Scilab Textbook Companion for Analog And Digital Electronics by U. A. Bakshi And A. P. Godse[1]

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

July 31, 2019

[1]Funded 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

List of Figures

Chapter 1

Special Diodes

Scilab code Exa 1.1 `current through LED`

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

Scilab code Exa 1.2 `transition capacitance and constant K`

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

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

Chapter 2

Frequency Response

Scilab code Exa 2.1 maximum voltage gain

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

Scilab code Exa 2.2 gain of the amplifier

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

Scilab code Exa 2.3 `gain of an amplifier`

```
//Example 2.3
clc
format(7)
a=200*sqrt(2)
disp(a,"We know that      A_mid = 3dB gain x sqrt(2)
   =")
am=282.84/(sqrt(1+(((10/2)^2))))
format(6)
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`

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

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

Scilab code Exa 2.5 `low frequency response of the FET amplifier`

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

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

Scilab code Exa 2.6 high frequency response of the amplifier

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

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

Scilab code Exa 2.7 high frequency response of the amplifier

```
//Example 2.7
clc
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)")
disp("       Av = -gm * RD")
disp("Here, RD should br replaced by RD || RL")
av=-6*2
disp(av,"Therefore,     Av = -gm*(RD||RL) =")
cin=2*(12+1)  // in pF
disp(cin,"C_in(miller)(in pF) = C_gd*(Av+1) = C_rss*(Av+1) =")
format(6)
cout=(2*13)/12  // in pF
disp(cout,"C_out(miller)(in pF) = C_gd*(Av+1) / Av = ")
disp("G_gs = C_iss - C_rss = 4 pF")
disp("We know analyze input and output network for critical frequency")
disp("     f_c(input) = 1 / 2*pi*RS*CT")
disp("                = 1 / 2*pi*RS*[C_gs+C_in(miller)]")
format(4)
fc1=(1/(2*%pi*100*30*10^-12))*10^-6  // in MHz
disp(fc1,"     f_c(input)(in MHz)= ")
```

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

Chapter 3

Feedback Amplifiers

Scilab code Exa 3.1 gain fL and fH

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

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

```
disp("Therefore,  1+beta*A = 100")
b=99/1000
format(6)
disp(b,"Therefore,      beta =")
disp("Gain of the amplifier with feedback is given as")
avf=1000/100
disp(avf,"      A_Vf = A_V / 1+beta*A_V =")
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")
vsf=10*100*10^-3  // in V
disp(vsf,"      V_sf(in V) = Vs * 100 =")
disp("(c) Second harmonic distortion is reduced by factor 1 + beta*A")
d2f=(0.1/100)*100 // in percentage
disp(d2f,"      D_2f(in percentage) = D_2 / 1+beta*A =")
```

Scilab code Exa 3.3 `beta and Af`

```
//Example 3.3
clc
disp("(a) We know that")
disp("             dAf/Af = 0.1/1+beta*A * dA/A")
disp("Therefore,  1+beta*A = 37.5")
b=(36.5/2000)*100 // in percentage
format(6)
disp(b,"Therefore,  beta(in percentage) =")
af=2000/(1+(0.01825*2000))
disp(af,"(b)      Af = A / 1+beta*A =")
```

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

```
//Example 3.4
clc
disp("Step 1: Identity topology")
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
   .")
disp("")
disp("Step 2 and Step 3: Find input and output
   circuit.")
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 Re1 across the output. The resultant
   circuit is shown in Fig.3.32")
disp("")
disp("Step 4: Find open loop voltage gain(A_v)")
format(5)
rl2=(4.7*10.1)/(4.7+10.1) // in k-ohm
disp(rl2,"  R_L2(in k-ohm) = R_c2 || (R_e1+Rf) =")
disp("  A_i2 = -hfe = -100")
disp("  R_i2 = hie = 1100 ohm")
format(7)
av2=(-100*3.21*10^3)/1100
disp(av2,"  A_v2 = A_i2*R_L2 / R_i2 =")
disp("  A_i1 = -hfe = -100")
format(5)
rl1=(22*220*22*1.100)/((220*22*1.100)+(22*22*1.100)
   +(22*220*1.100)+(22*220*22)) // in ohm
disp(rl1*10^3,"  R_L1(in ohm) = R_c1 || R3 || R4 ||
   R_i2 =")
ri1=1.1+(101*((0.1*10)/(0.1+10))) // in k-ohm
format(5)
disp(ri1,"  R_i1(in k-ohm) = hie + (1+hfe)*R_e1eff =
```

```
                    where Re1eff = (R_e1 || Rf)")
av1=(-100*995)/(11.099*10^3)
disp(av1,"Therefore,  A_v1 = A_i1*RL1 / Ri1 =")
disp("The overall voltage gain without feedback is
    given as,")
av=-291.82*-8.96
format(7)
disp(av,"  Av = A_v1 * A_v2 =")
disp("The overall voltage gain taking Rs in account
    is given as,")
aV=(2614.7*11.099*10^3)/((11.099*10^3)+100)
format(8)
disp(aV,"  Av = Vo / Vs = Av*R_i1 / R_i1+Rs =")
disp("")
disp("Step 5: Calculate beta")
disp("Looking at Fig.3.33.")
beta=100/(100+(10*10^3))
format(7)
disp(beta,"  beta = Vf / Vo =")
d=1+(0.0099*2591.35)
format(6)
disp(d,"  D = 1 + beta*Av =")
avf=2591.35/26.65
disp(avf,"  A_vf = Av/D =")
rif=26.65*11.099   // in k-ohm
format(8)
disp(rif,"  R_if(in k-ohm) = R_i1 * D =")
riff=(295.788*220*22)/((220*22)+(295.788*22)
    +(295.788*220))   // in k-ohm
format(6)
disp(riff,"  R''_if(in k-ohm) = R_if || R1 || R2 =")
disp("  R_of = Ro / D = infinity / D = infinity")
disp("Therefore,  R''_of = R''_o / D          where
    R''_o = R_L2")
roff=(3.21*10^3)/26.65   // in omh
format(7)
disp(roff,"Therefore,  R''_of(in ohm) = ")
```

Scilab code Exa 3.5 `Avf and Rif and Rof`

```
//Example 3.5
clc
disp("Step 1: Identity topology")
disp("  The feedback voltage is applied across R1
   (100 ohm), which is in series with input signal.
   Hence feedback is voltage series feedback.")
disp("")
disp("Step 2 and Step 3: Find input and output
   circuit")
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")
disp("")
disp("Step 4: Find the open loop voltage gain (Av)")
rl2=(4.7*4.8)/(4.7+4.8)  // in k-ohm
format(5)
disp(rl2,"  R_L2(in k-ohm) =")
disp("Since  h_oe = h_re = 0 we can use approximate
   analysis")
disp("  A_i2 = -hfe = -50")
disp("  R_i2 = hie = 1.1 k-ohm")
av2=(-50*2.37)/1.1
format(7)
disp(av2,"  A_v2 = A_i2*R_L2 / R_i2 =")
rl1=(10*47*33*1.1)/((47*33*1.1)+(10*33*1.1)
   +(10*47*1.1)+(10*47*33))  // in ohm
format(5)
disp(rl1*10^3,"  R_L1(in ohm) =")
```

```
disp("   A_i1 = -hfe = -50")
ri1=1.1+(51*((0.1*4.7)/(4.8)))   // in k-ohm
format(6)
disp(ri1,"   R_i1(in k-ohm) = hie + (1+hfe)*Re =")
av1=(-50*942)/(6.093*10^3)
format(5)
disp(av1,"   A_v1 = A_i1*R_L1 / R_i1 =")
av=-7.73*-107.73
format(7)
disp(av,"Therefore,   A_v = A_v1 * A_v2 =")
disp("")
disp("Step 5: Calculate beta and D")
disp("   beta = R1 / R1+R2 = 1/48")
d=1+(832.75/48)   // in ohm
format(6)
disp(d,"   D(in ohm) = 1 + A*beta =")
disp("")
disp("Step 5: Calculate A_vf, R_of and R_if")
avf=832.75/18.35
disp(avf,"   A_vf = A_v / D =")
rif=6.093*18.35   // in k-ohm
disp(rif,"   R_if(in k-ohm) = R_i1 * D =")
rof=(2.37*10^3)/18.35   // in ohm
format(7)
disp(rof,"   R_of(in ohm) = R_o / D =")
```

Scilab code Exa 3.6 `Avf and Rif and Rof`

```
//Example 3.6
clc
disp("Step 1: Identify topology")
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")
disp("")
```

```
disp("Step 2 and step 3: Find input and output circuit")
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")
disp("")
disp("Step 4: Find the open loop voltage gain (Av)")
rl2=(2.2*57.5)/(2.2+57.5)  // in k-ohm
format(6)
disp(rl2,"  R_L2(in k-ohm) = R_c2 || (Rf + R_e1) =")
disp("Since hoe*R_L2 = 10^-6*2.119 k-ohm = 0.002119 is less than 0.1 we use approximate analysis.")
disp("  A_i2 = -h_fe = -200")
disp("  R_i2 = hie = 2 k-ohm")
av2=(-200*2.119)/2
disp(av2,"  A_v2 = A_i2*R_L2 / R_i2 =")
rl1=(120*2)/(122)  // in k-ohm
disp(rl1,"  R_L1(in k-ohm) = R_C1 || R_i2 =")
disp("Since hoe*R_L1 = 10^-6*1.967 = 0.001967 is less than 0.1 we use approximate analysis.")
disp("  A_i1 = -hfe = -200")
ri1=2+(201*((1.5*56)/(57.5)))  // in k-ohm
format(7)
disp(ri1,"  R_i1(in k-ohm) = hie + (1+hfe)*Re =")
av1=(-200*1.967)/295.63
format(5)
disp(av1,"Therefore,  A_v1 = A_i1*R_L1 / R_i1 =")
disp("The overall gain without feedback is")
av=-1.33*-211.9
format(7)
disp(av,"  Av = A_v1 * A_v2 =")
disp("")
disp("Step 5: Calculate beta")
beta=1.5/57.5
```

```
format(6)
disp(beta,"  beta = Vf / Vo =")
disp("")
disp("Step 6: calculate D, A_vf, R_if, R_of")
d=1+(0.026*281.82)
disp(d,"  D = 1 + Av*beta =")
avf=281.82/8.327
disp(avf,"Therefore,  A_vf = Av / D =")
ri=(295.63*150)/(295.63+150)  // in k-ohm
format(5)
disp(ri,"  Ri(in k-ohm) = R_i1 || R =")
rif=99.5*8.327  // in k-ohm
format(7)
disp(rif,"  R_if(in k-ohm) = Ri *D =")
disp("  Ro = 1/hoe = 1 M-ohm")
rof=((1*10^6)/8.327)*10^-3  // in k-ohm
format(4)
disp(rof,"  R_of(in k-ohm) = Ro / D =")
ro=(1000*2.119)/(2.119+1000)  // in k-ohm
format(7)
disp(ro,"  R''o(in k-ohm) = Ro || R_c2 || (Rf+R_e1)
   = Ro || R_L2 =")
rof=(2.1145*10^3)/8.327  // in ohm
format(4)
disp(rof,"  R''_of(in ohm) = R''o / D =")
```

Scilab code Exa 3.7 `D and Avf and Rif and Rof`

```
//Example 3.7
clc
disp("Step 1: Identity topology")
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.")
disp("")
disp("Step 2 and Step 3: Find input and output
    circuit.")
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.")
disp("")
disp("Step 4: Replace FET with its equivalent
    circuit as shown in fig.3.51")
disp("")
disp("Step 5: Find open loop transfer gain.")
disp("  Av = Vo / Vs = A_v1*A_v2")
disp("  A_v2 = -u*R_L2 / R_L2+r_d")
rl2=(10.2*47)/(10.2+47)  // in k-ohm
format(5)
disp(rl2,"where  R_L2(in k-ohm) =")
av2=(-40*8.38)/(8.38+10)
format(7)
disp(av2,"Therefore,  A_v2 =")
disp("  A_v1 = u*R_Deff / r_d+R_Deff+(1+u)*R_seff")
rdeff=(47*1000)/(47+1000)  // in k-ohm
format(6)
disp(rdeff,"where  R_Deff(in k-ohm) = R_D || R_G2 ="
    )
disp("  R_seff = 200 || 10 K")
av1=(-40*44.98*10^3)/((10*10^3)+(44.89*10^3)
    +(41*((10*0.2)/(10.2))))
disp(av1,"  A_v1 =")  // answer in textbook is
    wrong
oav=-28.59*-18.237
format(7)
disp(oav,"Therefore,  Overall Av =")
disp("")
disp("Step 6: Calculate beta")
```

