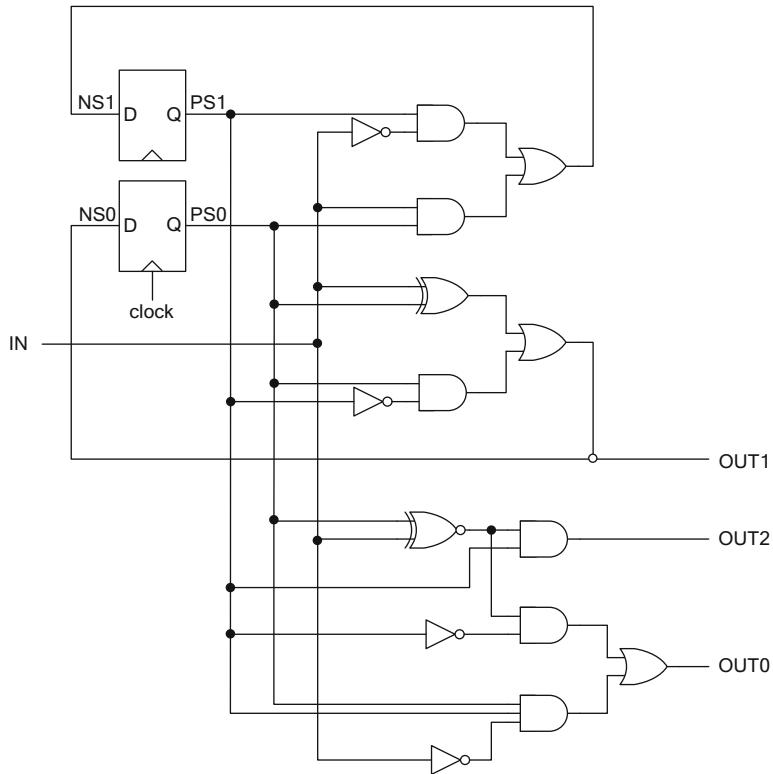
$$= (PS0 \oplus IN) + \overline{PS1}.PS0$$

$$OUT2 = PS1.\overline{PS0}.\overline{IN} + PS1.PS0.IN = PS1.(PS0 \oplus IN)$$

$$OUT1 = (PS0 \oplus IN) + \overline{PS1}.PS0 = NS0$$

$$OUT0 = \overline{PS1}.\overline{PS0}.\overline{IN} + \overline{PS1}.PS0.IN + PS1.PS0.\overline{IN} = \overline{PS1}.\overline{(PS0 \oplus IN)} + PS1.PS0.\overline{IN}$$

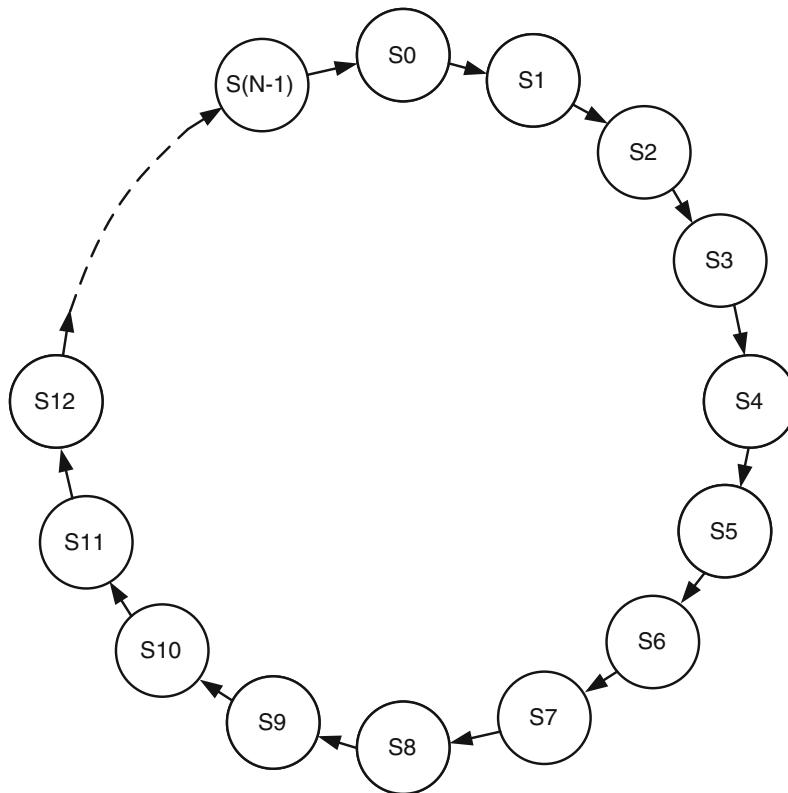
**Fig. 9.89** K-maps and SOP expressions for the Mealy-type machine in Fig. 9.86



**Fig. 9.90** Logic circuit of the Mealy-type machine in Fig. 9.86

## 9.20 Controller Design: Moore-Type State Machine Versus Counter-Decoder Scheme

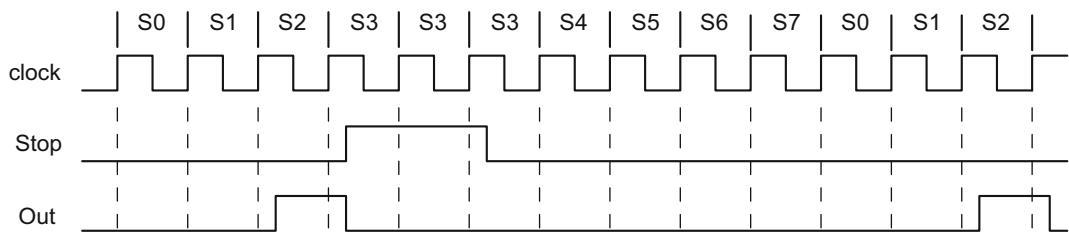
Both Mealy and Moore-type state machines have practical implementation limits when it comes to design. A large ring-style state machine composed of  $N$  states such as Fig. 9.91 may have multiple outputs attached to each state, making its implementation nearly impossible with conventional state machine implementation techniques. However, these types of designs are excellent candidates for the counter-decoder type of designs where each state in the state diagram is associated with a counter output value. Therefore, as the counter increments, the present state outputs of each state in Fig. 9.91 can simply be generated using a set of decoders connected at the counter output.



**Fig. 9.91** State diagram of a counter with  $N$  states

To illustrate this theory, a controller that generates the timing diagram in Fig. 9.92 will be implemented using both the Moore-type state machine and the counter-decoder approach.

As shown in the timing diagram below, this state machine generates a single active-high output,  $\text{Out} = 1$ , once in every eight cycles as long as  $\text{Stop} = 0$ . When  $\text{Stop} = 1$ , the machine stalls and retains its current state.

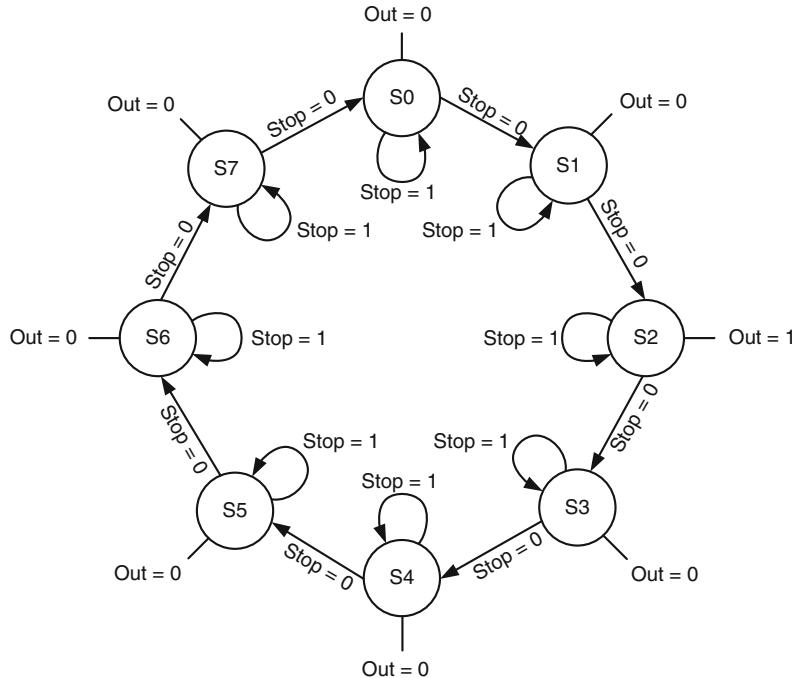


**Fig. 9.92** Timing diagram of a state machine with a single input, Stop, and a single output

Once the state assignments are made for each clock cycle of Fig. 9.92, the state diagram in Fig. 9.93 emerges for a Moore-type state machine.

