

Scilab Textbook Companion for
Analog And Digital Electronics
by U. A. Bakshi And A. P. Godse¹

Created by
Priyank Bangar
B.Tech
Electronics Engineering
NMIMS
College Teacher
No Teacher
Cross-Checked by

August 10, 2013

¹Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website <http://scilab.in>

Book Description

Title: Analog And Digital Electronics

Author: U. A. Bakshi And A. P. Godse

Publisher: Technical Publications, Pune

Edition: 1

Year: 2009

ISBN: 9788184316902

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

| | |
|--------------------------------|-----|
| List of Scilab Codes | 4 |
| 1 Special Diodes | 10 |
| 2 Frequency Response | 12 |
| 3 Feedback Amplifiers | 18 |
| 4 Oscillators | 41 |
| 5 Combinational Logic Circuits | 60 |
| 6 Sequential Logic Circuits | 84 |
| 7 Shift Registers | 86 |
| 8 Counters | 87 |
| 9 Op amp Applications | 119 |
| 10 Voltage Regulators | 138 |

List of Scilab Codes

| | | |
|----------|---|----|
| Exa 1.1 | current through LED | 10 |
| Exa 1.2 | transition capacitance and constant K | 10 |
| Exa 2.1 | maximum voltage gain | 12 |
| Exa 2.2 | gain of the amplifier | 12 |
| Exa 2.3 | gain of an amplifier | 13 |
| Exa 2.4 | low frequency response of the amplifier | 13 |
| Exa 2.5 | low frequency response of the FET amplifier | 14 |
| Exa 2.6 | high frequency response of the amplifier | 15 |
| Exa 2.7 | high frequency response of the amplifier | 16 |
| Exa 3.1 | gain fL and fH | 18 |
| Exa 3.2 | Vo and second harmonic distortion with feedback | 18 |
| Exa 3.3 | beta and Af | 19 |
| Exa 3.4 | beta and Av and Avf and Rif and Rof | 20 |
| Exa 3.5 | Avf and Rif and Rof | 22 |
| Exa 3.6 | Avf and Rif and Rof | 23 |
| Exa 3.7 | D and Avf and Rif and Rof | 25 |
| Exa 3.8 | voltage gain | 27 |
| Exa 3.9 | Avf | 28 |
| Exa 3.10 | Avf and Rif and Rof | 29 |
| Exa 3.11 | voltage gain and input and output resistance | 31 |
| Exa 3.12 | Ai and beta and Aif | 32 |
| Exa 3.13 | beta and Av and Avf and Rif and Rof | 34 |
| Exa 3.14 | gain and new bandwidth | 36 |
| Exa 3.15 | show voltage gain with feedback | 36 |
| Exa 3.16 | GMf and Rif and Rof | 37 |
| Exa 3.17 | feedback factor and Rif and Rof | 38 |
| Exa 3.18 | gain with feedback and new bandwidth | 39 |
| Exa 3.19 | overall voltage gain and bandwidth | 40 |

| | | |
|----------|---|----|
| Exa 4.1 | C and hfe | 41 |
| Exa 4.2 | frequency of oscillation | 42 |
| Exa 4.3 | R and C | 42 |
| Exa 4.4 | C and RD | 42 |
| Exa 4.5 | minimum and maximum R2 | 43 |
| Exa 4.6 | range over capacitor is varied | 44 |
| Exa 4.7 | frequency of oscillation | 44 |
| Exa 4.8 | frequency of oscillation | 45 |
| Exa 4.9 | calculate C | 45 |
| Exa 4.10 | series and parallel resonant frequency | 46 |
| Exa 4.11 | series and parallel resonant frequency | 46 |
| Exa 4.12 | Vary Barkhausen criterion and find frequency of oscillation | 47 |
| Exa 4.13 | minimum and maximum values of R2 | 49 |
| Exa 4.14 | frequency of oscillation and minimum hfe | 49 |
| Exa 4.15 | range of frequency of oscillation | 50 |
| Exa 4.16 | frequency of oscillation | 50 |
| Exa 4.17 | gain of the transistor | 51 |
| Exa 4.18 | new frequency and inductance | 51 |
| Exa 4.19 | new frequency of oscillation | 52 |
| Exa 4.20 | R and hfe | 52 |
| Exa 4.21 | component values of wien bridge | 53 |
| Exa 4.22 | values of C2 and new frequency of oscillation | 54 |
| Exa 4.23 | series and parallel resonant frequency and Qfactor | 55 |
| Exa 4.24 | change in frequency and trimmer capacitance | 55 |
| Exa 4.25 | design RC phase shift oscillator | 57 |
| Exa 4.26 | range of capacitor | 58 |
| Exa 4.27 | change in frequency of oscillation | 58 |
| Exa 5.1 | design a combinational logic circuit | 60 |
| Exa 5.2 | design a circuit with control line C and data lines | 61 |
| Exa 5.3 | design combinational circuit | 62 |
| Exa 5.4 | design logic circuit | 63 |
| Exa 5.5 | design circuit to detect invalid BCD number | 64 |
| Exa 5.6 | design two level combinational circuit | 65 |
| Exa 5.7 | design 32 to 1 multiplexer | 66 |
| Exa 5.8 | design a 32 to 1 multiplexer | 66 |
| Exa 5.9 | implement boolean function using 8to1 multiplexer | 67 |
| Exa 5.10 | implement boolean function using 4to1 multiplexer | 67 |

| | | |
|----------|---|----|
| Exa 5.11 | implement boolean function using 8to1 MUX | 69 |
| Exa 5.12 | implement boolean function using 4to1 MUX | 69 |
| Exa 5.13 | implement boolean function using 8to1 MUX | 70 |
| Exa 5.14 | implement boolean function using 8to1 MUX | 71 |
| Exa 5.15 | implement boolean function using 8to1 MUX | 72 |
| Exa 5.16 | determine boolean expression | 72 |
| Exa 5.17 | realize using 4 to 1 MUX | 73 |
| Exa 5.18 | design 1 to 8 DEMUX | 74 |
| Exa 5.19 | implement full subtractor | 74 |
| Exa 5.20 | construst 1 to 32 DEMUX | 75 |
| Exa 5.21 | design 3 to 8 decoder | 76 |
| Exa 5.22 | design 5 to 32 decoder | 77 |
| Exa 5.23 | design 4 line to 16 line decoder | 78 |
| Exa 5.24 | implement using 74LS138 | 78 |
| Exa 5.25 | implement full subtractor | 79 |
| Exa 5.26 | implement gray to binary code converter | 79 |
| Exa 5.27 | design 2 bit comparator | 80 |
| Exa 5.28 | design full adder circuit | 81 |
| Exa 5.29 | implement BCD to 7 segment decoder | 82 |
| Exa 5.31 | implement 32 input to 5 output encoder | 82 |
| Exa 6.4 | analyze the circuit | 84 |
| Exa 7.1 | determine number of flip flops needed | 86 |
| Exa 8.1 | count after 12 pulse | 87 |
| Exa 8.2 | count in binary | 87 |
| Exa 8.4 | draw logic diagram | 88 |
| Exa 8.5 | output frequency | 88 |
| Exa 8.6 | maximum operating frequency | 88 |
| Exa 8.8 | design 4 bit up down ripple counter | 89 |
| Exa 8.9 | design divide by 9 counter | 89 |
| Exa 8.10 | design a divide by 128 counter | 90 |
| Exa 8.11 | design divide by 78 counter | 90 |
| Exa 8.12 | design divide by 6 counter | 91 |
| Exa 8.13 | find fmax | 91 |
| Exa 8.14 | determine states | 92 |
| Exa 8.15 | design MOD 10 counter | 92 |
| Exa 8.16 | design down counter | 93 |
| Exa 8.17 | design programmable frequency divider | 94 |
| Exa 8.18 | design a counter | 94 |

| | | |
|----------|--|-----|
| Exa 8.19 | programmable frequency divider | 96 |
| Exa 8.20 | design divide by 2 and divide by 5 counter | 96 |
| Exa 8.21 | design mod 9 counter | 98 |
| Exa 8.22 | determine MOD number and counter range | 98 |
| Exa 8.23 | design a divide by 20 counter | 98 |
| Exa 8.24 | design divide by 96 counter | 99 |
| Exa 8.25 | design divide by 93 counter | 100 |
| Exa 8.26 | design divide by 78 counter | 100 |
| Exa 8.28 | 7490 IC | 100 |
| Exa 8.30 | sketch output waveforms of counter | 101 |
| Exa 8.31 | explain operation of circuit | 104 |
| Exa 8.32 | design divide by 40000 counter | 105 |
| Exa 8.33 | design modulo 11 counter | 106 |
| Exa 8.34 | design excess 3 decimal counter | 106 |
| Exa 8.35 | modulus greater than 16 | 107 |
| Exa 8.36 | modulo 8 counter | 108 |
| Exa 8.37 | synchronous decade counter | 108 |
| Exa 8.38 | flip flops | 110 |
| Exa 8.39 | design mod 5 synchronous counter | 112 |
| Exa 8.40 | design MOD 4 down counter | 114 |
| Exa 8.41 | design MOD 12 synchronous counter | 115 |
| Exa 8.42 | design 4 bit 4 state ring counter | 117 |
| Exa 8.44 | design 4 bit 8 state johnson counter | 117 |
| Exa 8.45 | johnson counter | 118 |
| Exa 9.1 | output voltage | 119 |
| Exa 9.3 | input bias current | 120 |
| Exa 9.4 | design inverting schmitt trigger | 120 |
| Exa 9.5 | threshold voltage | 121 |
| Exa 9.6 | tripping voltage | 121 |
| Exa 9.8 | time duration | 122 |
| Exa 9.9 | calculate R1 and R2 | 123 |
| Exa 9.10 | calculate trip point and hysteresis | 123 |
| Exa 9.11 | design schmitt trigger | 124 |
| Exa 9.12 | design op amp schmitt trigger | 125 |
| Exa 9.13 | VUT and VLT and frequency of oscillation | 126 |
| Exa 9.17 | design op amp circuit | 127 |
| Exa 9.18 | design monostable using IC 555 | 127 |
| Exa 9.19 | design a timer | 128 |

| | | |
|----------|--|-----|
| Exa 9.20 | draw timer using IC 555 | 129 |
| Exa 9.21 | frequency of output and duty cycle | 129 |
| Exa 9.26 | design astable multivibrator | 130 |
| Exa 9.27 | design astable multivibrator | 131 |
| Exa 9.28 | design astable mode to generate square wave | 132 |
| Exa 9.32 | define resolution | 133 |
| Exa 9.33 | output voltage | 133 |
| Exa 9.34 | find VoFS | 134 |
| Exa 9.35 | find out stepsize and analog output | 134 |
| Exa 9.36 | find output voltage | 134 |
| Exa 9.38 | find resolution and digital output | 135 |
| Exa 9.39 | calculate quantizing error | 136 |
| Exa 9.40 | dual scope ADC | 136 |
| Exa 9.41 | find digital output of ADC | 136 |
| Exa 9.42 | find conversion time | 137 |
| Exa 9.43 | find maximum frequency of input sine wave | 137 |
| Exa 10.1 | find line and load regulation and ripple refection | 138 |
| Exa 10.2 | design op amp series voltage regulator | 139 |
| Exa 10.4 | calculate output voltage | 139 |
| Exa 10.7 | determine regulated output voltage | 140 |
| Exa 10.8 | find the range | 140 |

List of Figures

| | |
|--|-----|
| 8.1 sketch output waveforms of counter | 104 |
|--|-----|

Chapter 1

Special Diodes

Scilab code Exa 1.1 current through LED

```
1 //Example 1.1
2 clc
3 format(5)
4 disp("Assume the drop across the LED as 2 V.")
5 disp("Therefore ,      VD = 2 V")
6 disp("From fig.1.11,  RS = 2.2 k-ohm and VS = 15 V")
7 is=(15-2)/(2.2) // in mA
8 disp(is,"Therefore ,      IS(mA) = VS-VD / RS =")
```

Scilab code Exa 1.2 transition capacitance and constant K

```
1 //Example 1.2
2 clc
3 disp("The transistor capacitance is given by,")
4 disp("  CT = C(0) / [1+|VR/VJ|^n]")
5 disp("Now  C(0) = 80pF, n = 1/3 as diffused junction
      ")
6 disp("      VR = 4.2 V,  VJ = 0.7 V")
```

```
7 ct=((80*10^-12)/((1+(4.2/0.7))^(1/3)))*10^12 // in
  pF
8 format(6)
9 disp(ct,"Therefore , CT(pF) = ")
10 disp("the transistor capacitance is also given by,")
11 disp(" CT = K / [VR+VJ]^n")
12 format(10)
13 k=(41.82*10^-12)*((4.2+0.7)^(1/3))
14 disp(k,"Therefore , K = ")
```

Chapter 2

Frequency Response

Scilab code Exa 2.1 maximum voltage gain

```
1 //Example 2.1
2 clc
3 format(7)
4 disp("We know that maximam voltage gain of voltage
      amplifier is given as")
5 mv=200*sqrt(2)
6 disp(mv,"Therefore , Maximum voltage gain = Gain at
      cut-off x sqrt(2) =")
```

Scilab code Exa 2.2 gain of the amplifier

```
1 //Example 2.2
2 clc
3 format(6)
4 disp("We know that ,")
5 a=100/sqrt(1+((1000/20)^2))
6 disp(a,"Below midband :           A = A_mid / sqrt(1+(f1
      /f)^2) =")
```

Scilab code Exa 2.3 gain of an amplifier

```
1 //Example 2.3
2 clc
3 format(7)
4 a=200*sqrt(2)
5 disp(a,"We know that      A_mid = 3dB gain x sqrt(2)
      =")
6 am=282.84/(sqrt(1+(((10/2)^2))))
7 format(6)
8 disp(am,"Above midband :      A = A_mid / sqrt(1+(
      f1/f)^2) =") // answer in textbook is wrong
```

Scilab code Exa 2.4 low frequency response of the amplifier

```
1 //Example 2.4
2 clc
3 format(6)
4 disp("It is necessary to analyze each network to
      determine the critical frequency of the amplifier
      ")
5 disp("(a) Input RC network")
6 fc1=1/(2*pi*[680+1031.7]*(0.1*10^-6))
7 disp(fc1,"      f_c(input)(in Hz) = 1 / 2*pi*[RS+(
      R1||R2||hie)]C1 =") // in Hz
8 disp("(b) Output RC network")
9 format(7)
10 fc2=1/(2*pi*((2.2+10)*10^3)*(0.1*10^-6))
11 disp(fc2,"      f_c(output)(in Hz) = 1 / 2*pi*(RC+
      RL)*C2 =") // in Hz
12 disp("(c) Bypass RC network")
```

```

13 rth=((68*22*0.680)/((22*0.680)+(68*0.680)+(68*22)))
    *10^3
14 disp(rth,"R_th(in ohm) = R1 || R2 || RS =")
15 format(6)
16 fc3=1/(2*%pi*17.23*10*10^-6)
17 disp(fc3,"          f_c(bypass)(in Hz) = 1 / 2*pi*[(
    R_th+hie/beta) || RE]*CE")
18 disp("We have calculated all the three critical
    frequencies :")
19 disp("(a) fc(input) = 929.8 Hz")
20 disp("(b) fc(output) = 130.45 Hz")
21 disp("(c) fc(bypass) = 923.7 Hz")

```

Scilab code Exa 2.5 low frequency response of the FET amplifier

```

1 //Example 2.5
2 clc
3 disp("It is necessary to analyze each network to
    determine the critical frequency of the amplifier
    ")
4 disp("(a) Input RC Network")
5 disp("          fc = 1 / 2*pi*R_in*C1")
6 format(6)
7 rin=(100*100)/(100+100)
8 disp(rin," where R_in(in M-ohm) = RG || R_in(gate) =
    RG || |VGS/IGSS| =")
9 format(5)
10 fc1=1/(2*%pi*50*10^6*0.001*10^-6)
11 disp(fc1," Therefore , fc(in Hz) =")
12 disp("(b) Output RC Network")
13 format(6)
14 fc2=1/(2*%pi*(24.2*10^3)*(1*10^-6))
15 disp(fc2,"          fc(in Hz) = 1 / 2*pi*(RD+RL)*C2 =")
16 disp("We have calculated two critical frequencies")
17 disp("(a) fc(input) = 3.18 Hz")

```

```
18 disp("(b) fc(output) = 6.577 Hz")
```

Scilab code Exa 2.6 high frequency response of the amplifier

```
1 //Example 2.6
2 clc
3 disp("Before calculating critical frequencies it is
      necessary to calculate mid frequency gain of the
      given circuit. This is required to calculate C_in(
      miller) and C_out(miller)")
4 disp("      Av = -hfe*Ro / Ri")
5 disp(" where Ri = hie || R1 || R2")
6 disp(" and Ro = RC || RL")
7 format(6)
8 av=(-100*1.8)/1.032
9 disp(av," Therefore , Av = -hfe(RC||RL) / hie || R1 || R2
      =")
10 disp("Negative sign indicates 180 degree shift
      between input and output")
11 format(7)
12 cin=(4*(174.4+1))*10^-3 // in nF
13 disp(cin," C_in(miller)(in nF) = C_bc*(Av+1) =")
14 cout=(4*175.4)/(174.4) // in pF
15 format(4)
16 disp(cout," C_out(miller)(in pF) = C_bc*(Av+1) /
      Av =")
17 disp("We now analyze input and output network for
      critical frequency.")
18 format(8)
19 fci=(1/(2*pi*410*0.7216*10^-9))*10^-3 // in kHz
20 disp(fci," f_c(input)(in kHz) = 1 / 2*pi*(Rs || R1
      || R2 || hie)*(C_be+C_in(miller)) =")
21 format(5)
22 fco=(1/(2*pi*((22*10^6)/(12.2*10^3))*(4*10^-12)))
      *10^-6 // in MHz
```



```

23 disp(fco,"      f_c(output)(in MHz) = 1 / 2*pi*(RC||RL
      )*C_out(miller) =")
24 disp("We have calculated both the critical
      frequencies")
25 disp("(a) f_c(input) = 537.947 kHz")
26 disp("(b) f_c(output) = 22.1 MHz")

```

Scilab code Exa 2.7 high frequency response of the amplifier

```

1 //Example 2.7
2 clc
3 disp("Before calculating critical frequencies it is
      necessary to calculate mid frequency gain of the
      given amplifier circuit. This is required to
      calculate C_in(miller) and C_out(miller)")
4 disp("      Av = -gm * RD")
5 disp("Here, RD should br replaced by RD || RL")
6 av=-6*2
7 disp(av," Therefore ,      Av = -gm*(RD||RL) =")
8 cin=2*(12+1) // in pF
9 disp(cin," C_in(miller)(in pF) = C_gd*(Av+1) = C_rss
      *(Av+1) =")
10 format(6)
11 cout=(2*13)/12 // in pF
12 disp(cout," C_out(miller)(in pF) = C_gd*(Av+1) / Av =
      ")
13 disp("G_gs = C_liss - C_rss = 4 pF")
14 disp("We know analyze input and output network for
      critical frequency")
15 disp("      f_c(input) = 1 / 2*pi*RS*CT")
16 disp("      = 1 / 2*pi*RS*[ C_gs+C_in(miller
      )]")
17 format(4)
18 fc1=(1/(2*pi*100*30*10^-12))*10^-6 // in MHz
19 disp(fc1,"      f_c(input)(in MHz)= ")

```

```
20 fc2=(1/(2*pi*((48.4*10^6)/(24.2*10^3))
    *(2.166*10^-12)))*10^-6 // in MHz
21 format(6)
22 disp(fc2,"      f_c(output)(in MHz) = 1 / 2*pi*(RD||RL
    )*C_out(miller) =")
23 disp("We have calculated both the critical
    frequencies :")
24 disp("(a) f_c(input) = 53 MHz")
25 disp("(b) f_c(output) = 36.74 MHz")
```

Chapter 3

Feedback Amplifiers

Scilab code Exa 3.1 gain fL and fH

```
1 //Example 3.1
2 clc
3 disp("(a) Gain with feedback")
4 format(5)
5 av=1000/(1+(0.05*1000))
6 disp(av,"          AV_mid = Av_mid / 1+beta*Av_mid =")
7 flf=50/(1+(0.05*1000)) // in Hz
8 disp(flf,"(b)          f_Lf(in Hz) = f_L / 1+beta*Av_mid
           =")
9 fhf=((50*10^3)*(1+(0.05*1000)))*10^-6 // in MHz
10 disp(fhf,"(c)          f_Hf(in MHz) = f_H * (1+beta*
           Av_mid) =")
```

Scilab code Exa 3.2 Vo and second harmonic distortion with feedback

```
1 //Example 3.2
2 clc
3 disp("(a) beta:          -40 = 20*log[1+beta*A]")
```

```

4 disp(" Therefore , 1+beta*A = 100")
5 b=99/1000
6 format(6)
7 disp(b," Therefore , beta =")
8 disp(" Gain of the amplifier with feedback is given
as")
9 avf=1000/100
10 disp(avf," A_Vf = A_V / 1+beta*A_V =")
11 disp("(b) To maintain output power 10 W, we should
maintain output voltage constant and to maintain
output constant with feedback gain required Vs is
")
12 vsf=10*100*10^-3 // in V
13 disp(vsf," V_sf(in V) = Vs * 100 =")
14 disp("(c) Second harmonic distortion is reduced by
factor 1 + beta*A")
15 d2f=(0.1/100)*100 // in percentage
16 disp(d2f," D_2f(in percentage) = D_2 / 1+beta*A
=")

```

Scilab code Exa 3.3 beta and Af

```

1 //Example 3.3
2 clc
3 disp("(a) We know that")
4 disp(" dAf/Af = 0.1/1+beta*A * dA/A")
5 disp(" Therefore , 1+beta*A = 37.5")
6 b=(36.5/2000)*100 // in percentage
7 format(6)
8 disp(b," Therefore , beta(in percentage) =")
9 af=2000/(1+(0.01825*2000))
10 disp(af," (b) Af = A / 1+beta*A =")