```
beta=200/(10.2*10^3)
disp(beta,"  beta = Vf / Vo =")
disp("")
disp("Step 7: Calculate D, A_vf, R_if, R''_of")
d=1+(0.0196*521.39)
format(6)
disp(d,"  D = 1 + Av*beta =")
avf=521.39/11.22
disp(avf,"  A_vf = Av / D =")
disp("  Ri = R_G = 1 M-ohm")
rif=11.22
disp(rif,"  R_if(in M-ohm) = Ri * D =")
disp("  Ro = r_d = 10 k-ohm")
ro=(10*8.38)/(18.38)  // in k-ohm
disp(ro,"  R''o(in k-ohm) = r_d || R_L2 =")
rof=(4.559*10^3)/11.22  // in ohm
format(4)
disp(rof,"  R''_of(in ohm) = R''o / D =")
```

Scilab code Exa 3.8 `voltage gain`

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

```
format(6)
disp(av,"  Av =")
avm=-33.11^3
disp(avm,"Av = Overall voltage gain = |A_vmid|^3 =") // answer in textbook is wrong
beta=50/(10^6)
format(7)
disp(beta,"  beta = Vf / Vo = -R_1 / R_g = -R_1 / R_1+R_2 =")
d=1+((-5*10^-5)*-36306)
format(6)
disp(d,"  D = 1 + |Av|*beta =")
avf=-36306/2.8153
disp(avf,"  A_vf = Av / D =")
```

Scilab code Exa 3.9 `Avf`

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

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

Scilab code Exa 3.10 Avf and Rif and Rof

```
//Example 3.10
disp("Step 1: Identify topology")
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.")
disp("")
disp("Step 2 and Step 3: Find input and output
   circuit.")
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.")
disp("")
disp("Step 4: Replace FET with its equivalent
   circuit as shown in fig.3.55.")
disp("")
disp("Step 5: Find open loop transfer gain.")
disp("  Av = Vo / Vs = A_v1 * A_v2")
disp("  A_v2 = -u*R_L2 / R_L2+r_d")
rl2=(10.3*22)/(10.3+22)  // in k-ohm
```

```
format(3)
disp(rl2,"where R_L2(in k-ohm) =")
av2=(-50*7)/17
format(6)
disp(av2," A_v2 =")
disp(" A_v1 = u*R_Deff / r_d+R_Deff+(1+u)*R_seff")
rdeff=(22*1000)/(22+1000)  // in k-ohm
disp(rdeff," R_Deff(in k-ohm) = R_D || R_G2 =")
disp(" R_seff = 330 || 10K")
av1=(-50*21.53)/(10+21.53+(51*((0.33*10)/(10+0.33))))
disp(av1,"Therefore, A_v1 =")
av=-20.59*-22.51
disp(av," Overall Av = A_v1 * A_v2 =")
disp("")
disp("Step 6: Calculate beta")
beta=330/(330+10000)
format(7)
disp(beta," beta = Vf / Vo = Rs / Rs+Rf =")
disp("")
disp("step 7: Calculate D, A_vf, R_if, R''_of")
d=1+(0.0319*463.5)
disp(d," D = 1 + Av*beta =")
avf=463.5/15.785
format(6)
disp(avf," A_vf = Av / D =")
disp("Ri = R_G = 1 M-ohm")
rif=15.785
format(7)
disp(rif," R_if(in k-ohm) = Ri * D =")
ro=(10*7)/(10+7)  // in k-ohm
format(6)
disp(ro," R''o(in k-ohm) = rd || R_L2 =")
rof=(4.118*10^3)/15.785  // in ohm
format(4)
disp(rof," R''_of(in ohm) = R''o / D =")
```

Scilab code Exa 3.11 voltage gain and input and output resistance

```
//Example 3.11
clc
disp("Step 1: Identify topology")
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.")
disp("")
disp("step 2 and Step 3: Find input and output circuit.")
disp(" To find input circuit, set Vo = 0 (connecting C2 to ground), which gives parllel combination of Re with Rf at E1. To find output ciruit, set Ii = 0 (opening the input node E1 at emitter of Q1), which gives series combination od Rf and R_e1 across the output. The resultant circuit is shown in fig.3.57")
disp("")
disp("Step 4: Find open loop voltage gain (Av)")
rl2=(4.7*3.42)/(4.7+3.42)  // in k-ohm
format(5)
disp(rl2," R_L2(in k-ohm) = R_c2 || (Rs+R) =")
disp(" A_i2 = -hfe = -50")
disp("R_i2 = hie = 1000 ohm = 1 k-ohm")
av2=-50*1.98
format(3)
disp(av2," A_v2 = A_i2*R_L2 / R_i2 =")
disp(" A_i1 = -hfe = -50")
format(7)
rl1=((10*100*22*1)/((100*22)+(10*22)+(10*100)+(10*100*22)))*10^3  // in ohm
disp(rl1," R_L1(in ohm) = R_c1 || R3 || R4 || R_i2
```

```
    =")
disp("  R_i1 = h_ie + (1+h_fe)*R_e1eff")
re1=1+(51*((3.3*0.12)/(3.42)))  // in k-ohm
format(4)
disp(re1,"where  R_e1eff(in k-ohm) = Rs || R =")
av1=(-50*865.46)/6900
format(5)
disp(av1,"  A_v1 = A_i1*R_L1 / R_i1 =")
disp("The overall voltage gain,")
av=-6.27*-99
format(7)
disp(av,"  Av = A_v1 * A_v2 =")
disp("")
disp("Step 5: Calculate beta")
beta=120/(120+3300)
format(6)
disp(beta,"  beta = Vf / Vo = Rs / Rs+R =")
disp("")
disp("Step 6: Calculate D, A_vf, R_if, R_of and R''
    _of")
d=1+(0.035*620.73)
format(7)
disp(d,"  D = 1 + Av*beta =")
avf=620.73/22.725
format(5)
disp(avf,"  A_vf = Av / D =")
rif=6.9*22.725  // in k-ohm
format(6)
disp(rif,"  R_if(in k-ohm) = R_i1 * D =")
disp("  R_of = Ro / D = infinity")
rof=(1.98*10^3)/22.725  // in ohm
disp(rof,"  R''_of(in ohm) = R''o / D = R_L2 / D =")
```

Scilab code Exa 3.12 Ai and beta and Aif

```
//Example 3.12
clc
disp("Step 1: Identify topology")
disp("  The feebdack is given from emitter of Q2 to the base of Q2. If Io = 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.")
disp("")
disp("Step 2 and Step 3: Find input and output ")
disp("  The input circuit of the amplifier without feedback is obtained by opening the output loop at the emitter of Q2(Io = 0). This places R''(5 K) in series with Re from base to emitter of Q1. The output circuit is found by shorting the input node, i.e. making Vi = 0. This places R'' (5 K) in parallel with Re. The resultant equivalent circuit is shown in fig.3.59 ")
disp("")
disp("Step 4: Find open circuit transfer gain.")
disp("  A_I = Io / Is = -Ic/I_b2 * I_b2/I_c1 * I_c1/I_b1 * I_b1/Is")
disp("We know that  -I_c2 / I_b2 = A_i2 = -hfe = -50 and")
disp("  -I_c1 / I_b1 = A_i1 = -hfe = 50")
disp("  I_c1 / I_b1 = 50")
disp("Looking at fig.3.59 we can write,")
disp("  I_b2 / I_c1 = -R_c1 / R_c1+R_i2 ")
ri2=1.5+(51*((5*0.5)/(5.5)))  // in k-ohm
format(8)
disp(ri2,"where  R_i2(in k-ohm) = h_ie + (1+h_fe)*(R_e2||R'') =")
x1=-2/(2+24.6818)
disp(x1,"  I_b2 / I_c1 =")
disp("  I_b1 / Is = R / R+R_i1     where R = Rs||(R''+R_e2) ")
r=((1*5.5)/(1+5.5))*10^3  // in ohm
```

```
format(9)
disp(r,"Therefore,  R(in ohm) =")
disp("and  R_i1 = h_ie + (1+h_fe)*R_e1 = 16.8 k-ohm"
    )
x1=846.1538/(846.1538+(16.8*10^3))
format(8)
disp(x1,"Therefore,  I_b1 / Is =")
ai=50*0.07495*50*0.04795
format(7)
disp(ai,"  A_I =")
disp("")
disp("Step 5: Calculate beta")
beta=500/(500+(5*10^3))
disp(beta,"  beta = If / Io = R_e2 / R_e2|R'' =")
disp("")
disp("Step 6: Calculate D, A_If")
d=1+(0.0909*8.9848)
disp(d,"  D = 1 + A_I*beta =")
aif=8.9848/1.8168
disp(aif,"  A_If = A_I / D =")
```

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

```
//Example 3.13
clc
disp("Step 1: Identify topology")
disp("  The feedback voltage is applied across R1
    (150 ohm), which is in series with input signal.
    Hence feedback is voltage series feedback.")
disp("")
disp("Step 2 and Step 3: Find input and output
    circuit")
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 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.3.61.")
disp("")
disp("Step 4: Find the open loop voltage gain (Av)")
rl2=(4.7*15.15)/(4.7+15.15)  // in k-ohm
format(5)
disp(rl2,"  RL2(in k-ohm) =")
disp("Since  hoe = hre = 0, we can use approximate analysis.")
disp("  A_i2 = -h_fe = -500")
disp("  R_i2 = h_ie = 1100 ohm")
av2=(-500*3.59*10^3)/1100
disp(av2,"  A_v2 = A_i2*R_L2 / R_i2 =")
rl1=((10*47*33*1.1)/((47*33*1.1)+(10*33*1.1)+(10*47*1.1)+(10*47*33)))*10^3  // in ohm
disp(rl1,"  R_L1(in ohm) = 10K || 47K || 33K || R_i2 =")
disp("  A_i1 = -h_fe = -500")
ri1=1.1+(501*((0.15*15)/(0.15+15)))  // in k-ohm
disp(ri1,"  R_i1(in k-ohm) = h_ie + (1+h_fe)*Re =")
av1=(-500*942)/(75.5*10^3)
format(6)
disp(av1,"  A_v1 = A_i1*R_L1 / R_i1 =")
av=-6.238*-1632
disp(av,"  Av = A_v1 * A_v2 =")
disp("")
disp("Step 5: Calculate beta and D")
beta=150/(150+15000)
format(7)
disp(beta,"  beta = R1 / R1+R2 =")
d=1+(10180*0.0099)
format(8)
disp(d,"  D = 1 + A*beta =")
disp("")
disp("Step 6: Calculate A_vf, R_of and R_if")
avf=10180/101.782
```

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

Scilab code Exa 3.14 `gain and new bandwidth`

```
//Example 3.14
clc
disp("Given: A_vmid = 500, f_L = 100 kHz, f_H = 20 kHz and beta = 0.01")
avf=500/(1+(0.01*500))
format(6)
disp(avf," A_vf = A_vmid / 1+beta*A_vmid =")
flf=100/(1+(0.01*500)) // in Hz
disp(flf," f_Lf(in Hz) = f_L / 1+beta*A_vmid =")
fhf=20*(1+(0.01*500)) // in kHz
disp(fhf," f_Hf(in kHZ) = f_H * (1 + beta*A_vmid) = ")
bw=120-0.01667 // in kHZ
format(9)
disp(bw," BW_f(in kHz) = f_Hf - f_Lf =")
```

Scilab code Exa 3.15 `show voltage gain with feedback`