The states  $S0$  and  $S1$  in the timing diagram are assigned to the first and second clock cycles, respectively. The third clock cycle is assigned to state  $S2$  where  $\text{Out} = 1$ . The fourth

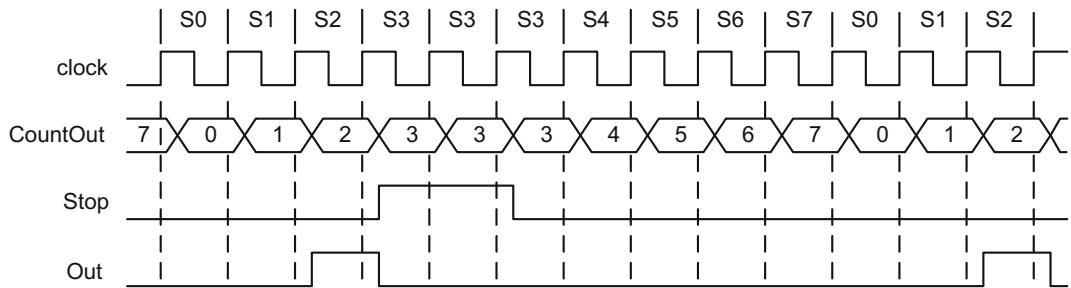
clock cycle corresponds to state S3. The machine stays in state S3 as long as Stop = 1. This ranges from the fourth to the sixth clock cycle in the timing diagram. The state assignments from the seventh to the tenth clock cycles become states S4, S5, S6 and S7. The eleventh clock cycle returns to state S0.



**Fig. 9.93** Moore-type state machine representation of the timing diagram in Fig. 9.92

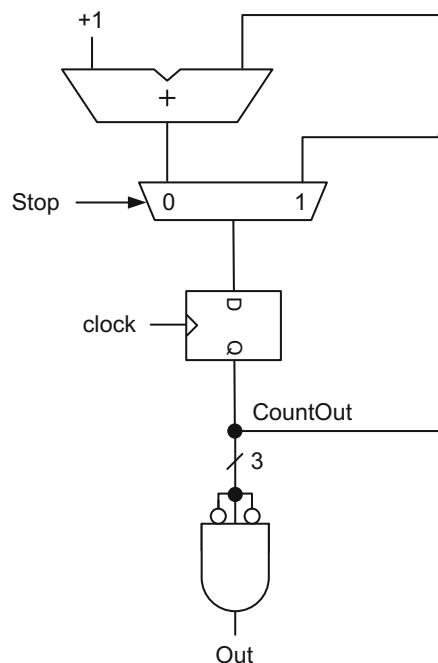
Implementing the state diagram in Fig. 9.93 follows a lengthy process of producing the state tables, transition tables, and K-maps that results in a total of four outputs (three flip-flop outputs due to eight states and one output for Out). However, using a counter-decoder approach minimizes this design task considerably and reveals a rather explicit circuit implementation.

When the timing diagram in Fig. 9.92 is redrawn to implement the counter-decoder design approach, it yields a simple three-bit counter which counts from zero to seven as shown in Fig. 9.94. The counter output, CountOut, is included in this figure to show the relationships between the state assignments, the input node, Stop, and the output node, Out. The figure also shows the clock cycle where the counter resets itself when its output reaches seven.



**Fig. 9.94** Timing diagram of a three-bit counter with a single input (Stop) and a single output

The first task for the design is to construct a three-bit up-counter as shown in Fig. 9.95. The counter in this figure is derived from a general counter topology, and it consists of a three-bit adder, three 2-1 MUXes and three flip-flops. A three-input AND gate is used as a decoder at the counter output to implement  $\text{Out} = 1$  when the CountOut node reaches 2. Therefore, this method follows a simple, step-by-step design approach in producing the final circuit that does not require implicit logic design techniques.



**Fig. 9.95** Counter-decoder representation of the timing diagram in Fig. 9.94

## 9.21 A Simple Memory Block

Small memory blocks can be assembled from one-bit registers shown in Fig. 9.73 to use in a variety of designs. For example, a 32-bit wide, 16-bit deep memory block shown in Fig. 9.96 can be built by stacking 16 rows of 32-bit registers on top of each other. The 32-bit register in each row has tri-state output buffers to be used during read as shown in Fig. 9.97.

All inputs to each column of Fig. 9.96 are connected together to write data. For example, the input terminal, IN[0], in Fig. 9.96 is connected to all the input pins, In[0], of rows 0 to 15 in Fig. 9.97 to write a single bit of data at a selected row. The same is true for the remaining inputs, IN[1] through IN[31].

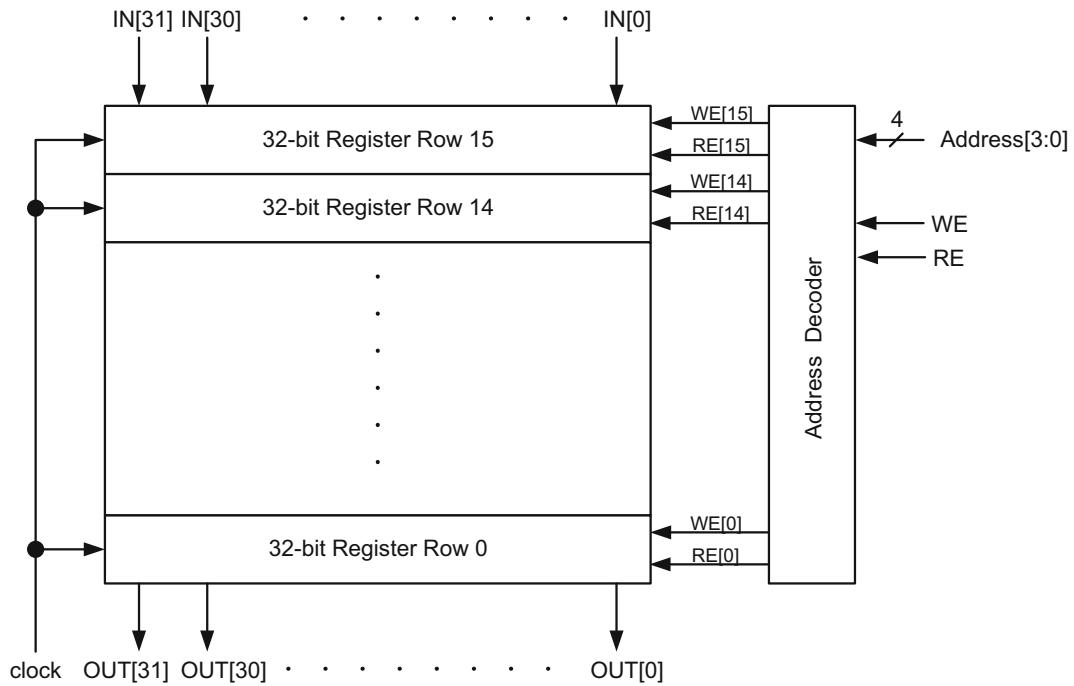
Similarly, all outputs of each column in Fig. 9.96 are connected together to read data from the memory block. For example, the output pin, OUT[0], is connected to all output pins, Out[0], from each row to read one bit of data from a selected row. The same is true for the remaining output pins, OUT[1] through OUT[31].

Every row of the memory block in Fig. 9.96 is accessed by individual Write Enable (WE) and Read Enable (RE) signals for writing or reading data, respectively.

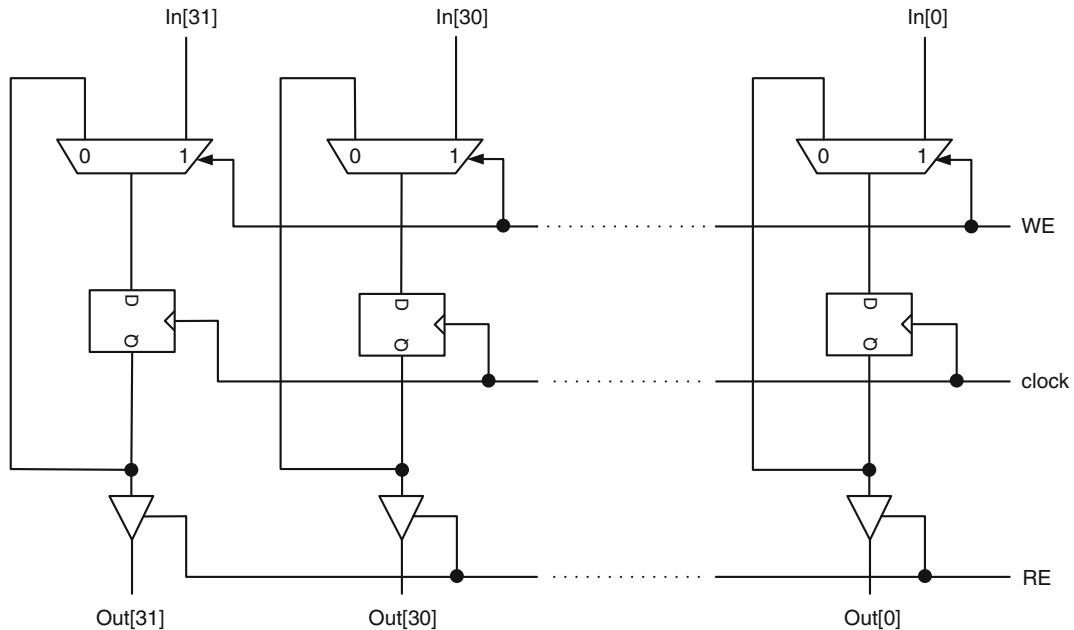
In order to generate the WE inputs, WE[0] to WE[15], an address decoder is used. This decoder enables only one row while deactivating all the other rows using a four-bit address, Address[3:0], and a single WE input according to the truth table in Fig. 9.98. For example, a 32-bit data is written to row 0 if WE = 1 and Address[3:0] = 0000 at the decoder input. However, WE = 0 blocks writing data to all rows of the memory block regardless of the input address as shown in the truth table in Fig. 9.99.