```

Scilab code Exa 3.4 beta and Av and Avf and Rif and Rof

```

1 //Example 3.4
2 clc
3 disp("Step 1: Identity topology")
4 disp(" The feedback voltage is applied across the
      resistance R_e1 and it is in series with input
      signal. Hence feedback is voltage series feedback
      .")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output
      circuit.")
7 disp(" To find input circuit , set Vo = 0 (
      connecting C2 to ground), which gives parallel
      combination of Re with Rf at E1. To find output
      circuit , set Ii = 0 (opening the input node E1 at
      emitter of Q1), which gives series combination
      of Rf and Rel across the output. The resultant
      circuit is shown in Fig.3.32")
8 disp("")
9 disp("Step 4: Find open loop voltage gain(A_v)")
10 format(5)
11 r12=(4.7*10.1)/(4.7+10.1) // in k-ohm
12 disp(r12," R_L2(in k-ohm) = R_c2 || (R_e1+Rf) =")
13 disp(" A_i2 = -hfe = -100")
14 disp(" R_i2 = hie = 1100 ohm")
15 format(7)
16 av2=(-100*3.21*10^3)/1100
17 disp(av2," A_v2 = A_i2*R_L2 / R_i2 =")
18 disp(" A_i1 = -hfe = -100")
19 format(5)
20 r11=(22*220*22*1.100)/((220*22*1.100)+(22*22*1.100)
      +(22*220*1.100)+(22*220*22)) // in ohm
21 disp(r11*10^3," R_L1(in ohm) = R_c1 || R3 || R4 ||
      R_i2 =")
22 ri1=1.1+(101*((0.1*10)/(0.1+10))) // in k-ohm
23 format(5)
24 disp(ri1," R_i1(in k-ohm) = hie + (1+hfe)*R_e1eff =

```

```

                                where Re1eff = (R_e1 || Rf)”)
25 av1=(-100*995)/(11.099*10^3)
26 disp(av1,” Therefore ,  A_v1 = A_i1*RL1 / Ri1 =”)
27 disp(”The overall voltage gain without feedback is
    given as ,”)
28 av=-291.82*-8.96
29 format(7)
30 disp(av,”  Av = A_v1 * A_v2 =”)
31 disp(”The overall voltage gain taking Rs in account
    is given as ,”)
32 aV=(2614.7*11.099*10^3)/((11.099*10^3)+100)
33 format(8)
34 disp(aV,”  Av = Vo / Vs = Av*R_i1 / R_i1+Rs =”)
35 disp(””)
36 disp(”Step 5: Calculate beta”)
37 disp(”Looking at Fig.3.33.”)
38 beta=100/(100+(10*10^3))
39 format(7)
40 disp(beta,”  beta = Vf / Vo =”)
41 d=1+(0.0099*2591.35)
42 format(6)
43 disp(d,”  D = 1 + beta*Av =”)
44 avf=2591.35/26.65
45 disp(avf,”  A_vf = Av/D =”)
46 rif=26.65*11.099 // in k-ohm
47 format(8)
48 disp(rif,”  R_if(in k-ohm) = R_i1 * D =”)
49 riff=(295.788*220*22)/((220*22)+(295.788*22)
    +(295.788*220)) // in k-ohm
50 format(6)
51 disp(riff,”  R’ ’_if(in k-ohm) = R_if || R1 || R2 =”)
52 disp(”  R_of = Ro / D = infinity / D = infinity”)
53 disp(” Therefore ,  R’ ’_of = R’ ’_o / D           where
    R’ ’_o = R_L2”)
54 roff=(3.21*10^3)/26.65 // in omh
55 format(7)
56 disp(roff,” Therefore ,  R’ ’_of(in ohm) = ”)

```

Scilab code Exa 3.5 Avf and Rif and Rof

```
1 //Example 3.5
2 clc
3 disp("Step 1: Identity topology")
4 disp(" The feedback voltage is applied across R1
      (100 ohm), which is in series with input signal.
      Hence feedback is voltage series feedback.")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output
      circuit")
7 disp(" To find input circuit , set Vo = 0, which
      gives parallel combination of R1 with R2 at E1 as
      shown in the fig.3.45. To find output circuit ,
      set Ii = 0 by opening the input node, E1 at
      emitter of Q1, which gives the series combination
      of R2 and R1 across the output. The resultant
      circuit is shown in fig.3.45")
8 disp("")
9 disp("Step 4: Find the open loop voltage gain (Av)")
10 r12=(4.7*4.8)/(4.7+4.8) // in k-ohm
11 format(5)
12 disp(r12," R_L2(in k-ohm) =")
13 disp("Since h_oe = h_re = 0 we can use approximate
      analysis")
14 disp(" A_i2 = -hfe = -50")
15 disp(" R_i2 = hie = 1.1 k-ohm")
16 av2=(-50*2.37)/1.1
17 format(7)
18 disp(av2," A_v2 = A_i2*R_L2 / R_i2 =")
19 r11=(10*47*33*1.1)/((47*33*1.1)+(10*33*1.1)
      +(10*47*1.1)+(10*47*33)) // in ohm
20 format(5)
21 disp(r11*10^3," R_L1(in ohm) =")
```

```

22 disp("  A_i1 = -hfe = -50")
23 ri1=1.1+(51*((0.1*4.7)/(4.8))) // in k-ohm
24 format(6)
25 disp(ri1,"  R_i1(in k-ohm) = hie + (1+hfe)*Re =")
26 av1=(-50*942)/(6.093*10^3)
27 format(5)
28 disp(av1,"  A_v1 = A_i1*R_L1 / R_i1 =")
29 av=-7.73*-107.73
30 format(7)
31 disp(av," Therefore ,  A_v = A_v1 * A_v2 =")
32 disp("")
33 disp("Step 5: Calculate beta and D")
34 disp("  beta = R1 / R1+R2 = 1/48")
35 d=1+(832.75/48) // in ohm
36 format(6)
37 disp(d,"  D(in ohm) = 1 + A*beta =")
38 disp("")
39 disp("Step 5: Calculate A_vf, R_of and R_if")
40 avf=832.75/18.35
41 disp(avf,"  A_vf = A_v / D =")
42 rif=6.093*18.35 // in k-ohm
43 disp(rif,"  R_if(in k-ohm) = R_i1 * D =")
44 rof=(2.37*10^3)/18.35 // in ohm
45 format(7)
46 disp(rof,"  R_of(in ohm) = R_o / D =")

```

Scilab code Exa 3.6 Avf and Rif and Rof

```

1 //Example 3.6
2 clc
3 disp("Step 1: Identify topology")
4 disp("  The feedback voltage is applied across R_e1
      = 1.5 k-ohm, which is in series with input signal
      . Hence feedback is voltage series feedback")
5 disp("")

```



```

6 disp("Step 2 and step 3: Find input and output
   circuit")
7 disp(" To find input circuit , set Vo = 0, which
   gives parallel combination of R_e1 with R_f at E1
   as shown in fig.3.47. To find ouput circuit , set
   I_i = 0 by opening the input node, E1 at emitter
   of Q1, which gives the series combination of R_f
   and R_e1 across the output. The resultant
   circuit is shown in fig.3.47")
8 disp("")
9 disp("Step 4: Find the open loop voltage gain (Av)")
10 r12=(2.2*57.5)/(2.2+57.5) // in k-ohm
11 format(6)
12 disp(r12," R_L2(in k-ohm) = R_c2 || (Rf + R_e1) =")
13 disp("Since hoe*R_L2 = 10^-6*2.119 k-ohm = 0.002119
   is less than 0.1 we use approximate analysis.")
14 disp(" A_i2 = -h_fe = -200")
15 disp(" R_i2 = hie = 2 k-ohm")
16 av2=(-200*2.119)/2
17 disp(av2," A_v2 = A_i2*R_L2 / R_i2 =")
18 r11=(120*2)/(122) // in k-ohm
19 disp(r11," R_L1(in k-ohm) = R_C1 || R_i2 =")
20 disp("Since hoe*R_L1 = 10^-6*1.967 = 0.001967 is
   less than 0.1 we use approximate analysis.")
21 disp(" A_i1 = -hfe = -200")
22 ri1=2+(201*((1.5*56)/(57.5))) // in k-ohm
23 format(7)
24 disp(ri1," R_i1(in k-ohm) = hie + (1+hfe)*Re =")
25 av1=(-200*1.967)/295.63
26 format(5)
27 disp(av1," Therefore , A_v1 = A_i1*R_L1 / R_i1 =")
28 disp("The overall gain without feedback is")
29 av=-1.33*-211.9
30 format(7)
31 disp(av," Av = A_v1 * A_v2 =")
32 disp("")
33 disp("Step 5: Calculate beta")
34 beta=1.5/57.5

```

```

35 format(6)
36 disp(beta," beta = Vf / Vo =")
37 disp("")
38 disp("Step 6: calculate D, A_vf, R_if, R_of")
39 d=1+(0.026*281.82)
40 disp(d," D = 1 + Av*beta =")
41 avf=281.82/8.327
42 disp(avf," Therefore, A_vf = Av / D =")
43 ri=(295.63*150)/(295.63+150) // in k-ohm
44 format(5)
45 disp(ri," Ri(in k-ohm) = R_i1 || R =")
46 rif=99.5*8.327 // in k-ohm
47 format(7)
48 disp(rif," R_if(in k-ohm) = Ri *D =")
49 disp(" Ro = 1/hoe = 1 M-ohm")
50 rof=((1*10^6)/8.327)*10^-3 // in k-ohm
51 format(4)
52 disp(rof," R_of(in k-ohm) = Ro / D =")
53 ro=(1000*2.119)/(2.119+1000) // in k-ohm
54 format(7)
55 disp(ro," R' 'o(in k-ohm) = Ro || R_c2 || (Rf+R_e1)
= Ro || R_L2 =")
56 rof=(2.1145*10^3)/8.327 // in ohm
57 format(4)
58 disp(rof," R' ' _of(in ohm) = R' ' o / D =")

```

Scilab code Exa 3.7 D and Avf and Rif and Rof

```

1 //Example 3.7
2 clc
3 disp("Step 1: Identity topology")
4 disp(" By shorting output voltage (Vo = 0),
feedback voltage Vf becomes zero and hence it is
voltage sampling. The feedback voltage is aaplied
in series with input voltage hence the topology

```

```

    is voltage series feedback.")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output
    circuit.")
7 disp(" To find input circuit , set Vo = 0. This
    places the parallel combination of resistor 10 K
    and 200 ohm at first source. To find output
    circuit , set Ii = 0. This places the resistor 10
    K and 200 ohm in series across the output. The
    resultant circuit is shown in fig.3.50.")
8 disp("")
9 disp("Step 4: Replace FET with its equivalent
    circuit as shown in fig.3.51")
10 disp("")
11 disp("Step 5: Find open loop transfer gain.")
12 disp(" Av = Vo / Vs = A_v1*A_v2")
13 disp(" A_v2 = -u*R_L2 / R_L2+r_d")
14 r12=(10.2*47)/(10.2+47) // in k-ohm
15 format(5)
16 disp(r12," where R_L2(in k-ohm) =")
17 av2=(-40*8.38)/(8.38+10)
18 format(7)
19 disp(av2," Therefore , A_v2 =")
20 disp(" A_v1 = u*R_DeFF / r_d+R_DeFF+(1+u)*R_seff")
21 rdeff=(47*1000)/(47+1000) // in k-ohm
22 format(6)
23 disp(rdeff," where R_DeFF(in k-ohm) = R_D || R_G2 =")
    )
24 disp(" R_seff = 200 || 10 K")
25 av1=(-40*44.98*10^3)/((10*10^3)+(44.89*10^3)
    +(41*((10*0.2)/(10.2))))
26 disp(av1," A_v1 =") // answer in textbook is
    wrong
27 oav=-28.59*-18.237
28 format(7)
29 disp(oav," Therefore , Overall Av =")
30 disp("")
31 disp("Step 6: Calculate beta")

```

```

32 beta=200/(10.2*10^3)
33 disp(beta," beta = Vf / Vo =")
34 disp("")
35 disp("Step 7: Calculate D, A_vf, R_if, R''_of")
36 d=1+(0.0196*521.39)
37 format(6)
38 disp(d," D = 1 + Av*beta =")
39 avf=521.39/11.22
40 disp(avf," A_vf = Av / D =")
41 disp(" Ri = R_G = 1 M-ohm")
42 rif=11.22
43 disp(rif," R_if(in M-ohm) = Ri * D =")
44 disp(" Ro = r_d = 10 k-ohm")
45 ro=(10*8.38)/(18.38) // in k-ohm
46 disp(ro," R''_o(in k-ohm) = r_d || R_L2 =")
47 rof=(4.559*10^3)/11.22 // in ohm
48 format(4)
49 disp(rof," R''_of(in ohm) = R''_o / D =")

```

Scilab code Exa 3.8 voltage gain

```

1 //Example 3.8
2 clc
3 disp("Here, output voltage is sampled and fed in
      series with the input signal. Hence the topology
      is voltage series feedback.")
4 disp(" The open loop voltage gain for one stage is
      given as,")
5 disp(" Av = -gm*R_eq")
6 req=(8*40*1000)/((40*1000)+(8*1000)+(8*40)) // in k
      -ohm
7 format(5)
8 disp(req," R_eq(in k-ohm) = r_d || R_d || (R_i1+R_2
      ) =")
9 av=-5*6.62

```

```

10 format(6)
11 disp(av," Av =")
12 avm=-33.11^3
13 disp(avm,"Av = Overall voltage gain = |A_vmid|^3 =")
    // answer in textbook is wrong
14 beta=50/(10^6)
15 format(7)
16 disp(beta," beta = Vf / Vo = -R_1 / R_g = -R_1 /
    R_1+R_2 =")
17 d=1+((-5*10^-5)*-36306)
18 format(6)
19 disp(d," D = 1 + |Av|*beta =")
20 avf=-36306/2.8153
21 disp(avf," A_vf = Av / D =")

```

Scilab code Exa 3.9 Avf

```

1 //Example 3.9
2 clc
3 disp("Here, output terminals are B and ground, thus
    the forward gain is the gain of Q1 and it is,")
4 disp(" A_BN = -33.11")
5 disp("However, Q2 and Q3 must be considered as a
    part of feedback loop")
6 disp("Here beta_BN = V_f / V_B = V_f/V_o * V_o/V_C
    * V_C/V_B")
7 disp("where V_B and V_C are voltages at point B and
    C, respectively.")
8 disp("Therefore, beta_BN = V_f/V_o * A_v3 * A_v2
    because V_o/V_C = A_v3 and V_C/V_B = A_v2
    ")
9 bbn=-(5*10^-5)*(33.11^2)
10 format(7)
11 disp(bbn,"Therefore, beta_BN = -R1/R_g * A_v3 *
    A_v2 =")

```

```

12 disp("Note that the loop gain - beta_BN * A_BN = A^3
    _Vo * R1/Rg = -1.815 = -A*beta")
13 disp("It should be clear tht regardless ofwhere the
    output terminals are taken , the loop gain is
    unchanged.")
14 avf=-33.11/2.815
15 format(6)
16 disp(avf,"Therefore ,  A_vf = A_BN / 1+A*beta =")

```

Scilab code Exa 3.10 Avf and Rif and Rof

```

1 //Example 3.10
2 disp("Step 1: Identify topology")
3 disp(" By shorting output voltage (Vo = 0) ,
    feedback voltage Vf becomes zero and hence it is
    voltage sampling. The feedback voltage is applied
    in series with the input voltage hence the
    topology is voltage series feedback.")
4 disp("")
5 disp("Step 2 and Step 3: Find input and output
    circuit.")
6 disp(" To find input circuit , set Vo = 0. This
    places the parallel combination of resistor 10 K
    and 300 ohm at first source. To find output
    circuit , set Ii = 0. This places the resistor 10K
    and 300 ohm in series across the output. The
    resultant circuit is shown in fig.3.54.")
7 disp("")
8 disp("Step 4: Replace FET with its equivalent
    circuit as shown in fig.3.55.")
9 disp("")
10 disp("Step 5: Find open loop transfer gain.")
11 disp(" Av = Vo / Vs = A_v1 * A_v2")
12 disp(" A_v2 = -u*R_L2 / R_L2+r_d")
13 r12=(10.3*22)/(10.3+22) // in k-ohm

```

```

14 format(3)
15 disp(r12," where R_L2(in k-ohm) =")
16 av2=(-50*7)/17
17 format(6)
18 disp(av2," A_v2 =")
19 disp(" A_v1 = u*R_DeFF / r_d+R_DeFF+(1+u)*R_seff")
20 rdeff=(22*1000)/(22+1000) // in k-ohm
21 disp(rdeff," R_DeFF(in k-ohm) = R_D || R_G2 =")
22 disp(" R_seff = 330 || 10K")
23 av1=(-50*21.53)/(10+21.53+(51*((0.33*10)/(10+0.33)))
    )
24 disp(av1," Therefore , A_v1 =")
25 av=-20.59*-22.51
26 disp(av," Overall Av = A_v1 * A_v2 =")
27 disp("")
28 disp("Step 6: Calculate beta")
29 beta=330/(330+10000)
30 format(7)
31 disp(beta," beta = Vf / Vo = Rs / Rs+Rf =")
32 disp("")
33 disp("step 7: Calculate D, A_vf, R_if, R''_of")
34 d=1+(0.0319*463.5)
35 disp(d," D = 1 + Av*beta =")
36 avf=463.5/15.785
37 format(6)
38 disp(avf," A_vf = Av / D =")
39 disp("Ri = R_G = 1 M-ohm")
40 rif=15.785
41 format(7)
42 disp(rif," R_if(in k-ohm) = Ri * D =")
43 ro=(10*7)/(10+7) // in k-ohm
44 format(6)
45 disp(ro," R''_o(in k-ohm) = rd || R_L2 =")
46 rof=(4.118*10^3)/15.785 // in ohm
47 format(4)
48 disp(rof," R''_of(in ohm) = R''_o / D =")

```

Scilab code Exa 3.11 voltage gain and input and output resistance

```
1 //Example 3.11
2 clc
3 disp("Step 1: Identify topology")
4 disp(" The feedback voltage is applied across the
      resistance R_e1 and it is in series with input
      signal. Hence feedback is voltage series feedback
      .")
5 disp("")
6 disp("step 2 and Step 3: Find input and output
      circuit.")
7 disp(" To find input circuit , set Vo = 0 (
      connecting C2 to ground), which gives parallel
      combination of R_e with R_f at E1. To find output
      circuit , set Ii = 0 (opening the input node E1 at
      emitter of Q1), which gives series combination of
      R_f and R_e1 across the output. The resultant
      circuit is shown in fig.3.57")
8 disp("")
9 disp("Step 4: Find open loop voltage gain (Av)")
10 r12=(4.7*3.42)/(4.7+3.42) // in k-ohm
11 format(5)
12 disp(r12," R_L2(in k-ohm) = R_c2 || (Rs+R) =")
13 disp(" A_i2 = -hfe = -50")
14 disp(" R_i2 = hie = 1000 ohm = 1 k-ohm")
15 av2=-50*1.98
16 format(3)
17 disp(av2," A_v2 = A_i2*R_L2 / R_i2 =")
18 disp(" A_i1 = -hfe = -50")
19 format(7)
20 r11=((10*100*22*1)/((100*22)+(10*22)+(10*100)
      +(10*100*22)))*10^3 // in ohm
21 disp(r11," R_L1(in ohm) = R_c1 || R3 || R4 || R_i2
```



```

    =")
22 disp(" R_i1 = h_ie + (1+h_fe)*R_e1eff")
23 re1=1+(51*((3.3*0.12)/(3.42))) // in k-ohm
24 format(4)
25 disp(re1," where R_e1eff(in k-ohm) = Rs || R =")
26 av1=(-50*865.46)/6900
27 format(5)
28 disp(av1," A_v1 = A_i1*R_L1 / R_i1 =")
29 disp("The overall voltage gain,")
30 av=-6.27*-99
31 format(7)
32 disp(av," Av = A_v1 * A_v2 =")
33 disp("")
34 disp("Step 5: Calculate beta")
35 beta=120/(120+3300)
36 format(6)
37 disp(beta," beta = Vf / Vo = Rs / Rs+R =")
38 disp("")
39 disp("Step 6: Calculate D, A_vf, R_if, R_of and R'
_of")
40 d=1+(0.035*620.73)
41 format(7)
42 disp(d," D = 1 + Av*beta =")
43 avf=620.73/22.725
44 format(5)
45 disp(avf," A_vf = Av / D =")
46 rif=6.9*22.725 // in k-ohm
47 format(6)
48 disp(rif," R_if(in k-ohm) = R_i1 * D =")
49 disp(" R_of = Ro / D = infinity")
50 rof=(1.98*10^3)/22.725 // in ohm
51 disp(rof," R' _of(in ohm) = R' _o / D = R_L2 / D =")

```

Scilab code Exa 3.12 Ai and beta and Aif

```

1 //Example 3.12
2 clc
3 disp("Step 1: Identify topology")
4 disp(" The feedback is given from emitter of Q2 to
   the base of Q2. If  $I_o = 0$  then feedback current
   through 5 K register is zero , hence it is current
   sampling. As feedback signal is mixed in shunt
   with input , the amplifier is current shunt
   feedback amplifier.")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output ")
7 disp(" The input circuit of the amplifier without
   feedback is obtained by opening the output loop
   at the emitter of Q2( $I_o = 0$ ). This places  $R''$ (5 K
   ) in series with  $R_e$  from base to emitter of Q1.
   The output circuit is found by shorting the input
   node , i.e. making  $V_i = 0$ . This places  $R''$  (5 K)
   in parallel with  $R_e$ . The resultant equivalent
   circuit is shown in fig.3.59 ")
8 disp("")
9 disp("Step 4: Find open circuit transfer gain.")
10 disp("  $A_i = I_o / I_s = -I_c/I_{b2} * I_{b2}/I_{c1} * I_{c1}/$ 
    $I_{b1} * I_{b1}/I_s$ ")
11 disp("We know that  $-I_{c2} / I_{b2} = A_{i2} = -h_{fe} = -50$ 
   and")
12 disp("  $-I_{c1} / I_{b1} = A_{i1} = -h_{fe} = 50$ ")
13 disp("  $I_{c1} / I_{b1} = 50$ ")
14 disp("Looking at fig.3.59 we can write,")
15 disp("  $I_{b2} / I_{c1} = -R_{c1} / R_{c1}+R_{i2}$  ")
16 ri2=1.5+(51*((5*0.5)/(5.5))) // in k-ohm
17 format(8)
18 disp(ri2," where  $R_{i2}(\text{in k-ohm}) = h_{ie} + (1+h_{fe})*$ 
    $R_{e2} || R'' =$ ")
19 x1=-2/(2+24.6818)
20 disp(x1,"  $I_{b2} / I_{c1} =$ ")
21 disp("  $I_{b1} / I_s = R / R+R_{i1}$  where  $R = R_s || (R$ 
    $''+R_{e2})$  ")
22 r=((1*5.5)/(1+5.5))*10^3 // in ohm

