Scilab Textbook Companion for Modern Digital Electronics by R. P. Jain[1]

Created by
Shikha Garg
DIGITAL ELECTRONICS
Instrumentation Engineering
THAPAR UNIVERSITY
College Teacher
Mr. Sunil Kumar Singla
Cross-Checked by

August 17, 2013

[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: Modern Digital Electronics

Author: R. P. Jain

Publisher: Tata McGraw - Hill Education, New Delhi

Edition: 4

Year: 2010

ISBN: 9780070669116

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

Chapter 1

Fundamental Concepts

Scilab code Exa 1.1 And Gate

```
//example 1.1//
clc
//clears the screen//
clear
//clears the existing variables//
disp('the locker door (Y) can be opened using one
   key (A) which is with you and the other key (B)
   which is with the bank executive. When both the
   keys are used, the locker door opens, i.e. the
   locker door can be opened (Y=1) only when both
   the keys are applied(A=B=1).Thus, this can be
   expressed as an AND operation')
disp('Y=A*B')
```

Scilab code Exa 1.3 or gate

```
//example 1.3//
clc
//clears the screen//
clear
//clears already existing variables//
disp('Let the temperature and pressure be converted
    into electrical signals and T=1 if temperature
    exceeds the specified limit and P=1 if pressure
    exceeds the specified limit. If T=1 or P=1 or
    both T and P are 1 then the alarm is required to
    be activated, i.e., the signal applied to the
    alarm Y=1. This operation can be expressed as an
    or operation.')
disp('Y=T or P')
disp('Y=T+P')
```

Scilab code Exa 1.7.a NAND gate

```
//example 1.7(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('when one of the logic input of 2-input NAND
    gate is 0, then irrespective of the other input,
    the output comes out to be 1. In fact, a NAND
    gate is disabled or inhibited if one of its
    inputs is connected to logic 0')
disp('Y=1')
```

Scilab code Exa 1.7.b NAND gate with one permanently connected to logic 1

```
//example 1.7(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('when one of the logic input of 2-input NAND
   gate is 1, then when A=1, Y=0 and if A=0, Y=1')
disp('Y=A''')
```

Scilab code Exa 1.9.a NOR gate connected to 0 logic as one input

```
//example 1.9(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('when one of the logic input of 2-input NOR
   gate is 0, then when A=1, Y=0 and if A=0, Y=1')
disp('Y=A''')
```

Scilab code Exa 1.9.b NOR gate with one input connected to logic 1

```
//example 1.9(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('when one of the logic input of 2-input NOR
    gate is 1, then irrespective of the other input,
    the output comes out to be 0. In fact, a NAND
    gate is disabled or inhibited if one of its
    inputs is connected to logic 1')
disp('Y=0')
disp('here the output of Y is 0 irrespective of
    input of A')
```

Scilab code Exa 1.11.a EXOR gate with one input permanently as logic 0

```
//example 1.11(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('If we connect one input of EX-OR gate to 0
    permanently, we observe that Y=A*0''+A''*0')
disp('thus, Y=A')
```

Scilab code Exa 1.11.b EXOR gate with one input permanently as logic 1

```
//example 1.11(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('If we connect one input of EX-OR gate to 1
    permanently, we observe that Y=A*1''+A''*1')
disp('thus, Y=A''')
```

Scilab code Exa 1.13.a EXNOR gate with one input permanently as logic 0

```
//example 1.13(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('If we connect one input of EX-NOR gate to 0
    permanently, we observe that Y=A*0+A''*0''')
disp('thus, Y=A''')
```

Scilab code Exa 1.13.b EXNOR gate with one input permanently as logic 1

```
//example 1.13(b)//
clc
//clears the screen//
clear
//clears already existing variables//
```

```
disp('If we connect one input of EX-NOR gate to 1
   permanently, we observe that Y=A*1+A''*1''')
disp('thus, Y=A')
```

Chapter 2

Number System and Codes

Scilab code Exa 2.1 decimal equivalent

```
//example 2.1//
//decimal to binary conversion//
ans=bin2dec('11111')
//decimal equivalent of binary number//
disp(ans)
//answer in decimal form//
```

Scilab code Exa 2.2.a conversion

```
//example 2.2(a)//
//decimal to binary conversion//
ans=bin2dec('110101')
//decimal equivalent of binary number//
disp(ans)
//answer in decimal form//
```