The RE inputs, RE[0] through RE[15], use address decoders similar to Fig. 9.98 and Fig. 9.99 to read a block of data from a selected row. The read operation is achieved with a valid input address and RE = 1 according to the truth table in Fig. 9.100. An RE = 0 entry disables reading data from any row regardless of the value of the input address as shown in Fig. 9.101.

Therefore, one must provide a valid input address and the control signals, RE and WE, to perform a read or a write operation, respectively. The WE = 0 and RE = 1 combination reads data from the selected row. Similarly, the WE = 1 and RE = 0 combination writes data to a selected row. The WE = 0 and RE = 0 combination disables both reading and writing to the memory block. The control input entry, WE = 1 and RE = 1, is not allowed, and it should be interpreted as memory read.



**Fig. 9.96** A  $32 \times 16$  memory and the truth table of its address decoder



**Fig. 9.97** A 32-bit register slice at every row of Fig. 9.96

Address	WE[15]	WE[14]	WE[13]		WE[2]	WE[1]	WE[0]
0 0 0 0	0	0	0	.	0	0	1
0 0 0 1	0	0	0	.	0	1	0
0 0 1 0	0	0	0	.	1	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
1 1 1 0	0	1	0	.	0	0	0
1 1 1 1	1	0	0	.	0	0	0

**Fig. 9.98** The address decoder for the  $32 \times 16$  memory in Fig. 9.96 when WE = 1

Address	WE[15]	WE[14]	WE[13]		WE[2]	WE[1]	WE[0]
0 0 0 0	0	0	0	.	0	0	0
0 0 0 1	0	0	0	.	0	0	0
0 0 1 0	0	0	0	.	0	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
1 1 1 0	0	0	0	.	0	0	0
1 1 1 1	0	0	0	.	0	0	0

**Fig. 9.99** The address decoder for the  $32 \times 16$  memory in Fig. 9.96 when WE = 0

Address	RE[15]	RE[14]	RE[13]		RE[2]	RE[1]	RE[0]
0 0 0 0	0	0	0	.	0	0	1
0 0 0 1	0	0	0	.	0	1	0
0 0 1 0	0	0	0	.	1	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
1 1 1 0	0	1	0	.	0	0	0
1 1 1 1	1	0	0	.	0	0	0

**Fig. 9.100** The address decoder for the  $32 \times 16$  memory in Fig. 9.96 when RE = 1

Address	RE[15]	RE[14]	RE[13]		RE[2]	RE[1]	RE[0]
0 0 0 0	0	0	0	.....	0	0	0
0 0 0 1	0	0	0	.....	0	0	0
0 0 1 0	0	0	0	.....	0	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
1 1 1 0	0	0	0	.....	0	0	0
1 1 1 1	0	0	0	.....	0	0	0

**Fig. 9.101** The address decoder for the  $32 \times 16$  memory in Fig. 9.96 when RE = 0

## 9.22 A Design Example

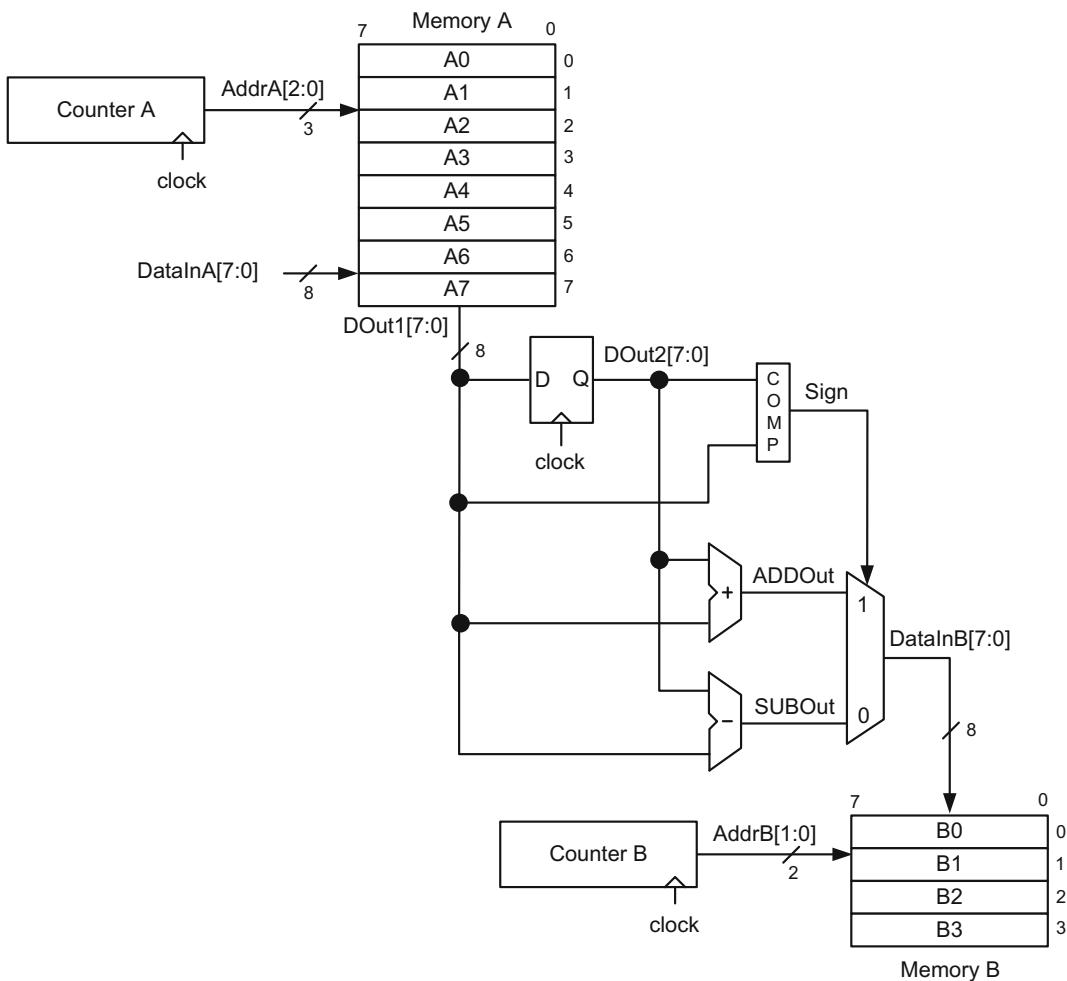
This design example combines the data-path and controller design concepts described earlier in this chapter. It also introduces the use of important sequential logic blocks, such as flip-flop, register, counter and memory, into the design.

Every design starts with gathering the small or large logic blocks to construct a data-path and setting up a proper data-flow according to the design specifications. Once the data-path is set, then the precise data movements from one logic block to the next is shown using a timing diagram. Any change in the data-path should be included in the timing diagram or vice versa.

When the data-path design and its timing diagram are complete and fully associate with each other, the next step in the design process is to build the controller circuit that governs the data-flow. To define the states of the controller, the clock periods that generate different sets of controller outputs are separated from each other and named as distinct states. Similarly, the clock periods with identical controller outputs are combined under the same state. The controller design can be Moore-type or Mealy-type according to the design needs.

The example design in this section reads two eight-bit data packets from an  $8 \times 8$  source memory (memory A), processes them and stores the result in an  $8 \times 4$  target memory (memory B). The processing part depends on the relative contents of each data packet: if the contents of the first data packet are larger than the second, the contents of the data packets are added. Otherwise, they are subtracted from each other before the result is stored.

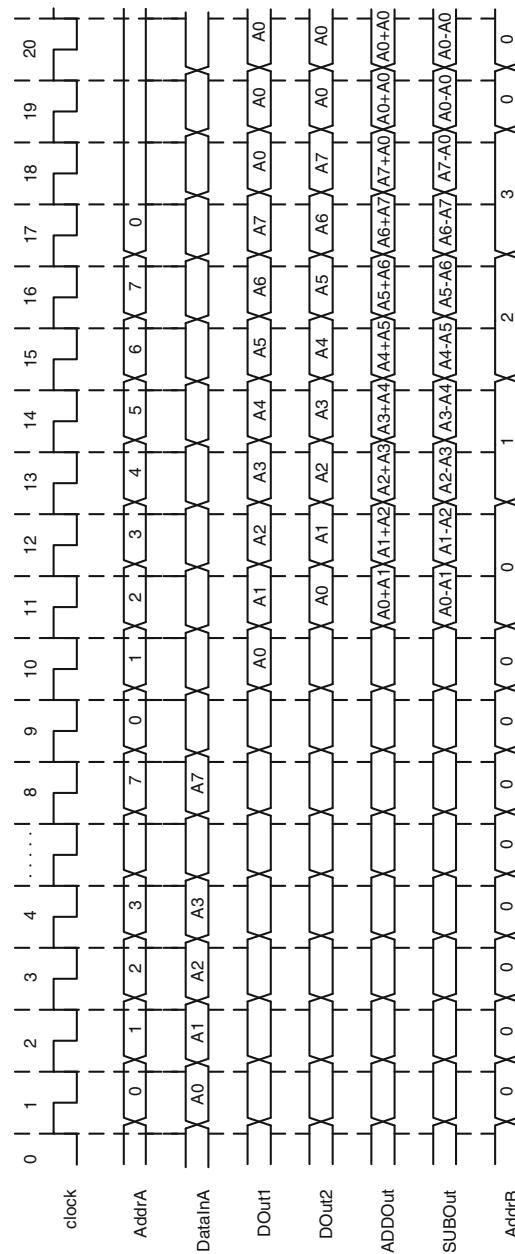
The block diagram in Fig. 9.102 demonstrates the data-path required for this memory-to-memory data transfer as described above. The timing diagram in Fig. 9.103



**Fig. 9.102** Data-path of a memory-memory data transfer

needs to accompany the data-flow in Fig. 9.102 since it depicts precise data movements at each clock cycle.