```

```

23 format(9)
24 disp(r,"Therefore , R(in ohm) =")
25 disp("and R_i1 = h_ie + (1+h_fe)*R_e1 = 16.8 k-ohm"
      )
26 x1=846.1538/(846.1538+(16.8*10^3))
27 format(8)
28 disp(x1,"Therefore , I_b1 / I_s =")
29 ai=50*0.07495*50*0.04795
30 format(7)
31 disp(ai," A_I =")
32 disp("")
33 disp("Step 5: Calculate beta")
34 beta=500/(500+(5*10^3))
35 disp(beta," beta = I_f / I_o = R_e2 / R_e2|R' ' =")
36 disp("")
37 disp("Step 6: Calculate D, A_If")
38 d=1+(0.0909*8.9848)
39 disp(d," D = 1 + A_I*beta =")
40 aif=8.9848/1.8168
41 disp(aif," A_If = A_I / D =")

```

Scilab code Exa 3.13 beta and Av and Avf and Rif and Rof

```

1 //Example 3.13
2 clc
3 disp("Step 1: Identify topology")
4 disp(" The feedback voltage is applied across R1
      (150 ohm), which is in series with input signal.
      Hence feedback is voltage series feedback.")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output
      circuit")
7 disp(" To find input circuit , set Vo = 0, which
      gives parallel combination of R1 with R2 at E1 as
      shown in the fig.3.61. To find output circuit ,

```

set $I_i = 0$ by opening the input node, E1 at emitter of Q1, which gives the series combination of R2 and R1 across the output. The resultant circuit is shown in fig.3.61.”)

```

8 disp(“”)
9 disp(“Step 4: Find the open loop voltage gain (Av)”)
10 r12=(4.7*15.15)/(4.7+15.15) // in k-ohm
11 format(5)
12 disp(r12,“ RL2(in k-ohm) =”)
13 disp(“Since hoe = hre = 0, we can use approximate
analysis.”)
14 disp(“ Ai2 = -h_fe = -500”)
15 disp(“ Ri2 = h_ie = 1100 ohm”)
16 av2=(-500*3.59*10^3)/1100
17 disp(av2,“ Av2 = Ai2*RL2 / Ri2 =”)
18 r11=((10*47*33*1.1)/((47*33*1.1)+(10*33*1.1)
+(10*47*1.1)+(10*47*33)))*10^3 // in ohm
19 disp(r11,“ RL1(in ohm) = 10K || 47K || 33K || Ri2
=”)
20 disp(“ Ai1 = -h_fe = -500”)
21 ri1=1.1+(501*((0.15*15)/(0.15+15))) // in k-ohm
22 disp(ri1,“ Ri1(in k-ohm) = h_ie + (1+h_fe)*Re =”)
23 av1=(-500*942)/(75.5*10^3)
24 format(6)
25 disp(av1,“ Av1 = Ai1*RL1 / Ri1 =”)
26 av=-6.238*-1632
27 disp(av,“ Av = Av1 * Av2 =”)
28 disp(“”)
29 disp(“Step 5: Calculate beta and D”)
30 beta=150/(150+15000)
31 format(7)
32 disp(beta,“ beta = R1 / R1+R2 =”)
33 d=1+(10180*0.0099)
34 format(8)
35 disp(d,“ D = 1 + A*beta =”)
36 disp(“”)
37 disp(“Step 6: Calculate Avf, R_of and Rif”)
38 avf=10180/101.782

```

```

39 format(4)
40 disp(avf," A_vf = Av / D =")
41 rif=75.5*101.782*10^-3 // in M-ohm
42 format(6)
43 disp(rif," R_if(in M-ohm) = R_i1 * D =")
44 rof=(3.59*10^3)/101.782
45 disp(rof," R_of(in ohm) = Ro / D = R_L2 / D =")

```

Scilab code Exa 3.14 gain and new bandwidth

```

1 //Example 3.14
2 clc
3 disp("Given: A_vmid = 500, f_L = 100 kHz, f_H = 20
      kHz and beta = 0.01")
4 avf=500/(1+(0.01*500))
5 format(6)
6 disp(avf," A_vf = A_vmid / 1+beta*A_vmid =")
7 flf=100/(1+(0.01*500)) // in Hz
8 disp(flf," f_Lf(in Hz) = f_L / 1+beta*A_vmid =")
9 fhf=20*(1+(0.01*500)) // in kHz
10 disp(fhf," f_Hf(in kHz) = f_H * (1 + beta*A_vmid) =
      ")
11 bw=120-0.01667 // in kHz
12 format(9)
13 disp(bw," BW_f(in kHz) = f_Hf - f_Lf =")

```

Scilab code Exa 3.15 show voltage gain with feedback

```

1 //Example 3.15
2 clc
3 disp("Step 1: Identify topology")
4 disp(" By shorting output(Vo = 0), feedback voltage
      does not become zero. By opening the output loop

```

```

    feedback becomes zero and hence it is current
    sampling. The feedback is applied in series with
    the input signal, hence topology used is current
    series feedback.")
5 disp("")
6 disp("Step 2 and Step 3: Find input and output
    circuit.")
7 disp(" To find input circuit , set Io = 0. This
    places Re in series with input. To find output
    circuit Ii = 0. This places Re in output side.
    The resultant circuit is shown in fig.3.63.")
8 disp("")
9 disp("Step 4: Replace transistor with its h-
    parameter equivalent as shown in fig.3.64.")
10 disp("")
11 disp("Step 5: Find open loop transfer gain.")
12 disp(" From quation(5) of section 3.9.1 we have")
13 disp(" A_vf = Io*R_L / Vs = G_Mf * R_L")
14 disp("      = -h_fe*R_L / R' 's+h_ie+(1+h_fe)*Re")
15 disp(" Here R' 's = Rs || R1 || R2")
16 disp("      = Rs || Rb           because R_b = R1
    || R2")
17 disp(" Therefore , Vo / Vs = Vo/Vi * Vi/Vs")
18 disp(" where Vi / Vs = Rb / Rs+Rb")
19 disp(" Therefore , Vo / Vs = (-h_fe*R_L / R' 's+h_ie
    +(1+h_fe)*Re) * (Rb / Rs+Rb)")
20 disp(" Dividing both numerator and denominator by Rs+
    Rb we get ,")
21 disp(" A_vf = Vo / Vs = [-h_fe*Rc*(Rb/Rb+Rs)] / R' '
    s+h_ie+(1+h_fe)*Re because RL = Rc")
22 disp("      = -h_fe*Rc*[1/1+(Rs/Rb)] / R' 's+h_ie
    +(1+h_fe)*Re")

```

Scilab code Exa 3.16 GMf and Rif and Rof

```

1 //Example 3.16
2 clc
3 disp("Refer example 3.15")
4 disp("  A_vf = -h_fe*Rc*[1/1+(Rs/Rb)] / R' 's+h_ie
      +(1+h_fe)*Re      where R' 's = Rs || R1 || R2")
5 avf=(-50*(1.8*10^3)*[1/(1+(1000/4272))])
      /(810+1000+((1+50)*1000))
6 format(5)
7 disp(avf,"  A_vf =")
8 gmf=-1.38/(1.8*10^3)
9 format(9)
10 disp(gmf,"  G_Mf = A_vf / R_L =")
11 disp("  beta = Vf / Io = Ie*Re / Io = -Io*Re / Io =
      -Re = -1 K")
12 disp("  G_Mf = G_M / 1+beta*G_M")
13 gm=1/((1/(-7.66*10^-4))+1000)
14 format(10)
15 disp(gm,"Therefore ,  G_M =")
16 d=1+(-1000*-3.2735*10^-3)
17 format(7)
18 disp(d,"  D = 1 + G_M*beta =")
19 ri=(1+1.36) // in k-ohm
20 format(5)
21 disp(ri,"  R_i(in k-ohm) = Rs+(h_ie+Re) || R_D =")
22 rif=2.36*4.2735 // in k-ohm
23 format(3)
24 disp(rif,"  R_if(in k-ohm) = R_i * D =")
25 disp("  R_o = infinity")
26 disp("  R_of = R_o * D = infinity")
27 disp("  R' ' _of = R_of || R_L = R_L = 1.8 k-ohm")

```

Scilab code Exa 3.17 feedback factor and Rif and Rof

```

1 //Example 3.17
2 clc

```

```

3 disp(" Refer example 3.15")
4 disp("  A_vf = -h_fe*Rc*[1/1+(Rs/Rb)] / R' 's+h_ie
      +(1+h_fe)*Re      where R' 's = Rs || R1 || R2")
5 avf=(-50*(4*10^3)*[1/(1+(1000/9000))])
      /(900+1000+((1+150)*1000))
6 format(6)
7 disp(avf,"  A_vf =")
8 gmf=-1.177/(4*10^3)
9 format(9)
10 disp(gmf,"  G_Mf = A_vf / R_L =")
11 disp("  beta = Vf / Io = Ie*Re / Io = -Io*Re / Io =
      -Re = -1 K")
12 disp("  G_Mf = G_M / 1+beta*G_M")
13 gm=1/((1/(-2.943*10^-4))+1000)
14 format(9)
15 disp(gm," Therefore ,  GM =")
16 d=1+(-1000*-4.17*10^-4)
17 format(6)
18 disp(d,"  D = 1 + GM*beta =")
19 ri=1+((2*9)/(2+9)) // in k-ohm
20 disp(ri,"  R_i(in k-ohm) = Rs+(h_ie+Re) || R.D =")
21 rif=2.636*1.417 // in k-ohm
22 format(6)
23 disp(rif,"  R_if(in k-ohm) = R_i * D =")
24 disp("  R_o = infinity")
25 disp("  R_of = R_o * D = infinity")
26 disp("  R' ' _of = R_of || R_L = R_L = 4 k-ohm")

```

Scilab code Exa 3.18 gain with feedback and new bandwidth

```

1 //Example 3.18.
2 clc
3 disp(" Given: A_v mid = 40, f_L = 100 Hz, f_H = 15
      kHz and beta = 0.01")
4 avf=400/(1+(0.01*400))

```



```

5 format(3)
6 disp(avf," A_vf = A_v mid / 1+beta*A_v mid =")
7 flf=100/(1+(0.01*400))
8 disp(flf," f_Lf = f_L / 1+beta*A_v mid =")
9 fhf=(15)*(1+(0.01*400)) // in kHz
10 disp(fhf," f_Hf(in kHz) = f_H * (1+beta*A_v mid) =")
    )
11 bw=75-0.02 // in kHz
12 format(6)
13 disp(bw," BWf(in kHz) = f_Hf - f_Lf =")

```

Scilab code Exa 3.19 overall voltage gain and bandwidth

```

1 //Example 3.19
2 clc
3 disp("Given: Av = 10, BW = 1*10^3, n =3")
4 disp("(i) Overall voltage gain")
5 disp("The gain of cascaded amplifier without
    feedback = 10*10*10 = 1000")
6 avf=1000/(1+(0.1*1000))
7 format(4)
8 disp(avf," A_vf = Av / 1+Av*beta =")
9 disp("(ii) Bandwidth of cascaded stage")
10 disp("Bandwidth of cascaded amplifier without
    feedback")
11 bw=((1*10^6)*sqrt((2^(1/3))-1))*10^-3 // in kHz
12 format(7)
13 disp(bw," BW(cascade)(in kHz) = BW*sqrt(2^(1/n) -
    1) =")
14 bwf=(509.82*10^3*(1+(0.1*1000)))*10^-6 // in MHz
15 format(6)
16 disp(bwf," BWf(in MHz) = BW * (1 + beta*A_v mid) =
    ")

```

Chapter 4

Oscillators

Scilab code Exa 4.1 C and hfe

```
1 //Example 4.1
2 clc
3 disp("Referring to equation(1),")
4 ri=(25*57*1.8)/((57*1.8)+(25*1.8)+(25*57)) // in k-
    ohm
5 format(6)
6 disp(ri," R' ' _i (in k-ohm) = R1 || R2 || h_ie =")
7 disp("Now R' ' _i + R3 = R")
8 r3=7.1-1.631 // in k-ohm
9 format(5)
10 disp(r3," Therefore , R3(in k-ohm) = R - R' ' _i =")
11 k=20/7.1
12 format(6)
13 disp(k," K = R_C / R =")
14 disp("Now f = 1 / 2*pi*R*C*sqrt(6+4K)")
15 c=(1/(sqrt(6+(4*2.816))*2*%pi*7.1*10*10^6))*10^12
    // in pF
16 format(8)
17 disp(c," Therefore , C(in pF) =")
18 disp(" h_fe >= 4K + 23 + 29/K")
19 hfe=(4*2.816)+23+(29/2.816)
```

```
20 format(7)
21 disp(hfe," h_fe >=")
```

Scilab code Exa 4.2 frequency of oscillation

```
1 //Example 4.2
2 clc
3 disp("The given values are , R = 4.7 k-ohm and C =
      0.47 uF")
4 f=1/(2*%pi*sqrt(6)*(4.7*10^3)*(0.47*10^-6)) // in
      Hz
5 format(7)
6 disp(f," f(in Hz) = 1 / 2*pi*sqrt(6)*R*C =")
```

Scilab code Exa 4.3 R and C

```
1 //Example 4.3
2 clc
3 disp("f = 1 kHz")
4 disp("Now f = 1 / 2*pi*sqrt(6)*R*C")
5 disp("Choose C = 0.1 uF")
6 r=1/(sqrt(6)*2*%pi*0.1*1*10^-3) // in ohm
7 format(8)
8 disp(r,"Therefore , R(in ohm) = ")
9 disp("Choose R = 680 ohm standard value")
```

Scilab code Exa 4.4 C and RD

```
1 //Example 4.4
2 clc
```

```

3 disp("Using the expression for the frequency")
4 disp("Now, f = 1 / 2*pi*R*C*sqrt(6)")
5 f=(1/(sqrt(6)*2*%pi*9.7*5*10^6))*10^9 // in nF
6 format(5)
7 disp(f,"Therefore, C(in nF) =")
8 disp("Now using the equation(27)")
9 disp(" |A| = g_m * R_L")
10 disp("Therefore, |A| >= 29")
11 disp("Therefore, g_m * R_L >= 29")
12 r1=(29/(5000*10^-6))*10^-3 // in k-ohm
13 format(4)
14 disp(r1,"Therefore, R_L(in k-ohm) >= 29 / g_m =")
15 disp(" R_L = R_D*r_d / R_D+r_d")
16 rd=(40)/4.8823
17 format(5)
18 disp(rd," Therefore, R_D(in k-ohm) = ")
19 disp(" While for minimum value of R_L = 5.8 k-ohm")
20 disp(" R_D = 6.78 k-ohm")

```

Scilab code Exa 4.5 minimum and maximum R2

```

1 //Example 4.5
2 clc
3 disp("The frequency of the oscillator is given by,")
4 disp(" f = 1 / 2*pi*sqrt(R1*R2*C1*C2)")
5 disp(" For f = 10 kHz,")
6 r2=(1/(4*(%pi^2)*(100*10^6)*(10*10^3)*(0.001*10^-12)
7 )) // in k-ohm
8 format(6)
9 disp(r2,"Therefore, R2(in k-ohm) =")
10 disp(" For f = 50 kHz,")
11 r2=(1/(4*(%pi^2)*(2500*10^6)*(10*10^3)
12 *(0.001*10^-12))) // in k-ohm
13 format(6)
14 disp(r2,"Therefore, R2(in k-ohm) =")

```

```
13 disp("So minimum value of R2 is 1.013 k-ohm while
    the maximum value of R2 is 25.33 k-ohm")
```

Scilab code Exa 4.6 range over capacitor is varied

```
1 //Example 4.6
2 clc
3 disp("The frequency is given by,")
4 disp(" f = 1 / 2*pi*sqrt(C*L_eq)")
5 leq=(2*10^-3)+(20*10^-6)
6 format(8)
7 disp(leq," where L_eq = L1 + L2 =")
8 disp("For f = f_max = 2050 kHz")
9 format(5)
10 c=(1/(4*(%pi^2)*((2050*10^3)^2)*0.00202))*10^12 //
    in pF
11 disp(c," Therefore , C(in pF) =")
12 disp("For f = f_min = 950 kHz")
13 c=(1/(4*(%pi^2)*((950*10^3)^2)*0.00202))*10^12 //
    in pF
14 format(6)
15 disp(c," Therefore , C(in pF) =")
16 disp("Hence C must be varied from 2.98 pF to 13.89
    pF, to get the required frequency variation.")
```

Scilab code Exa 4.7 frequency of oscillation

```
1 //Example 4.7
2 clc
3 disp("The given values are,")
4 disp(" L1 = 0.5 mH, L2 = 1 mH, C = 0.2 uF")
5 disp("Now f = 1 / 2*pi*sqrt(C*L_eq)")
6 leq=0.5+1 // in mH
```

```

7 disp(1eq,"and L_eq(in mH) = L1 + L2 =")
8 f=(1/(2*%pi*sqrt(1.5*0.2*10^-9)))*10^-3 // in kHz
9 format(5)
10 disp(f,"Therefore , f(in kHz) =")

```

Scilab code Exa 4.8 frequency of oscillation

```

1 //Example 4.8
2 clc
3 disp("The equivalent capacitance is given by,")
4 ceq=(150*1.5*10^-21)/((150*10^-12)+(1.5*10^-9)) //
   in F
5 format(12)
6 disp(ceq," C_eq(in F) = C1*C2 / C1+C2 =")
7 disp("Now, f = 1 / 2*pi*sqrt(L*C_eq)")
8 f=(1/(2*%pi*sqrt(50*136.363*10^-18)))*10^-6 // in
   MHz
9 format(6)
10 disp(f," f(in MHz) =")

```

Scilab code Exa 4.9 calculate C

```

1 //Example 4.9
2 clc
3 disp("The given values are,")
4 disp(" L = 100 uH, C1 = C2 = C and f = 500
   kHz")
5 disp("Now, f = 1 / 2*pi*sqrt(L*C_eq)")
6 ceq=1/(4*(%pi^2)*(100*10^-6)*((500*10^3)^2)) // in
   F
7 format(11)
8 disp(ceq,"Therefore , C_eq(in F) =")

```

```

9 disp("but      C_eq = C1*C2 / C1+C2      and C1 = C2 = C
      ")
10 disp("Therefore ,   C_eq = C / 2")
11 c=1.0132*2
12 format(6)
13 disp(c,"Therefore ,   C(in nF) =")

```

Scilab code Exa 4.10 series and parallel resonant frequency

```

1 //Example 4.10
2 clc
3 fs=(1/(2*%pi*sqrt(0.4*0.085*10^-12)))*10^-6 // in
      MHz
4 format(6)
5 disp(fs,"(i)   f_s (in MHz) = 1 / 2*pi*sqrt(L*C) =")
6 ceq=0.085/1.085 // in pF
7 disp(ceq,"(ii)  C_eq(in pF) = C*C_M / C+C_M =")
8 fp=(1/(2*%pi*sqrt(0.4*0.078*10^-12)))*10^-6 // in
      MHz (the answer in textbook is wrong)
9 disp(fp,"Therefore ,   f_p(in MHz) = 1 / 2*pi*sqrt(L*
      C_eq) =")
10 inc=((0.899-0.856)/0.856)*100 // in percentage
11 disp(inc,"(iii)  %increase =")
12 q=(2*%pi*0.4*0.856*10^6)/(5*10^3)
13 format(8)
14 disp(q,"(iv)   Q = omega_s*L / R = 2*pi*f_s*L / R =")

```

Scilab code Exa 4.11 series and parallel resonant frequency

```

1 //Example 4.11
2 clc
3 disp("      C_M = 2 pF")
4 fs=(1/(2*%pi*sqrt(2*0.01*10^-12)))*10^-6 // in MHz

```

```

5 format(6)
6 disp(fs,"Now f_s (in MHz) = 1 / 2*pi*sqrt(L*C) =")
7 ceq=(2*0.01*10^-24)/(2.01*10^-12) // in F
8 format(9)
9 disp(ceq," C_eq (in F) = C_M*C / C_M+C =")
10 fp=(1/(2*%pi*sqrt(2*9.95*10^-15)))*10^-6 // in MHz
11 format(6)
12 disp(fp," f_p = 1 / 2*pi*sqrt(L*C_eq) =")
13 disp("So f_s and f_p values are almost same.")