Scilab code Exa 2.2.b conversion

```
//example 2.2(b)//
//decimal to binary conversion//
ans=bin2dec('101101')
//decimal equivalent of binary number//
disp(ans)
//answer in decimal form//
```

Scilab code Exa 2.2.c conversion

```
//example 2.2(c)//
//decimal to binary conversion//
ans=bin2dec('11111111')
//decimal equivalent of binary number//
disp(ans)
//answer in decimal form//
```

Scilab code Exa 2.2.d conversion

```
//example 2.2(d)//
```

```
//decimal to binary conversion//
ans=bin2dec('00000000')
//decimal equivalent of binary number//
disp(ans)
//answer in decimal form//
```

Scilab code Exa 2.3.a decimal representation

```
//example 2.3(a)//
clc
//clears the command window//
clear
//clears//
p =1;
//initialising//
q =1;
z =0;
b =0;
w =0;
f =0;
//bin= input (      Enter the binary no to be converted to its decimal equivalent :     )
//accepting the binary input from user//
bin =101101.10101;
d = modulo(bin,1) ;
//separating the decimal part and the integer part//
d= d *10^10;
a = floor(bin) ;
//removing the decimal part//
while (a >0)
// Loop to take the binary bits of integer into a matrix//
r = modulo (a ,10) ;
b(1,q) = r ;
a=a/10;
```

```
a=floor( a ) ;
q = q +1;
end
for m =1: q -1
// multipliying the bits of integer with their position values and adding//
c=m -1;
f=f+b(1,m)*(2^c);
end
while (d >0)
   // Loop to take the binary bits of decimal into a matrix//
    e = modulo (d ,2)
    w (1 , p ) = e
    d = d /10;
    d = floor ( d )
    p = p +1;
    end
for n =1: p -1
//multipliying the bits of decimal with their position values and adding//
z = z + w (1 , n ) *(0.5) ^(11 - n ) ;
end
z = z *10000;
//rounding of to 4 decimal values//
z = round ( z ) ;
z = z /10000;
x=f+z;
disp('The Decimal equivalent of the Binary number given is');
disp(x);
//Displaying the final result//
```

Scilab code Exa 2.3.b decimal representation

```
//example 2.3 (b)//
clc
//clears the command window//
clear
//clears//
p =1;
//initialising//
q =1;
z =0;
b =0;
w =0;
f =0;
//bin= input (      Enter the binary no to be converted to its decimal equivalent :     )
//accepting the binary input from user//
bin =1100.1011;
d = modulo(bin,1) ;
//separating the decimal part and the integer part//
d= d *10^10;
a = floor(bin) ;
//removing the decimal part//
while (a >0)
// Loop to take the binary bits of integer into a matrix//
r = modulo (a ,10) ;
b(1,q) = r ;
a=a/10;
a=floor( a ) ;
q = q +1;
end
for m =1: q -1
// multipliying the bits of integer with their position values and adding//
c=m -1;
f=f+b(1,m)*(2^c);
end
```

```
while (d >0)
   // Loop to take the binary bits of decimal into a matrix//
    e = modulo (d ,2)
    w (1 , p ) = e
    d = d /10;
    d = floor ( d )
    p = p +1;
    end
for n =1: p -1
//multipliying the bits of decimal with their position values and adding//
z = z + w (1 , n ) *(0.5) ^(11 - n ) ;
end
z = z *10000;
//rounding of to 4 decimal values//
z = round ( z ) ;
z = z /10000;
x=f+z;
disp('The Decimal equivalent of the Binary number given is');
disp(x);
//Displaying the final result//
```

Scilab code Exa 2.3.c decimal conversion

```
//example 2.3(c)//
clc
//clears the command window//
clear
//clears//
p =1;
//initialising//
```

```
q =1;
z =0;
b =0;
w =0;
f =0;
//bin= input (      Enter the binary no to be
   converted to its decimal equivalent :     )
//accepting the binary input from user//
bin =1001.0101;
d = modulo(bin,1) ;
//separating the decimal part and the integer part//
d= d *10^10;
a = floor(bin) ;
//removing the decimal part//
while (a >0)
// Loop to take the binary bits of integer into a
   matrix//
r = modulo (a ,10) ;
b(1,q) = r ;
a=a/10;
a=floor( a ) ;
q = q +1;
end
for m =1: q -1
// multipliying the bits of integer with their
   position values and adding//
c=m -1;
f=f+b(1,m)*(2^c);
end
while (d >0)
   // Loop to take the binary bits of decimal into a
      matrix//
    e = modulo (d ,2)
    w (1 , p ) = e
    d = d /10;
    d = floor ( d )
    p = p +1;
    end
```

```
for n =1: p -1
//multipliying the bits of decimal with their
   position values and adding//
z = z + w (1 , n ) *(0.5) ^(11 - n ) ;
end
z = z *10000;
//rounding of to 4 decimal values//
z = round ( z ) ;
z = z /10000;
x=f+z;
disp('The Decimal equivalent of the Binary number
   given is');
disp(x);
//Displaying the final result//
```