Initially, counter A generates the addresses 0 to 7 for the memory A and writes the data packets A0 to A7 through its DataInA[7:0] port. This is shown in the timing diagram in Fig. 9.103 from clock cycles 1 through 8. When this task is complete, counter A resets and reads the first data packet A0 from AddrA[2:0] = 0 in clock cycle 9. In the next clock cycle, A0 becomes available at DOut1[7:0], and the counter A increments by one. In cycle 11, AddrA[2:0] becomes 2, the data packet A1 is read from DOut1[7:0], and the data packet A0 transfers to DOut2[7:0]. In this cycle, the contents of the data packets A0 and A1 are compared with each other by subtracting A1 (at DOut1) from A0 (at DOut2). If the contents of A0 are less than A1, then the sign bit, Sign, of  $(A_0 - A_1)$  becomes negative. Sign = 1 selects  $(A_0 + A_1)$  at ADDOut[7:0] and routes this value to DataInB[7:0]. However, if the

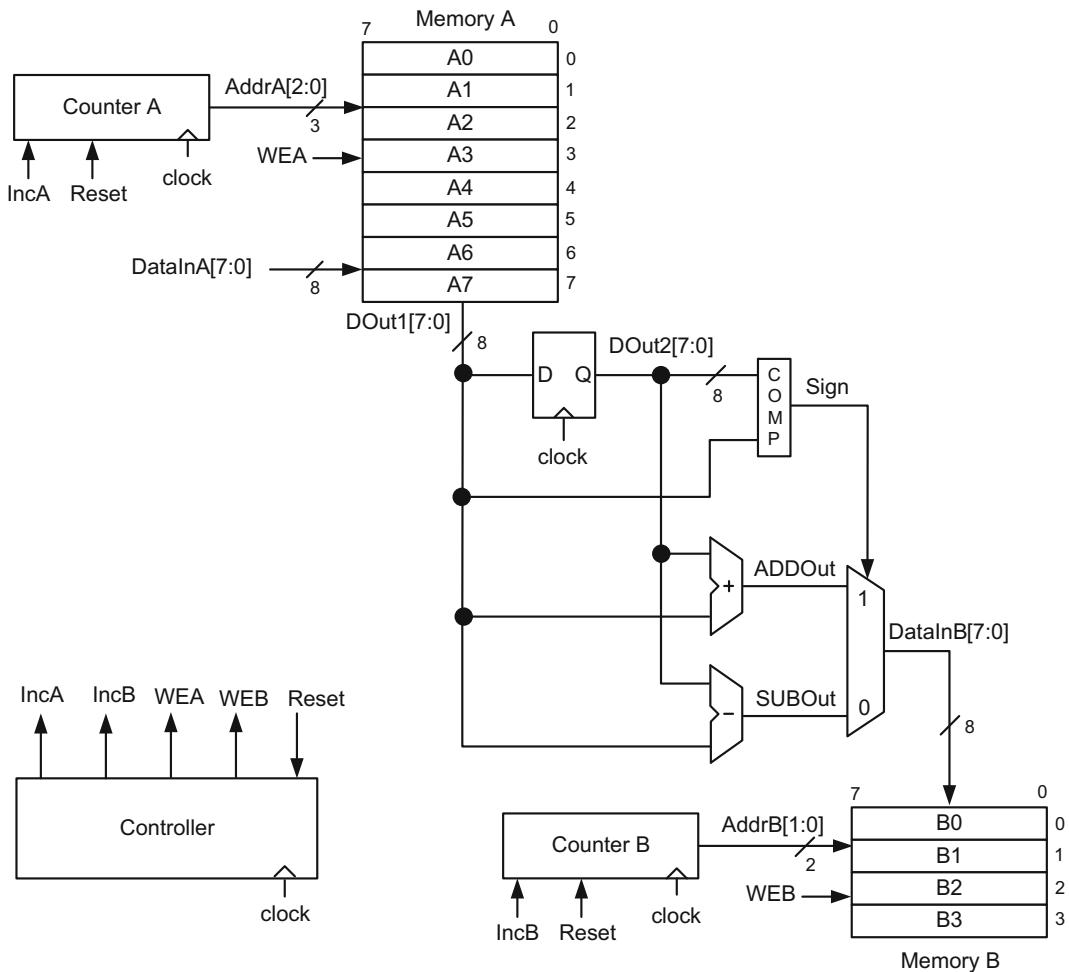


**Fig. 9.103** Timing diagram of the memory-memor data transfer in Fig. 9.102

contents of A0 are greater than A1,  $(A0 - A1)$  becomes positive. Sign = 0 selects  $(A0 - A1)$  and routes this value from SUBOut[7:0] to DataInB[7:0]. The result at DataInB[7:0] is written at AddrB[1:0] = 0 of memory B at the positive edge of clock cycle 12. In the same cycle, A1 is transferred to DOut2[7:0], and A2 becomes available at DOut1[7:0]. A comparison between A1 and A2 takes place, and either  $(A1 + A2)$  or  $(A1 - A2)$  is written to memory B depending on the value of the Sign node. However, this is an unwarranted step in the data transfer process because the design requirement states that the comparison has to be done only once between data packets from memory A. Since A1 is used in an earlier comparison with A0, A1 cannot be used in a subsequent comparison with A2, and neither  $(A1 + A2)$  nor  $(A1 - A2)$  should be written to memory B. The remaining clock cycles from 13 through 18 compare the values of A2 with A3, A4 with A5, and A6 with A7, and write the added or subtracted results to memory B. After clock cycle 19, all operations on this data-path suspend, the counters are reset, and all writes to the memory core are disabled.

To govern the data-flow in Fig. 9.103, a Moore-type state machine (or a counter-decoder-type controller) is used. A Mealy-type state machine for a controller design is usually avoided because the present state inputs to this type of state machine may change during the clock period and result in jittery outputs.

The inclusion of the controller identifies the control signals that govern the data-flow in Fig. 9.104. These signals increment the counters A and B, and enable writes to memory A or B when necessary. The timing diagram in Fig. 9.103 is also expanded to include the control signals, IncA, IncB, WEA and WEB, as shown in Fig. 9.105. Therefore, it provides a complete picture of the data transfer process from memory A to memory B in contrast to the earlier timing diagram in Fig. 9.103.