```

Scilab code Exa 4.12 Verify Barkhausen criterion and find frequency of oscillation

```

1 //example 4.12.
2 clc
3 disp("From the given information we can write,")
4 disp(" A = -16*10^6/j*omega and beta =
5 10^3/[2*10^3+j*omega]^2")
6 disp("To verify the Barkhausen condition means to
7 verify whether |A*beta| = 1 at a frequency for
8 which A*beta = 0 degree. Let us express , A*beta
9 in its rectangular form.")
10 disp(" A*beta = -16*10^6*10^3 / j*omega*[2*10^3+j*
11 omega]^2 = -16*10^9 / j*omega*[4*10^6+4*10^3*j*
12 omega+(j*omega)^2]")
13 disp(" = -16*10^9 / j*omega*[4*10^6+4*10^3*j
14 *omega-omega^2] as j^2 = -1")
15 disp(" = -16*10^9 / 4*10^6*j*omega+4*10^3*j
16 ^2*omega^2-j*omega^3")
17 disp(" = -16*10^9 / j*omega*[4*10^6-omega
18 ^2]-[omega^2*4*10^3]")
19 disp(" Rationalising the denominator function we get ,
20 ")
21 disp(" A*beta = -16*10^9[-omega^2*4*10^3 - j*omega
22 *[4*10^6-omega^2]] / [-[omega^2*4*10^3]-j*omega

```



```

    *[4*10^6 - omega^2]] * [- omega^2*4*10^3 - j*omega
    *[4*10^6 - omega^2]]")
12 disp("Using (a-b)(a+b) = a^2 - b^2 in the
    denominator,")
13 disp(" A*beta = 16*10^9[omega^2*4*10^3+j*omega
    *[4*10^6 - omega^2]] / [-omega^2*4*10^3]^2 - [j*
    omega*[4*10^6 - omega^2]^2")
14 disp(" A*beta = 16*10^9[omega^2*4*10^3+j*omega
    *[4*10^6 - omega^2]] / 16*10^6*omega^4 + omega
    ^2(4*10^6 - omega^2)^2")
15 disp("Now to have A*beta = 0 degree, the imaginary
    part of A*beta must be zero. This is possible
    when,")
16 disp(" Therefore, omega*(4*10^6 - omega^2) = 0")
17 disp(" Therefore, omega = 0 or 4*10^6 - omega^2 =
    0")
18 disp(" Therefore, omega^2 = 4*10^6
    Neglecting zero value of frequency")
19 disp(" Therefore, omega = 2*10^3 rad/sec")
20 disp("At this frequency |A*beta| can be obtained as,
    ")
21 disp(" |A*beta| = 16*10^9[4*10^3*omega^2] /
    16*10^6*omega^4+omega^2[4*10^6 - omega^2]^2
    at omega = 2*10^3")
22 ab=(2.56*10^20)/(2.56*10^20)
23 disp(ab," |A*beta| =")
24 disp(" Therefore, At omega = 2*10^3 rad/sec, A*beta
    = 0 degree as imaginary part is zero while |A*
    beta| = 1. Thus Barkhausen Criterion is satisfied
    .")
25 disp("The frequency at which circuit will oscillate
    is the value of omega for which |A*beta| = 1 and
    A*beta = 0 degree at the same time")
26 disp(" i.e. omega = 2*10^3 rad/sec")
27 disp(" But omega = 2*pi*f")
28 f=(2*10^3)/(2*pi) // in Hz
29 format(9)
30 disp(f," Therefore, f(in Hz) = omega / 2pi =")

```

Scilab code Exa 4.13 minimum and maximum values of R2

```
1 //Example 4.13
2 clc
3 disp("The frequency of the oscillator is given by,")
4 disp(" f = 1 / 2*pi*sqrt(R1*R2*C1*C2)")
5 disp(" For f = 20 kHz,")
6 r2=(1/(4*(%pi^2)*((20*10^3)^2)*(10*10^3)
      *((0.001*10^-6)^2)))*10^-3
7 format(5)
8 disp(r2," Therefore , R2(in k-ohm) =")
9 disp(" For f = 70 kHz,")
10 r2=(1/(4*(%pi^2)*((70*10^3)^2)*(10*10^3)
      *((0.001*10^-6)^2)))*10^-3
11 format(6)
12 disp(r2," Therefore , R2(in k-ohm) =")
13 disp("So minimum value of R2 is 0.517 k-ohm while
      the maximum value of R2 is 6.33 k-ohm")
```

Scilab code Exa 4.14 frequency of oscillation and minimum hfe

```
1 //Example 4.14.
2 clc
3 disp("R = 6 k-ohm, C = 1500 pF, R_C = 18 k-ohm")
4 k=18/6
5 disp(k,"Now K = R_C / R =")
6 disp(" Therefore , f = 1 / 2*pi*R*C*sqrt(6+4K)")
7 f=(1/(2*%pi*(6*10^3)*(1500*10^-12)*sqrt(6+12)))
      *10^-3 // in kHz
8 format(6)
9 disp(f," f(in kHz) =")
```

```

10 hfe=(4*3)+23+(29/3)
11 disp(hfe," (h_fe)min = 4K + 23 + 29/K =")

```

Scilab code Exa 4.15 range of frequency of oscillation

```

1 //Example 4.15.
2 clc
3 format(6)
4 disp("For a Wien bridge oscillator,")
5 disp(" f = 1 / 2*pi*R*C")
6 fm=(1/(2*pi*(100*10^3)*(50*10^-12)))*10^-3 // in
    kHz
7 disp(fm,"Therefore, f_max(in kHz) =")
8 fmi=(1/(2*pi*(100*10^3)*(500*10^-12)))*10^-3
9 disp(fmi,"and f_min(in kHz) =")
10 fn=31.83+50
11 disp(fn,"Now f_new(in kHz) = f_max + 50*10^3 =")
12 disp("The corresponding R = R'' with an additional
    resistance R_x in parallel")
13 disp("Therefore, f = 1 / 2*pi*R''*C")
14 r=(1/(2*pi*(50*10^-12)*(81.83*10^3)))*10^-3 // in
    k-ohm
15 disp(r,"Therefore, R''(in k-ohm) =")
16 rx=1/((1/38.89)-(1/100)) // in k-ohm
17 disp("Therefore, R'' = R*R_x / R+R_x")
18 disp(rx,"Therefore, R_x(in k-ohm) =
    in parallel with 100 k-ohm")

```

Scilab code Exa 4.16 frequency of oscillation

```

1 //Example 4.16.
2 clc
3 format(6)

```

```

4 disp("For a Hartley oscillator the frequency is
      given by,")
5 disp("  f = 1 / 2*pi*sqrt(L_eq*C)          where L_eq
      = L1+L2")
6 leq=20+5 // in mH
7 disp(leq,"Therefore ,  L_eq(in mH) = 20+5 =")
8 f=(1/(2*%pi*sqrt(25*500*10^-15)))*10^-3 // in kHz
9 disp(f,"Therefore ,  f(in kHz) =")

```

Scilab code Exa 4.17 gain of the transistor

```

1 //Example 14.7
2 clc
3 disp("For a Hartley oscillator ,")
4 disp("  f = 1 / 2*pi*sqrt(L_eq*C)          where L_eq
      = L1 + L2 + 2M")
5 leq=(1/(4*(%pi^2)*((168*10^3)^2)*(50*10^-12)))*10^3
      // in mH
6 format(6)
7 disp(leq,"Therefore ,  L_eq(in mH) =")
8 l2=((17.95*10^-3)-(15*10^-3)-(5*10^-6))*10^3 // in
      mH
9 disp(l2,"Therefore ,  L2(in mH) =")
10 hfe=((15*10^-3)+(5*10^-6))/((2.945*10^-3)+(5*10^-6))
11 format(5)
12 disp(hfe,"Now      h_fe = L1+M / L2+M =")

```

Scilab code Exa 4.18 new frequency and inductance

```

1 //Example 4.18
2 clc
3 disp("For a Colpitts oscillator ,")
4 disp("  f = 1 / 2*pi*sqrt(L*C_eq)")

```

```

5 disp(" where C_eq = C1*C2 / C1+C2 but C1 = C2 =
      0.001 uF")
6 ceq=((0.001*10^-6)^2)/(2*(0.001*10^-6)) // in F
7 format(7)
8 disp(ceq," Therefore , C_eq(in F) =")
9 disp(" L = 5*10^-6 H")
10 f=(1/(2*%pi*sqrt(25*10^-16)))*10^-6 // in MHz
11 format(6)
12 disp(f," Therefore , f(in MHz) =")
13 disp("Now L is doubled i.e. 10 uH")
14 f1=(1/(2*%pi*sqrt(50*10^-16)))*10^-6 // in MHz
15 format(5)
16 disp(f1," Therefore , f(in MHz) =")
17 nf= 2*3.183
18 format(6)
19 disp(nf,"New frequency(in MHz) = 2*3.183 =")
20 l=(1/(4*(%pi^2)*((6.366*10^6)^2)*(5*10^-10)))*10^6
    // in uH
21 format(5)
22 disp(l," Therefore , L(in uH) =")

```

Scilab code Exa 4.19 new frequency of oscillation

```

1 //Example 4.19
2 clc
3 disp("For a Clapp oscillator ,")
4 disp(" f = 1 / 2*pi*sqrt(L*C3)")
5 disp(" where C3 = 63 pF")
6 f=(1/(2*%pi*sqrt(315*10^-18)))*10^-6 // in MHz
7 format(6)
8 disp(f," Therefore , f(in MHz) =")

```

Scilab code Exa 4.20 R and hfe

```

1 //Example 4.20
2 clc
3 disp("Referring to equation(1) of section 4.5.3, the
      input impedance is given by,")
4 disp("R' ' _i = R1 || R2 || h_ie")
5 disp("Now R1 = 25 k-ohm, R2 = 47 k-ohm, and h_ie
      = 2 k-ohm")
6 format(7)
7 ri=(25*47*2)/((47*2)+(25*2)+(25*47)) // in k-ohm
8 disp(ri,"Therefore, R' ' _i(in k-ohm) =")
9 disp(" K = R_C / R")
10 disp("Now R_C = 10 k-ohm ... given")
11 disp("Now f = 1 / 2*pi*R*C*sqrt(6+4K)")
12 disp("Therefore, R*sqrt(6+4K) = 31830.989")
13 disp("Now K = R_C / R = 10*10^3 / R")
14 disp("Therefore, R*sqrt(6+(40*10*10^3/R)) =
      31830.989")
15 disp("Therefore, R^2*(6+(40*10*10^3/R)) =
      (31830.989)^2")
16 R=poly(0,'R')
17 p1=6*R^2+(40*10^3)*R-(31830.989)^2
18 t1=roots(p1)
19 ans1=t1(1)
20 format(6)
21 disp((-ans1)*10^-3,"Therefore, R(in k-ohm)=
      Neglecting negative value")
22 k=10/16.74
23 format(7)
24 disp(k,"Therefore, K = R_C / R =")
25 disp("Therefore, h_fe >= 4K + 23 + 29/K")
26 hfe=(4*0.5973)+23+(29/0.5973)
27 format(6)
28 disp(hfe," h_fe >=")

```

Scilab code Exa 4.21 component values of wien bridge

```

1 //Example 4.21
2 clc
3 disp("The frequency is given by,")
4 disp("  f = 1 / 2*pi*R*C")
5 disp("Let the resistance value to be selected as,")
6 disp("  R1 = R2 = R = 50 k-ohm")
7 disp("  f = 1 / 2*pi*50*10^3*C")
8 f=(1/(2*%pi*(50*10^3)*100))*10^9 // in nF
9 format(6)
10 disp(f,"  f(in nF) =")
11 disp("and  f_max = 1 / 2*pi*50*10^3*C")
12 c=(1/(2*%pi*(50*10^3)*(10*10^3)))*10^9 // in pF
13 disp(c,"  C(in nF) =")
14 disp("Thus to vary the frequency from 100 Hz to 10
      kHz, the capacitor range should be selected as
      0.318 nF to 31.83 nF")
15 disp("Similarly keeping the capacitor value constant
      , the range of the resistance values can be
      obtained.")

```

Scilab code Exa 4.22 values of C2 and new frequency of oscillation

```

1 //Example 4.22
2 clc
3 disp("  f = 2.5 MHz,  L = 10 uH,  C1 = 0.02 uF")
4 disp("For Colpitts oscillator , the frequency is
      given by,")
5 disp("  f = 1 / 2pi*sqrt(L*C_eq)")
6 ceq=(1/(4*(%pi^2)*((2.5*10^6)^2)*(10*10^-6)))*10^12
      // in pF
7 format(8)
8 disp(ceq," Therefore ,  C_eq(in pF) =")
9 disp("(i)But  C_eq = C1*C2 / C1+C2")
10 c2=((0.02*10^-6)/49.348)*10^9 // in nF
11 format(7)

```

```

12 disp(c2," Therefore , C2(in nF) =") // answer in
    textbook is wrong
13 disp("(ii) L = 2*10 = 20 uH")
14 disp("and C_eq = 405.284 pF")
15 f=(1/(2*pi*sqrt(20*405.284*10^-18)))*10^-6 // in
    MHz
16 disp(f," f(in MHz) = 1 / 2*pi*sqrt(L*C_eq) =")

```

Scilab code Exa 4.23 series and parallel resonant frequency and Qfactor

```

1 //Example 4.23.
2 clc
3 f=(1/(2*pi*sqrt(0.33*0.065*10^-12)))*10^-6 // in
    MHz
4 format(6)
5 disp(f,"(i) f(in MHz) = 1 / 2*pi*sqrt(L*C) =")
6 ceq=0.065/1.065 // in pF
7 disp(ceq,"(ii) C_eq(in pF) = C*C_M / C+C_M =")
8 fp=(1/(2*pi*sqrt(0.33*0.061*10^-12)))*10^-6 // in
    MHz
9 disp(fp,"(i) f_p(in MHz) = 1 / 2*pi*sqrt(L*C_eq) =")
10 pi=((1.121-1.087)/1.087)*100 // in percentage
11 disp(pi,"(iii) % increase =")
12 q=(2*pi*1.087*0.33*10^6)/(5.5*10^3)
13 format(8)
14 disp(q,"(iv) Q = omega_x*L / R = 2*pi*f_s*L / R =")

```

Scilab code Exa 4.24 change in frequency and trimmer capacitance

```

1 //Examble 4.24
2 clc

```



```

3 disp("(i) Assume one particular coupling direction
   for which,")
4 disp(" L_eq = L1 + L2 + 2M = 0.25 mH")
5 format(8)
6 f=(1/(2*pi*sqrt(0.25*100*10^-15)))*10^-6 // in MHz
7 disp(f,"Therefore, f(in MHz) = 1 / 2*pi*sqrt(L_eq*C
   ) =")
8 disp("Let the direction of coupling is reversed,")
9 disp(" L_eq = L1 + L2 - 2M = 0.15 mH")
10 fd=(1/(2*pi*sqrt(0.15*100*10^-15)))*10^-6 // in
   MHz
11 format(7)
12 disp(fd,"Therefore, f''(in MHz) = 1 / 2*pi*sqrt(
   L_eq*C) =")
13 pc=((1.2994-1.00658)/1.00658)*100 // in percentage
14 format(6)
15 disp(pc,"Therefore, % change = f''-f/f * 100 =")
16 disp("(ii) Let us assume direction of coupling such
   that,")
17 disp(" L_eq = L1 + L2 + 2M = 0.25 mH")
18 disp(" C_t = Trim capacitor = 100 pF")
19 disp("Therefore, C_eq = C*C_t / C+C_t = 50 pF")
20 f1=(1/(2*pi*sqrt(0.25*50*10^-15)))*10^-6 // in MHz
21 format(7)
22 disp(f1,"Therefore, f = 1 / 2*pi*sqrt(L_eq*C_eq) =")
   )
23 disp("If now direction of coupling is reversed,")
24 disp(" L_eq = L1 + L2 - 2M = 0.15 mH")
25 f2=(1/(2*pi*sqrt(0.15*50*10^-15)))*10^-6 // in MHz
26 format(8)
27 disp(f2,"Therefore, f'' = 1 / 2*pi*sqrt(L_eq*C_eq)
   =")
28 pc1=((1.83776-1.4235)/1.4235)*100
29 format(7)
30 disp(pc1,"Therefore, % change = f''-f/f * 100 =")

```

Scilab code Exa 4.25 design RC phase shift oscillator

```

1 //Example 4.25
2 clc
3 disp("For RC phase shift oscillator ,")
4 disp("  hfe = 4K + 23 + 29/K          ... given
      hfe = 150")
5 disp(" Therefore , 150 = 4K + 23 + 29/K")
6 disp(" Therefore , 4K2 - 127K + 29 = 0")
7 K=poly(0, 'K')
8 p1=4*K2-127*K+29
9 t1=roots(p1)
10 format(6)
11 disp(t1," Therefore , K =")
12 disp("  f = 1 / 2*pi*R*C*sqrt(6+4K)      ... given
      f = 5 kHz")
13 disp(" Therefore , Choose C = 100 pF")
14 r=(1/(2*%pi*(1000*10-12)*(5*103)*sqrt(6+(4*0.23))))
      )*10-3 // in k-ohm
15 format(3)
16 disp(r," Therefore , R(in k-ohm) =")
17 disp("  K = RC / R i.e. RC = KR = 2.7 k-ohm")
18 disp(" Neglecting effect of biasing resistances
      assuming them to be large and selecting
      transistor with hie = 2 k-ohm")
19 disp("  R' ' i = hie = 2 k-ohm")
20 disp(" Therefore , Last resistance in phase network")
21 r3=12-2
22 disp(r3,"  R3 = R - R' ' i =")
23 disp(" Using the back to back connected zener diodes
      of 9.3 V (Vz) each at the output of emitter
      follower and using this at the output of the
      oscillator , the output amplitude can be
      controlled to 10 V i.e. 20 V peak to peak. The

```

```

    zener diode 9.3V and forward biased diode of 0.7
    V gives total 10 V")
24 disp("The designed circuit is shown in fig.4.49")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 4.26 range of capacitor

```

1 //Example 4.26
2 clc
3 disp(" L1 = 20 uH, L2 = 2 mH")
4 leq=((20*10^-6)+(2*10^-3))*10^3 // in mH
5 format(7)
6 disp(leq," Therefore , L_eq(in mH) = L1 + L2 =")
7 disp(" For f = f_max = 2.5 MHz")
8 disp(" f = 1 / 2*pi*sqrt(C*L_eq)")
9 c=(1/(4*(%pi^2)*((2.5*10^6)^2)*(2.002*10^-3)))*10^12
    // in pF
10 format(7)
11 disp(c," Therefore , C(in pF) =")
12 disp(" For f = f_min = 1 MHz")
13 disp(" f = 1 / 2*pi*sqrt(C*L_eq)")
14 c1=(1/(4*(%pi^2)*((1*10^6)^2)*(2.002*10^-3)))*10^12
    // in pF
15 format(8)
16 disp(c1," Therefore , C(in pF) =")
17 disp("Thus C must be varied from 2.0244 pF to
    12.6525 pF")

```

Scilab code Exa 4.27 change in frequency of oscillation

```

1 //Example 4.27

```

```

2  clc
3  ceq=((0.02*12*10^-24)/(12.02*10^-12))*10^12 // in
    pF
4  format(8)
5  disp(ceq," C_eq(in pF) = C1*C2 / C1+C2 =")
6  fs=(1/(2*pi*sqrt(50*0.02*10^-15)))*10^-6 // in MHz
7  format(7)
8  disp(fs," Therefore , f_s(in MHz) = 1 / 2*pi*sqrt(L*
    C1) =")
9  fp=(1/(2*pi*sqrt(50*0.01996*10^-15)))*10^-6 // in
    MHz
10 format(7)
11 disp(fp," Therefore , f_p(in MHz) = 1 / 2*pi*sqrt(L*
    C_eq) =")
12 disp("Let C_s = 5 pF connected across the crystal")
13 c2=12+5
14 disp(c2," Therefore , C''2(in pF) = C2 + C_x =")
15 format(10)
16 ceq1=0.019976
17 disp(ceq1," Therefore , C''_eq(in pF) = C1*C''2 / C1+
    C''2 =")
18 fp1=5.03588
19 disp(fp1," Therefore , f''_p(in MHz) = 1 / 2*pi*sqrt(
    L*C_eq) =")
20 disp("New C_x = 6 pF is connected then,")
21 c21=12+6
22 disp(c21," C''''2(in pF) = C2 + C_x =")
23 ceq2=0.0199778
24 disp(ceq2," Therefore , C''''_eq(in pF) = C1*C''''2 /
    C1+C''''2 =")
25 fp2=5.035716
26 disp(fp2," Therefore , f''''_p(in MHz) = 1 / 2*pi*
    sqrt(L*C''''_eq) =")
27 c=(5.03588-5.035716)*10^6
28 disp(c," Therefore , Change(in Hz) = f''_p - f''''_p
    =")

```

Chapter 5

Combinational Logic Circuits

Scilab code Exa 5.1 design a combinational logic circuit

```
1 //Example 5.1
2 clc
3 disp("Given problem specific that there are three
    input variables and one output variable. We
    assign A, B and C letter symbols to three input
    variables and assign Y letter symbol to one
    output variable. The relationship between input
    variables and output variable can be tabulated as
    shown in truth table 5.1")
4 disp("  A    B    C    Y")
5 disp("  0    0    0    0")
6 disp("  0    0    1    0")
7 disp("  0    1    0    0")
8 disp("  0    1    1    1")
9 disp("  1    0    0    0")
10 disp("  1    0    1    1")
11 disp("  1    1    0    1")
12 disp("  1    1    1    1")
13 disp("Now we obtain the simplified Boolean
    expression for output variable Y using K-map
    simplification.")
```

```

14 disp("      BC      BC' '      B' 'C' '      B' 'C'")
15 disp("A      0      0      1      0")
16 disp("A' '      0      1      1      1")
17 disp(" Y = AC + BC + AB")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.2 design a circuit with control line C and data lines

```

1 //Example 5.2
2 clc
3 disp("The truth table for the given problem is as
      shown below.")
4 disp(" C      D3      D2      D1      Output")
5 disp(" 0      x      x      x      0")
6 disp(" 0      0      0      0      0")
7 disp(" 0      0      0      1      1")
8 disp(" 0      0      1      0      1")
9 disp(" 0      1      0      0      1")
10 disp("")
11 disp("K-map simplification")
12 disp("      D1' 'D2' '      D1' 'D2      D1D2      D1D2
      ' '")
13 disp("C' 'D3' '      0      0      0      0")
14 disp("C' 'D3      0      0      0      0")
15 disp("CD3      1      X      X      X")
16 disp("CD3' '      0      1      X      1")
17 disp("")
18 disp("Therefore ,      Y = CD3 + CD2 + CD1")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.3 design combinational circuit

```

1 //Example 5.3
2 clc
3 disp("Truth table")
4 disp("      Input      Output")
5 disp("  Decimal Digit      Digit 1      Digit 0")
6 disp("  A  B  C  D      Y7  Y6  Y5  Y4  Y3  Y2  Y1
7 disp("  0  0  0  0      0  0  0  0  0  0  0
8 disp("  0  0  0  1      0  0  0  0  0  1  0
9 disp("  0  0  1  0      0  0  0  1  0  0  0
10 disp("  0  0  1  1      0  0  0  1  0  1  0
11 disp("  0  1  0  0      0  0  1  0  0  0  0
12 disp("  0  1  0  1      0  0  1  0  0  1  0
13 disp("  0  1  1  0      0  0  1  1  0  0  0
14 disp("  0  1  1  1      0  0  1  1  0  1  0
15 disp("  1  0  0  0      0  1  0  0  0  0  0
16 disp("  1  0  0  1      0  1  0  0  0  1  0
17 disp("")
18 disp("Here Y0 = D, Y1 = 0, Y2 = D, Y3 = 0, Y4 =
      C, Y5 = B, Y6 = A and Y7 = 0. Therefore, the
      given circuit can be obtained from the input
      lines without using any logic gates")

```

Scilab code Exa 5.4 design logic circuit

```
1 //Example 5.4
2 clc
3 disp("Let us consider D for Door, I for ignition, L
      for Light. Then conditions to activate the alarm
      are:")
4 disp("1. The headlights are ON while the ignition is
      OFF.")
5 disp("   i.e. L = 1, I = 0 and D may be anything.")
6 disp("2. The door is open while the ignition is ON")
7 disp("   i.e. D = 1, I = 1, L may be anything.")
8 disp("Also alarm will sound if logic circuit output
      is zero.")
9 disp("Therefore, output for above condition is zero
      and for rest of the condition it is 1 which is
      summarized in the following table.")
10 disp("  D    I    L    Y")
11 disp("  0    0    0    1")
12 disp("  X    0    1    0")
13 disp("  0    1    0    1")
14 disp("  0    1    1    1")
15 disp("  1    0    0    1")
16 disp("  1    1    X    0")
17 disp("Therefore, K-map for logic circuit.")
18 disp("      I''L''    I''L    IL    IL''")
19 disp("D''    1    0    1    1")
20 disp("D    1    0    0    0")
21 disp("Output = Y = I''L'' + D''I")
22 disp("As AND-OR logic can be directly replaced by
      NAND-NAND, logic circuit using only NAND gates is
      as shown in fig.5.9 and fig.5.10")
```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 5.5 design circuit to detect invalid BCD number