```
//Example 3.15
clc
disp("Step 1: Identify topology")
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.")
disp("")
disp("Step 2 and Step 3: Find input and output circuit.")
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.")
disp("")
disp("Step 4: Replace transistor with its h-parameter equivalent as shown in fig.3.64.")
disp("")
disp("Step 5: Find open loop transfer gain.")
disp("  From quation(5) of section 3.9.1 we have")
disp("   A_vf = Io*R_L / Vs = G_Mf * R_L")
disp("         = -h_fe*R_L / R''s+h_ie+(1+h_fe)*Re")
disp("Here  R''s = Rs || R1 || R2")
disp("             = Rs || Rb          because R_b = R1 || R2")
disp("Therefore,  Vo / Vs = Vo/Vi * Vi/Vs")
disp("where       Vi / Vs = Rb / Rs+Rb")
disp("Therefore,  Vo / Vs = (-h_fe*R_L / R''s+h_ie+(1+h_fe)*Re) * (Rb / Rs+Rb)")
disp("Dividing both numerator and denominator by Rs+Rb we get,")
disp("   A_vf = Vo / Vs = [-h_fe*Rc*(Rb/Rb+Rs)] / R''s+h_ie+(1+h_fe)*Re     because RL = Rc")
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`

```
//Example 3.16
clc
disp("Refer example 3.15")
disp("  A_vf = -h_fe*Rc*[1/1+(Rs/Rb)] / R''s+h_ie+(1+h_fe)*Re     where R''s = Rs||R1||R2")
avf=(-50*(1.8*10^3)*[1/(1+(1000/4272))])/(810+1000+((1+50)*1000))
format(5)
disp(avf,"  A_vf =")
gmf=-1.38/(1.8*10^3)
format(9)
disp(gmf,"  G_Mf = A_vf / R_L =")
disp("  beta = Vf / Io = Ie*Re / Io = -Io*Re / Io = -Re = -1 K")
disp("  G_Mf = G_M / 1+beta*G_M")
gm=1/((1/(-7.66*10^-4))+1000)
format(10)
disp(gm,"Therefore,  G_M =")
d=1+(-1000*-3.2735*10^-3)
format(7)
disp(d,"  D = 1 + G_M*beta =")
ri=(1+1.36)  // in k-ohm
format(5)
disp(ri,"  R_i(in k-ohm) = Rs+(h_ie+Re) || R_D =")
rif=2.36*4.2735  // in k-ohm
format(3)
disp(rif,"  R_if(in k-ohm) = R_i * D =")
disp("  R_o = infinity")
disp("  R_of = R_o * D = infinity")
disp("  R''_of = R_of || R_L = R_L = 1.8 k-ohm")
```

Scilab code Exa 3.17 `feedback factor and Rif and Rof`

```
//Example 3.17
clc
```

```
disp("Refer example 3.15")
disp("   A_vf = -h_fe*Rc*[1/1+(Rs/Rb)] / R''s+h_ie
   +(1+h_fe)*Re       where R''s = Rs||R1||R2")
avf=(-50*(4*10^3)*[1/(1+(1000/9000))])
   /(900+1000+((1+150)*1000))
format(6)
disp(avf,"   A_vf =")
gmf=-1.177/(4*10^3)
format(9)
disp(gmf,"   G_Mf = A_vf / R_L =")
disp("   beta = Vf / Io = Ie*Re / Io = -Io*Re / Io =
   -Re = -1 K")
disp("   G_Mf = G_M / 1+beta*G_M")
gm=1/((1/(-2.943*10^-4))+1000)
format(9)
disp(gm,"Therefore,  G_M =")
d=1+(-1000*-4.17*10^-4)
format(6)
disp(d,"   D = 1 + G_M*beta =")
ri=1+((2*9)/(2+9))   // in k-ohm
disp(ri,"   R_i(in k-ohm) = Rs+(h_ie+Re) || R_D =")
rif=2.636*1.417   // in k-ohm
format(6)
disp(rif,"   R_if(in k-ohm) = R_i * D =")
disp("   R_o = infinity")
disp("   R_of = R_o * D = infinity")
disp("   R''_of = R_of || R_L = R_L = 4 k-ohm")
```

Scilab code Exa 3.18 gain with feedback and new bandwidth

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

```
format(3)
disp(avf,"  A_vf = A_v mid / 1+beta*A_v mid =")
flf=100/(1+(0.01*400))
disp(flf,"  f_Lf = f_L / 1+beta*A_v mid =")
fhf=(15)*(1+(0.01*400))  // in kHz
disp(fhf,"  f_Hf(in kHz) = f_H * (1+beta*A_v mid) =")
bw=75-0.02  // in kHz
format(6)
disp(bw,"  BW_f(in kHz) = f_Hf - f_Lf =")
```

Scilab code Exa 3.19 `overall voltage gain and bandwidth`

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

Chapter 4

Oscillators

Scilab code Exa 4.1 `C and hfe`

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

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

Scilab code Exa 4.2 frequency of oscillation

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

Scilab code Exa 4.3 R and C

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

Scilab code Exa 4.4 C and RD

```
//Example 4.4
clc
```

```
disp("Using the expression for the frequency")
disp("Now,  f = 1 / 2*pi*R*C*sqrt(6)")
f=(1/(sqrt(6)*2*%pi*9.7*5*10^6))*10^9  // in nF
format(5)
disp(f,"Therefore,  C(in nF) =")
disp("Now using the equation(27)")
disp("   |A| = g_m * R_L")
disp("Therefore,   |A| >= 29")
disp("Therefore,   g_m * R_L >= 29")
rl=(29/(5000*10^-6))*10^-3  // in k-ohm
format(4)
disp(rl,"Therefore,  R_L(in k-ohm) >= 29 / g_m =")
disp("   R_L = R_D*r_d / R_D+r_d")
rd=(40)/4.8823
format(5)
disp(rd,"   Therefore,  R_D(in k-ohm) = ")
disp("While for minimum value of R_L = 5.8 k-ohm")
disp("                               R_D = 6.78 k-ohm")
```

Scilab code Exa 4.5 `minimum and maximum R2`

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

```
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

```
//Example 4.6
clc
disp("The frequency is given by,")
disp("   f = 1 / 2*pi*sqrt(C*L_eq)")
leq=(2*10^-3)+(20*10^-6)
format(8)
disp(leq,"where  L_eq = L1 + L2 =")
disp("For  f = f_max = 2050 kHz")
format(5)
c=(1/(4*(%pi^2)*((2050*10^3)^2)*0.00202))*10^12  //
    in pF
disp(c,"Therefore,  C(in pF) =")
disp("For  f = f_min = 950 kHz")
c=(1/(4*(%pi^2)*((950*10^3)^2)*0.00202))*10^12  //
    in pF
format(6)
disp(c,"Therefore,  C(in pF) =")
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

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

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

Scilab code Exa 4.8 `frequency of oscillation`

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

Scilab code Exa 4.9 `calculate C`

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

```
disp("but    C_eq = C1*C2 / C1+C2    and C1 = C2 = C")
disp("Therefore,  C_eq = C / 2")
c=1.0132*2
format(6)
disp(c,"Therefore,  C(in nF) =")
```

Scilab code Exa 4.10 `series and parallel resonant freqency`

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

Scilab code Exa 4.11 `series and parallel resonant freqency`

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

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

Scilab code Exa 4.12 `Varify Barkhausen criterion and find frequency of oscillation`

```
//example 4.12.
clc
disp("From the given information we can write,")
disp("     A = -16*10^6/j*omega  and  beta =
   10^3/[2*10^3+j*omega]^2")
disp("To verify the Barkhausen condition means to
   verify whether |A*beta| = 1 at a frequency for
   which A*beta = 0 degree. Let us express, A*beta
   in its rectangluar form.")
disp("  A*beta = -16*10^6*10^3 / j*omega*[2*10^3+j*
   omega]^2 = -16*10^9 / j*omega*[4*10^6+4*10^3*j*
   omega+(j*omega)^2]")
disp("          = -16*10^9 / j*omega*[4*10^6+4*10^3*j
   *omega-omega^2]       as j*2 = -1")
disp("          = -16*10^9 / 4*10^6*j*omega+4*10^3*j
   ^2*omega^2-j*omega^3]")
disp("          = -16*10^9 / j*omega*[4*10^6-omega
   ^2]-[omega^2*4*10^3]")
disp("Rationalising the denominator function we get,
   ")
disp("  A*beta = -16*10^9[-omega^2*4*10^3 - j*omega
   *[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]]")
disp("Using (a-b)(a+b) = a^2 - b^2 in the denominator,")
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")
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")
disp("Now to have A*beta = 0 degree, the imaginary part of A*beta must be zero. This is possible when,")
disp("Therefore,  omega*(4*10^6 - omega^2) = 0")
disp("Therefore,  omega = 0  or  4*10^6 - omega^2 = 0")
disp("Therefore,  omega^2 = 4*10^6       Neglecting zero value of frequency")
disp("Therefore,  omega = 2*10^3 rad/sec")
disp("At this frequency |A*beta| can be obtained as,")
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")
ab=(2.56*10^20)/(2.56*10^20)
disp(ab,"  |A*beta| =")
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.")
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")
disp("i.e.       omega = 2*10^3  rad/sec")
disp("But        omega = 2*pi*f")
f=(2*10^3)/(2*%pi)  // in Hz
format(9)
disp(f,"Therefore,  f(in Hz) = omega / 2pi =")
```

Scilab code Exa 4.13 minimum and maximum values of R2

```
//Example 4.13
clc
disp("The frequency of the oscillator is given by,")
disp("   f = 1 / 2*pi*sqrt(R1*R2*C1*C2)")
disp("For  f = 20 kHz,")
r2=(1/(4*(%pi^2)*((20*10^3)^2)*(10*10^3)
   *((0.001*10^-6)^2)))*10^-3
format(5)
disp(r2,"Therefore,  R2(in k-ohm) =")
disp("For  f = 70 kHz,")
r2=(1/(4*(%pi^2)*((70*10^3)^2)*(10*10^3)
   *((0.001*10^-6)^2)))*10^-3
format(6)
disp(r2,"Therefore,  R2(in k-ohm) =")
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

```
//Example 4.14.
clc
disp("R = 6 k-ohm,  C = 1500 pF,  R_C = 18 k-ohm")
k=18/6
disp(k,"Now     K = R_C / R =")
disp("Therefore,   f = 1 / 2*pi*R*C*sqrt(6+4K)")
f=(1/(2*%pi*(6*10^3)*(1500*10^-12)*sqrt(6+12)))
   *10^-3  // in kHZ
format(6)
disp(f,"  f(in kHz) =")
hfe=(4*3)+23+(29/3)
```

```
disp(hfe,"   (h_fe)min = 4K + 23 + 29/K =")
```

Scilab code Exa 4.15 `range of frequency of oscillation`

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

Scilab code Exa 4.16 `frequency of oscillation`

```
//Example 4.16.
clc
format(6)
```

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

Scilab code Exa 4.17 `gain of the transistor`

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

Scilab code Exa 4.18 `new frequency and inductance`

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

```
disp(" where   C_eq = C1*C2 / C1+C2   but C1 = C2 =
   0.001 uF")
ceq=((0.001*10^-6)^2)/(2*(0.001*10^-6))  // in F
format(7)
disp(ceq," Therefore ,  C_eq(in F) =")
disp("  L = 5*10^-6 H")
f=(1/(2*%pi*sqrt(25*10^-16)))*10^-6  // in MHz
format(6)
disp(f," Therefore ,  f(in MHz) =")
disp("Now L is doubled i.e. 10 uH")
f1=(1/(2*%pi*sqrt(50*10^-16)))*10^-6  // in MHz
format(5)
disp(f1," Therefore ,  f(in MHz) =")
nf= 2*3.183
format(6)
disp(nf,"New frequency(in MHz) = 2*3.183 =")
l=(1/(4*(%pi^2)*((6.366*10^6)^2)*(5*10^-10)))*10^6
   // in uH
format(5)
disp(l," Therefore ,  L(in uH) =")
```

Scilab code Exa 4.19 new frequency of oscillation

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

Scilab code Exa 4.20 R and hfe