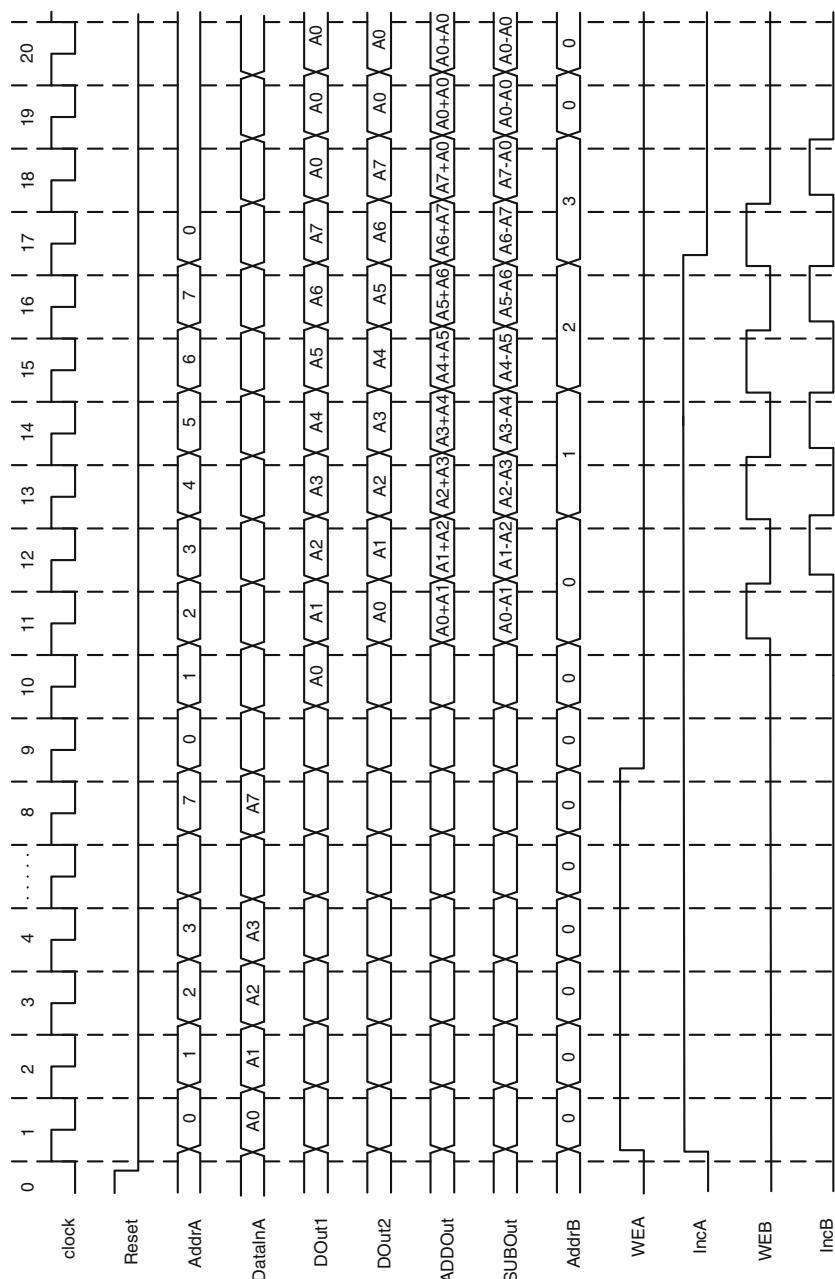


**Fig. 9.104** Complete block diagram of the memory-memory data transfer with the controller

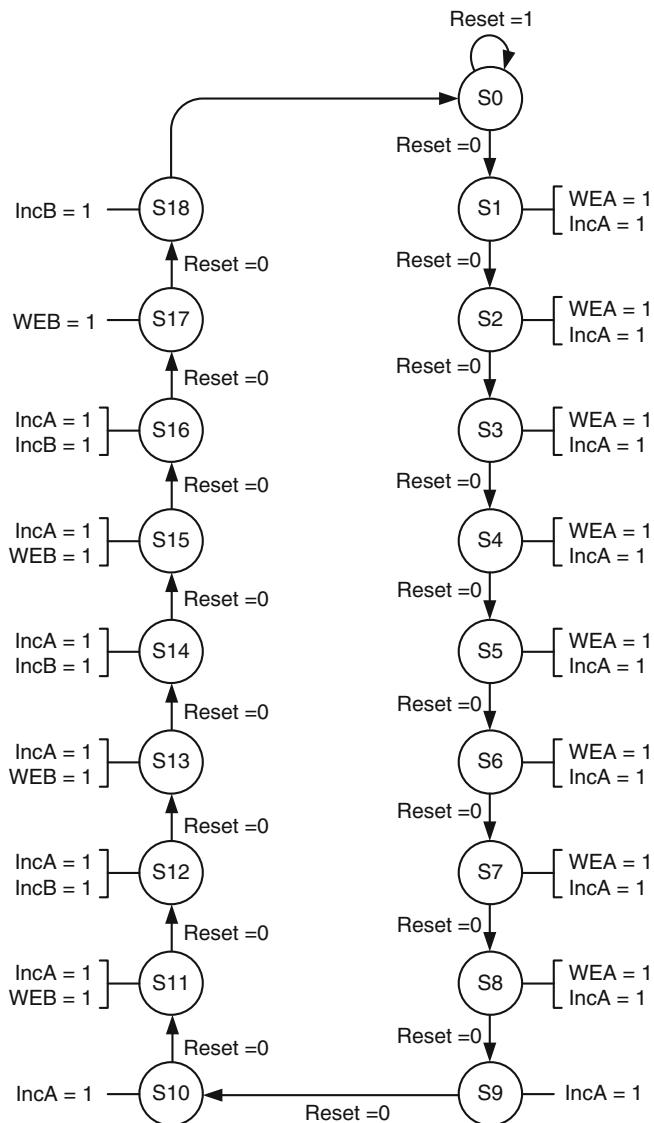
The controller in Fig. 9.104 is implemented by a Moore-type state machine in Fig. 9.106 and counter-decoder-type design in Fig. 9.107.

In the Moore type design, the states from S1 through S18 are assigned to each clock cycle of the timing diagram in Fig. 9.105. Subsequently, the values of the present state outputs, WEA, IncA, WEB and IncB, at each clock cycle are read off from the timing diagram and attached to each state in Fig. 9.106. The reset state, S0, is included in the Moore machine in case the data-path receives an external reset signal to interrupt an ongoing data transfer process. Whichever state the machine may be, Reset = 1 always transitions the state of the machine to S0 state. These transitions are not included in Fig. 9.106 for simplicity.

The counter-decoder style design in Fig. 9.107 consists of a five-bit counter and four decoders to generate WEA, IncA, WEB and IncB control signals. To show the operation of

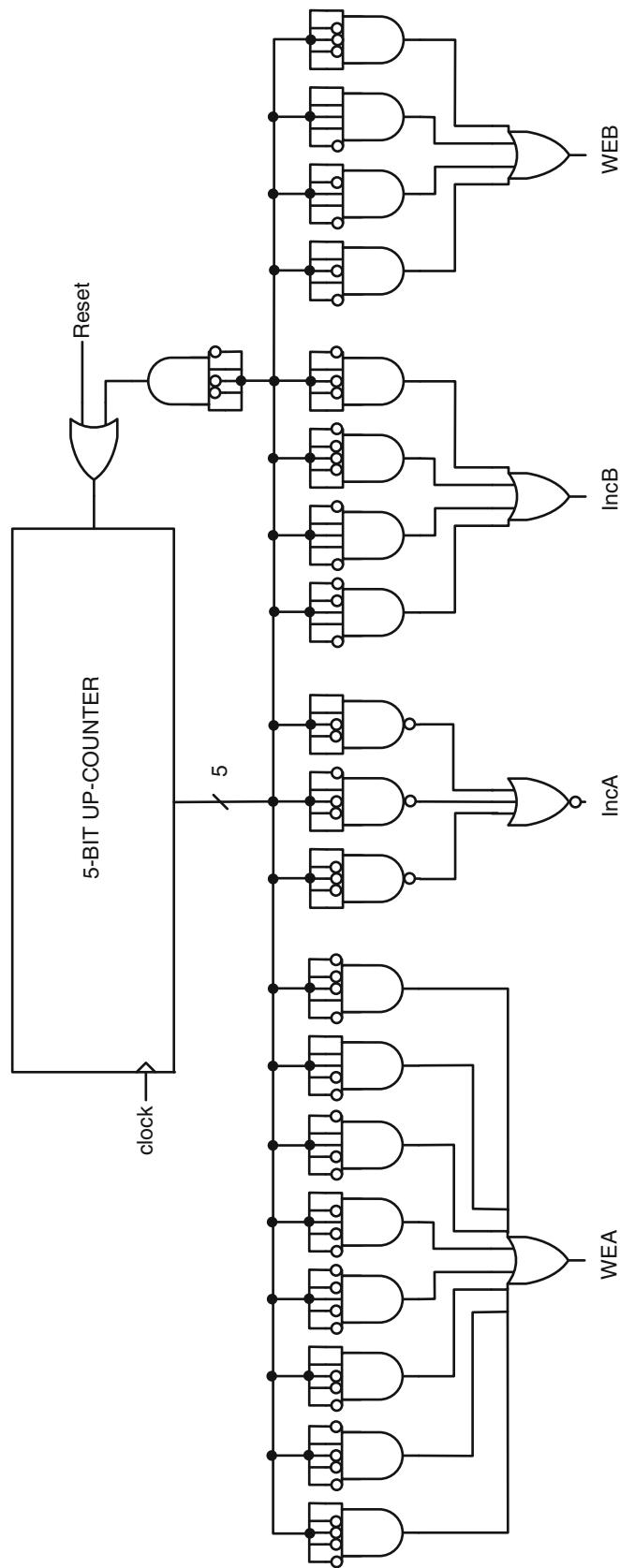


**Fig. 9.105** Complete timing diagram of the memory-memor transfer in Fig. 9.104



**Fig. 9.106** Moore-type state diagram of the controller unit in Fig. 9.104

this design to generate WEA, for example, this particular decoder includes eight five-input AND gates, one for each clock cycle from cycle number one to cycle number eight in order to keep  $WEA = 1$  in Fig. 9.105. The five-bit counter implicitly receives a reset signal from its output when it reaches clock cycle 18 and resets counter A, counter B and the rest of the system in Fig. 9.104.



**Fig. 9.107** Counter-decoder schematic of the controller unit in Fig. 9.104

## Review Questions

**1.** Implement the following gates:

- (a) Implement a two-input XOR gate using two-input NAND gates and inverters.
- (b) Implement a two-input AND gate using two-input XNOR gates and inverters.