```
1 //Example 5.5
2 clc
3 disp("Truth table")
4 disp("  Dec    A  B  C  D    Output Y")
5 disp("  0     0  0  0  0    0")
6 disp("  1     0  0  0  1    0")
7 disp("  2     0  0  1  0    0")
8 disp("  3     0  0  1  1    0")
9 disp("  4     0  1  0  0    0")
10 disp("  5     0  1  0  1    0")
11 disp("  6     0  1  1  0    0")
12 disp("  7     0  1  1  1    0")
13 disp("  8     1  0  0  0    0")
14 disp("  9     1  0  0  1    0")
15 disp(" 10     1  0  1  0    1")
16 disp(" 11     1  0  1  1    1")
17 disp(" 12     1  1  0  0    1")
18 disp(" 13     1  1  0  1    1")
19 disp(" 14     1  1  1  0    1")
20 disp(" 15     1  1  1  1    1")
21 disp("")
22 disp("K-map simplification")
23 disp("      C'D'   C'D    CD    CD' ")
24 disp("A'B'   0      0      0      0")
25 disp("A'B    0      0      0      0")
```

```

26 disp("AB      1      1      1      1")
27 disp("AB' ,    0      0      1      1")
28 disp("  Y = AB + AC")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.6 design two level combinational circuit

```

1 //Example 5.6
2 clc
3 disp("  Input 1 -> Pressure in fuel tank")
4 disp("  Input 2 -> Pressure in oxidizer tank")
5 disp("  Input = 1 Indicates pressure is equal to or
      above the required minimum")
6 disp("          = 0 Otherwise")
7 disp("  Input 3 -> From timer")
8 disp("if input 3 = 1 Indicates that there are less
      than or exactly 10 minutes for lift off")
9 disp("          = 0 Otherwise")
10 disp("  Output -> Panel light , if light goes on
      then")
11 disp("  Output = 1")
12 disp("else Output = 0")
13 disp("")
14 disp("Truth table")
15 disp("Let input 1 = A,  input 2 = B,  input 3 = C.")
16 disp("  Inputs      Output")
17 disp("  A      B      C      Y")
18 disp("  0      0      0      1")
19 disp("  0      0      1      0")
20 disp("  0      1      0      1")
21 disp("  0      1      1      0")
22 disp("  1      0      0      1")
23 disp("  1      0      1      0")

```

```

24 disp(" 1      1      0      0")
25 disp(" 1      1      1      1")
26 disp("")
27 disp("K-map simplification")
28 disp("      B' 'C' '      B' 'C      BC      BC' ' '")
29 disp("A' '      1      0      0      1")
30 disp("A      1      0      1      0")
31 disp(" Y = ABC + A' 'B' 'C' ' + B' 'C' ' '")
32 disp("      = ABC + C' '(B' '+A' 'B)")
33 disp("      = ABC + C' '(B' '+A' ' '      [A' '+A' 'B =
      A + B] ")
34 disp("      = ABC + C' '(A' 'B' ' ' ")
35 disp("      = A' 'B' ' XOR C' ' ")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.7 design 32 to 1 multiplexer

```

1 //Example 5.7
2 clc
3 disp(" Fig. 5.20 shows a 32 to 1 multiplexer using 74
      LS150 ICs. ")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.8 design a 32 to 1 multiplexer

```

1 //Example 5.8
2 clc
3 disp(" Fig. 5.21 shows a 32 to 1 multiplexer using
      four 8 to 1 multiplexers and 2 to 4 decoder.. ")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.9 implement boolean function using 8to1 multiplexer

```
1 //Example 5.9
2 clc
3 disp("The function can be implemented with a 8 to 1
    multiplexer , as shown in fig. 5.22. Three
    variables A, B and C are applied to the select
    lines . The minterms to be included (1, 3, 5 and
    6) are chosen by making their corresponding input
    lines equal to 1. Mintems 0, 2, 4 and 7 are not
    included by making their input lines equal to 0."
    )
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.10 implement boolean function using 4to1 multiplexer

```
1 //Example 5.10
2 clc
3 disp(" Fig. 5.23 shows the implementation of function
    with 4 to 1 multiplexer. Two of the variables , B
    and C, are applied to the selection lines. B is
    connected to S1 and C is connected to S0. The
    inputs for multiplexer are derived from the
    implementation table.")
4 disp("Truth table")
5 disp("Minterm      A  B  C      F")
```

```

6 disp("    0      0 0 0  0")
7 disp("    1      0 0 1  1")
8 disp("    2      0 1 0  0")
9 disp("    3      0 1 1  1")
10 disp("    4      1 0 0  0")
11 disp("    5      1 0 1  1")
12 disp("    6      1 1 0  1")
13 disp("    7      1 1 1  0")
14 disp("")
15 disp("Implementation table")
16 disp("      D0 D1 D2 D3")
17 disp("A' '  0  1  2  3  Row 1")
18 disp("A   4  5  6  7  Row 2")
19 disp("      0  1  A  A' ")
20 disp("")
21 disp("As shown in fig. 5.23(c) the implementation
      table is nothing but the list of the inputs of
      the multiplexers and under them list of all the
      minterms in two rows. The first row lists all
      those minterms where A is complemented, and the
      second row lists all the minterms with A
      uncomplemented. The minterms given in the
      function are circled and then each column is
      inserted separately as follows.")
22 disp("1. If the two minterms in a column are not
      circled, 0 is applied to the corresponding
      multiplexer input (see column 1).")
23 disp("2. If the two minterms in a column are circled
      , 1 is applied to the corresponding multiplexer
      input (see column 2).")
24 disp("3. If the minterm in the second row is circled
      and minterms in the first row is not circled, A
      is applied to the corresponding multiplexer input
      (see column 3).")
25 disp("4. If the minterm in the first row is circled
      and minterm in the second row is not circled, A'
      is applied to the corresponding multiplexer
      input (see column 4).")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.11 implement boolean function using 8to1 MUX

```
1 //Example 5.11
2 clc
3 disp("Fig 5.25 shows the implementation of given
   Boolean function with 8:1 multiplexer.")
4 disp("Implementation table")
5 disp("      D0  D1  D2  D3  D4  D5  D6  D7")
6 disp("A' '   0   1   2   3   4   5   6   7")
7 disp("A      8   9  10  11  12  13  14  15")
8 disp("      1   1   0   A' '   A' '   0   0   A")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.12 implement boolean function using 4to1 MUX

```
1 //Example 5.12
2 clc
3 disp("The function has four variables. To implement
   this function we require 8 : 1 multiplexer. i.e.,
   two 4 : 1 multiplexers. We have already seen how
   to construct 8 : 1 multiplexer using two 4 : 1
   multiplexers. The same concept is used here to
   implement given Boolean function.")
4 disp("")
5 disp("Implementation table")
6 disp("      D0  D1  D2  D3  D4  D5  D6  D7")
```

```

7 disp("A' ' 0 1 2 3 4 5 6 7")
8 disp("A 8 9 10 11 12 13 14 15")
9 disp(" A' ' 1 A' ' 0 1 0 1 0")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.13 implement boolean function using 8to1 MUX

```

1 //Example 5.13
2 clc
3 disp("The given Boolean expression is not in
      standard SOP form. Let us first convert this in
      standard form.")
4 disp(" F(A, B, C, D) = A' 'BD' '(C+C' ') + ACD(B+B' ')
      + B' 'CD(A+A' ') + A' 'C' 'D(B+B' ')")
5 disp("          = A' 'BCD' ' + A' 'BC' 'D' ' + ABCD
      + AB' 'CD + AB' 'CD + A' 'B' 'CD + A' 'BC' 'D + A' 'B' '
      C' 'D")
6 disp("          = A' 'BCD' ' + A' 'BC' 'D' ' + ABCD
      + AB' 'CD + A' 'B' 'CD + A' 'BC' 'D + A' 'B' 'C' 'D")
7 disp("")
8 disp("The truth table for this standard SOP form can
      be given as")
9 disp(" No.  Minterms    A  B  C  D  Y")
10 disp(" 0                0  0  0  0  0")
11 disp(" 1    A' 'B' 'C' 'D    0  0  0  1  1")
12 disp(" 2                0  0  1  0  0")
13 disp(" 3    A' 'B' 'CD        0  0  1  1  1")
14 disp(" 4    A' 'BC' 'D' '      0  1  0  0  1")
15 disp(" 5    A' 'BC' 'D         0  1  0  1  1")
16 disp(" 6    A' 'BCD' '         0  1  1  0  1")
17 disp(" 7                0  1  1  1  0")
18 disp(" 8                1  0  0  0  0")
19 disp(" 9                1  0  0  1  0")

```

```

20 disp(" 10          1 0 1 0 0")
21 disp(" 11    AB' 'CD    1 0 1 1 1")
22 disp(" 12          1 1 0 0 0")
23 disp(" 13          1 1 0 1 0")
24 disp(" 14          1 1 1 0 0")
25 disp(" 15    ABCD      1 1 1 1 1")
26 disp("")
27 disp("From the truth table Boolean function can be
      implemented using 8 : 1 multiplexer as follows :")
28 disp("Implementation table :")
29 disp("      D0  D1  D2  D3  D4  D5  D6  D7")
30 disp("A' '  0   1   2   3   4   5   6   7")
31 disp("A      8   9   10  11  12  13  14  15")
32 disp("      0   A' '  0   1   A' '  A' '  A' '  A")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.14 implement boolean function using 8to1 MUX

```

1 //Example 5.14.
2 clc
3 disp("Here, instead of minterms, maxterms are
      specified. Thus, we have to circle maxterms which
      are not included in the Boolean function. Fig.
      5.28 shows the implementation of Boolean function
      with 8 : 1 multiplexer.")
4 disp("")
5 disp("Implementation table :")
6 disp("      D0  D1  D2  D3  D4  D5  D6  D7")
7 disp("A' '  0   1   2   3   4   5   6   7")
8 disp("A      8   9   10  11  12  13  14  15")
9 disp("      0   A' '  A' '  A  A' '  A  0  1")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.15 implement boolean function using 8to1 MUX

```
1 //Example 5.15
2 clc
3 disp("In the given Boolean function three don't
   care conditions are also specified. We know that
   don't care conditions can be treated as either 0
   s or 1s. Fig. 5.29 shows the implementation of
   given Boolean function using 8 : 1 multiplexer.")
4 disp("")
5 disp("Implementation table :")
6 disp("      D0  D1  D2  D3  D4  D5  D6  D7")
7 disp("A''   0   1   2   3   4   5   6   7")
8 disp("A     8   9   10  11  12  13  14  15")
9 disp("      1   0   1   1   A   A   1   0")
10 disp("")
11 disp("In this example, by taking don't care
   conditions 8 and 14 we have eliminated A'' term
   and hence the inverter.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.16 determine boolean expression

```
1 //Example 15.6
2 clc
3 disp("      D''   D")
4 disp("D     0     1")
```

```

5 disp("0   2   3")
6 disp("0   4   5")
7 disp("1   6   7")
8 disp("D   8   9")
9 disp("1  10  11")
10 disp("D' 12  13")
11 disp("0  14  15")
12 disp("")
13 disp("Here, implementation table is listed for least
      significant bit i.e. D. The first column list
      all minterms with D is complementated and the
      second column lists all the minterms with D
      uncomplemented, as shown in fig. 5.30(a). Then
      according to data inputs given to the multiplexer
      minterms are circled applying following rules.")
14 disp("1. If multiplexer input is 0, don't circle
      any minterm in the corresponding row.")
15 disp("2. If multiplexer input 1, circle both the
      minterms in the corresponding row.")
16 disp("3. If multiplexer input is D, circle the
      minterm belongs to cloumn D in the corresponding
      row.")
17 disp("4. If multiplexer input is D'', circle the
      minterm belongs to column D'' in the
      corresponding row.")
18 disp("This is illustrated in fig. 5.30(b). Now
      circled minterms can be written to get Boolean
      expression as follows :")
19 disp(" Y = A' 'B' 'C' 'D + A' 'BCD' ' + A' 'BCD + AB' 'C' '
      D + AB' 'CD' ' + AB' 'CD + ABC' 'D' '")

```

Scilab code Exa 5.17 realize using 4 to 1 MUX

```

1 //Example 5.17
2 clc

```

```

3 disp("          D0  D1  D2  D3")
4 disp("w' 'x' ' '    0   1   2   3")
5 disp("w' 'x          4   5   6   7")
6 disp("wx' '          8   9  10  11")
7 disp("wx           12  13  14  15")
8 disp("")
9 disp("D0 = w' 'x + wx' ' = w XOR x")
10 disp("D1 = w' 'x' ' ' + wx' ' = x' '")
11 disp("D2 = w' 'x + wx' ' = w XOR x")
12 disp("D3 = w' 'x + wx' ' + wx = x + wx' ' = w + x")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.18 design 1 to 8 DEMUX

```

1 //Example 5.18
2 clc
3 disp("The cascading of demultiplexers is similar to
      the cascading of decoder. Fig. 5.33 shows
      cascading of two 1 : 4 demultiplexers to form 1 :
      8 demultiplexer.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.19 implement full subtractor

```

1 //Example 5.19
2 clc
3 disp("Let us see the truth table of full subtractor.
      ")
4 disp("  A  B  Bin  D  Bout")

```

```

5 disp(" 0 0 0 0 0")
6 disp(" 0 0 1 1 1")
7 disp(" 0 1 0 1 1")
8 disp(" 0 1 1 0 1")
9 disp(" 1 0 0 1 0")
10 disp(" 1 0 1 0 0")
11 disp(" 1 1 0 0 0")
12 disp(" 1 1 1 1 1")
13 disp("")
14 disp("For full subtractor difference D function can
      be written as  $D = f(A, B, C) = \text{summation } m(1, 2, 4, 7)$  and Bout function can be written as")
15 disp("      Bout =  $F(A, B, C) = \text{summation } m(1, 2, 3, 7)$ ")
16 disp("With Din input 1, demultiplexer gives minterms
      at the output so by logically ORing required
      minterms we can implement Boolean functions for
      full subtractor. Fig. 5.34 shows the
      implementation of full subtractor using
      demultiplexer.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.20 construst 1 to 32 DEMUX

```

1 //Exmaple 5.20
2 clc
3 disp("The fig. 5.37 shows the implementation of 1 to
      32 demultiplexer using two 74X154 ICs. Here, the
      most significant bit of select signal (A4) is
      used to enable either upper 1 to 16 demultiplexer
      or lower 1 to 16 demultiplexer. The data input
      and other select signals are connected parallel
      to both the demultiplexer ICs. When  $A4 = 0$ , upper

```

demultiplexer is enabled and the data input is routed to the output corresponds to the status of A0 A1 A2 and A3 lines. When A4 = 1, lower multiplexer is enabled and the data input is routed to the output corresponds to the status of A0 A1 A2 and A3 lines.”)

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.21 design 3 to 8 decoder

```

1 //Example 5.21
2 clc
3 disp(" Fig. 5.40 shows 3 to 8 line decoder. Here, 3
  inputs are decoded into eight outputs, each
  output represent one of the minterms of the 3
  input variables. The three inverters provide the
  complement of the inputs, and each one of the
  eight AND gates generates one of the minterms.
  Enable input is provided to activate decoded
  output based on data inputs A, B and C. The table
  shows the truth table for 3 to 8 decoder.")
4 disp("")
5 disp("Truth table for a 3 to 8 decoder")
6 disp("  Inputs      |           Outputs")
7 disp("EN  A  B  C  |  Y7  Y6  Y5  Y4  Y3  Y2  Y1  Y0
  ")
8 disp("0   X  X  X  |  0   0   0   0   0   0   0   0")
9 disp("1   0  0  0  |  0   0   0   0   0   0   0   1")
10 disp("1   0  0  1  |  0   0   0   0   0   0   1   0")
11 disp("1   0  1  0  |  0   0   0   0   0   1   0   0")

```

```

    )
12 disp("1  0  1  1  |  0  0  0  0  1  0  0  0")
    )
13 disp("1  1  0  0  |  0  0  0  1  0  0  0  0")
    )
14 disp("1  1  0  1  |  0  0  1  0  0  0  0  0")
    )
15 disp("1  1  1  0  |  0  1  0  0  0  0  0  0")
    )
16 disp("1  1  1  1  |  1  0  0  0  0  0  0  0")
    )

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.22 design 5 to 32 decoder

```

1 //Example 5.22
2 clc
3 disp("The Fig. 5.45 shows the construction of 5 to
      32 decoder using four 74LS138s and half 74LS139.
      The half section of 74LS139 IC used as a 2 to 4
      decoder to decode the two higher order inputs , D
      and E. The four outputs of this decoder are used
      to enable one of the four 3 to 8 decoders. The
      three lower inputs A, B and C are connected in
      parallel to four 3 to 8 decoders. This means that
      the same output pin of each of the four 3 to 8
      decoders is selected but only one is enable. The
      remaining enables signals of four 3 to 8 decoders
      ICs are connected in parallel to construct
      enable signals for 5 to 32 decoder.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.23 design 4 line to 16 line decoder

```
1 //Example 5.23
2 clc
3 disp("4 line to 16 line decoder using 1 line to 4
   line decoder")
4 disp("As shown in fig. 5.46 five numbers of 2 : 4
   decoder are required to design 4 : 16 decoder.
   Decoder 1 is used to enable one of the decoder 2,
   3, 4 and 5. Inputs of first decoder are the A
   and B MSB inputs of 4 : 16 decoder. The inputs of
   decoder are connected together forming C and D
   LSB inputs of 4 : 16 decoder.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.24 implement using 74LS138

```
1 //Example 5.24
2 clc
3 disp("In this example, we use IC 74LS138, 3 : 8
   decoder to implement multiple output function.
   The outputs of 74LS138 are active low, therefore,
   SOP function (function F1) can be implemented
   using NAND gate and POS function (function F2)
   can be implemented using AND gate, as shown in
   fig.5.50")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.25 implement full subtractor

```
1 //Example 5.25
2 clc
3 disp("The truth table for full subtractor is as
      shown below")
4 disp("")
5 disp(" Inputs      Outputs")
6 disp("A  B  Bin    D  Bout")
7 disp("0  0  0     0  0")
8 disp("0  0  1     1  1")
9 disp("0  1  0     1  1")
10 disp("0  1  1     0  1")
11 disp("1  0  0     1  0")
12 disp("1  0  1     0  0")
13 disp("1  1  0     0  0")
14 disp("1  1  1     1  1")
15 disp("")
16 disp("Implementation of full subtractor using 3 : 8
      decoder is shown in fig. 5.51")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.26 implement gray to binary code converter

```
1 //Example 5.26
2 clc
```



```

3 disp("The truth table for 3-bit binary to gray code
      converter is as shown below")
4 disp("")
5 disp("A  B  C      G2  G1  G0")
6 disp("0  0  0      0   0   0")
7 disp("0  0  1      0   0   1")
8 disp("0  1  0      0   1   1")
9 disp("0  1  1      0   1   0")
10 disp("1  0  0      1   1   0")
11 disp("1  0  1      1   1   1")
12 disp("1  1  0      1   0   1")
13 disp("1  1  1      1   0   0")
14 disp("")
15 disp("The fig. 5.52 shows the implementation of 3-
      bit binary to gray code converter using 3:8
      decoder. As outputs of 74138 are active low we
      have to use NAND gate instead of OR gate. The
      active low output from the decoder forces output(
      s) of connected NAND gate(s) to become HIGH, thus
      implementing the function.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.27 design 2 bit comparator

```

1 //Example 5.27
2 clc
3 disp("")
4 disp("A1  A0  B1  B0      A>B  A=B  A<B")
5 disp("0   0   0   0      0     0    0")
6 disp("0   0   0   1      0     0    1")
7 disp("0   0   1   0      0     0    1")
8 disp("0   0   1   1      0     0    1")
9 disp("0   1   0   0      1     0    0")

```

```

10 disp(" 0   1   0   1       0   1   0")
11 disp(" 0   1   1   0       0   0   1")
12 disp(" 0   1   1   1       0   0   1")
13 disp(" 1   0   0   0       1   0   0")
14 disp(" 1   0   0   1       1   0   0")
15 disp(" 1   0   1   0       0   1   0")
16 disp(" 1   0   1   1       0   0   1")
17 disp(" 1   1   0   0       1   0   0")
18 disp(" 1   1   0   1       1   0   0")
19 disp(" 1   1   1   0       1   0   0")
20 disp(" 1   1   1   1       0   1   0")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.28 design full adder circuit

```

1 //Example 5.28
2 clc
3 disp("Truth table for full adder is as shown below."
   )
4 disp("  Inputs          Outputs")
5 disp(" A   B   Cin      Carry  Sum")
6 disp(" 0   0   0         0     0")
7 disp(" 0   0   1         0     1")
8 disp(" 0   1   0         0     1")
9 disp(" 0   1   1         1     0")
10 disp(" 1   0   0         0     1")
11 disp(" 1   0   1         1     0")
12 disp(" 1   1   0         1     0")
13 disp(" 1   1   1         1     1")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 5.29 implement BCD to 7 segment decoder

```
1 //Example 5.29
2 clc
3 disp("BCD-to-common anode 7-segment decoder")
4 disp(" Digit      A  B  C  D      a  b  c  d  e  f  g")
5 disp("  0          0  0  0  0      0  0  0  0  0  0  1")
6 disp("  1          0  0  0  1      1  0  0  1  1  1  1")
7 disp("  2          0  0  1  0      0  0  1  0  0  1  0")
8 disp("  3          0  0  1  1      0  0  0  0  1  1  0")
9 disp("  4          0  1  0  0      1  0  0  0  1  1  0")
10 disp("  5          0  1  0  1      0  1  0  0  1  0  0")
11 disp("  6          0  1  1  0      0  1  0  0  0  0  0")
12 disp("  7          0  1  1  1      0  0  0  1  1  1  1")
13 disp("  8          1  0  0  0      0  0  0  0  0  0  0")
14 disp("  9          1  0  0  1      0  0  0  0  1  0  0")
```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 5.31 implement 32 input to 5 output encoder