Scilab code Exa 2.3.d decimal representation

```
//example 2.3(d)//
clc
//clears the command window//
clear
//clears//
p =1;
//initialising//
q =1;
z =0;
b =0;
w =0;
f =0;
//bin= input (     Enter the binary no to be
   converted to its decimal equivalent :     )
//accepting the binary input from user//
bin =0.10101;
```

```
d = modulo(bin,1) ;
//separating the decimal part and the integer part//
d= d *10^10;
a = floor(bin) ;
//removing the decimal part//
while (a >0)
// Loop to take the binary bits of integer into a matrix//
r = modulo (a ,10) ;
b(1,q) = r ;
a=a/10;
a=floor( a ) ;
q = q +1;
end
for m =1: q -1
// multipliying the bits of integer with their position values and adding//
c=m -1;
f=f+b(1,m)*(2^c);
end
while (d >0)
   // Loop to take the binary bits of decimal into a matrix//
    e = modulo (d ,2)
    w (1 , p ) = e
    d = d /10;
    d = floor ( d )
    p = p +1;
    end
for n =1: p -1
//multipliying the bits of decimal with their position values and adding//
z = z + w (1 , n ) *(0.5) ^(11 - n ) ;
end
z = z *10000;
//rounding of to 4 decimal values//
z = round ( z ) ;
z = z /10000;
```

```
x=f+z;
disp('The Decimal equivalent of the Binary number
   given is');
disp(x);
//Displaying the final result//
```

Scilab code Exa 2.4 binary conversion

```
//example 2.4//
ans=dec2bin(13)
//conversion of decimal number to binary//
disp(ans)
//answer in binary form//
```

Scilab code Exa 2.5 conversion

```
//example 2.5//
clc
//clears the command window//
clear
//clears all the variables//
q =0;
b =0;
s =0;
//a=input(  Enter the decimal no to be converted to
    its binary equivalent:    ) ;
//accepting the decimal input from user//
```

```
a =0.65625;
d = modulo (a ,1) ;
//separating the decimal part and the integer part//
a = floor ( a ) ;
//removing the decimal part//
while (a >0)
//taking integer part into a matrix and convert to
    equivalent binary//
x = modulo (a ,2) ;
b = b + (10^ q ) * x ;
a = a /2;
a = floor ( a ) ;
q = q +1;
end
for i =1:10
//For values after decimal point converting to
    binary//
d = d *2;
q = floor ( d ) ;
s = s + q /(10^ i ) ;
     if d >=1 then
             d =d -1;
     end
end
k=b+s;
disp('The binary equivalent of the given decimal
    number is =');
disp(k);
//displaying the final result .
```

Scilab code Exa 2.6.a dec to bin

```
//example 2.6(a)//
```

```
clc
//clears the command window//
clear
//clears all the variables//
q =0;
b =0;
s =0;
//a=input( Enter the decimal no to be converted to its binary equivalent: ) ;
//accepting the decimal input from user//
a =25.5;
d = modulo (a ,1) ;
//separating the decimal part and the integer part//
a = floor ( a ) ;
//removing the decimal part//
while (a >0)
//taking integer part into a matrix and convert to equivalent binary//
x = modulo (a ,2) ;
b = b + (10^ q ) * x ;
a = a /2;
a = floor ( a ) ;
q = q +1;
end
for i =1:10
//For values after decimal point converting to binary//
d = d *2;
q = floor ( d ) ;
s = s + q /(10^ i ) ;
    if d >=1 then
            d =d -1;
    end
end
k=b+s;
disp('The binary equivalent of the given decimal number is =');
disp(k);
```

```
//displaying the final result//
```