**2.** Simplify the equation below:

$$\text{out} = (\overline{A + B}) \cdot (\overline{\overline{A} + \overline{B}})$$

**3.** Simplify the equation below:

$$\text{out} = (\overline{A} + C) \cdot (\overline{A} + \overline{C}) \cdot (\overline{A + B + \overline{C}})$$

**4.** Obtain the SOP and POS expressions for the following function:

$$\text{out} = (A \cdot B + C) \cdot (B + A \cdot C)$$

**5.** Implement the following function using NAND gates and inverters:

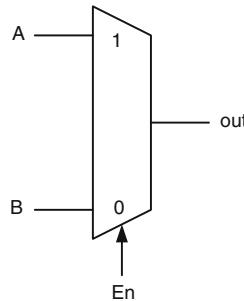
$$\text{out} = A \cdot \overline{C} + B \cdot C + \overline{A} \cdot \overline{B} \cdot D$$

**6.** Implement the following function using NOR gates and inverters:

$$\text{out} = (A \oplus B) \cdot (\overline{C \oplus D})$$

**7.** Implement the following 2-1 multiplexer using AND and OR gates:

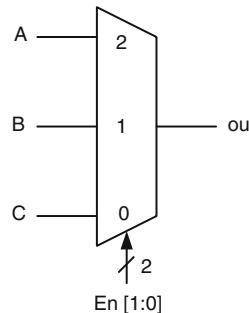
Note that the function of this complex gate must produce the following:  
If  $\text{En} = 1$  then  $\text{out} = A$  else (when  $\text{En} = 0$ )  $\text{out} = B$ .



- 8.** Implement the following 3-1 multiplexer using AND and OR gates:

Note that the function of this complex gate must produce the following:

If  $En = 2$  then  $out = A$ ; if  $En = 1$  then  $out = B$  else (when  $En = 0$  or  $En = 3$ )  $out = C$ .



- 9.** Implement a two-bit ripple-carry adder with inputs  $A[1:0]$  and  $B[1:0]$  and an output  $C[1:0]$  using one-bit half- and full-adders. Preserve the overflow bit at the output as  $C[2]$ .
- 10.** Implement a two-bit ripple-carry subtractor with inputs  $A[1:0]$  and  $B[1:0]$  and an output  $C[1:0]$  using one-bit half and full-adders. Preserve the overflow bit at the output as  $C[2]$ .
- 11.** Implement a two-bit multiplier with inputs  $A[1:0]$  and  $B[1:0]$  and an output  $C[3:0]$  using one-bit half and full-adders.
- 12.** Construct a four-bit comparator with inputs  $A[3:0]$  and  $B[3:0]$  using a subtractor. The comparator circuit should identify the following cases as active-high outputs:  
 $A[3:0] = B[3:0]$   
 $A[3:0] > B[3:0]$   
 $A[3:0] < B[3:0]$

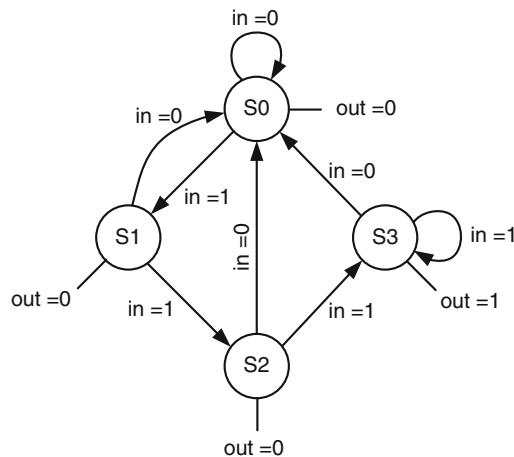
- 13.** Implement a two-bit decoder that produces four outputs.

When enabled the decoder generates the following outputs:

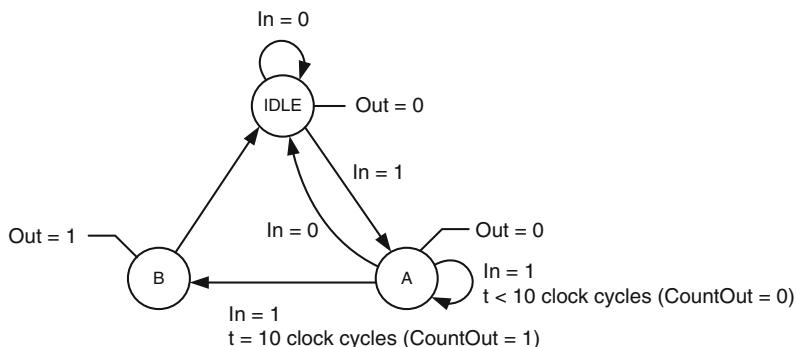
$$\begin{aligned} \text{in}[1:0] = 0 \text{ then } \text{out}[3:0] &= 1 \\ \text{in}[1:0] = 1 \text{ then } \text{out}[3:0] &= 2 \\ \text{in}[1:0] = 2 \text{ then } \text{out}[3:0] &= 4 \\ \text{in}[1:0] = 3 \text{ then } \text{out}[3:0] &= 8 \end{aligned}$$

When disabled the  $\text{out}[3:0]$  always equals to zero regardless of the input value.

14. Design a 64-bit adder in ripple-carry form and compare it against carry-look-ahead, carry-select, and carry-look-ahead/carry-select hybrids in terms of speed and the number of gates which define the circuit area. Divide the 64-bit carry-look-ahead/carry-select hybrid into four-bit, eight-bit, 16-bit and 32-bit carry-look-ahead segments. Indicate which carry-look-ahead/carry-select hybrid produces the optimum design.
15. Implement the following Moore machine:



16. Implement the following Moore machine using a timer. The timer is initiated when In = 1. The state machine goes to the A-state and stays there for 10 cycles. In the tenth cycle, the state machine transitions to the B-state and stays in this state for only one cycle before switching to the IDLE-state. One implementation scheme is to construct a four-bit up-counter to generate the timer. When the counter output reaches 9, the decoder at the output of the counter informs the state machine to switch from the A-state to the B-state.



17. The following decoder needs to be implemented using only two-input NAND gates and inverters.

A	B	C	Out
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	3
1	0	1	2
1	1	0	1
1	1	1	0

$T_{NAND} = 500 \text{ ps}$  (two-input NAND propagation delay)

$T_{INV} = 100 \text{ ps}$  (inverter propagation delay)

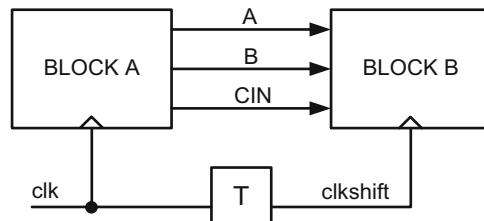
$t_{CLKQ} = 200 \text{ ps}$  (clock-q delay)

$t_{SU} = 200 \text{ ps}$  (set-up time)

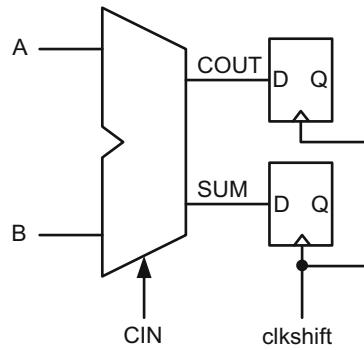
$t_H = 300 \text{ ps}$  (hold time)

- (a) Implement this decoder between two flip-flop boundaries.
- (b) Find the maximum clock frequency using a timing diagram.
- (c) Now the clock is shifted by 500 ps at the receiving flip-flop boundary. Show whether or not you have any hold violation using a timing diagram.

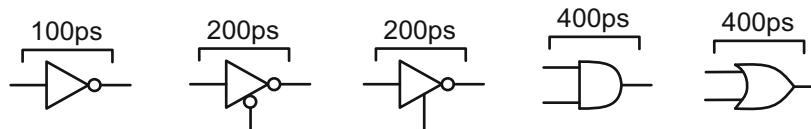
18. The block diagram is given below:



Block B contains one-bit adder with  $SUM = A \oplus B \oplus CIN$  and  $COUT = A \cdot B + CIN \cdot (A + B)$ , and two flip-flops as shown below:

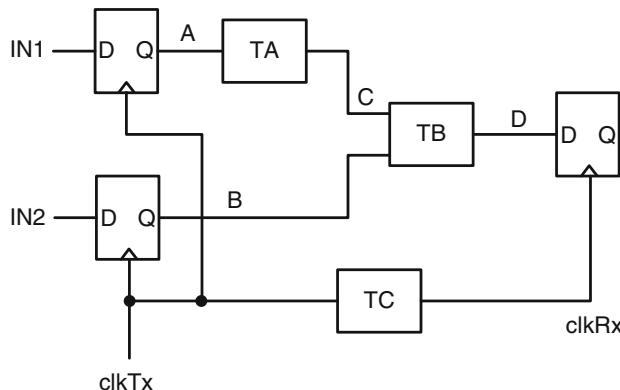


- (a) Using the gates with propagation delays below, determine the setup time for the A, B, and CIN inputs with respect to clkshift.



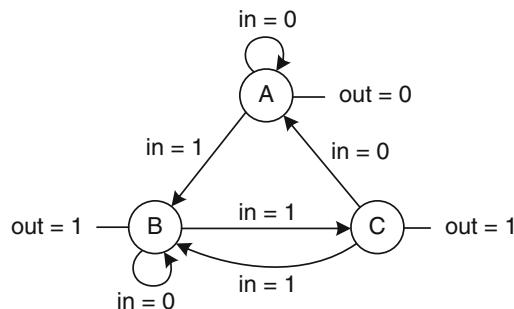
- (b) Assuming  $T = 0$  ns and  $T_{CLK}$  (clock period) = 5 ns, if data at A, B and CIN become valid and stable 4 ns after the positive edge of clkshift, will there be any timing violations? Assume  $t_H$  (hold time) = 3 ns for the flip-flop.  
(c) How can you eliminate the timing violations? Show your calculations and draw a timing diagram with no timing violations.