```
1 //Example 5.31
2 clc
3 disp(" Fig.5.75 shows how four 74LS148 can be
      connected to accept 32 inputs and produce a 5-bit
```

encoded output, A0 – A4. EO'' signal is connected to the EI'' input of the next lower priority encoder and EI'' input of the highest priority encoder is grounded. Therefore, at any time only one encoder is enabled. Since, the A2 – A0 outputs of at the most one 74LS148 will be enabled at a time, the outputs of the individual 74LS148s can be ORed to produce A2 – A0. Likewise, the individual GS'' outputs can be combined in a 4 to 2 encoder to produce A4 and A3. The GS output for 32-bit encoder is produced by ORing GS'' outputs of all encoders. ")

This code can be downloaded from the website www.scilab.in

Chapter 6

Sequential Logic Circuits

Scilab code Exa 6.4 analyze the circuit

```
1 //Example 6.4
2 clc
3 disp("To analyze the circuit means to drive the
      truth table for it.")
4 disp("We have, D = Input XOR Q_n")
5 disp("")
6 disp("CLK      Input      Q_n      D = input XOR Q_n
      Q_n+1")
7 disp("down      0          0          0
      0")
8 disp("down      0          1          1
      1")
9 disp("down      1          0          1
      1")
10 disp("down     1          1          0
      0")
11 disp("")
12 disp("In the circuit fig. 6.53, output does not
      change when input is 0 and it toggles when input
      is 1. This is the characteristics of T flip-flop.
      Hence, the given circuit is T flip-flop")
```

```
constructed using D flip-flop.”)
```

This code can be downloaded from the website www.scilab.in

Chapter 7

Shift Registers

Scilab code Exa 7.1 determine number of flip flops needed

```
1 //Example 7.1
2 clc
3 disp("(i) A 6-bit binary number requires register
      with 6 flip-flops.")
4 disp("")
5 disp("(ii) (32)10 = (100000)2. The number of bits
      required to represent 32 in binary are six,
      therefore, 6 flip-flops are needed to construct a
      register capable of storing 32 decimal.")
6 disp("")
7 disp("(iii) (F)16 = (1111)2. The number of bits
      required to represent (F)16 in binary are four,
      therefore four flip-flops are needed to construct
      a register capable of storing (F)16")
8 disp("")
9 disp("(iv) (10)8 = (1000)2. The number of bits
      required to represent (10)8 in binary are four,
      therefore, four flip-flops are needed to
      construct a register capable of storing (10)8.")
```

Chapter 8

Counters

Scilab code Exa 8.1 count after 12 pulse

```
1 //Example 8.1
2 clc
3 disp("After 12 pulses , the count will be (1100)2 , i
   .e. 12 in decimal.")
```

Scilab code Exa 8.2 count in binary

```
1 //Example 8.2
2 clc
3 a=dec2bin(144)
4 disp(a,"decimal (144) =")
5 disp("Since counter is a 5-bit counter , it resets
   after 2^5 = 32 clock pulses.")
6 disp("Dividing 144 by 32 we get quotient 2 and
   remainder 6")
7 disp("Therefore , counter resets four times and then
   it counts remaining 16 clock pulses. Thus, the
   count will be binary (110000), i.e., 16 in
   decimal")
```

Scilab code Exa 8.4 draw logic diagram

```
1 //Example 8.4
2 clc
3 disp("When flip-flops are negatively edge triggered ,
      the Q output of previous stage is connected to
      the clock input of the next stage. Fig. 8.5 shows
      3-stage asynchronous counter with negative edge
      triggered flip-flops.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.5 output frequency

```
1 //example 8.5
2 clc
3 of=50/14
4 format(5)
5 disp(of,"Output frequency = 50 kHz / 14 =")
```

Scilab code Exa 8.6 maximum operating frequency

```
1 //Example 8.6
2 clc
3 disp("We know that MOD-32 uses five flip-flops. With
      t_pd = 50 ns, the f_max for ripple counter can
      be given as,")
4 fm=(1/(250*10^-9))*10^-6
```

```
5 disp(fm,"f_max(ripple) = ")
```

Scilab code Exa 8.8 design 4 bit up down ripple counter

```
1 //Example 8.8
2 clc
3 disp("The 4-bit counter needs four flip-flops. The
      circuit for 4-bit up/down ripple counter is
      similar to 3-bit up/down ripple counter except
      that 4-bit counter has one more flip-flop and its
      clock driving circuiting.")
4 disp(" The fig. 8.14 shows the 4-bit up/down ripple
      counter.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.9 design divide by 9 counter

```
1 //Example 8.9
2 clc
3 disp("Internal structure of 7492 is as shown in fig
      .8.16.")
4 disp("")
5 disp("The circuit diagram for divide-by-9 counter is
      as shown in fig.8.17.")
```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.10 design a divide by 128 counter

```
1 //Example 8.10
2 clc
3 disp("Since 128 = 16 x 8, a divide-by-16 counter
      followed by a divide-by-8 counter will become a
      divide-by-128 counter. IC 7493 is a 4-bit binary
      counter (i.e. mod-16 or divide-by-16), therefore,
      two IC packages will be required.")
4 disp("The circuit diagram is as shown in fig.8.18.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.11 design divide by 78 counter

```
1 //Example 8.11
2 clc
3 disp("Since 78 = 13 x 6, we have to use 7493 as mod-
      13 and 7492 as mod-6 counters. For the mod-13
      counter QD, QC and QA outputs of 7493 are ANDed
      and used to clear the count when the count
      reaches 1101. For the mod-6 counter, clock is
      applied to B input of 7492.")
4 disp(" The circuit diagram is as shown in the fig.
      8.19")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.12 design divide by 6 counter

```
1 //Example 8.12
2 clc
3 disp("The fig.8.20 shows divided-by-6 (MOD 6)
      counter using 7493. As shown in the fig.8.20, the
      clock is applied to inout B of IC 7493 and the
      output count sequence is taken from QD, QC and QB.
      As soon as count is 110, i.e. QD and QC = 1, the
      internal NAND gate output goes low and it resets
      the counter.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.13 find fmax

```
1 //Example 8.13
2 clc
3 disp("For a synchronous counter the total delay that
      must be allowed between input clock pulses is
      equal to flip-flop t_pd + AND gate t_pd. Thus
      T_clock >= 50 + 20 = 70 ns and so the counter has
      ")
4 fm=(1/(70*10^-9))*10^-6
5 format(5)
6 disp(fm," f_max(in MHz) =")
7 disp("We know that MOD-16 ripple counter used four
      flip-flops. With flip-flop t_pd = 50 ns, the
      f_max for ripple counter can be given as,")
8 fma=(1/(4*(50*10^-9)))*10^-6
9 format(3)
10 disp(fma," f_max(ripple)(in MHz) =")
```

Scilab code Exa 8.14 determine states

```
1 //Example 8.14
2 clc
3 disp("IC 74191 is a 4-bit counter. Thus it is MOD-16
      counter. However, we require MOD-11 counter. The
      difference between 16 and 11 is 5. Hence 5 steps
      must be skipped from the full modulus sequence.
      This can be achieved by presetting counter to
      value 5. Each time when counter recycles it
      starts counting from 5 upto 16 on each full cycle
      . Therefore, each full cycle of the counter
      consists of 11 states.")
```

Scilab code Exa 8.15 design MOD 10 counter

```
1 //Example 8.15
2 clc
3 disp("IC 74191 is a 4-bit counter. Thus it is MOD-16
      counter. However, we require MOD-10 counter. The
      difference between 16 and 10 is 6. Hence 6 steps
      must be skipped from the full modulus sequence.
      This can be achieved by presetting counter to
      value 6. Each time when counter recycles it
      starts counting from 6 upto 16 on each full cycle
      . Therefore, each full cycle of the counter
      consists of 10 states.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.16 design down counter

```

1 //Example 8.16
2 clc
3 disp("The fig 8.36 shows the connections for 74LS191
      to get desire operation. We can design the
      combinational circuit for such counter from the
      truth table shown below.")
4 disp("")
5 disp("Q3  Q2  Q1  Q0    Y")
6 disp("0   0   0   0    0")
7 disp("0   0   0   1    0")
8 disp("0   0   1   0    0")
9 disp("0   0   1   1    1")
10 disp("0   1   0   0    1")
11 disp("0   1   0   1    1")
12 disp("0   1   1   0    1")
13 disp("0   1   1   1    1")
14 disp("1   0   0   0    1")
15 disp("1   0   0   1    1")
16 disp("1   0   1   0    1")
17 disp("1   0   1   1    1")
18 disp("1   1   0   0    1")
19 disp("1   1   0   1    1")
20 disp("1   1   1   0    0")
21 disp("1   1   1   1    0")
22 disp("")
23 disp("K=map simplification")
24 disp("          Q1'Q0'  Q1'Q0  Q1Q0  Q1Q0'")
25 disp("Q3'Q2'  0      0      1      0")
26 disp("Q3'Q2   1      1      1      1")
27 disp("Q3Q2     1      1      0      0")
28 disp("Q3Q2'    1      1      1      1")
29 disp("")
30 disp("Therefore, PL' = Y = Q3'Q1Q0 + Q3'Q2 +
      Q3Q1' + Q3Q2'")
31 disp("After switch ON, if the counter output is
      other than 1101 through 0011, the PL' goes low

```

and count 1101 is loaded in the counter. The counter is then decremented on the occurrence of clock pulses. When counter reaches 0010, the PL' again goes low and count 1101 is loaded in the counter")

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.17 design programmable frequency divider

```
1 //Example 8.17
2 clc
3 disp("The IC 74191 is a 4-bit binary counter ,
    therefore f_out = f_CLK / 16 in up and down
    counter mode. If f_CLK = 500 Hz and f_out = 50 Hz
    we need mod 10 (500/50) counter. The fig. 8.39
    shows the mod-10 counter using IC 74191")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.18 design a counter

```
1 //Example 8.18
2 clc
3 disp("The fig shows the connections for 74LS191 to
    get desire operation. We can design the
    combinational circuit for such counter from the
    truth table shown below.")
```

```

4 disp("")
5 disp("Q3 Q2 Q1 Q0 Y")
6 disp("0 0 0 0 0")
7 disp("0 0 0 1 0")
8 disp("0 0 1 0 0")
9 disp("0 0 1 1 1")
10 disp("0 1 0 0 1")
11 disp("0 1 0 1 1")
12 disp("0 1 1 0 1")
13 disp("0 1 1 1 1")
14 disp("1 0 0 0 1")
15 disp("1 0 0 1 1")
16 disp("1 0 1 0 1")
17 disp("1 0 1 1 1")
18 disp("1 1 0 0 1")
19 disp("1 1 0 1 1")
20 disp("1 1 1 0 1")
21 disp("1 1 1 1 0")
22 disp("")
23 disp("K=map simplification")
24 disp("          Q1'Q0'  Q1'Q0  Q1Q0  Q1Q0'")
25 disp("Q3'Q2'  0      0      1      0")
26 disp("Q3'Q2    1      1      1      1")
27 disp("Q3Q2      1      1      0      1")
28 disp("Q3Q2'    1      1      1      1")
29 disp("")
30 disp("Therefore,  $PL' = Y = Q3'Q1Q0 + Q3'Q2 +$   

 $Q3Q1' + Q3Q2' + Q2Q1Q0'$ ")
31 disp("After switch ON, if the counter output is  

other than 1110 through 0011, the PL' goes low  

and count 1110 is loaded in the counter. The  

counter is then decremented on the occurrence of  

clock pulses. When counter reaches 0010, the PL'  

again goes low and count 1110 is loaded in the  

counter")

```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.19 programmable frequency divider

```
1 //Example 8.19
2 clc
3 disp("IC 74191 is a 4-bit binary counter. Thus it
    divides the input frequency by 16. However, we
    can design MOD-N counter using IC 74191. For MOD-
    N counter the output frequency will be  $f_{out} =$ 
 $f_{in} / N$ . Thus by changing N we can change the
    output frequency. The fig.8.40 shows the
    programmable frequency divider using IC 74191.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.20 design divide by 2 and divide by 5 counter

```
1 //Example 8.20
2 clc
3 disp("Fig. 8.41 shows Dividing-by-2 for up counting"
    )
4 disp("Divide-by-2 is a mod-2 counter. Since, after
    preset above counter goes through 2 states 1110
    and 1111, it is a mod-2 counter. Thus, above
    circuit is a divide-by-2 counter for up counting
    mode.")
```

```

5 disp("")
6 disp("Divide-by-5 for down counting mode:")
7 disp("")
8 disp("Q3  Q2  Q1  Q0    Y")
9 disp("0   0   0   0    0")
10 disp("0   0   0   1    0")
11 disp("0   0   1   0    0")
12 disp("0   0   1   1    0")
13 disp("0   1   0   0    0")
14 disp("0   1   0   1    0")
15 disp("0   1   1   0    0")
16 disp("0   1   1   1    0")
17 disp("1   0   0   0    0")
18 disp("1   0   0   1    0")
19 disp("1   0   1   0    0")
20 disp("1   0   1   1    1")
21 disp("1   1   0   0    1")
22 disp("1   1   0   1    1")
23 disp("1   1   1   0    1")
24 disp("1   1   1   1    1")
25 disp("")
26 disp("K-map simplification")
27 disp("      Q1',Q0',  Q1',Q0  Q1Q0  Q1Q0',")
28 disp("Q3',Q2',  0      0      0      0")
29 disp("Q3',Q2  0      0      0      0")
30 disp("Q3Q2      1      1      1      1")
31 disp("Q3Q2',  0      0      1      0")
32 disp("")
33 disp("Therefore ,  Y = Q3Q2 + Q3Q1Q0")

```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.21 design mod 9 counter

```
1 //Example 8.21
2 clc
3 disp("IC 74191 is a 4-bit counter. Thus it is MOD-16
      counter. However, we require MOD-9 counter. The
      difference between 16 and 9 is 7. Hence 7 steps
      must be skipped from the full modulus sequence.
      This can be achieved by presetting counter to
      value 7. Each time when counter recycles it
      starts counting from 7 upto 16 on each full cycle
      . Therefore, each full cycle of the counter
      consists of 9 states.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.22 determine MOD number and counter range

```
1 //Example 8.22
2 clc
3 disp("Clock frequency = 256 kHz")
4 disp("Output frequency = 2 kHz")
5 format(4)
6 mn=256/2
7 disp(mn,"Therefore, Mod number = n =")
8 disp("Therefore, Counter is MOD-128 counter")
9 disp("Mod-128 counter can count the numbers from 0
      to 127.")
```

Scilab code Exa 8.23 design a divide by 20 counter

```
1 //Example 8.23
```

```

2  clc
3  disp("Internal structure of 7490 ripple counter IC
      is as shown in fig. 8.50")
4  disp("")
5  disp("We know that , one IC can work as mod-10 (BCD)
      counter. Therefore , we need two ICs. The counter
      will go through states 0-19 and should be reset
      on state 20. i.e.")
6  disp("          QD  QC  QB  QA          QD  QC  QB  QA")
7  disp("          0   0   1   0          0   0   0   0")
8  disp("          7490(2)          7490(1)")
9  disp("")
10 disp("The diagram of divide-by-20 counter using IC
      7490 is as shown in fig.8.51")

```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.24 design divide by 96 counter

```

1  //Example 8.24
2  clc
3  disp("IC 7490 is a decade counter. When two such ICs
      are cascaded , it becomes a divide-by-100 counter
      . To get a divide-by-96 counter , the counter is
      reset as soon as it becomes 1001 0110. The
      diagram is shown in fig.8.52.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.25 design divide by 93 counter

```
1 //Example 8.25.
2 clc
3 disp("IC 7490 is a decade counter. Whentwo such ICs
    are cascaded , it becomes a divide-by-100 counter .
    To get a divide-by-93 counter , the counter is
    reset as soon as ot becomes 1001 0011. The
    diagram is as shown in fig.8.53")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.26 design divide by 78 counter

```
1 //Example 8.26
2 clc
3 disp("IC 7490 is a decade counter. When two such ICs
    are cascaded , it becomes a divide-by-100 counter
    . To get a divide by 78 or MOD-78 counter , the
    counter is reset as soon as ot becomes 0111 1000
    as shown in fig.8.54")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.28 7490 IC

```
1 //Example 8.28
2 clc
3 disp("If the QD output is connected to A input of
    7490 IC and, input clock is applied to B input
```

```

        divide by ten square wave is obtained at output
        QA.")
4  disp(" Clock      Outputs")
5  disp("           QA  QD  QC  QB")
6  disp("    0      L   L   L   L")
7  disp("    1      L   L   L   H")
8  disp("    2      L   L   H   L")
9  disp("    3      L   L   H   H")
10 disp("    4      L   H   L   L")
11 disp("    5      H   L   L   L")
12 disp("    6      H   L   L   H")
13 disp("    7      H   L   H   L")
14 disp("    8      H   L   H   H")
15 disp("    9      H   H   L   L")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.30 sketch output waveforms of counter

```

1  //example 10.9
2
3  clc;
4  clear;
5  close;
6  c = [0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
       0]; //taking the values for a mod -6 counter
7  q = [0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1
       0];
8  a = [0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0
       0];
9  b = [0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1
       0];
10 y1=q;
11 y2=a;

```

```

12 y3=b;
13 y11p=1;
14 y22p=1;
15 y33p=1;
16 y44p=1;
17 cp=1;
18 yf1p=1;
19 for i=1:25 // making arrays to draw the output
20     if y1(i)==1 then
21         for o=1:100
22             y11(y11p)=1;
23             y11p=y11p+1;
24         end
25     else
26         for o=1:100
27             y11(y11p)=0;
28             y11p=y11p+1;
29         end
30
31 end
32 if y2(i)==1 then
33     for o=1:100
34         y21(y22p)=1;
35         y22p=y22p+1;
36     end
37 else
38     for o=1:100
39         y21(y22p)=0;
40         y22p=y22p+1;
41     end
42
43 end
44 if y3(i)==1 then
45     for o=1:100
46         y31(y33p)=1;
47         y33p=y33p+1;
48     end
49 else

```

```

50         for o=1:100
51             y31(y33p)=0;
52             y33p=y33p+1;
53         end
54
55     end
56     if c(i)==1 then
57         for o=1:100
58             c1(cp)=1;
59             cp=cp+1;
60         end
61     else
62         for o=1:100
63             c1(cp)=0;
64             cp=cp+1;
65         end
66     end
67
68 end
69 z=[2 2];
70 subplot(4,1,1); //ploting the out put
71 title('Timing Diagram');
72 plot(c1);
73 plot(z);
74 ylabel('QA');
75 subplot(4,1,2);
76 plot(y11);
77 ylabel('QB');
78 plot(z);
79 subplot(4,1,3);
80 plot(y21);
81 ylabel('QC');
82 plot(z);
83 subplot(4,1,4);
84 plot(z);
85 ylabel('QD');
86 plot(y31);
87 disp("The counter goes through states 0000 (Decimal

```

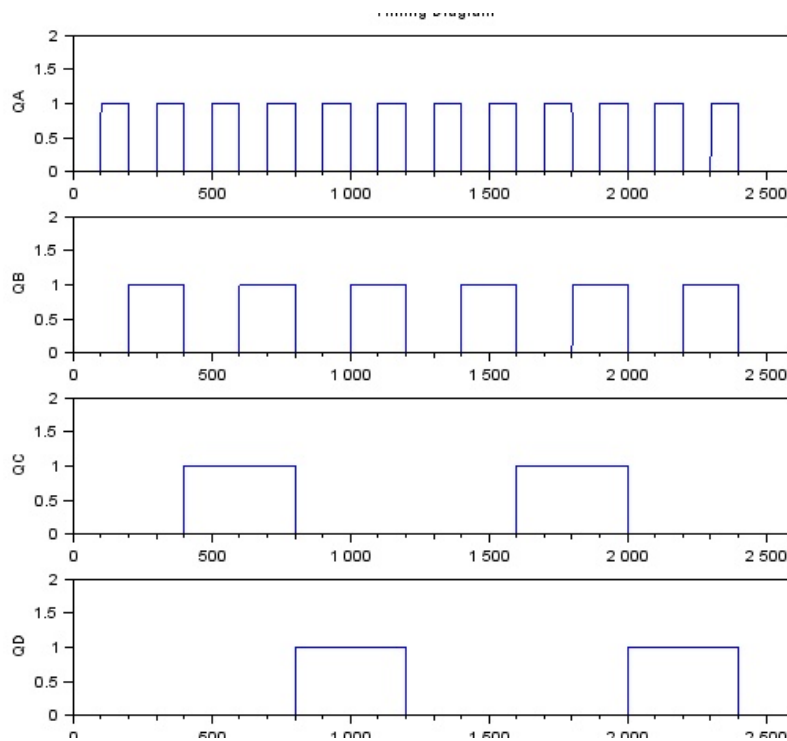



Figure 8.1: sketch output waveforms of counter

0) to 1011 (Decimal 11), i.e., through 12 states. Thus it is a MOD-12 counter.”)

Scilab code Exa 8.31 explain operation of circuit

```

1 //Example 8.31
2 clc
3 disp("The fig. 8.63 shows the cascaded connection of
  4-bit binary counters. Let us see the circuit
  operation. The counter IC1 operates as a counter
  for countin in the UP direction since CLEAR =

```

LOAD = 1. When the count reaches the maximum value (1111) its RC (Ripple Carry Output) goes HIGH which makes P and T (Enable) inputs of IC2 HIGH for one clock cycle advancing its output by 1 and making Q outputs of IC1, 0 at the next clock cycle. After this clock cycle P = T = 0 for IC2 and IC1 will go on counting the pulses. When the outputs of IC1 and IC2 both reach the maximum count, RC outputs of both of these ICs will go HIGH. This will make P = T of IC3 HIGH and therefore, in the next clock cycle IC3 count will be incremented and simultaneously IC1 and IC2 will be cleared. This way the counting will continue.”)

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.32 design divide by 40000 counter

```

1 //Example 8.32
2 clc
3 disp("Cascading four 74161 (each 4-bit) counters we
   get 16 (4 x 4) bit counter as shown in fig 8.63."
   )
4 disp("Therefore, we get 2^16 = 65,536 modulus
   counter")
5 disp("However, we require divide-by-40,000 counter.
   The difference between 65,536 and 40,000 is
   25,536, which is the number of states those must
   be skipped from the full modulus sequence. This
   can be achieved by presetting the counting from
   25,536 upto 65,536 on each full cycle. Therefore,
   each full cycle of the counter consists of 40,000
   states.")