```
//Example 4.20
clc
disp("Refering to equation(1) of section 4.5.3, the input impedance is given by,")
disp("R''_i = R1 || R2 || h_ie")
disp("Now  R1 = 25 k-ohm,  R2 = 47 k-ohm,  and  h_ie = 2 k-ohm")
format(7)
ri=(25*47*2)/((47*2)+(25*2)+(25*47))  // in k-ohm
disp(ri,"Therefore,  R''_i(in k-ohm) =")
disp("   K = R_C / R")
disp("Now  R_C = 10 k-ohm          ...given")
disp("Now  f = 1 / 2*pi*R*C*sqrt(6+4K)")
disp("Therefore,  R*sqrt(6+4K) = 31830.989")
disp("Now  K = R_C / R = 10*10^3 / R")
disp("Therefore,  R*sqrt(6+(40*10*10^3/R)) = 31830.989")
disp("Therefore,  R^2*(6+(40*10*10^3/R)) = (31830.989)^2")
R=poly(0,'R')
p1=6*R^2+(40*10^3)*R-(31830.989)^2
t1=roots(p1)
ans1=t1(1)
format(6)
disp((-ans1)*10^-3,"Therefore,  R(in k-ohm)=            Neglecting negative value")
k=10/16.74
format(7)
disp(k,"Therefore,  K = R_C / R =")
disp("Therefore,  h_fe >= 4K + 23 + 29/K")
hfe=(4*0.5973)+23+(29/0.5973)
format(6)
disp(hfe,"  h_fe >=")
```

Scilab code Exa 4.21 `component values of wien bridge`

```
//Example 4.21
clc
disp("The frequency is given by,")
disp("   f = 1 / 2*pi*R*C")
disp("Let the resistance value to be selected as,")
disp("   R1 = R2 = R = 50 k-ohm")
disp("   f = 1 / 2*pi*50*10^3*C")
f=(1/(2*%pi*(50*10^3)*100))*10^9   // in nF
format(6)
disp(f,"   f(in nF) =")
disp("and   f_max = 1 / 2*pi*50*10^3*C")
c=(1/(2*%pi*(50*10^3)*(10*10^3)))*10^9   // in pF
disp(c,"   C(in nF) =")
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")
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

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

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

Scilab code Exa 4.23 series and parallel resonant freqency and Qfactor

```
//Example 4.23.
clc
f=(1/(2*%pi*sqrt(0.33*0.065*10^-12)))*10^-6  // in MHz
format(6)
disp(f,"(i)  f(in MHz) = 1 / 2*pi*sqrt(L*C) =")
ceq=0.065/1.065  // in pF
disp(ceq,"(ii)  C_eq(in pF) = C*C_M / C+C_M =")
fp=(1/(2*%pi*sqrt(0.33*0.061*10^-12)))*10^-6  // in MHz
disp(fp,"(i)  f_p(in MHz) = 1 / 2*pi*sqrt(L*C_eq) =")
pi=((1.121-1.087)/1.087)*100  // in percentage
disp(pi,"(iii)  % increase =")
q=(2*%pi*1.087*0.33*10^6)/(5.5*10^3)
format(8)
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

```
//Examle 4.24
clc
```

```
disp("(i)  Assume one perticular coupling direction for which,")
disp("  L_eq = L1 + L2 + 2M = 0.25 mH")
format(8)
f=(1/(2*%pi*sqrt(0.25*100*10^-15)))*10^-6  // in MHz
disp(f,"Therefore,  f(in MHz) = 1 / 2*pi*sqrt(L_eq*C) =")
disp("Let the direction of coupling is reversed,")
disp("  L_eq = L1 + L2 - 2M = 0.15 mH")
fd=(1/(2*%pi*sqrt(0.15*100*10^-15)))*10^-6  // in MHz
format(7)
disp(fd,"Therefore,  f''(in MHz) = 1 / 2*pi*sqrt(L_eq*C) =")
pc=((1.2994-1.00658)/1.00658)*100  // in percentage
format(6)
disp(pc,"Therefore,  % change = f''-f/f * 100 =")
disp("(ii)  Let us assume direction of coupling such that,")
disp("  L_eq = L1 + L2 + 2M = 0.25 mH")
disp("   C_t = Trim capacitor = 100 pF")
disp("Therefore,  C_eq = C*C_t / C+C_t = 50 pF")
f1=(1/(2*%pi*sqrt(0.25*50*10^-15)))*10^-6  // in MHz
format(7)
disp(f1,"Therefore,  f = 1 / 2*pi*sqrt(L_eq*C_eq) =")
disp("If now direction of coupling is reversed,")
disp("  L_eq = L1 + L2 - 2M = 0.15 mH")
f2=(1/(2*%pi*sqrt(0.15*50*10^-15)))*10^-6  // in MHz
format(8)
disp(f2,"Therefore,  f'' = 1 / 2*pi*sqrt(L_eq*C_eq) =")
pc1=((1.83776-1.4235)/1.4235)*100
format(7)
disp(pc1,"Therefore,  % change = f''-f/f * 100 =")
```

Scilab code Exa 4.25 `design RC phase shift oscillator`

```
//Example 4.25
clc
disp("For RC phase shhift oscillator,")
disp("  h_fe = 4K + 23 + 29/K              ...given h_fe = 150")
disp("Therefore,  150 = 4K + 23 + 29/K")
disp("Therefore,  4K^2 - 127K + 29 = 0")
K=poly(0,'K')
p1=4*K^2-127*K+29
t1=roots(p1)
format(6)
disp(t1,"Therefore, K =")
disp("  f = 1 / 2*pi*R*C*sqrt(6+4K)          ...given f = 5 kHz")
disp("Therefore,Choose    C = 100 pF")
r=(1/(2*%pi*(1000*10^-12)*(5*10^3)*sqrt(6+(4*0.23))))*10^-3  // in k-ohm
format(3)
disp(r,"Therefore,  R(in k-ohm) =")
disp("  K = R_C / R  i.e.  R_C = KR = 2.7 k-ohm")
disp("Neglecting effect of biasing resistances assuming them to be large and selecting transistor with h_ie = 2 k-ohm")
disp("  R''_i = h_ie = 2 k-ohm")
disp("Therefore,Last resistance in phase network")
r3=12-2
disp(r3,"  R3 = R - R''_i =")
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")
disp("The designed circuit is shown in fig.4.49")
```

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

Scilab code Exa 4.26 `range of capacitor`

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

Scilab code Exa 4.27 `change in frequency of oscillation`

```
//Example 4.27
```

```
clc
ceq=((0.02*12*10^-24)/(12.02*10^-12))*10^12  // in pF
format(8)
disp(ceq,"  C_eq(in pF) = C1*C2 / C1+C2 =")
fs=(1/(2*%pi*sqrt(50*0.02*10^-15)))*10^-6  // in MHz
format(7)
disp(fs,"Therefore,  f_s(in MHz) = 1 / 2*pi*sqrt(L*C1) =")
fp=(1/(2*%pi*sqrt(50*0.01996*10^-15)))*10^-6  // in MHz
format(7)
disp(fp,"Therefore,  f_p(in MHz) = 1 / 2*pi*sqrt(L*C_eq) =")
disp("Let C_s = 5 pF connected across the crystal")
c2=12+5
disp(c2,"Therefore,  C''2(in pF) = C2 + C_x =")
format(10)
ceq1=0.019976
disp(ceq1,"Therefore,  C''_eq(in pF) = C1*C''2 / C1+C''2 =")
fp1=5.03588
disp(fp1,"Therefore,  f''_p(in MHz) = 1 / 2*pi*sqrt(L*C_eq) =")
disp("New C_x = 6 pF is connected then,")
c21=12+6
disp(c21,"  C''''2(in pF) = C2 + C_x =")
ceq2=0.0199778
disp(ceq2,"Therefore,  C''''_eq(in pF) = C1*C''''2 / C1+C''''2 =")
fp2=5.035716
disp(fp2,"Therefore,  f''''_p(in MHz) = 1 / 2*pi*sqrt(L*C''''_eq) =")
c=(5.03588-5.035716)*10^6
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

```
//Example 5.1
clc
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")
disp("  A     B     C     Y")
disp("  0     0     0     0")
disp("  0     0     1     0")
disp("  0     1     0     0")
disp("  0     1     1     1")
disp("  1     0     0     0")
disp("  1     0     1     1")
disp("  1     1     0     1")
disp("  1     1     1     1")
disp("Now we obtain the simplified Boolean
   expression for output variable Y using K-map
   simplification.")
```

```
disp("     BC      BC''          B''C''          B''C")
disp("A    0          0              1               0")
disp("A''    0          1              1               1")
disp("   Y = AC + BC + AB")
```

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

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

```
//Example 5.2
clc
disp("The truth table for the given problem is as
    shown below.")
disp("   C      D3      D2      D1      Output")
disp("   0      x       x       x           0")
disp("   0      0       0       0           0")
disp("   0      0       0       1           1")
disp("   0      0       1       0           1")
disp("   0      1       0       0           1")
disp("")
disp("K-map simplification")
disp("             D1''D2''        D1''D2        D1D2        D1D2
    ''")
disp("C''D3''        0                0            0            0")
disp("C''D3        0                0            0            0")
disp("CD3          1                X            X            X")
disp("CD3''        0                1            X            1")
disp("")
disp("Therefore,      Y = CD3 + CD2 + CD1")
```

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

Scilab code Exa 5.3 `design combinational circuit`

```
//Example 5.3
clc
disp("Truth table")
disp("        Input                          Output")
disp("   Decimal Digit          Digit 1          Digit 0")
disp("   A   B   C   D       Y7   Y6   Y5   Y4   Y3   Y2   Y1   Y0")
disp("   0   0   0   0       0    0    0    0    0    0    0    0")
disp("   0   0   0   1       0    0    0    0    0    1    0    1")
disp("   0   0   1   0       0    0    0    1    0    0    0    0")
disp("   0   0   1   1       0    0    0    1    0    1    0    1")
disp("   0   1   0   0       0    0    1    0    0    0    0    0")
disp("   0   1   0   1       0    0    1    0    0    1    0    1")
disp("   0   1   1   0       0    0    1    1    0    0    0    0")
disp("   0   1   1   1       0    0    1    1    0    1    0    1")
disp("   1   0   0   0       0    1    0    0    0    0    0    0")
disp("   1   0   0   1       0    1    0    0    0    1    0    1")
disp("")
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`

```
//Example 5.4
clc
disp("Let us consider D for Door, I for ignition, L
    for Light. Then conditions to activate the alarm
    are:")
disp("1. The headlights are ON while the ignition is
    OFF.")
disp("    i.e. L = 1, I = 0 and D may be anything.")
disp("2. The ddor is open while the ignition is ON")
disp("    i.e. D = 1, I = 1, L may be anything.")
disp("Also alarm will sound if logic circuit output
    is zero.")
disp("Therefore, output for above condition is zero
    and for rest of the condition it is 1 which is
    summarized in the following table.")
disp("  D    I    L    Y")
disp("  0    0    0    1")
disp("  X    0    1    0")
disp("  0    1    0    1")
disp("  0    1    1    1")
disp("  1    0    0    1")
disp("  1    1    X    0")
disp("Therefore, K-map for logic circuit.")
disp("        I''L''    I''L    IL    IL''")
disp("D''      1         0       1      1")
disp("D        1         0       0      0")
disp("  Output = Y = I''L'' + D''I")
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 wwww.scilab.in This code can be downloaded from the website wwww.scilab.in

Scilab code Exa 5.5 `design circuit to detect invalid BCD number`

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

```
disp("AB          1              1            1            1")
disp("AB''         0              0            1            1")
disp("   Y = AB + AC")
```