**19.** A schematic is given below:



- (a) Assume  $t_{SU}$  (setup time) = 200 ps,  $t_H$  (hold time) = 200 ps,  $t_{CLK-Q}$  (clock-to-q delay) = 300 ps for the flip-flop, and  $T_A$  = 1000 ps,  $T_B$  = 100 ps for the internal logic blocks on the schematic. Show if there is any timing violation or timing slack by drawing a detailed timing diagram when  $T_C$  = 0 ps.  
(b) What happens if  $T_C$  = 400 ps? Show it in a separate timing diagram.

- 20.** The state diagram of a Moore machine is given below:

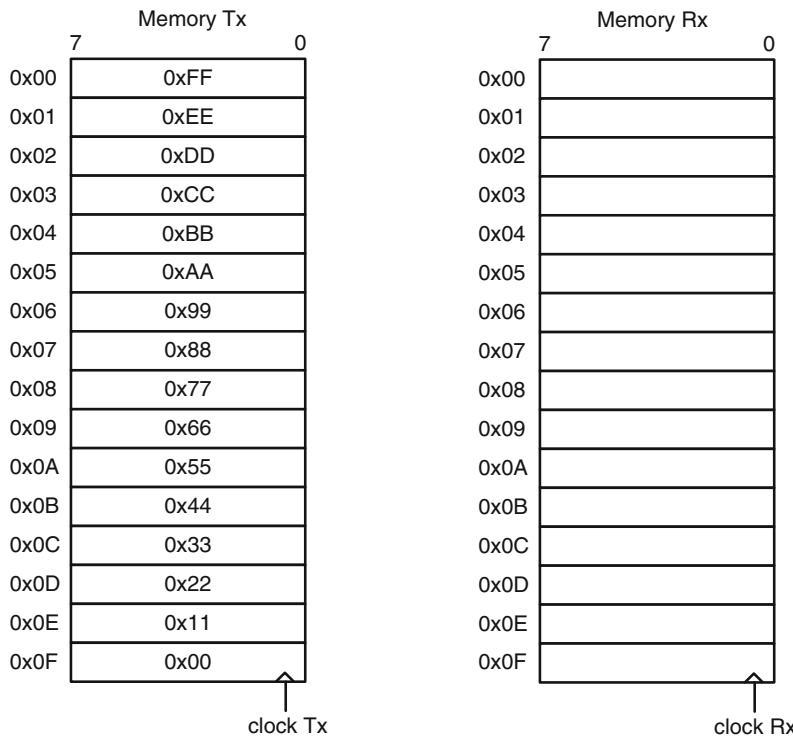


The assignment of the states A, B and C are indicated as follows:

states	PS[1]	PS[0]
A	0	0
B	0	1
C	1	1

- (a) Implement this state machine using inverters, two-input and three-input AND gates and two-input OR gates.
- (b) Find the maximum operating frequency of the implementation in part (a) if the following timing assignments are applied:  
 $t_{SU}$  (setup time) = 100 ps,  $t_H$  (hold time) = 100 ps,  $t_{CLK-Q}$  (clock-to-q delay) = 200 ps,  $T_{INV}$  = 200 ps,  $T_{AND2}$  = 300 ps,  $T_{AND3}$  = 400 ps,  $T_{OR2}$  = 400 ps.  
 Here,  $T_{INV}$ ,  $T_{AND2}$ ,  $T_{AND3}$ ,  $T_{OR2}$  correspond to the inverter, two-input and three-input AND gates and two-input OR gate, respectively.

- 21.** Data is transferred from Memory Tx to Memory Rx starting from the address  $0 \times 00$  and ending at the address  $0 \times 0F$  as shown below. Once a valid address is produced for Memory Tx, the data becomes available at the next positive clock edge. On the other hand, data is written to the Memory Rx at the positive edge of the clock when a valid address is produced. The operating clock frequency of Memory Tx is twice as high as that of Memory Rx.



- (a) Assuming address generators for Memory Tx and Memory Rx start generating valid addresses at the same positive clock edge, show which data is actually stored in Memory Rx using a timing diagram. Indicate all the address and data values for Memory Tx and Memory Rx in the timing diagram.
- (b) Now, assume that the operating clock frequency of Memory Tx is four times as high as the clock frequency of Memory Rx and the write takes place at the negative edge of the clock for Memory Rx when a valid address is present. Redraw the timing diagram indicating all address and data values transferred from one memory to the next.
22. Serial data is transferred to program four eight-bit registers. The start of the transfer is indicated by a seven-bit sequence = {1010101} immediately followed by the address of the register (two bits) and the data (eight bits). The transfer stops after programming the last register. After this point, all other incoming bits at the serial input is ignored. Design this interface by developing a data-path and a timing diagram simultaneously. Implement the state diagram. Can this controller be implemented by a counter-decoder scheme?

# Index

## A

Acceptor, 35  
Accelerometer, 129, 131  
Active device interface (with a bipolar transistor), 104, 108, 113  
Active region, 44, 46, 48  
Adder, 201, 228, 231, 232, 234, 235, 238, 241, 257  
Address decoder, 276–279  
Analog microphone, 183  
Analog-to-Digital Converter (ADC), 157–160, 163, 165–167, 183, 185, 191  
AND gate, 202, 203, 205, 210–212, 222, 225, 231, 238, 239  
Array multiplier, 201, 244–246  
As atom, 34

## B

Barrel shifter, 201, 241–244  
Base current, 43–45, 90  
Base-emitter voltage, 45  
B atom, 34  
Bipolar transistor, 43, 46, 47, 49, 104, 108, 109, 112–115, 118–120  
Block diagram, 264, 269, 279, 283  
Boltzmann constant, 37  
Boolean algebra, 208  
Breakdown voltage, 189  
Buffer, 204, 207, 247, 276  
Buzzer, 177, 178

## C

Capacitance, 68, 76, 131  
Capacitor, 2–4, 10, 13, 14  
Carry-look-ahead adder, 201, 233, 235, 238–240  
Carry-select adder, 201, 229, 234, 235, 239  
Clock, 201, 247–257, 259–263, 273, 274, 279, 280, 282  
CMOS inverter, 62, 63  
CMOS NAND logic gate, 63, 64  
CMOS NOR logic gate, 65, 66  
CMOS-TTL interface, 118, 119  
Collector, 43–45  
Collector-base voltage, 45  
Collector current, 90, 91, 94

Collector-emitter voltage, 44

Combinational logic, 201, 214, 223, 225, 232, 249, 250,

252–255, 263, 267, 268

Complementary MOS (CMOS), 62

Complemented logic gate, 205

Complex CMOS logic gate implementation, 66

Complex plane, 20, 21, 23, 26

Conduction band (EC), 35, 123, 125, 127

Constant coefficient differential equation, 4, 15

Controller design, 279, 282

Counter, 194–197, 201, 262, 263, 272–275, 279, 280, 282

Counter-decoder design, 201, 272–275, 282, 283, 286

Crystalline structure, 33–35

Current gain, 45, 52

Current-voltage characteristics, 58, 60

Cut-off region, 44, 46

## D

Data-path, 202, 256, 279, 280, 282, 283

DC motor control, 178, 181

Decoder, 201, 225, 275, 283

De Morgan's theorem, 210

Denominator, 22, 24

D flip-flop, 250–253

Difference amplifier, 138, 142

Digital-to-Analog Converter (DAC), 162, 163, 166, 169, 170

Diode, 37–40, 42, 89–91, 94, 97, 99–101, 116, 117, 120

D latch, 247, 248, 250

Down-rounding, 163, 165, 167

Drain current (ID), 57, 59

Drain-source saturation voltage ( $V_{DSAT}$ ), 57, 59–61

Drain-source voltage ( $V_{DS}$ ), 57–60, 83

## E

Effective resistance, 198

Electromechanical, 175, 176

Electron, 33–36, 58, 123–125, 127, 129, 132

Electron mobility ( $\mu N$ ), 76

Embedded system, 123, 127, 132

Emitter, 43–45

Emitter current, 90, 91, 99, 100

Encoder, 201, 223, 224  
 Energy band bending, 127  
 Energy band gap (EG), 35  
 Equivalent circuit, 10, 13, 15, 16, 22, 24, 43, 124, 128  
 Equivalent resistor, 40, 52  
 Exclusive NOR gate, 65  
 Exclusive OR gate, 204  
 Exponential, 37, 44  
 Extrinsic semiconductor, 34, 35

**F**

Fall delay, 73, 74  
 Fall time, 72–74, 76  
 Fermi level ( $E_F$ ), 35, 36, 125  
 First-order passive circuits, 3  
 Flash-type ADC, 160–162, 165  
 Forward-bias, 51  
 Forward voltage, 36–38  
 4-1 multiplexer, 222  
 4-1 MUX, 222, 223  
 Free electron, 34, 35  
 Frequency domain analysis, 1, 19, 24  
 Front-end electronics, 183, 185, 189, 194, 196, 201  
 Full adder, 229, 234, 246  
 Full-wave rectifier, 40, 41

**G**

Galvanometric voltage drop, 123, 124  
 Gate, 57, 59, 62, 63, 65–68, 73, 74, 76–78, 201–208, 231, 247, 252  
 Gate oxide capacitance ( $C_{OX}$ ), 58, 60  
 Gate-source voltage ( $V_{GS}$ ), 62  
 General solution, 4, 7