```

Scilab code Exa 8.33 design modulo 11 counter

```
1 //Example 8.33
2 clc
3 disp("Although the 74X163 is a modulo-16 counter, it
      can be made to count in a modulus less than 16
      by using the CLR'' or LD'' input to shorten the
      normal counting sequence. The fig.8.69 shows
      circuit connections for modulo-11 counter. Here,
      load input is activated upon activation of RCO (
      ripple-carry-output). Since load input is
      adjusted to state 5, counter counts from 5 to 15
      and then starts at 5 again, for a total of 11
      states per counting cycle.")
4 disp("")
5 disp("We can also design modulo-11 counter using CLR
      '' input as shown in fig.8.70. here, NAND gate is
      used to detect state 10 and force the next state
      to 0. A 2-input gate is used to detect state 10
      (binary 1010) by connecting Q1 and Q3 to the
      inputs of the NAND gate.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.34 design excess 3 decimal counter

```
1 //Example 8.34
2 clc
3 disp("An excess-3 decimal counter should start
      counting from count 3 (binary 0011) and count
```

upto count 12 (binary 1100). Starting count is adjusted by loading 0011 at load inputs. To recycle count from 1100 to 0011, Q3 and Q2 output are connected as inputs for 2-input NAND gate. Thus, NAND gate detects state 1100 and forces 0011 to be loaded as the next state.”)

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.35 modulus greater than 16

```
1 //Example 8.35
2 clc
3 disp("A binary counter with a modulus greater than
    16 can be built by cascading 74X163s. When
    counters are cascaded, CLK, CLR'' and LD'' of all
    the 74X163s are connected in parallel, so that
    all of them count or are cleared or loaded at the
    same time. The RCO signal drives the ENT input
    of the next counter. The fig.8.73 shows modulo-60
    counter. To have a modulo 60 count we need at
    least 6-bit counter, thus two 74X163s are
    cascaded. Counter is designed to count from 196
    to 255. The MAXCNT signal detects the state 255
    and stops the counter until GO'' is asserted. When
    GO'' is asserted the counter is reloaded with
    196 (binary 1100 0100) and counts upto 255. To
    enable counting, CNTEN is connected to the ENP
    inputs in parallel. A NAND gate asserts RELOAD''
    to get back to state 196 only if GO'' is asserted
    and the counter is in state 255.”)
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.36 modulo 8 counter

```
1 //Example 8.36
2 clc
3 disp("A binary counter may be combined with a
  decoder to obtain a set of 1-out-of-M coded
  signals , where one signal is asserted in each
  count state. This is useful when counters are
  used to control a set of devices , where a
  different devices is enabled in each counter
  state.")
4 disp("The fig.8.74 shows how a 74X163 connected as a
  modulo-8 counter can be combined with a 74X138
  3-8 decoder to provide eight signals , each one
  representing a counter state.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.37 synchronous decade counter

```
1 //Example 8.37
2 clc
3 disp("Excitation table")
4 disp("Present State           Next State
  Flip-flop Inputs")
5 disp("QD QC QB QA   Q_D+1 Q_C+1 Q_B+1 Q_A+1
  JK_D JK_C JK_B JK_A")
```

```

6  disp("0  0  0  0      0      0      0      1
      0  0      0  1")
7  disp("0  0  0  1      0      0      1      0
      0  0      1  1")
8  disp("0  0  1  0      0      0      1      1
      0  0      0  1")
9  disp("0  0  1  1      0      1      0      0
      0  1      1  1")
10 disp("0  1  0  0      0      1      0      1
      0  0      0  1")
11 disp("0  1  0  1      0      1      1      0
      0  0      1  1")
12 disp("0  1  1  0      0      1      1      1
      0  0      0  1")
13 disp("0  1  1  1      1      0      0      0
      1  1      1  1")
14 disp("1  0  0  0      1      0      0      1
      0  0      0  1")
15 disp("1  0  0  1      0      0      0      0
      1  0      0  1")
16 disp("1  0  1  0      X      X      X      X
      X  X      X  1")
17 disp("1  0  1  1      X      X      X      X
      X  X      X  1")
18 disp("1  1  0  0      X      X      X      X
      X  X      X  X")
19 disp("1  1  0  1      X      X      X      X
      X  X      X  X")
20 disp("1  1  1  0      X      X      X      X
      X  X      X  X")
21 disp("1  1  1  1      X      X      X      X
      X  X      X  X")
22 disp("")
23 disp("K-map Simplification")
24 disp("      For JK_D")
25 disp("      QB' 'QA' '  QB' 'QA  QBQA  QBQA' '")
26 disp("QB' 'QC' '  0      0      0      0")
27 disp("QB' 'QC      0      0      1      0")

```

```

28 disp("QDQC      X      X      X      X")
29 disp("QDQC' ,      0      1      X      X")
30 disp("JK_D = QA QD + QA QB QC")
31 disp("")
32 disp("                          For JK_C")
33 disp("          QB' ,QA' ,      QB' ,QA      QBQA      QBQA' ,")
34 disp("QD' ,QC' ,      0      0      1      0")
35 disp("QD' ,QC      0      0      1      0")
36 disp("QDQC      X      X      X      X")
37 disp("QDQC' ,      0      0      X      X")
38 disp("JK_C = QA QB")
39 disp("")
40 disp("                          For JK_B")
41 disp("          QB' ,QA' ,      QB' ,QA      QBQA      QBQA' ,")
42 disp("QD' ,QC' ,      0      1      1      0")
43 disp("QD' ,QC      0      1      1      0")
44 disp("QDQC      X      X      X      X")
45 disp("QDQC' ,      0      0      X      X")
46 disp("JK_B = QA QD' ")
47 disp("")
48 disp("                          For JK_A")
49 disp("          QB' ,QA' ,      QB' ,QA      QBQA      QBQA' ,")
50 disp("QD' ,QC' ,      1      1      1      1")
51 disp("QD' ,QC      1      1      1      1")
52 disp("QDQC      X      X      X      X")
53 disp("QDQC' ,      1      1      X      X")
54 disp("JK_A = 1")
55 disp("")
56 disp("Fig shows the logic diagram for the
      synchronous decade counter using JK flip-flop")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.38 flip flops

```

1 //Example 8.38
2 clc
3 disp("Excitation table")
4 disp(" Input          Present State          Next State
      Flip-flop Inputs")
5 disp("UP/DOWN' ,      QC  QB  QA          Q_C+1  Q_B+1
      Q_A+1  JK_C  JK_B  JK_A")
6 disp("  UD")
7 disp("  0          0  0  0          1      1
      1          1  1  1")
8 disp("  0          0  0  1          0      0
      0          0  0  1")
9 disp("  0          0  1  0          0      0
      1          0  1  1")
10 disp("  0          0  1  1          0      1
      0          0  0  1")
11 disp("  0          1  0  0          0      1
      1          1  1  1")
12 disp("  0          1  0  1          1      0
      0          0  0  1")
13 disp("  0          1  1  0          1      0
      1          0  1  1")
14 disp("  0          1  1  1          1      1
      0          0  0  1")
15 disp("  1          0  0  0          0      0
      1          0  0  1")
16 disp("  1          0  0  1          0      1
      0          0  1  1")
17 disp("  1          0  1  0          0      1
      1          0  0  1")
18 disp("  1          0  1  1          1      0
      0          1  1  1")
19 disp("  1          1  0  0          1      0
      1          0  0  1")
20 disp("  1          1  0  1          1      1
      0          0  1  1")
21 disp("  1          1  1  0          1      1
      1          0  0  1")

```



```

22 disp("      1          1  1  1          0      0
           0          1  1  1          1")
23 disp("")
24 disp("K-map Simplification")
25 disp("                For JK_C")
26 disp("                QB' 'QA' '   QB' 'QA   QBQA   QBQA' ' '")
27 disp("QD' 'QC' '   1          0          0          0")
28 disp("QD' 'QC      1          0          0          0")
29 disp("QDQC         0          0          1          0")
30 disp("QDQC' '      0          0          1          0")
31 disp("JK_C =UD' '   QB' '   QB' '   + UD QB QA")
32 disp("")
33 disp("                For JK_B")
34 disp("                QB' 'QA' '   QB' 'QA   QBQA   QBQA' ' '")
35 disp("QD' 'QC' '   1          0          0          1")
36 disp("QD' 'QC      1          0          0          1")
37 disp("QDQC         0          1          1          0")
38 disp("QDQC' '      0          1          1          0")
39 disp("TB =UD' '   QA' '   + UD QA")
40 disp("")
41 disp("                For JK_A")
42 disp("                QB' 'QA' '   QB' 'QA   QBQA   QBQA' ' '")
43 disp("QD' 'QC' '   1          1          1          1")
44 disp("QD' 'QC      1          1          1          1")
45 disp("QDQC         1          1          1          1")
46 disp("QDQC' '      1          1          1          1")
47 disp("TA = 1")
48 disp("")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.39 design mod 5 synchronous counter

1 //Example 8.39

```

2  clc
3  disp("For mod-5 counter we require 3 flip-flops.")
4  disp("Excitation table")
5  disp("
           Present State           Next State
           Flip-flop Inputs")
6  disp("
           QC  QB  QA           Q_A+1  Q_B+1
           Q_C+1  T_A  T_B  T_C")
7  disp("  0           0  0  0           0      0
           1           0  0  1")
8  disp("  1           0  0  1           0      1
           0           0  1  1")
9  disp("  2           0  1  0           0      1
           1           0  0  1")
10 disp("  3           0  1  1           1      0
           0           1  1  1")
11 disp("  4           1  0  0           0      0
           0           1  0  0")
12 disp("")
13 disp("K-map Simplification")
14 disp("           QB' 'QC' '  QB' 'QC  QBQC  QBQC' ' '")
15 disp("QA' '           0           0           1           0")
16 disp("QA           1           X           X           X")
17 disp("T_A = QA + QB QC")
18 disp("")
19 disp("           QB' 'QC' '  QB' 'QC  QBQC  QBQC' ' '")
20 disp("QA' '           0           1           1           0")
21 disp("QA           0           X           X           X")
22 disp("T_B = QC")
23 disp("")
24 disp("           QB' 'QC' '  QB' 'QC  QBQC  QBQC' ' '")
25 disp("QA' '           1           1           1           1")
26 disp("QA           0           X           X           X")
27 disp("T_C = QA' '")
28 disp("")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.40 design MOD 4 down counter

```

1 //Example 8.40
2 clc
3 disp("Excitation table")
4 disp("Present State      Next State      Flip-
   flop Inputs")
5 disp("  QC  QB      A+   B+      J_A   K_A
   J_B   K_B")
6 disp("  0   0      1    1      1     X
   1     X")
7 disp("  0   1      0    0      0     X
   X     1")
8 disp("  1   0      0    1      X     1
   1     X")
9 disp("  1   1      1    0      X     0
   X     1")
10 disp("")
11 disp("K-map Simplification")
12 disp("  For J_A")
13 disp("      B'   B")
14 disp("A'   1   0")
15 disp("A    X   X")
16 disp("J_A = B' ")
17 disp("")
18 disp("  For K_A")
19 disp("      B'   B")
20 disp("A'   X   X")
21 disp("A    1   0")
22 disp("K_A = B' ")
23 disp("")

```

```

24 disp(" For J_B")
25 disp("      B' ' B")
26 disp("A' ' 1 X")
27 disp("A 1 X")
28 disp("J_B = 1")
29 disp("")
30 disp(" For K_B")
31 disp("      B' ' B")
32 disp("A' ' X 1")
33 disp("A X 1")
34 disp("K_B = 1")
35 disp("")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.41 design MOD 12 synchronous counter

```

1 //Example 8.41
2 clc
3 disp("Mod-12 synchronous counter using D flip-flop :
      ")
4 disp("Let      Number of flip-flop required = n")
5 disp("                2^n >= 12")
6 disp("                n = 4")
7 disp("Excitation table")
8 disp("Present State      Next State      ")
9 disp("QD  QC  QB  QA      Q_D+1  Q_C+1  Q_B+1  Q_A+1")
10 disp("0   0   0   0      0       0       0       1")
11 disp("0   0   0   1      0       0       1       0")
12 disp("0   0   1   0      0       0       1       1")
13 disp("0   0   1   1      0       1       0       0")
14 disp("0   1   0   0      0       1       0       1")
15 disp("0   1   0   1      0       1       1       0")
16 disp("0   1   1   0      0       1       1       1")

```

```

17 disp(" 0   1   1   1       1       0       0       0")
18 disp(" 1   0   0   0       1       0       0       1")
19 disp(" 1   0   0   1       1       0       1       0")
20 disp(" 1   0   1   0       1       0       1       1")
21 disp(" 1   0   1   1       0       0       0       0")
22 disp("")
23 disp("K-map Simplification")
24 disp("                For D_A")
25 disp("                QB'QA'  QB'QA  QBQA  QBQA'")
26 disp("QD'QC'      1       0       0       1")
27 disp("QD'QC       1       0       0       1")
28 disp("QDQC        X       X       X       X")
29 disp("QDQC'      1       0       0       1")
30 disp("D_A = QA'")
31 disp("")
32 disp("                For D_B")
33 disp("                QB'QA'  QB'QA  QBQA  QBQA'")
34 disp("QD'QC'      0       1       0       1")
35 disp("QD'QC       0       1       0       1")
36 disp("QDQC        X       X       X       X")
37 disp("QDQC'      0       1       0       1")
38 disp("D_B = QB' QaA + QA' QB")
39 disp("      = QA XOR QB")
40 disp("")
41 disp("                For D_C")
42 disp("                QB'QA'  QB'QA  QBQA  QBQA'")
43 disp("QD'QC'      0       0       1       0")
44 disp("QD'QC       1       1       0       1")
45 disp("QDQC        X       X       X       X")
46 disp("QDQC'      0       0       0       0")
47 disp("D_C = QC QB' + QC QA' + QD' QC' QB QA")
48 disp("")
49 disp("                For D_D")
50 disp("                QB'QA'  QB'QA  QBQA  QBQA'")
51 disp("QD'QC'      0       0       0       0")
52 disp("QD'QC       0       0       1       0")
53 disp("QDQC        X       X       X       X")
54 disp("QDQC'      1       1       0       1")

```

```
55 disp("D_D = QD QB' ' + QC QB QA + QD QA' '")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.42 design 4 bit 4 state ring counter

```
1 //Example 8.42
2 clc
3 disp("The fig.8.99 shows the circuit diagram for a
    4-bit , 4-state ring counter with a single
    circulating 1. Here , 74X194 universal shift
    register is connected so that it normally
    performs a left-shift . However , when RESET is
    asserted it loads 0001. Once RESET is negated ,
    the 74194 shifts left on each clock pulse . The
    D_SL serial input is connected to the leftmost
    output (Q3 : MSB) , so the next states are 0010 ,
    0100 , 1000 , 0001 , 0010 , ..... Thus the counter
    counter visits four unique states before
    repeating.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 8.44 design 4 bit 8 state johnson counter

```
1 //Example 8.44
2 clc
3 disp("Johnson counter is basically a twisted ring
    counter . The fig.8.104(a) shows the basic circuit
    for a Johnson counter . The table shows the
    states of a 4-bit Johnson counter.")
```

```

4 disp("")
5 disp("States of 4-bit Johnson counter")
6 disp("State name      Q3  Q2  Q1  Q0")
7 disp("    S1          0  0  0  0")
8 disp("    S2          0  0  0  1")
9 disp("    S3          0  0  1  1")
10 disp("    S4          0  1  1  1")
11 disp("    S5          1  1  1  1")
12 disp("    S6          1  1  1  0")
13 disp("    S7          1  1  0  0")
14 disp("    S8          1  0  0  0")
15 disp("")
16 disp("This counter can be modified to have self
      correcting Johnson counter as shown in fig.8.104(
      c). Here, the connections are made such that
      circuit oads 0001 as the next state whenever the
      current state is 0XX0.")

```

This code can be downloaded from the website www.scilab.in This code

can be downloaded from the website www.scilab.in

Scilab code Exa 8.45 johnson counter

```

1 //Example 8.45
2 clc
3 disp("Johnson counter will produce a modulus of 2xn
      where n is the number of stages (i.e. flip-flops)
      in the counter. Therefore, Mod 10 requires 5
      flip-flops and Mod 16 requires 8 flip-flops.")

```

Chapter 9

Op amp Applications

Scilab code Exa 9.1 output voltage

```
1 //Example 9.1
2 clc
3 disp("The differential amplifier is represented as
      shown in fig. 9.5.")
4 disp("(i) CMRR = 100")
5 vd=300-240
6 disp(vd,"      Vd(in uV) = V1 - V2 = ")
7 vc=(300+240)/2
8 disp(vc,"      Vc(in uV) = V1+V2 / 2 =")
9 disp("CMRR = Ad / Ac")
10 ac=5000/100
11 disp(ac," Therefore , Ac =")
12 format(6)
13 vo=((5000*60)+(50*270))*10^-3
14 disp(vo," Therefore , Vo(in mV) = Ad*Vd + Ac*Vc =")
15 disp("(ii) CMRR = 10^5")
16 ac=5000/(10^5)
17 disp(ac," Therefore , Ac = Ad / CMRR =")
18 vo=((5000*60)+(0.05*270))*10^-3
19 format(9)
20 disp(vo," Therefore , Vo(in mV) = Ad*Vd + Ac*Vc =")
```


21 `disp("Ideally Ac must be zero and output should be only Ad*Vd which is 5000*60*10-6 i.e. 300 mV. It can be seen that higher the value of CMRR, the output is almost proportional to the difference voltage Vd, rejecting the common mode signal. So ideal value of CMRR for a differential amplifier is infinity.")`

Scilab code Exa 9.3 input bias current

```

1 //Example 9.3
2 clc
3 disp("I_iOS = 20 nA, I_b = 60 nA")
4 disp("Now I_iOS = I_b1 - I_b2 = 20")
5 disp("I_b = I_b1+I_b2 / 2 = 60")
6 disp("Therefore, I_b1 + I_b2 = 120")
7 disp("Therefore, 2*I_b1 = 140")
8 disp("Therefore, I_b1 = 70 nA, I_b2 = 50 nA")

```

Scilab code Exa 9.4 design inverting schmitt trigger

```

1 //Example 9.4
2 clc
3 disp("V_UT = +4 V, V_LT = -4 V, Supply = +- 15 V")
4 disp("+- V_sat = 0.9 x [Supply] = +- 13.5 V = Vo")
5 disp("For op-amp 741, I_B(max) = 500 nA")
6 disp("Therefore, I2 = 100I_B(max) = 50 uA")
7 r2=(4/(50*10-6))*10-3
8 disp(r2,"Therefore, R2(in k-ohm) = V_UT / I2 =")
9 i2=(4/(82*103))*10-6
10 format(6)
11 disp(i2,"Recalculating I2, I2 = V_UT / R2 =")
12 r1=((13.5-4)/(48.78*10-6))*10-3

```

```

13 format(7)
14 disp(r1,"Therefore , R1 =  $V_o - V_{UT} / I_2 = +V_{sat} - V_{UT}$ 
      /  $I_2 =$ ")
15 disp("The designed circuit is shown in fig")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.5 threshold voltage

```

1 //Example 9.5
2 clc
3 disp("VCC = +15 V")
4 vsat=0.9*15
5 format(5)
6 disp(vsat,"Therefore , Vsat(in V) = 0.9 VCC =")
7 disp(" R1 = 51 k-ohm, R2 = 120 ohm")
8 vut=(13.5*120)/((51*103)+120)
9 format(8)
10 disp(vut,"VUT(in V) = +Vsat*R2 / R1+R2 =")
11 vlt=(-13.5*120)/((51*103)+120)
12 disp(vlt,"VLT(in V) = -Vsat*R2 / R1+R2 =")
13 h=(0.03169*2)*103
14 format(6)
15 disp(h,"H(in mV) = VUT - VLT =")

```

Scilab code Exa 9.6 tripping voltage

```

1 //Example 9.6
2 clc
3 disp("As input is applied to the non-inverting
      terminal, the circuit is non-inverting Schmitt
      trigger.")