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

Scilab code Exa 5.6 `design two level combinational circuit`

```
//Example 5.6
clc
disp("    Input 1 -> Pressure in fuel tank")
disp("    Input 2 -> Pressure in oxidizer tank")
disp("    Input = 1 Indicates pressure is equal to or
    above the required minimum")
disp("           = 0 Otherwise")
disp("    Input 3 -> From timer")
disp("if input 3 = 1 Indicates that there are less
   than or exactly 10 minutes for lift off")
disp("           = 0 Otherwise")
disp("      Output -> Panel light , if light goes on
   then")
disp("      Output = 1")
disp("else Output = 0")
disp("")
disp("Truth table")
disp("Let input 1 = A,  input 2 = B,  input 3 = C.")
disp("     Inputs      Output")
disp("   A    B    C     Y")
disp("   0    0    0     1")
disp("   0    0    1     0")
disp("   0    1    0     1")
disp("   0    1    1     0")
disp("   1    0    0     1")
disp("   1    0    1     0")
```

```
disp("   1      1      0       0")
disp("   1      1      1       1")
disp("")
disp("K-map simplification")
disp("         B''C''          B''C      BC       BC''")
disp("A''        1               0       0         1")
disp("A          1               0       1         0")
disp("   Y = ABC + A''B''C'' + B''C''")
disp("     = ABC + C''(B''+A''B)")
disp("     = ABC + C''(B''+A'')              [A''+A''B = 
    A + B]")
disp("     = ABC + C''(A''B'')")
disp("     = A''B'' XOR C''")
```

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

Scilab code Exa 5.7 `design 32 to 1 multiplexer`

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

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

Scilab code Exa 5.8 `design a 32 to 1 multiplexer`

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

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

Scilab code Exa 5.9 `implement boolean function using 8to1 multiplexer`

```
//Example 5.9
clc
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 wwww.scilab.in

Scilab code Exa 5.10 `implement boolean function using 4to1 multiplexer`

```
//Example 5.10
clc
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.")
disp("Truth table")
disp("Minterm     A   B   C     F")
```

```
disp("    0          0   0   0      0")
disp("    1          0   0   1      1")
disp("    2          0   1   0      0")
disp("    3          0   1   1      1")
disp("    4          1   0   0      0")
disp("    5          1   0   1      1")
disp("    6          1   1   0      1")
disp("    7          1   1   1      0")
disp("")
disp("Implementation table")
disp("      D0   D1   D2   D3")
disp("A''    0     1     2     3     Row 1")
disp("A     4     5     6     7     Row 2")
disp("      0     1     A     A''")
disp("")
disp("As shown in fig. 5.23(c) the implementation
   table is nothing but the list of the inputs of
   the miltiplexers 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.")
disp("1. If the two minterms in a column are not
   circled, O is applied to the corresponding
   multiplexer input (see column 1).")
disp("2. If the two minterms in a column are circled
   , 1 is applied to the corresponding multiplexer
   input (see column 2).")
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).")
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 wwww.scilab.in

Scilab code Exa 5.11 `implement boolean function using 8to1 MUX`

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

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

Scilab code Exa 5.12 `implement boolean function using 4to1 MUX`

```
//Example 5.12
clc
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.")
disp("")
disp("Implementation table")
disp("      D0  D1  D2  D3  D4  D5  D6  D7")
```

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

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

Scilab code Exa 5.13 `implement boolean function using 8to1 MUX`

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

```
disp("   10                    1   0   1   0   0")
disp("   11      AB''CD         1   0   1   1   1")
disp("   12                    1   1   0   0   0")
disp("   13                    1   1   0   1   0")
disp("   14                    1   1   1   0   0")
disp("   15      ABCD          1   1   1   1   1")
disp("")
disp("From the truth table Boolean function can be
   implemented using 8 : 1 multiplexer as follows :"
   )
disp("Implementation table :")
disp("        D0   D1   D2   D3   D4   D5   D6   D7")
disp("A''      0    1    2    3    4    5    6    7")
disp("A       8    9    10   11   12   13   14   15")
disp("        0    A''  0    1    A''  A''  A''  A")
```

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

Scilab code Exa 5.14 `implement boolean function using 8to1 MUX`

```
//Example 5.14.
clc
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.")
disp("")
disp("Implementation table :")
disp("        D0   D1   D2   D3   D4   D5   D6   D7")
disp("A''      0    1    2    3    4    5    6    7")
disp("A       8    9    10   11   12   13   14   15")
disp("        0    A''  A''  A    A''  A    0    1")
```

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

Scilab code Exa 5.15 `implement boolean function using 8to1 MUX`

```
//Example 5.15
clc
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.")
disp("")
disp("Implementation table :")
disp("        D0   D1   D2   D3   D4   D5   D6   D7")
disp("A''      0    1    2    3    4    5    6    7")
disp("A       8    9    10   11   12   13   14   15")
disp("        1    0    1    1    A    A    1    0")
disp("")
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 wwww.scilab.in

Scilab code Exa 5.16 `determine boolean expression`

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

```
disp("0    2     3")
disp("0    4     5")
disp("1    6     7")
disp("D    8     9")
disp("1    10    11")
disp("D''   12    13")
disp("0    14    15")
disp("")
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.")
disp("1. If multiplexer input is 0, don''t circle any minterm in the corresponding row.")
disp("2. If multiplexer input 1, circle both the minterms in the corresponding row.")
disp("3. If multiplexer input is D, circle the minterm belongs to cloumn D in the corresponding row.")
disp("4. If multiplexer input is D'', circle the minterm belongs to column D'' in the corresponding row.")
disp("This is illustrated in fig. 5.30(b). Now circled minterms can be written to get Boolean expression as follows :")
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

```
//Example 5.17
clc
```

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

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

Scilab code Exa 5.18 `design 1 to 8 DEMUX`

```
//Example 5.18
clc
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 wwww.scilab.in

Scilab code Exa 5.19 `implement full subtractor`

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

```
disp("   0   0    0    0    0")
disp("   0   0    1    1    1")
disp("   0   1    0    1    1")
disp("   0   1    1    0    1")
disp("   1   0    0    1    0")
disp("   1   0    1    0    0")
disp("   1   1    0    0    0")
disp("   1   1    1    1    1")
disp("")
disp("For full subtractor difference D function can be written as D = f(A, B, C) = summation m(1, 2, 4, 7) and Bout function can be written as")
disp("      Bout = F(A, B, C) = summation m(1, 2, 3, 7)")
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 wwww.scilab.in

Scilab code Exa 5.20 `construst 1 to 32 DEMUX`

```
//Exmaple 5.20
clc
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
miltiplexer 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 wwww.scilab.in

Scilab code Exa 5.21 `design 3 to 8 decoder`

```
//Example 5.21
clc
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.")
disp("")
disp("Truth table for a 3 to 8 decoder")
disp("   Inputs     |               Outputs")
disp("EN  A  B  C  |  Y7  Y6  Y5  Y4  Y3  Y2  Y1  Y0
   ")
disp("0   X  X  X  |  0   0   0   0   0   0   0   0"
   )
disp("1   0  0  0  |  0   0   0   0   0   0   0   1"
   )
disp("1   0  0  1  |  0   0   0   0   0   0   1   0"
   )
disp("1   0  1  0  |  0   0   0   0   0   1   0   0"
```

```
    )
disp("1    0  1  1  |  0   0   0   0   1   0   0   0"
    )
disp("1    1  0  0  |  0   0   0   1   0   0   0   0"
    )
disp("1    1  0  1  |  0   0   1   0   0   0   0   0"
    )
disp("1    1  1  0  |  0   1   0   0   0   0   0   0"
    )
disp("1    1  1  1  |  1   0   0   0   0   0   0   0"
    )
```

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

Scilab code Exa 5.22 `design 5 to 32 decoder`

```
//Example 5.22
clc
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 wwww.scilab.in

Scilab code Exa 5.23 `design 4 line to 16 line decoder`

```
//Example 5.23
clc
disp("4 line to 16 line decoder using 1 line to 4
   line decoder")
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 wwww.scilab.in

Scilab code Exa 5.24 `implement using 74LS138`

```
//Example 5.24
clc
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 wwww.scilab.in

Scilab code Exa 5.25 `implement full substractor`

```
//Example 5.25
clc
disp("The truth table for full subtractor is as shown below")
disp("")
disp(" Inputs        Outputs")
disp("A  B  Bin     D  Bout")
disp("0  0   0      0   0")
disp("0  0   1      1   1")
disp("0  1   0      1   1")
disp("0  1   1      0   1")
disp("1  0   0      1   0")
disp("1  0   1      0   0")
disp("1  1   0      0   0")
disp("1  1   1      1   1")
disp("")
disp("Implementation of full subtractor using 3 : 8 decoder is shown in fig. 5.51")
```

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

Scilab code Exa 5.26 `implement gray to binary code converter`

```
//Example 5.26
clc
```

```
disp("The truth table for 3-bit binary to gray code converter is as shown below")
disp("")
disp("A  B  C     G2  G1  G0")
disp("0  0  0     0   0   0")
disp("0  0  1     0   0   1")
disp("0  1  0     0   1   1")
disp("0  1  1     0   1   0")
disp("1  0  0     1   1   0")
disp("1  0  1     1   1   1")
disp("1  1  0     1   0   1")
disp("1  1  1     1   0   0")
disp("")
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 wwww.scilab.in

Scilab code Exa 5.27 `design 2 bit comparator`

```
//Example 5.27
clc
disp("")
disp("A1   A0   B1   B0      A>B   A=B   A<B")
disp("0    0    0    0        0     0     0")
disp("0    0    0    1        0     0     1")
disp("0    0    1    0        0     0     1")
disp("0    0    1    1        0     0     1")
disp("0    1    0    0        1     0     0")
```

```
disp("0   1   0   1      0    1    0")
disp("0   1   1   0      0    0    1")
disp("0   1   1   1      0    0    1")
disp("1   0   0   0      1    0    0")
disp("1   0   0   1      1    0    0")
disp("1   0   1   0      0    1    0")
disp("1   0   1   1      0    0    1")
disp("1   1   0   0      1    0    0")
disp("1   1   0   1      1    0    0")
disp("1   1   1   0      1    0    0")
disp("1   1   1   1      0    1    0")
```

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

Scilab code Exa 5.28 `design full adder circuit`

```
//Example 5.28
clc
disp("Truth table for full adder is as shown below."
   )
disp("   Inputs         Outputs")
disp("A   B   Cin     Carry   Sum")
disp("0   0    0        0      0")
disp("0   0    1        0      1")
disp("0   1    0        0      1")
disp("0   1    1        1      0")
disp("1   0    0        0      1")
disp("1   0    1        1      0")
disp("1   1    0        1      0")
disp("1   1    1        1      1")
```

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

Scilab code Exa 5.29 `implement BCD to 7 segment decoder`

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

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

Scilab code Exa 5.31 `implement 32 input to 5 output encoder`

```
//Example 5.31
clc
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 producedby ORing GS
'' outputs of all encoders.")
```

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

Chapter 6

Sequential Logic Circuits

Scilab code Exa 6.4 `analyze the circuit`

```
//Example 6.4
clc
disp("To analyze the circuit means to drive the truth table for it.")
disp("We have,  D = Input XOR Q_n")
disp("")
disp("CLK     Input     Q_n     D = input XOR Q_n     Q_n+1")
disp("down      0         0                 0           0")
disp("down      0         1                 1           1")
disp("down      1         0                 1           1")
disp("down      1         1                 0           0")
disp("")
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 circui is T flip-flop
```

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

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

Chapter 7

Shift Registers

Scilab code Exa 7.1 `determine number of flip flops needed`

```
//Example 7.1
clc
disp("(i) A 6-bit binary number requires register
    with 6 flip-flops.")
disp("")
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 od storing 32 decimal.")
disp("")
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")
disp("")
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`

```
//Example 8.1
clc
disp("After 12 pulses, the count will be (1100)_2, i
   .e. 12 in decimal.")
```

Scilab code Exa 8.2 `count in binary`

```
//Example 8.2
clc
a=dec2bin(144)
disp(a,"decimal (144) =")
disp("Since counter is a 5-bit counter, it resets
   after 2^5 = 32 clock pulses.")
disp("Dividing 144 by 32 we get quotient 2 and
   remainder 6")
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`

```
//Example 8.4
clc
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 wwww.scilab.in

Scilab code Exa 8.5 `output frequency`

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

Scilab code Exa 8.6 `maximum operating frequency`

```
//Example 8.6
clc
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,")
fm=(1/(250*10^-9))*10^-6
```

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

Scilab code Exa 8.8 `design 4 bit up down ripple counter`

```
//Example 8.8
clc
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.")
disp(" The fig. 8.14 shows the 4-bit up/down ripple
    counter.")
```