**H**

Half adder, 228  
 Half-wave rectifier, 38, 39, 41  
 Hall-effect device, 132, 133, 182, 183, 196  
 Hall-effect sensor, 196  
 Hall-effect switch, 196  
 Hall-effect tachometer, 196  
 Hall voltage ( $V_H$ ), 132, 182  
 Hardware, 191  
 Hold time, 251, 254, 255  
 Hold violation, 255–258  
 Hole, 36, 60, 125, 127  
 Hole mobility ( $\mu_p$ ), 76

**I**

ID-VDS curve, 59  
 Inductance, 2, 14, 24, 28, 31, 198  
 Inductor, 2, 6, 14, 15  
 Initial condition, 8, 10, 11, 15, 16

Input current, 116, 119, 121  
 Input Current High (I<sub>IH</sub>), 103, 111  
 Input Current Low (I<sub>IL</sub>), 102, 111  
 Input current measurement, 102, 103  
 Input impedance( $Z_{IN}(s)$ ), 13, 135, 138  
 Input resistance ( $R_{IN}$ ), 153  
 Input voltage ( $V_{IN}(s)$ ), 13, 90, 91, 108, 111, 116–119  
 Input Voltage High (V<sub>IH</sub>), 103, 111  
 Input Voltage Low (V<sub>IL</sub>), 102, 111  
 Input voltage measurement, 103  
 Instability, 20  
 Intrinsic, 33, 35  
 Inverter, 205, 207, 208, 222, 247, 257  
 Inverting amplifier, 135–137, 142, 144  
 Isolation circuit, 138

**J**

Junction, 36, 37, 43–46

**K**

Karnaugh map, 201, 214  
 Kirchoff's voltage law, 3, 6, 10, 14  
 K-map, 214–219, 221, 224, 227, 266, 267, 270, 271, 274

**L**

Ladder-type DAC, 170, 171  
 Laplace transform, 1–4, 15, 22, 23  
 Leakage current, 37, 44  
 Light Emitting Diode (LED), 42, 176, 177, 183  
 Light level measurement, 190, 191  
 Linear shifter, 241, 242  
 Load line, 47–50  
 Logic gate, 201, 204, 205, 208, 211, 257

**M**

Magnetic field, 132, 133  
 Magnetic tachometer, 196  
 Mealy machine, 268  
 Memory, 201, 247, 249, 276–284  
 Minimization, 201, 214, 216, 217  
 Moore machine, 263–268, 270, 283  
 2-1 multiplexer, 220  
 Multiplier, 245  
 2-1 MUX, 220, 221, 234, 239, 241, 257, 259, 275

**N**

NAND gate, 205, 223, 257, 258  
 Natural frequency, 12–15, 19  
 N-channel MOSFET (NMOSFET), 57–59, 65  
 Next state, 263, 264, 266, 268  
 NMOS transistor, 58, 59, 62, 63, 65–67, 69, 75, 76  
 NMOS transistor threshold voltage ( $V_{TN}$ ), 58

- NMOS tree, 63, 65–67, 69  
Non-inverting amplifier, 137, 145  
NOR gate, 206  
NPN transistor, 43, 44, 48, 108, 109, 118, 120  
N-type semiconductor, 35, 36, 125, 127  
Numerator, 13
- O**  
One bit full adder, 226–229, 232, 257  
One-bit half adder, 228, 229  
Open-collector, 93–96  
Operational amplifier (Opamp), 135, 137, 142, 144–146, 150, 151  
Optoelectronic tachometer, 194–196  
Opto-isolator, 176–178, 181  
OR gate, 203, 210–212, 222, 223, 232, 239  
Oscillation, 18, 21  
Oscillatory, 20, 21, 27  
Output, 202–208, 211, 212, 214, 215, 217, 218, 220, 221, 223, 225–227, 229, 232, 234, 241–243, 246, 248, 250, 251, 257, 259, 261–264, 266, 268, 269, 272–276, 282, 283, 285  
Output current, 91, 108, 116, 119, 121  
Output Current High (IOH), 91–94, 97, 104, 111–113  
Output Current Low (IOL), 91–95, 103, 105, 112  
Output current measurement, 103  
Output impedance, 135, 151  
Output I-V characteristics, 37, 46, 48, 49  
Output voltage, 94, 101, 117, 118, 120  
Output Voltage High (VOH), 91, 104, 112  
Output Voltage Low (VOL), 94, 103  
Output voltage measurement, 103, 104
- P**  
Parallel plates, 131  
Particular solution, 4, 7  
Passive device interface (with a resistor), 104, 107, 116  
Passive elements, 12  
Passive network, 14, 19–21  
P-channel MOSFET (PMOSFET), 59–61  
Photo detector, 191, 192  
Photodiode, 125–128  
Photon, 125, 129  
Photo-resistor, 129  
Photo-transistor, 176, 177  
Photovoltaic voltage drop, 128  
Piezoelectric, 129, 130  
PMOS transistor, 60–65, 67–69, 76  
PMOS transistor threshold voltage ( $V_{TP}$ ), 60  
PMOS tree, 63–66, 69  
PN junction, 36, 37  
PNP transistor, 43  
Poles, 19–21, 25–27
- POS, 212, 214, 215, 217–219, 222, 229  
Power supply voltage ( $V_{CC}$ ), 47, 90, 91, 94, 104, 107, 111, 117  
Power supply voltage ( $V_{DD}$ ), 58, 61, 63  
Present state, 263, 264, 266, 268, 269, 272, 282, 283  
Product of sums, 212  
P-type semiconductor, 35, 36, 125–127  
Pull-down resistor, 116, 117  
Pull-up resistor, 94, 104–108  
Pulse Width Modulation (PWM) circuit, 178–181
- R**  
Ramp-type ADC, 162–165  
Read enable, 276  
Rectifying diode, 38  
Reference signal, 138  
Reference voltage, 158, 160  
Register, 201, 259–262, 276, 279  
Relay, 175, 176  
Resistor, 1–3, 6, 14  
Reverse-bias, 37, 44  
Reverse voltage, 36, 37  
Ripple-carry adder, 201, 229, 230, 234  
Rise delay, 73  
Rise time, 72
- S**  
Sample and hold, 157, 160, 163  
Sampling, 157–160  
Saturation, 90, 94, 95, 97, 98, 100, 101, 113, 114, 118  
Saturation region, 44, 45  
Schmitt trigger, 147–150  
2s complement addition, 240, 241  
Semiconductor, 33–35  
Sensor, 127, 135–145  
Sequential logic, 201, 279  
Servo control, 182  
Set-up slack, 258  
Set-up time, 248, 251, 252, 254, 255  
Set-up violation, 254–256  
Shifter, 241  
Shift register, 201, 261, 262  
Silicon, 33, 38  
Sinusoidal, 38–41  
Solar cell, 127, 128  
Source, 57, 59–61  
Square waveform generator, 150–152  
Stability, 19, 22–24  
State diagram, 263–265, 268, 269, 272–274  
State machine, 201, 263, 267, 268, 272–274, 282, 283  
State table, 264–266, 269, 270, 274  
Subtractor, 201, 241  
Successive approximation, 165, 166, 168

Successive approximation ADC, 165, 167  
 Summation amplifier, 141  
 Sum of products (SOP), 212, 214–216, 218–222, 224, 227, 266, 267, 270, 271

**T**  
 Temperature measurement, 186, 187  
 Tensile stress, 129, 130  
 Thermocouple, 123, 124, 185–187  
 Time constant, 13, 14, 62  
 Time-domain analysis, 1, 22, 23  
 Timing diagram, 201, 202, 249, 252, 254–256, 258–263, 273–275, 279–283  
 Timing methodology, 250, 253  
 Timing violations, 201, 254  
 Transfer function, 19, 24  
 Transistor length, 76  
 Transistor-Transistor Logic (TTL), 89, 90, 92, 102, 104–106, 108–113, 115, 116, 118–120  
 Transistor width, 75, 76  
 Transition table, 266, 269, 270, 274  
 Trans-resistance amplifier, 144, 145  
 Tri-state, 207, 208, 247, 251, 276  
 Truth table, 201–203, 205, 206, 211–217, 224, 225, 227, 276, 277  
 TTL-CMOS interface, 104, 107–109  
 TTL inverter, 89–96, 105, 107–109  
 TTL NAND logic gate, 97–99  
 TTL NOR logic gate, 99–102

**U**  
 Unity gain, 138, 145  
 Up-counter, 162, 163, 165  
 Up-rounding, 163, 165–167

**V**  
 Valence band ( $E_V$ ), 36, 125, 127  
 Voltage amplifier, 135, 137, 139, 142, 145  
 Voltage comparator, 146, 147

**W**  
 Waveform, 5, 6, 8, 9, 11, 12, 17–21, 27  
 Weighted sum DAC, 169  
 Work function, 123, 124  
 Worst-case charge path, 64, 72  
 Worst-case discharge path, 65, 72  
 Write enable, 259, 276

**X**  
 XNOR gate, 206, 207  
 XOR gate, 204, 206, 231, 232

**Z**  
 Zener diode, 42  
 Zeros, 4, 19, 22, 24