```

```

4 disp(" R1 = 100 k-ohm, R2 = 1 k-ohm")
5 vut=13.5*(1/100)
6 format(6)
7 disp(vut," Therefore , V_UT(in V) = +V_sat * R2/R1 ="
      )
8 vlt=-13.5*(1/100)
9 disp(vlt," Therefore , V_LT(in V) = -V_sat * R2/R1 ="
      )

```

Scilab code Exa 9.8 time duration

```

1 //Example 9.8
2 clc
3 disp("LTP = -1.5 V and H = 2 V")
4 disp("Now H = UTP - LTP")
5 disp("Therefore , 2 = UTP - (-1.5)")
6 disp("Therefore , UTP = 0.5 V")
7 disp("In the fig.9.47, the angle theta can be
      obtained from equation of sine wave. Sine wave is
      represented as,")
8 disp("V_in = V_p*sin(pi+thata) when pi < omega*t
      < 2pi")
9 disp("At LTP, -1.5 = 5*sin(pi+theta)")
10 disp(" = - 5*sin(theta)")
11 disp("Therefore , sin(theta) = 0.3")
12 t=asind(0.3)
13 format(6)
14 disp(t,"Therefore , theta(in degree) =")
15 disp("The time period of sine wave is,")
16 T=1
17 disp(T," T(in ms) = 1/f =")
18 disp("At UTP, 0.5 = V_p*sin(theta)")
19 disp("Therefore , 0.5 = 5 * sin(theta)")
20 disp("Therefore , sin(theta) = 0.1")
21 t=asind(0.1)

```

```

22 disp(t,"Therefore , theta(in degree) =")
23 disp("The time T1 for output is from 5.739 degree to
      (180 degree + 17.45 degree)")
24 t1=197.45-5.739
25 format(7)
26 disp(t1,"Therefore , T1(in degree) =")
27 T1=(191.71/360)
28 disp(T1,"i.e. T1(ms) =")
29 t2=1-0.5325
30 disp(t2,"and T2(in ms) = T - T1 =")

```

Scilab code Exa 9.9 calculate R1 and R2

```

1 //Example 9.9
2 clc
3 disp("Given +V_sat = 12 V, -V_sat = -12 V,
      V_H = 6 V")
4 disp("We know that hysteresis width is given as")
5 disp(" V_H = (R2/R1+R2)[+V_sat-V_sat]")
6 disp("Therefore , R2 / R1+R2 = V_H / +V_sat-V_sat")
7 r=6/(24)
8 disp(r,"Therefore , R2 / R1+R2 =")
9 disp("Therefore , R2 = 0.25R1 + 0.25R2")
10 disp("Therefore , 0.75R2 = 0.25R1")
11 r2=0.25/0.75
12 format(7)
13 disp(r2,"Therefore , R2 / R1 =")
14 disp("Assuming R2 = 10 k-ohm")
15 r1=(10000/0.3333)*10^-3
16 format(3)
17 disp(r1," R1(in k-ohm) =")

```

Scilab code Exa 9.10 calculate trip point and hysteresis

```

1 //Example 9.10
2 clc
3 disp("From fig 9.45, R1 = 68 k-ohm, R2 = 1.5 k-ohm
      and V_sat = 13.5 V")
4 vut=(1.5/(1.5+68))*13.5
5 format(7)
6 disp(vut,"V_UT(in V) = R2/R1+R2 * V_sat =")
7 vlt=(-1.5/(1.5+68))*13.5
8 disp(vlt,"V_LT(in V) = -R2/R1+R2 * V_sat =")
9 h=2*0.2913
10 disp(h,"Therefore, H(in V) = V_UT - V_LT =")
11 disp("Now H = (2*R2 / R1+R2) * V_sat")
12 disp("For minimum H, R2 must be minimum and R1 must
      be maximum")
13 r2min=((1.5)-(0.05*1.5))
14 format(6)
15 disp(r2min,"Therefore, R2_min(in k-ohm) = R2 - 5%*
      R2 =")
16 r2max=((68)+(0.05*68))
17 disp(r2max,"Therefore, R1_max(in k-ohm) = R1 + 5%*
      R1 =")
18 hm=((2*1.425)/(71.4+1.425))*13.5
19 disp(hm,"Therefore, H_min(in V) =")

```

Scilab code Exa 9.11 design schmitt trigger

```

1 //Example 9.11
2 clc
3 disp("Choose op-amp LM318 with V_sat as +- 13.5 V
      with supply voltage +- 15 V")
4 disp("V_UT = + 5 V")
5 disp("Now V_UT = (R2 / R1+R2)*V_sat")
6 disp("Therefore, 5 = (R2 / R1+R2)*13.5")
7 disp("Therefore, R1 + R2 = 2.7*R2")
8 disp("Therefore, R1 = 1.7*R2")

```

```

9 disp("Choose R2 = 10 k-ohm")
10 r1=1.7*10
11 disp(r1,"Therefore , R1(in k-ohm) =")
12 disp("The designed circuit is shown in fig.9.46")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.12 design op amp schmitt trigger

```

1 //Example 9.12
2 clc
3 disp("For the Schmitt trigger")
4 disp("V_UT = 2 V, V_LT = -4 V, +V_sat = +-10
      V")
5 disp("For unequal UTP and LTP values , a modified
      circuit is required as shown in the fig.9.52.")
6 disp("The voltage V1 decides the UTP and LTP levels.
      Applying KVL to the output circuit and
      neglecting op-amp input current we can write,")
7 disp("-IR2 - IR1 - x + V0 = 0")
8 disp("Therefore , I = V0-x / R1+R2")
9 disp("And V1 = IR1 + x")
10 disp("Therefore , V1 = (V0-x/R1+R2)*R1 + x")
11 disp("For +V_sat = 10 V,")
12 disp("V1 = V_UT = 2 V,")
13 disp("V0 = 10 V")
14 disp("Therefore , 2 = (10-x/R1+R2)*R1 + x (1)")
15 disp("For -V_sat = -10 V,")
16 disp("V1 = V_LT = -4 V,")
17 disp("V0 = -10 V")
18 disp("Therefore , -4 = (-10-x/R1+R2)*R1 + x (2)"
      )
19 disp("Subtracting equations (2) and (1),")
20 disp("Therefore , 6 = 20*R1 / R1+R2")

```

```

21 disp(" Therefore ,   R1+R2 = 3.333*R1")
22 disp(" Therefore ,   R2 = 2.333*R1      (3)")
23 disp(" Substituting (3) in equation (1)")
24 disp(" 2 = ((10-x)*R1 / 3.333*R1) + x")
25 disp(" Therefore ,   2.333*x = -3.3334")
26 x=-3.3334/2.333
27 format(7)
28 disp(x," Therefore ,   x =")
29 disp("So actually polarity of the voltage source 'x
      ' must be opposite to what is assumed earlier as
      shown in fig.9.52.")
30 disp(" Choose      R1 = 1 k-ohm      hence  R2 = 2.333 k-
      ohm")
31 disp(" Therefore ,   R_comp = R1 || R2 = 0.7 k-ohm")
32 disp("Now as long as V_in is less than V_UT, the
      output is at +V_sat = 10 V and when V_in > V_UT,
      the output switches from +V_sat to -V_sat. While
      as long as V_in > V_LT, the output is at -V_sat =
      -10 V and when V_in < V_LT, the output switches
      from -V_sat to +V_sat.")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.13 VUT and VLT and frequency of oscillation

```

1 //Example 9.13
2 clc
3 disp("(a) We know that ,")
4 vut=(86*15)/(86+100)
5 format(5)
6 disp(vut," V_UT(in V) = R1*+V_sat / R1+R2 =")
7 vlt=(86*-15)/(86+100)
8 disp("(b) We know that ,")
9 disp(vlt," V_LT(in V) = R1*-V_sat / R1+R2 =")

```

```

10 disp("(c) We know that ,")
11 f0=1/0.02
12 disp(f0," f0 (in Hz) = 1 / 2*Rf*C_in*[+V_sat-V_LT/+
      V_sat-V_UT] =")

```

Scilab code Exa 9.17 design op amp circuit

```

1 //Example 9.17
2 clc
3 disp("The monostable multivibrator using op-amp
      produces the pulse waveform. The pulse width is
      given by,")
4 disp(" T = RC*ln[1+V_D1/V_sat / 1-beta]")
5 disp(" where V_D1 = 0.7 V, +v_sat = +-12 V for op-
      amp 741")
6 disp(" beta = R2 / R1+R2 = 0.5 with R1 = R2")
7 t=1/(2*10^3)
8 format(6)
9 disp(t,"T(in sec) = 1/f")
10 disp(" Choose C = 0.1 uF")
11 disp(" Therefore , 5*10^-4 = R*0.1*10^-6*ln
      [1+(0.7/12)/1-0.5]")
12 disp(" Therefore , R = 6.7 k-ohm")
13 disp(" Choose R1 = R2 = 10 k-ohm")
14 disp("The designed circuit is shown in fig.9.63")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.18 design monostable using IC 555

```

1 //Example 9.18
2 clc

```



```

3 disp("The required pulse width is ,")
4 disp(" W = 10 ms")
5 disp("The pulse width is given by ,")
6 disp(" W = 1.1*R*C")
7 disp(" Therefore , 10*10-3 = 1.1*R*C")
8 disp(" Therefore , RC = 9.0909*10-3")
9 disp(" Choose C = 0.1 uF")
10 disp(" Therefore , R = 90.909 k-ohm ~ 91 k-ohm")
11 disp("The designed circuit is shown in fig.9.78")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.19 design a timer

```

1 //Example 9.19
2 clc
3 disp("Fig. 9.79 shows monostable circuit used to
   drive the relay.")
4 disp("This relay should be energized for 5 second to
   hold heater 'ON' for 5 seconds. Thus, T_ON for
   monostable is 5 seconds.")
5 disp("We know that the pulse width is given by ,")
6 disp("          W = 1.1 RC")
7 disp(" Therefore , 5 = 1.1 RC")
8 disp("Now, there are two unknowns. In this case , we
   have to select value for capacitor and with the
   selected value we have to find the value of
   resistance from the formula.")
9 disp(" Therefore , If capacitor value is 10 uF")
10 disp(" then 5 = 1.1*R*10 uF")
11 r=(5/(1.1*10*10-6))*10-3
12 format(7)
13 disp(r," Therefore , R(in k-ohm) =")
14 disp("The calculated value is not standard value ,

```

but we can adjust this value by connecting variable resistance i.e. potentiometer.”)

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.20 draw timer using IC 555

```
1 //Example 9.20
2 clc
3 disp("The requirement is that the door must be open
      for 15 sec after receiving a trigger signal and
      then gets shut door automatically. This requires
      IC 555 in a monostable mode with a pulse width of
      15 sec.")
4 disp("Therefore , W = 15 sec")
5 disp("Now          W = 1.1 RC")
6 disp("Therefore , 15 = 1.1 RC")
7 disp("Choose      C = 100 uF")
8 r=(15/(1.1*100*10^-6))*10^-3
9 format(8)
10 disp(r,"Therefore , R(in k-ohm) =")
11 disp("The designed circuit is shown in the fig. 9.80
      ")
12 disp("The supply voltage 10 or 15 V has no effect on
      the operation of the circuit or the values of R
      and C selected.")
```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.21 frequency of output and duty cycle

```

1  ///Example 9.21
2  clc
3  disp("The frequency of output is given by,")
4  f=(1.44/(12*0.01*10^-3))*10^-3
5  format(3)
6  disp(f," f(in kHz) = 1.44 / (R_A+2*R_B)*C =")
7  disp("The duty cycle is given by,")
8  d=8/12
9  format(7)
10 disp(d," D = R_A+R_B / R_A+2*R_B =")
11 disp("Thus the duty cycle 66.67%")

```

Scilab code Exa 9.26 design astable multivibrator

```

1  //Example 9.26
2  clc
3  disp("f = 1 kHz")
4  disp("D = 75% = 0.75")
5  disp("Now f = 1.44 / (R_A+2*R_B)*C")
6  disp("Therefore, 1*10^3 = 1.44 / (R_A+2*R_B)*C")
7  disp("Therefore, (R_A+2*R_B)*C = 1.44*10^-3
      ....(1)")
8  disp("Therefore, while %D = ((R_A+R_B)/(R_A+2*R_B))
      *100")
9  disp("Therefore, 0.75 = R_A+R_B / R_A+2*R_B")
10 disp("Therefore, R_A+2*R_B = (R_A+R_B)/0.75")
11 disp("Therefore, R_A+2*R_B = 1.33*(R_A+R_B)")
12 disp("Therefore, 0.66*R_B = 0.33*R_A")
13 disp("Therefore, R_B = 0.5*R_A
      ....(2)")
14 disp("Choose C = 0.1 uF")
15 disp("Substituting in (1),")
16 disp("(R_A+2*R_B)*0.1*10^-6 = 1.44*10^-3")
17 disp("Therefore, R_A+2*R_B = 14400
      ....(3)")

```

```

18 disp("Substituting (2) in (3),")
19 disp("R_A + 2(0.5*R_A) = 14400")
20 ra=(14400/2)*10^-3
21 format(4)
22 disp(ra,"Therefore , R_A(in k-ohm) =")
23 rb=0.5*7.2
24 disp(rb,"Therefore , R_B(in k-ohm) =")
25 disp("and C = 0.1 uF")
26 disp("Hence the circuit diagram is as shown in fig
      .9.100")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.27 design astable multivibrator

```

1 //Example 9.27
2 clc
3 disp("T_ON = 0.6 ms, T = 1 ms")
4 d=0.6*100
5 disp(d,"Therefore , D(in percentage) = t_ON / T =")
6 disp("Now D = R_A+R_B / R_A+2*R_B = 0.6")
7 disp("Therefore , R_A+R_B = 0.6*R_A + 1.2*R_B")
8 disp("Therefore , 0.4*R_B = 0.2*R_B")
9 disp("Therefore , R_B = 2*R_A .....(1)")
10 disp("      f = 1.44 / (R_A+2*R_B)*C = 1/T = 1000"
      )
11 disp("Choose C = 0.1 uF")
12 disp("Therefore , R_A+2*R_B = 14400")
13 disp("Using (1) , 5*R_A = 14400")
14 ra=(14400/5)*10^-3
15 format(5)
16 disp(ra,"Therefore , R_A(in k-ohm) =")
17 rb=2.88*2
18 disp(rb," R_B(in k-ohm) =")

```

19 `disp("The circuit is shown in the fig.9.101")`

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.28 design astable mode to generate square wave

```
1 //Example 9.28
2 clc
3 disp("T_ON = T_OFF = 0.5 ms")
4 disp("Therefore , T = T_ON + T_OFF = 1 ms")
5 disp("i.e. f = 1/T = 1 kHz")
6 disp("Now T_d = T_OFF = 0.69*R_B*C")
7 disp("Choose C = 0.1 uF")
8 rb=((0.5*10^-3)/(0.69*0.1*10^-6))*10^-3
9 format(6)
10 disp(rb,"Therefore , R_B(in k-ohm) =")
11 disp("Now duty cycle is 50% so R_A = R_B = 7.246 k-ohm")
12 disp("Practically a modified circuit is required for
    50% duty cycle where diode is connected across
    R_B and charging takes place through R_A and
    diode. And R_B must be equal to sum of R_A and
    diode forward resistance. So to have perfect
    square wave, R_A is kept variable i.e. pot of say
    10 k-ohm in this case. It is then adjusted to
    obtain precise square wave. The resistance
    required in series with LED to be connected is,")
13 disp("R = V_0-V_LED / I_LED")
14 disp("Assuming V_LED = 0.7 V")
15 r=(5-0.7)/(50*10^-3)
16 format(3)
17 disp(r,"Current limiting R(ohm) = ")
18 disp("The voltage of R is")
19 disp("P = (50*10^-3)^2 * 100")
```

```

20 p = ((50*10^-3)^2)*100
21 disp(p,"P(in W) =")
22 disp("Both resistors R can be of 1/4 W")
23 disp("The required circuit is shown in the fig.9.102
      ")

```

This code can be downloaded from the website www.scilab.in

Scilab code Exa 9.32 define resolution

```

1 //Example 9.32
2 clc
3 disp("For the given DAC,")
4 disp(" n = Number of bits = 8")
5 disp("(i) Resolution = 2^n = 2^8 = 256")
6 disp("i.e. the output voltage can have 256 different
      values including zero.")
7 disp("(ii) V_0FS = Full scale output voltage")
8 disp("          = 2.55 V")
9 disp("Therefore, Resolution = V_0FS / 2^n - 1 =
      2.55 / 2^8 - 1 = 10mV / 1LSB")
10 disp("Thus an input change of 1LSB causes the output
      to change by 10mV")

```

Scilab code Exa 9.33 output voltage

```

1 //Example 9.33
2 clc
3 disp("For given DAC,")
4 disp(" n = 4")
5 disp("Therefore, V_0FS = 15 V")

```

```

6 disp(" Therefore , Resolution = V_0FS / 2^n - 1 = 1V
    / LSB")
7 disp(" Therefore , V0 = Resolution * D")
8 disp(" Now D = Decimal of 0110 = 6")
9 disp(" Therefore , V0 = 1V / LSB * 6 = 6 V")

```

Scilab code Exa 9.34 find VoFS

```

1 //Example 9.34
2 clc
3 disp(" Resolution = V_0FS / 2^n - 1")
4 disp(" Therefore , 20 = V_0FS / 2^n - 1")
5 disp(" Therefore , V_0FS = 5.1 V")
6 disp(" D = Equivalent of 10000000 = 128")
7 disp(" Therefore , V0 = Resolution * D = 20 * 128 =
    2.56 V")

```

Scilab code Exa 9.35 find out stepsize and analog output

```

1 //Example 9.35
2 clc
3 disp(" For given DAC, n = 4, V_0FS = +5 V")
4 disp(" Resolution = V_0FS / 2^n - 1 = 1/3 V/LSB")
5 disp(" Therefore , V0 = Resolution * D")
6 disp(" For D = Decimal of 10000 = 8")
7 disp(" V0 = 1/3 * 8 = 2.6667 V")
8 disp(" For D = Decimal of 1111 = 15")
9 disp(" V0 = 1/3 * 15 = 5 V")

```

Scilab code Exa 9.36 find output voltage

```

1 //Example 9.36
2 clc
3 disp("For 12-bit DAC, step size is 8 mV")
4 v=(8*10^-3)*((2^12)-1)
5 format(6)
6 disp(v," V_0FS = 8 mV * 2^12 - 1 =")
7 r=((8*10^-3)/32.76)*100
8 format(8)
9 disp(r,"% Resolution = 8mV/32.76V * 100 =")
10 q=(8*10^-3)*1389
11 format(7)
12 disp(q,"The output voltage for the input
    010101101101 is = 8mV * 1389 =")

```

Scilab code Exa 9.38 find resolution and digital output

```

1 //Example 9.38.
2 clc
3 disp("(a) From equation(1) we have,")
4 r=2^8
5 format(4)
6 disp(r," Resolution = 2^8 =")
7 disp("and from equation(2) we have,")
8 disp(" Resolution = 5.1V/(2^8 - 1) = 20 mV/LSB")
9 disp("Therefore, we can say that to change output by
    1 LSB we have to change input by 20 mV")
10 disp("(b) For 1.28 V analog input, digital output
    can be calculated as,")
11 d=1.28/(20*10^-3)
12 format(3)
13 disp(d,"D (in LSBs) = 1.28V / 20 mV/LSB =")
14 disp("The binary equivalent of 64 is 0100 0000")

```

Scilab code Exa 9.39 calculate quantizing error

```
1 //Example 9.39
2 clc
3 disp("From equation(3) we get")
4 qe=(4.095/(4095*2))*10^3
5 format(4)
6 disp(qe,"Q-E(in mV) = 4.095 / (4096-1)*2 =")
```

Scilab code Exa 9.40 dual scope ADC

```
1 //Example 9.40
2 clc
3 disp("We know that,")
4 disp("t2 = (V1/VR)*t1")
5 t2=83.33
6 format(6)
7 disp(t2,"(i) t2(in ms) = (100/100)*83.33 =")
8 disp("(ii) V1 = 200 mV")
9 t2=83.33*2
10 disp(t2,"Therefore , t2(in ms) = (200/100)*83.33 =")
```

Scilab code Exa 9.41 find digital output of ADC

```
1 //Example 8.41
2 clc
3 disp("The digital output is given as,")
4 disp("Digital output = (Counts/Second)*t1*(V_i/V_R)"
5      )
6 disp("Now Clock frequency = 12 kHz")
7 disp("i.e. = 12000 counts/second")
8 d=12000*83.33*(100/100)*10^-3
9 format(5)
```

```
9 disp(d," Therefore , Digital output(in counts) =  
    12000*83.33*(100/100)*10-3 =")
```

Scilab code Exa 9.42 find conversion time

```
1 //Example 9.42.  
2 clc  
3 disp("          f = 1 MHz")  
4 disp(" Therefore , T = 1/f = 1 / 1*106 = 1 usec")  
5 disp("          n = 8")  
6 tc=1*(8+1)  
7 format(2)  
8 disp(tc," Therefore , T_C(in usec) = T*(n+1) =")
```

Scilab code Exa 9.43 find maximum frequency of input sine wave

```
1 //Example 9.43  
2 clc  
3 disp("The maximum frequency is given by,")  
4 f=1/(2*%pi*(9*10-6)*28)  
5 format(6)  
6 disp(f," f_max(in Hz) = 1 / 2*pi*(T_C)*2n =")
```

Chapter 10

Voltage Regulators

Scilab code Exa 10.1 find line and load regulation and ripple refection

```
1 //Example 10.1
2 clc
3 disp("Z_Z = 7 ohm, R3 = 330 ohm, V_0 = 4.7 V,
      V_in = 15 V")
4 disp("The specified change in V_in is 10%,")
5 vin=0.1*15
6 format(4)
7 disp(vin," Therefore , deltaV_in(in V) = 10% of V_in
      =")
8 vo=(1.5*7)/330
9 format(8)
10 disp(vo," Therefore , deltaV_0(in V) = deltaV_in*Z_Z
      / R3 =")
11 lr=0.03181*100/4.7
12 format(6)
13 disp(lr," Therefore , Line regulation(in percentage)
      = deltaV_0*100 / V_0 =")
14 disp("For I_L(max) = 50 mA,")
15 dvo=(20*7*50*10^-3)/330
16 format(8)
17 disp(dvo," Therefore , deltaV_0(in V) = I_L(max)*R_S*
```

```

    Z_Z / R3 =")
18 lr=0.02121*100/4.7
19 format(7)
20 disp(lr,"Therefore , Line regulation(in precentage)
    = deltaV_0*100 / V_0 =")
21 disp("Now    V_R(out) = V_R(in)*Z_Z / R3")
22 zz=7/330
23 format(8)
24 disp(zz,"Therefore , V_R(out)/V_R(in) = Z_Z/R3 =")
25 rr=20*log10(0.02121)
26 format(6)
27 disp(rr,"Therefore , RR(in dB) = 20*log(0.02121) = "
    )

```

Scilab code Exa 10.2 design op amp series voltage regulator

```

1 //Example 10.4
2 clc
3 disp("R1 = 5 k-ohm, R2 = 10 k-ohm")
4 disp("The IC is 7808 i.e. V_reg = +8 V")
5 vt=8*(3)
6 format(3)
7 disp(vt,"Therefore , V_out(in V) = V_reg*[1 + R2/R1]
    =")
8 disp("Now R2 = 1 k-ohm then,")
9 vo=8*(1+(1/5))
10 format(4)
11 disp(vo,"V_out(in V) = 8*[1 + 1/5] =")
12 disp("Thus the V_out can be varied from 9.6 V to 24
    V, by varing R2 from 1 k-ohm to 10 k-ohm.")

```

Scilab code Exa 10.4 calculate output voltage

```

1 //Example 10.4
2 clc
3 disp("R1 = 5 k-ohm,      R2 = 10 k-ohm")
4 disp("The IC is 7808 i.e. V_reg = +8 V")
5 vo=8*3
6 format(3)
7 disp(vo,"Therefore ,   V_out(in V) = V_reg*[1 + R2/R1]
      =")
8 disp("Now  R2 = 1 k-ohm then,")
9 vou=8*(1+(1/5))
10 format(4)
11 disp(vou,"V_out(in V) =")
12 disp("Thus the V_out can be varied from 9.6 V to 24
      V, by varing R2 from 1 k-ohm to 10 k-ohm")

```

Scilab code Exa 10.7 determine regulated output voltage

```

1 //Example 10.7
2 clc
3 disp("The resistance used are,")
4 disp("  R1 = 220 ohm  and  R2 = 1.5 k-ohm")
5 disp(" while for LM 317,  I_ADJ = 100 uA")
6 disp(" Therefore ,   V_0 = 1.25*[1+R2/R1] + I_ADJ*R2")
7 vo=(1.25*(1+((1.5*10^3)/220)))+(100*1.5*10^-3)
8 format(5)
9 disp(vo,"Therefore ,   V_0(in V) =")

```

Scilab code Exa 10.8 find the range

```

1 //Example 10.8
2 clc
3 disp("For LM 317, the current I_ADJ = 100 uA")
4 disp("When R2 is maximum i.e. R2 = 0 then,")

```

```
5 disp(" V_0 = 1.25*[1+R2/R1] + I_ADJ*R2 = 1.25 V")
6 disp("When R2 is maximum, i.e. R2 = 10 k-ohm then")
7 vo=(1.25*(1+((10*10^3)/820)))+(100*10*10^-3)
8 format(6)
9 disp(vo," V_0(in V) = ")
10 disp("Thus the output voltage can be varied in the
      range 1.25 V to 17.49 V")
```