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

Scilab code Exa 8.9 `design divide by 9 counter`

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

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

Scilab code Exa 8.10 `design a divide by 128 counter`

```
//Example 8.10
clc
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.")
disp("The circuit diagram is as shown in fig.8.18.")
```

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

Scilab code Exa 8.11 `design divide by 78 counter`

```
//Example 8.11
clc
disp("Since 78 = 13 x 6, we have to use 7493 as mod-
   a3 and 7492 as mod-6 counters. For the mod-13
   counter QD, QC and QA outputs of 7493 ans 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.")
disp(" The circuit diagram is as shown in the fig.
   8.19")
```

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

Scilab code Exa 8.12 `design divide by 6 counter`

```
//Example 8.12
clc
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 sequenceis 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 wwww.scilab.in

Scilab code Exa 8.13 `find fmax`

```
//Example 8.13
clc
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")
fm=(1/(70*10^-9))*10^-6
format(5)
disp(fm,"  f_max(in MHz) =")
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,")
fma=(1/(4*(50*10^-9)))*10^-6
format(3)
disp(fma,"  f_max(ripple)(in MHz) =")
```

Scilab code Exa 8.14 determine states

```
//Example 8.14
clc
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

```
//Example 8.15
clc
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 wwww.scilab.in

Scilab code Exa 8.16 `design down counter`

```
//Example 8.16
clc
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.")
disp("")
disp("Q3   Q2   Q1   Q0     Y")
disp("0    0    0    0      0")
disp("0    0    0    1      0")
disp("0    0    1    0      0")
disp("0    0    1    1      1")
disp("0    1    0    0      1")
disp("0    1    0    1      1")
disp("0    1    1    0      1")
disp("0    1    1    1      1")
disp("1    0    0    0      1")
disp("1    0    0    1      1")
disp("1    0    1    0      1")
disp("1    0    1    1      1")
disp("1    1    0    0      1")
disp("1    1    0    1      1")
disp("1    1    1    0      0")
disp("1    1    1    1      0")
disp("")
disp("K=map simplification")
disp("            Q1''Q0''   Q1''Q0   Q1Q0   Q1Q0''")
disp("Q3''Q2''        0          0        1       0")
disp("Q3''Q2          1          1        1       1")
disp("Q3Q2            1          1        0       0")
disp("Q3Q2''          1          1        1       1")
disp("")
disp("Therefore,  PL'' = Y = Q3''Q1Q0 + Q3''Q2 + Q3Q1'' + Q3Q2''")
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 wwww.scilab.in This code can be downloaded from the website wwww.scilab.in

Scilab code Exa 8.17 `design programmable frequency divider`

```
//Example 8.17
clc
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 wwww.scilab.in

Scilab code Exa 8.18 `design a counter`

```
//Example 8.18
clc
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.")
```

```
disp("")
disp("Q3   Q2   Q1   Q0      Y")
disp("0    0    0    0       0")
disp("0    0    0    1       0")
disp("0    0    1    0       0")
disp("0    0    1    1       1")
disp("0    1    0    0       1")
disp("0    1    0    1       1")
disp("0    1    1    0       1")
disp("0    1    1    1       1")
disp("1    0    0    0       1")
disp("1    0    0    1       1")
disp("1    0    1    0       1")
disp("1    0    1    1       1")
disp("1    1    0    0       1")
disp("1    1    0    1       1")
disp("1    1    1    0       1")
disp("1    1    1    1       0")
disp("")
disp("K-map simplification")
disp("             Q1''Q0''   Q1''Q0   Q1Q0   Q1Q0''")
disp("Q3''Q2''         0          0        1       0")
disp("Q3''Q2           1          1        1       1")
disp("Q3Q2             1          1        0       1")
disp("Q3Q2''           1          1        1       1")
disp("")
disp("Therefore,  PL'' = Y = Q3''Q1Q0 + Q3''Q2 +
   Q3Q1'' + Q3Q2'' + Q2Q1Q0''")
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 wwww.scilab.in This code can be downloaded from the website wwww.scilab.in

Scilab code Exa 8.19 `programmable frequency divider`

```
//Example 8.19
clc
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 frequncy divider using IC 74191.")
```

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

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

```
//Example 8.20
clc
disp("Fig. 8.41 shows Dividing-by-2 for up counting"
   )
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.")
```

```
disp("")
disp("Divide-by-5 for down counting mode:")
disp("")
disp("Q3   Q2   Q1   Q0      Y")
disp("0    0    0    0       0")
disp("0    0    0    1       0")
disp("0    0    1    0       0")
disp("0    0    1    1       0")
disp("0    1    0    0       0")
disp("0    1    0    1       0")
disp("0    1    1    0       0")
disp("0    1    1    1       0")
disp("1    0    0    0       0")
disp("1    0    0    1       0")
disp("1    0    1    0       0")
disp("1    0    1    1       1")
disp("1    1    0    0       1")
disp("1    1    0    1       1")
disp("1    1    1    0       1")
disp("1    1    1    1       1")
disp("")
disp("K=map simplification")
disp("              Q1''Q0''   Q1''Q0   Q1Q0   Q1Q0''")
disp("Q3''Q2''         0          0        0       0")
disp("Q3''Q2           0          0        0       0")
disp("Q3Q2             1          1        1       1")
disp("Q3Q2''           0          0        1       0")
disp("")
disp("Therefore,  Y = Q3Q2 + Q3Q1Q0")
```

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

Scilab code Exa 8.21 `design mod 9 counter`

```
//Example 8.21
clc
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 wwww.scilab.in

Scilab code Exa 8.22 `determine MOD number and counter range`

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

Scilab code Exa 8.23 `design a divide by 20 counter`

```
//Example 8.23
```

```
clc
disp("Internal structure of 7490 ripple counter IC
   is as shown in fig. 8.50")
disp("")
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.")
disp("          QD  QC  QB  QA        QD  QC  QB  QA")
disp("          0   0   1   0         0   0   0   0")
disp("             7490(2)               7490(1)")
disp("")
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 wwww.scilab.in This code can be downloaded from the website wwww.scilab.in

Scilab code Exa 8.24 `design divide by 96 counter`

```
//Example 8.24
clc
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 wwww.scilab.in

Scilab code Exa 8.25 design divide by 93 counter

```
//Example 8.25.
clc
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 wwww.scilab.in

Scilab code Exa 8.26 design divide by 78 counter

```
//Example 8.26
clc
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 wwww.scilab.in

Scilab code Exa 8.28 7490 IC

```
//Example 8.28
clc
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.")
disp("Clock         Outputs")
disp("              QA  QD  QC  QB")
disp("   0          L   L   L   L")
disp("   1          L   L   L   H")
disp("   2          L   L   H   L")
disp("   3          L   L   H   H")
disp("   4          L   H   L   L")
disp("   5          H   L   L   L")
disp("   6          H   L   L   H")
disp("   7          H   L   H   L")
disp("   8          H   L   H   H")
disp("   9          H   H   L   L")
```

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

Scilab code Exa 8.30 `sketch output waveforms of counter`

```
//example 10.9

clc;
clear;
close;
c = [0 1 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
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];
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];
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];
y1=q;
y2=a;
```

```
y3=b;
y11p=1;
y22p=1;
y33p=1;
y44p=1;
cp=1;
yf1p=1;
for i=1:25     // making arrays to draw the output
    if y1(i)==1 then
        for o=1:100
        y11(y11p)=1;
        y11p=y11p+1;
        end
    else
        for o=1:100
        y11(y11p)=0;
        y11p=y11p+1;
        end

end
if y2(i)==1 then
        for o=1:100
        y21(y22p)=1;
        y22p=y22p+1;
        end
    else
        for o=1:100
        y21(y22p)=0;
        y22p=y22p+1;
        end

end
if y3(i)==1 then
        for o=1:100
        y31(y33p)=1;
        y33p=y33p+1;
        end
    else
```

```
        for o=1:100
        y31(y33p)=0;
        y33p=y33p+1;
        end

end
if c(i)==1 then
        for o=1:100
        c1(cp)=1;
        cp=cp+1;
        end
    else
        for o=1:100
        c1(cp)=0;
        cp=cp+1;
        end
end

end
z=[2 2];
subplot(4,1,1); //ploting the out put
title('Timing Diagram');
plot(c1);
plot(z);
ylabel('QA');
subplot(4,1,2);
plot(y11);
ylabel('QB');
plot(z);
subplot(4,1,3);
plot(y21);
ylabel('QC');
plot(z);
subplot(4,1,4);
plot(z);
ylabel('QD');
plot(y31);
disp("The counter goes through states 0000 (Decimal
```

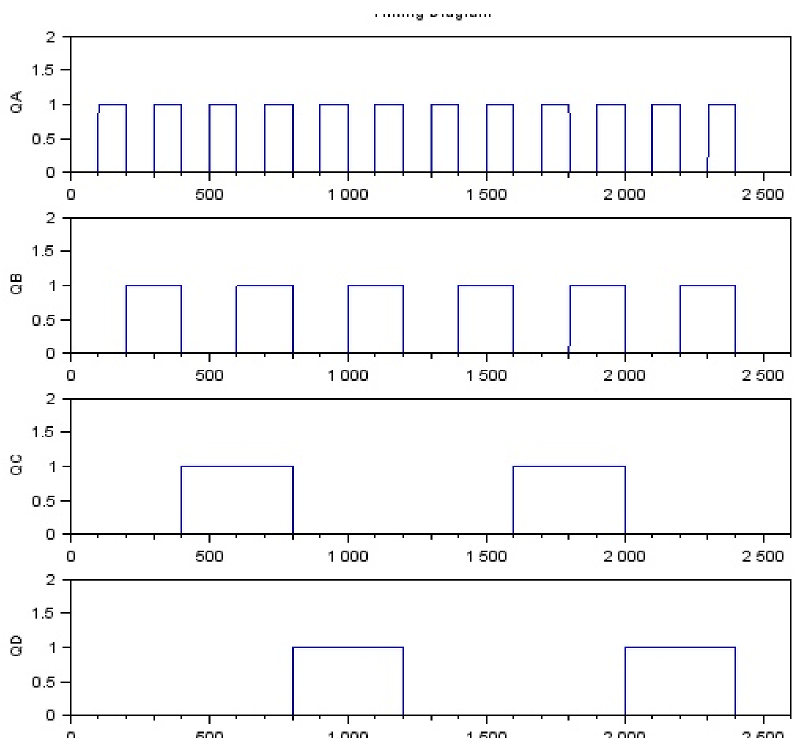

Figure 8.1: sketch output waveforms of counter

```
    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`

```
//Example 8.31
clc
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 countion 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 wwww.scilab.in

Scilab code Exa 8.32 `design divide by 40000 counter`

```
//Example 8.32
clc
disp("Cascading four 74161 (each 4-bit) counters we
    get 16 (4 x 4) bit counter as shown in fig 8.63."
    )
disp("Therefore, we get 2^16 = 65,536 modulus
    counter")
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 ful cycle. Therefore,
    each full cycle of the counter consists of 40,000
     states.")
```

Scilab code Exa 8.33 design modulo 11 counter

```
//Example 8.33
clc
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.")
disp("")
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 wwww.scilab.in

Scilab code Exa 8.34 design excess 3 decimal counter

```
//Example 8.34
clc
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 wwww.scilab.in

Scilab code Exa 8.35 `modulus greater than 16`

```
//Example 8.35
clc
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 util 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 assets 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 wwww.scilab.in

Scilab code Exa 8.36 `modulo 8 counter`

```
//Example 8.36
clc
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.")
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 wwww.scilab.in

Scilab code Exa 8.37 `synchronous decade counter`

```
//Example 8.37
clc
disp("Excitation table")
disp("Present State                 Next State
                    Flip-flop Inputs")
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")
```