Scilab code Exa 2.6.b dec to bin

```
//example 2.6(b)//
clc
//clears the command window//
clear
//clears all the variables//
q =0;
b =0;
s =0;
//a=input( Enter the decimal no to be converted to
    its binary equivalent:    ) ;
//accepting the decimal input from user//
a =10.625;
d = modulo (a ,1) ;
//separating the decimal part and the integer part//
a = floor ( a ) ;
//removing the decimal part//
while (a >0)
//taking integer part into a matrix and convert to
   equivalent binary//
x = modulo (a ,2) ;
b = b + (10^ q ) * x ;
a = a /2;
a = floor ( a ) ;
q = q +1;
end
for i =1:10
//For values after decimal point converting to
   binary//
d = d *2;
```

```
q = floor ( d ) ;
s = s + q /(10^ i ) ;
     if d >=1 then
             d =d -1;
     end
end
k=b+s;
disp('The binary equivalent of the given decimal
   number is =');
disp(k);
//displaying the final result .
```

Scilab code Exa 2.6.c dec to bin

```
//example 2.6(c)//
clc
//clears the command window//
clear
//clears all the variables//
q =0;
b =0;
s =0;
//a=input( Enter the decimal no to be converted to
    its binary equivalent: ) ;
//accepting the decimal input from user//
a =0.6875;
d = modulo (a ,1) ;
//separating the decimal part and the integer part//
a = floor ( a ) ;
//removing the decimal part//
while (a >0)
//taking integer part into a matrix and convert to
   equivalent binary//
```

```
x = modulo (a ,2) ;
b = b + (10^ q ) * x ;
a = a /2;
a = floor ( a ) ;
q = q +1;
end
for i =1:10
//For values after decimal point converting to
   binary//
d = d *2;
q = floor ( d ) ;
s = s + q /(10^ i ) ;
    if d >=1 then
            d =d -1;
    end
end
k=b+s;
disp('The binary equivalent of the given decimal
   number is =');
disp(k);
//displaying the final result//
```

Scilab code Exa 2.8.a ones complement

```
//example 2.8(a)//
//one's complement of binary number//
clc
//clears the screen//
clear
//clears all the existing variables//
x=bin2dec('0100111001')
//entering the data in binary form//
ans=dec2bin(bitcmp(x,10))
```

```
disp(ans);
//result will be displayed//
```

Scilab code Exa 2.8.b ones complement of unsigned no

```
//example 2.8(b)//
//one's complement of binary number//
clc
//clears the screen//
clear
//clears all the existing variables//
x=bin2dec('11011010')
//entering the data in binary form//
ans=dec2bin(bitcmp(x,8))
disp(ans);
//result will be displayed//
```

Scilab code Exa 2.9.a ones comp

```
//example 2.9(a)//
//representation in 1's complement form//
clc
//clears the screen//
clear
//clears all the existing variables//
x=7
//x=+7//y=-x//
xb=dec2bin(7)
```

```
//xb means binary conversion of x to its one's complement form//
xc=dec2bin(bitcmp(7,4))
//xc means conversion of y in its one's complement form//
disp('displaying result of +7 and -7 in ones complement form')
disp(x)
printf("=")
disp(xb)
//displaying answer in one's complement form//
disp(-x)
printf("=")
disp(xc)
//answer in one's complement form//
```

Scilab code Exa 2.9.b ones complement of signed no

```
//example 2.9(b)//
//representation in 1's complement form//
clc
//clears the window//
clear
//clears all the existing variables//
x=8
//x=+8//y=-x//
xb=dec2bin(8)
//xb means binary conversion of x to its one's complement form//
xc=dec2bin(bitcmp(8,5))
//xc means conversion of y in its one's complement form//
disp(x)
```

```
printf("=")
disp(xb)
//displaying answer in one's complement form//
disp(-x)
printf("=")
disp(xc)
//answer in one's complement form//
```

Scilab code Exa 2.9.c ones complement of signed no

```
//example 2.9(c)//
//representation in 1's complement form//
clc
//clears the window//
clear
//clears all the existing variables//
x=15
//x=+15//y=-x//
xb=dec2bin(15)
//xb means binary conversion of x to its one's complement form//
xc=dec2bin(bitcmp(15,5))
//xc means conversion of y in its one's complement form//
disp(x)
printf("=")
disp(xb)
//displaying answer in one's complement form//
disp(-x)
printf("=")
disp(xc)
//answer in one's complement form//
```