```
disp("0    0    0    0         0        0        0        1
            0       0       0       1")
disp("0    0    0    1         0        0        1        0
            0       0       1       1")
disp("0    0    1    0         0        0        1        1
            0       0       0       1")
disp("0    0    1    1         0        1        0        0
            0       1       1       1")
disp("0    1    0    0         0        1        0        1
            0       0       0       1")
disp("0    1    0    1         0        1        1        0
            0       0       1       1")
disp("0    1    1    0         0        1        1        1
            0       0       0       1")
disp("0    1    1    1         1        0        0        0
            1       1       1       1")
disp("1    0    0    0         1        0        0        1
            0       0       0       1")
disp("1    0    0    1         0        0        0        0
            1       0       0       1")
disp("1    0    1    0         X        X        X        X
            X       X       X       1")
disp("1    0    1    1         X        X        X        X
            X       X       X       1")
disp("1    1    0    0         X        X        X        X
            X       X       X       X")
disp("1    1    0    1         X        X        X        X
            X       X       X       X")
disp("1    1    1    0         X        X        X        X
            X       X       X       X")
disp("1    1    1    1         X        X        X        X
            X       X       X       X")
disp("")
disp("K-map Simplification")
disp("                          For JK_D")
disp("             QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''        0         0       0      0")
disp("QD''QC        0         0       1      0")
```

```
disp("QDQC          X          X         X       X")
disp("QDQC''          0          1         X       X")
disp("JK_D = QA QD + QA QB QC")
disp("")
disp("                        For JK_C")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''          0          0         1       0")
disp("QD''QC          0          0         1       0")
disp("QDQC          X          X         X       X")
disp("QDQC''          0          0         X       X")
disp("JK_C = QA QB")
disp("")
disp("                        For JK_B")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''          0          1         1       0")
disp("QD''QC          0          1         1       0")
disp("QDQC          X          X         X       X")
disp("QDQC''          0          0         X       X")
disp("JK_B = QA QD''")
disp("")
disp("                        For JK_A")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''          1          1         1       1")
disp("QD''QC          1          1         1       1")
disp("QDQC          X          X         X       X")
disp("QDQC''          1          1         X       X")
disp("JK_A = 1")
disp("")
disp("Fig shows the logic diagram for the
   synchronous decade counter using JK flip-flop")
```

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

Scilab code Exa 8.38 `flip flops`

```
//Example 8.38
clc
disp("Excitation table")
disp(" Input        Present State         Next State         Flip-flop Inputs")
disp("UP/DOWN''       QC   QB   QA          Q_C+1   Q_B+1   Q_A+1     JK_C   JK_B   JK_A")
disp("  UD")
disp("  0            0    0    0           1       1       1         1      1      1")
disp("  0            0    0    1           0       0       0         0      0      1")
disp("  0            0    1    0           0       0       1         0      1      1")
disp("  0            0    1    1           0       1       0         0      0      1")
disp("  0            1    0    0           0       1       1         1      1      1")
disp("  0            1    0    1           1       0       0         0      0      1")
disp("  0            1    1    0           1       0       1         0      1      1")
disp("  0            1    1    1           1       1       0         0      0      1")
disp("  1            0    0    0           0       0       1         0      0      1")
disp("  1            0    0    1           0       1       0         0      1      1")
disp("  1            0    1    0           0       1       1         0      0      1")
disp("  1            0    1    1           1       0       0         1      1      1")
disp("  1            1    0    0           1       0       1         0      0      1")
disp("  1            1    0    1           1       1       0         0      1      1")
disp("  1            1    1    0           1       1       1         0      0      1")
```

```
disp("   1                 1    1    1              0         0         0          1        1        1")
disp("")
disp("K-map Simplification")
disp("                        For JK_C")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''      1         0       0      0")
disp("QD''QC        1         0       0      0")
disp("QDQC          0         0       1      0")
disp("QDQC''        0         0       1      0")
disp("JK_C =UD'' QB'' QB'' + UD QB QA")
disp("")
disp("                        For JK_B")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''      1         0       0      1")
disp("QD''QC        1         0       0      1")
disp("QDQC          0         1       1      0")
disp("QDQC''        0         1       1      0")
disp("TB =UD'' QA'' + UD QA")
disp("")
disp("                        For JK_A")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''      1         1       1      1")
disp("QD''QC        1         1       1      1")
disp("QDQC          1         1       1      1")
disp("QDQC''        1         1       1      1")
disp("TA = 1")
disp("")
```

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

Scilab code Exa 8.39 `design mod 5 synchronous counter`

```
//Example 8.39
```

```
clc
disp("For mod-5 counter we require 3 flip-flops.")
disp("Excitation table")
disp("                    Present State          Next State              Flip-flop Inputs")
disp("                    QC  QB  QA          Q_A+1   Q_B+1   Q_C+1      T_A      T_B      T_C")
disp("   0                 0   0   0              0       0       1        0        0        1")
disp("   1                 0   0   1              0       1       0        0        1        1")
disp("   2                 0   1   0              0       1       1        0        0        1")
disp("   3                 0   1   1              1       0       0        1        1        1")
disp("   4                 1   0   0              0       0       0        1        0        0")
disp("")
disp("K-map Simplification")
disp("            QB''QC''   QB''QC   QBQC   QBQC''")
disp("QA''           0          0       1      0")
disp("QA             1          X       X      X")
disp("T_A = QA + QB QC")
disp("")
disp("            QB''QC''   QB''QC   QBQC   QBQC''")
disp("QA''           0          1       1      0")
disp("QA             0          X       X      X")
disp("T_B = QC")
disp("")
disp("            QB''QC''   QB''QC   QBQC   QBQC''")
disp("QA''           1          1       1      1")
disp("QA             0          X       X      X")
disp("T_C = QA''")
disp("")
```

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

Scilab code Exa 8.40 `design MOD 4 down counter`

```
//Example 8.40
clc
disp("Excitation table")
disp("Present State        Next State          Flip-flop Inputs")
disp("   QC  QB              A+      B+          J_A     K_A      J_B     K_B")
disp("   0     0             1       1           1       X        1       X")
disp("   0     1             0       0           0       X        X       1")
disp("   1     0             0       1           X       1        1       X")
disp("   1     1             1       0           X       0        X       1")
disp("")
disp("K-map Simplification")
disp("  For J_A")
disp("       B''    B")
disp("A''     1      0")
disp("A      X      X")
disp("J_A = B''")
disp("")
disp("  For K_A")
disp("       B''    B")
disp("A''     X      X")
disp("A      1      0")
disp("K_A = B''")
disp("")
```

```
disp("   For  J_B")
disp("        B''    B")
disp("A''     1      X")
disp("A      1      X")
disp("J_B  =  1")
disp("")
disp("   For  K_B")
disp("        B''    B")
disp("A''     X      1")
disp("A      X      1")
disp("K_B  =  1")
disp("")
```

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

Scilab code Exa 8.41 design MOD 12 synchronous counter

```
//Example 8.41
clc
disp("Mod-12 synchronous counter using D flip-flop :
   ")
disp("Let     Number of flip-flop required = n")
disp("                                        2^n >= 12")
disp("                                           n = 4")
disp("Excitation table")
disp("Present State                  Next State          ")
disp("QD  QC  QB  QA     Q_D+1  Q_C+1  Q_B+1  Q_A+1")
disp("0   0   0   0       0      0      0      1")
disp("0   0   0   1       0      0      1      0")
disp("0   0   1   0       0      0      1      1")
disp("0   0   1   1       0      1      0      0")
disp("0   1   0   0       0      1      0      1")
disp("0   1   0   1       0      1      1      0")
disp("0   1   1   0       0      1      1      1")
```

```
disp("0    1    1    1         1         0         0         0")
disp("1    0    0    0         1         0         0         1")
disp("1    0    0    1         1         0         1         0")
disp("1    0    1    0         1         0         1         1")
disp("1    0    1    1         0         0         0         0")
disp("")
disp("K-map Simplification")
disp("                        For D_A")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''       1         0       0      1")
disp("QD''QC         1         0       0      1")
disp("QDQC           X         X       X      X")
disp("QDQC''         1         0       0      1")
disp("D_A = QA''")
disp("")
disp("                        For D_B")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''       0         1       0      1")
disp("QD''QC         0         1       0      1")
disp("QDQC           X         X       X      X")
disp("QDQC''         0         1       0      1")
disp("D_B = QB'' QaA + QA'' QB")
disp("    = QA XOR QB")
disp("")
disp("                        For D_C")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''       0         0       1      0")
disp("QD''QC         1         1       0      1")
disp("QDQC           X         X       X      X")
disp("QDQC''         0         0       0      0")
disp("D_C = QC QB'' + QC QA'' + QD'' QC'' QB QA")
disp("")
disp("                        For D_D")
disp("            QB''QA''  QB''QA  QBQA  QBQA''")
disp("QD''QC''       0         0       0      0")
disp("QD''QC         0         0       1      0")
disp("QDQC           X         X       X      X")
disp("QDQC''         1         1       0      1")
```

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

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

Scilab code Exa 8.42 `design 4 bit 4 state ring counter`

```
//Example 8.42
clc
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
   preforms 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 wwww.scilab.in

Scilab code Exa 8.44 `design 4 bit 8 state johnson counter`

```
//Example 8.44
clc
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.")
```

```
disp("")
disp("States of 4-bit Johnson counter")
disp("State name    Q3  Q2  Q1  Q0")
disp("     S1        0   0   0   0")
disp("     S2        0   0   0   1")
disp("     S3        0   0   1   1")
disp("     S4        0   1   1   1")
disp("     S5        1   1   1   1")
disp("     S6        1   1   1   0")
disp("     S7        1   1   0   0")
disp("     S8        1   0   0   0")
disp("")
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 wwww.scilab.in This code can be downloaded from the website wwww.scilab.in

Scilab code Exa 8.45 `johnson counter`

```
//Example 8.45
clc
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

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

```
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`

```
//Example 9.3
clc
disp("I_iOS = 20 nA,   I_b = 60 nA")
disp("Now   I_iOS = I_b1 - I_b2 = 20")
disp("      I_b = I_b1+I_b2 / 2 = 60")
disp("Therefore,  I_b1 + I_b2 = 120")
disp("Therefore,  2*I_b1 = 140")
disp("Therefore,  I_b1 = 70 nA,   I_b2 = 50 nA")
```

Scilab code Exa 9.4 `design inverting schmitt trigger`

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

```
format(7)
disp(r1,"Therefore,  R1 = Vo-V_UT / I2 = +V_sat-V_UT
    / I2 =")
disp("The designed circuit is shown in fig")
```

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

Scilab code Exa 9.5 `threshold voltage`

```
//Example 9.5
clc
disp("V_CC = +15 V")
vsat=0.9*15
format(5)
disp(vsat,"Therefore,  V_sat(in V) = 0.9 V_CC =")
disp("  R1 = 51 k-ohm,  R2 = 120 ohm")
vut=(13.5*120)/((51*10^3)+120)
format(8)
disp(vut,"V_UT(in V) = +V_sat*R2 / R1+R2 =")
vlt=(-13.5*120)/((51*10^3)+120)
disp(vlt,"V_LT(in V) = -V_sat*R2 / R1+R2 =")
h=(0.03169*2)*10^3
format(6)
disp(h,"H(in mV) = V_UT - V_LT =")
```

Scilab code Exa 9.6 `tripping voltage`

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

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

Scilab code Exa 9.8 time duration

```
//Example 9.8
clc
disp("LTP = -1.5 V   and   H = 2 V")
disp("Now            H = UTP - LTP")
disp("Therefore,   2 = UTP - (-1.5)")
disp("Therefore,   UTP = 0.5 V")
disp("In the fig.9.47, the angle theta can be obtained from equation of sine wave. Sine wave is represented as,")
disp("V_in = V_p*sin(pi+thata)     when   pi < omega*t < 2pi")
disp("At LTP,     -1.5 = 5*sin(pi+theta)")
disp("                 = - 5*sin(theta)")
disp("Therefore,   sin(theta) = 0.3")
t=asind(0.3)
format(6)
disp(t,"Therefore,   theta(in degree) =")
disp("The time period of sine wave is,")
T=1
disp(T,"     T(in ms) = 1/f =")
disp("At UTP,     0.5 = V_p*sin(theta)")
disp("Therefore, 0.5 = 5 * sin(theta)")
disp("Therefore, sin(theta) = 0.1")
t=asind(0.1)
```

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

Scilab code Exa 9.9 `calculate R1 and R2`

```
//Example 9.9
clc
disp("Given +V_sat = 12 V, -V_sat = -12 V, V_H = 6 V")
disp("We know that hysteresis width is given as")
disp(" V_H = (R2/R1+R2)[+V_sat-V_sat]")
disp("Therefore, R2 / R1+R2 = V_H / +V_sat-V_sat")
r=6/(24)
disp(r,"Therefore, R2 / R1+R2 =")
disp("Therefore, R2 = 0.25R1 + 0.25R2")
disp("Therefore, 0.75R2 = 0.25R1")
r2=0.25/0.75
format(7)
disp(r2,"Therefore, R2 / R1 =")
disp("Assuming R2 = 10 k-ohm")
r1=(10000/0.3333)*10^-3
format(3)
disp(r1," R1(in k-ohm) =")
```

Scilab code Exa 9.10 `calculate trip point and hysteresis`

```
//Example 9.10
clc
disp("From fig 9.45 ,  R1 = 68 k-ohm ,  R2 = 1.5 k-ohm
    and  V_sat = 13.5 V")
vut=(1.5/(1.5+68))*13.5
format(7)
disp(vut,"V_UT(in V) = R2/R1+R2 * V_sat =")
vlt=(-1.5/(1.5+68))*13.5
disp(vlt,"V_LT(in V) = -R2/R1+R2 * V_sat =")
h=2*0.2913
disp(h,"Therefore ,  H(in V) = V_UT - V_LT =")
disp("Now    H = (2*R2 / R1+R2) * V_sat")
disp("For minimum H, R2 must be minimum and R1 must
    be maximum")
r2min=((1.5)-(0.05*1.5))
format(6)
disp(r2min,"Therefore ,  R2_min(in k-ohm) = R2 - 5%*
    R2 =")
r2max=((68)+(0.05*68))
disp(r2max,"Therefore ,  R1_max(in k-ohm) = R1 + 5%*
    R1 =")
hm=((2*1.425)/(71.4+1.425))*13.5
disp(hm,"Therefore ,  H_min(in V) =")
```

Scilab code Exa 9.11 design schmitt trigger

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

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

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

Scilab code Exa 9.12 `design op amp schmitt trigger`

```
//Example 9.12
clc
disp("For the Schmitt trigger")
disp("V_UT = 2 V,      V_LT = -4 V,      +-V_sat = +-10
   V")
disp("For unequal UTP and LTP values , a modified
    circuit is required as shown in the fig.9.52.")
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,")
disp("-IR2 - IR1 - x + V0 = 0")
disp("Therefore ,   I = V0-x / R1+R2")
disp("And   V1 = IR1 + x")
disp("Therefore ,   V1 = (V0-x/R1+R2)*R1 + x")
disp("For    +V_sat = 10 V,")
disp("V1 = V_UT = 2 V,")
disp("V0 = 10 V")
disp("Therefore ,   2 = (10-x/R1+R2)*R1 + x        (1)")
disp("For -V_sat = -10 V,")
disp("V1 = V_LT = -4 V,")
disp("V0 = -10 V")
disp("Therefore ,   -4 = (-10-x/R1+R2)*R1 + x       (2)"
    )
disp("Subtracting equations (2) and (1),")
disp("Therefore ,   6 = 20*R1 / R1+R2")
```

```
disp("Therefore ,  R1+R2 = 3.333*R1")
disp("Therefore ,  R2 = 2.333*R1      (3)")
disp("Substituting (3) in equation (1)")
disp("2 = ((10-x)*R1 / 3.333*R1) + x")
disp("Therefore ,  2.333*x = -3.3334")
x=-3.3334/2.333
format(7)
disp(x,"Therefore ,  x =")
disp("So actually polarity of the voltage source ''x
    '' must be opposite to what is assumed earlier as
     shown in fig.9.52.")
disp("Choose    R1 = 1 k-ohm    hence  R2 = 2.333 k-
    ohm")
disp("Therefore ,  R_comp = R1 || R2 = 0.7 k-ohm")
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 wwww.scilab.in

Scilab code Exa 9.13 `VUT and VLT and frequency of oscillation`

```
//Example 9.13
clc
disp("(a) We know that ,")
vut=(86*15)/(86+100)
format(5)
disp(vut,"  V_UT(in V) = R1*+V_sat / R1+R2 =")
vlt=(86*-15)/(86+100)
disp("(b) We know that ,")
disp(vlt,"  V_LT(in V) = R1*-V_sat / R1+R2 =")
```

```
disp("(c) We know that,")
f0=1/0.02
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`

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

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

Scilab code Exa 9.18 `design monostable using IC 555`

```
//Example 9.18
clc
```

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

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

Scilab code Exa 9.19 `design a timer`

```
//Example 9.19
clc
disp("Fig. 9.79 shows monostable circuit used to
   drive the relay.")
disp("This relay should be energized for 5 second to
    hold heater ''ON'' for 5 seconds. Thus, T_ON for
    monostable is 5 seconds.")
disp("We know that the pulse width is given by,")
disp("               W = 1.1 RC")
disp("Therefore,   5 = 1.1 RC")
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.")
disp("Therefore,   If capacitor value is 10 uF")
disp("then     5 = 1.1*R*10 uF")
r=(5/(1.1*10*10^-6))*10^-3
format(7)
disp(r,"Therefore,   R(in k-ohm) =")
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 wwww.scilab.in

Scilab code Exa 9.20 `draw timer using IC 555`

```
//Example 9.20
clc
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.")
disp("Therefore,  W = 15 sec")
disp("Now         W = 1.1 RC")
disp("Therefore, 15 = 1.1 RC")
disp("Choose      C = 100 uF")
r=(15/(1.1*100*10^-6))*10^-3
format(8)
disp(r,"Therefore,  R(in k-ohm) =")
disp("The designed circuit is shown in the fig. 9.80
    ")
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 wwww.scilab.in

Scilab code Exa 9.21 `frequency of output and duty cycle`

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

Scilab code Exa 9.26 `design astable multivibrator`

```
//Example 9.26
clc
disp("f = 1 kHz")
disp("D = 75% = 0.75")
disp("Now  f = 1.44 / (R_A+2*R_B)*C")
disp("Therefore,  1*10^3 = 1.44 / (R_A+2*R_B)*C")
disp("Therefore,  (R_A+2*R_B)*C = 1.44*10^-3  ....(1)")
disp("Therefore, while  %D = ((R_A+R_B)/(R_A+2*R_B))*100")
disp("Therefore,  0.75 = R_A+R_B / R_A+2*R_B")
disp("Therefore,  R_A+2*R_B = (R_A+R_B)/0.75")
disp("Therefore,  R_A+2*R_B = 1.33*(R_A+R_B)")
disp("Therefore,  0.66*R_B = 0.33*R_A")
disp("Therefore,  R_B = 0.5*R_A  ....(2)")
disp("Choose  C = 0.1 uF")
disp("Substituting in (1),")
disp("(R_A+2*R_B)*0.1*10^-6 = 1.44*10^-3")
disp("Therefore,  R_A+2*R_B = 14400  ....(3)")
```

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

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

Scilab code Exa 9.27 `design astable multivibrator`

```
//Example 9.27
clc
disp("T_ON = 0.6 ms, T = 1 ms")
d=0.6*100
disp(d,"Therefore, D(in percentage) = t_ON / T =")
disp("Now D = R_A+R_B / R_A+2*R_B = 0.6")
disp("Therefore, R_A+R_B = 0.6*R_A + 1.2*R_B")
disp("Therefore, 0.4*R_B = 0.2*R_B")
disp("Therefore, R_B = 2*R_A        ....(1)")
disp("          f = 1.44 / (R_A+2*R_B)*C = 1/T = 1000"
    )
disp("Choose C = 0.1 uF")
disp("Therefore, R_A+2*R_B = 14400")
disp("Using (1),    5*R_A = 14400")
ra=(14400/5)*10^-3
format(5)
disp(ra,"Therefore,     R_A(in k-ohm) =")
rb=2.88*2
disp(rb,"  R_B(in k-ohm) =")
```

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

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

Scilab code Exa 9.28 `design astable mode to generate square wave`

```
//Example 9.28
clc
disp("T_ON = T_OFF = 0.5 ms")
disp("Therefore,  T = T_ON + T_OFF = 1 ms")
disp("i.e.     f = 1/T = 1 kHz")
disp("Now  T_d = T_OFF = 0.69*R_B*C")
disp("Choose     C = 0.1 uF")
rb=((0.5*10^-3)/(0.69*0.1*10^-6))*10^-3
format(6)
disp(rb,"Therefore,  R_B(in k-ohm) =")
disp("Now duty cycle is 50% so  R_A = R_B = 7.246 k-ohm")
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,")
disp("R = V_0-V_LED / I_LED")
disp("Assuming  V_LED = 0.7 V")
r=(5-0.7)/(50*10^-3)
format(3)
disp(r,"Current limiting  R(ohm) = ")
disp("The voltage of R is")
disp("P = (50*10^-3)^2 * 100")
```

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

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

Scilab code Exa 9.32 `define resolution`

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

Scilab code Exa 9.33 `output voltage`

```
//Example 9.33
clc
disp("For given DAC,")
disp(" n = 4")
disp("Therefore,  V_0FS = 15 V")
```

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

Scilab code Exa 9.34 `find VoFS`

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

Scilab code Exa 9.35 `find out stepsize and analog output`

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

Scilab code Exa 9.36 `find output voltage`

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

Scilab code Exa 9.38 `find resolution and digital output`

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

Scilab code Exa 9.39 calculate quantizing error

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

Scilab code Exa 9.40 dual scope ADC

```
//Example 9.40
clc
disp("We know that,")
disp("t2 = (V1/VR)*t1")
t2=83.33
format(6)
disp(t2,"(i) t2(in ms) = (100/100)*83.33 =")
disp("(ii) V1 = 200 mV")
t2=83.33*2
disp(t2,"Therefore, t2(in ms) = (200/100)*83.33 =")
```

Scilab code Exa 9.41 find digital output of ADC

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

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

Scilab code Exa 9.42 `find conversion time`

```
//Example 9.42.
clc
disp("                f = 1 MHz")
disp("Therefore,  T = 1/f = 1 / 1*10^6 = 1 usec")
disp("                n = 8")
tc=1*(8+1)
format(2)
disp(tc,"Therefore,  T_C(in usec) = T*(n+1) =")
```

Scilab code Exa 9.43 `find maximum frequency of input sine wave`

```
//Example 9.43
clc
disp("The maximum frequency is given by,")
f=1/(2*%pi*(9*10^-6)*2^8)
format(6)
disp(f,"f_max(in Hz) = 1 / 2*pi*(T_C)*2^n =")
```

Chapter 10

Voltage Regulators

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

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

```
Z_Z / R3 =")
lr=0.02121*100/4.7
format(7)
disp(lr,"Therefore, Line regulation(in precentage) = deltaV_0*100 / V_0 =")
disp("Now   V_R(out) = V_R(in)*Z_Z / R3")
zz=7/330
format(8)
disp(zz,"Therefore, V_R(out)/V_R(in) = Z_Z/R3 =")
rr=20*log10(0.02121)
format(6)
disp(rr,"Therefore, RR(in dB) = 20*log(0.02121) = ")
```

Scilab code Exa 10.2 design op amp series voltage regulator

```
//Example 10.4
clc
disp("R1 = 5 k-ohm, R2 = 10 k-ohm")
disp("The IC is 7808 i.e. V_reg = +8 V")
vt=8*(3)
format(3)
disp(vt,"Therefore, V_out(in V) = V_reg*[1 + R2/R1] =")
disp("Now R2 = 1 k-ohm then,")
vo=8*(1+(1/5))
format(4)
disp(vo,"V_out(in V) = 8*[1 + 1/5] =")
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

```
//Example 10.4
clc
disp("R1 = 5 k-ohm,     R2 = 10 k-ohm")
disp("The IC is 7808 i.e. V_reg = +8 V")
vo=8*3
format(3)
disp(vo,"Therefore,  V_out(in V) = V_reg*[1 + R2/R1]
   =")
disp("Now  R2 = 1 k-ohm then,")
vou=8*(1+(1/5))
format(4)
disp(vou,"V_out(in V) =")
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`

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

Scilab code Exa 10.8 `find the range`

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

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