Scilab code Exa 2.10.a twos complement

```
//example 2.10(a)//
//find 2's complement of binary number//
clc
//clears the window//
clear
//clears all the existing variables//
x=01001110
//the number//
xd=bin2dec('01001110')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 01001110 is : ')
disp(xc1)
//answer in 2's complement form//
```

Scilab code Exa 2.10.b twos complement of binary no

```
//example 2.10(b)//
//find 2's complement of binary number//
clc
//clears the window//
clear
```

```
//clears all the existing variables//
x=00110101
//the number//
xd=bin2dec('00110101')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 00110101 is : ')
disp(xc1)
//answer in 2's complement form//
```

Scilab code Exa 2.11.a complement

```
//example 2.11(a)//
//find 2's complement of binary number//
clc
//clears the window//
clear
//clears all the existing variables//
x=01100100
//the number//
xd=bin2dec('01100100')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 01100100 is : ')
disp(xc1)
```

```
//answer in 2's complement form//
```

Scilab code Exa 2.11.b complement

```
//example 2.11(b)//
//find 2's complement of binary number//
clc
//clears the window//
clear
//clears all the existing variables//
x=10010010
//the number//
xd=bin2dec('10010010')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 10010010 is : ')
disp(xc1)
//answer in 2's complement form//
```

Scilab code Exa 2.11.c complement

```
//example 2.11(c)//
//find 2's complement of binary number//
clc
//clears the window//
```

```
clear
//clears all the existing variables//
x=11011000
//the number//
xd=bin2dec('11011000')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 11011000 is : ')
disp(xc1)
//answer in 2's complement form//
```

Scilab code Exa 2.11.d twos complement of binary number

```
//example 2.11(d)//
//find 2's complement of binary number//
clc
//clears the window//
clear
//clears all the existing variables//
x=01100111
//the number//
xd=bin2dec('01100111')
//binary to decimal conversion//
xc=bitcmp(xd,8)
//one's complement of the number//
xp=xc+1
xc1=dec2bin(xp)
//2's complement of the number//
disp('2s complement of 01100111 is : ')
```

```
disp(xc1)
//answer in 2's complement form//
```

Scilab code Exa 2.13.a addition

```
//example 2.13(a)//
//addition of binary number//
clc
//clears the screen//
clear
//clears already existing variables//
x=bin2dec('1011')
//binary to decimal conversion//
y=bin2dec('1100')
z=x+y
//addition//
a=dec2bin(z)
//decimal to binary conversion//
disp('the addition of given numbers is:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.13.b addition of binary numbers

```
//example 2.13(b)//
//addition of binary number//
clc
//clears the screen//
clear
```

```
//clears already existing variables//
x=bin2dec('0101')
//binary to decimal conversion//
y=bin2dec('1111')
z=x+y
//addition//
a=dec2bin(z)
//decimal to binary conversion//
disp('the addition of given numbers is:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.14 addition of 4 binary numbers

```
//example 2.14//
//addition of binary numbers//
clc
//clears the screen//
clear
//clears already existing variables//
x=bin2dec('01101010')
//x is the first number in addition//
//binary to decimal conversion//
y=bin2dec('00001000')
//y is the second number in addition//
t=bin2dec('10000001')
//t is the third number in addition//
k=bin2dec('11111111')
//k is the fourth number we have to end//
z=x+y+t+k
//addition//
a=dec2bin(z)
//decimal to binary conversion//
```

```
disp('the addition of given numbers is:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.15 subtraction of binary numbers

```
//example 2.15//
//subtraction of two binary number//
clc
//clears the screen//
clear
//clears the existing variables//
x=bin2dec('1011')
//x is the minuend//
//binary to decimal conversion//
y=bin2dec('0110')
//y is the subtrahend//
z=x-y
//subtraction//
disp('the subtraction of given numbers is:')
ans=dec2bin(z)
//decimal to binary conversion//
disp(ans)
//answer in binary form//
```

Scilab code Exa 2.16 multiplication of binary numbers

```
//example 2.16//
//multiplication in binary form//
```

```
clc
//clears the screen//
clear
//clears all the existing variables//
x=bin2dec('1001')
//first number to be multiplied is x//
//binary to decimal conversion//
y=bin2dec('1101')
//second number to be multiplied is y//
z=x*y
//multiplication//
a=dec2bin(z)
//decimal to binary conversion//
disp('the multiplication of given numbers results in
    :')
disp(a)
//answer in binary number//
```

Scilab code Exa 2.17 division of binary numbers

```
//example 2.17//
//division in binary//
clc
//clears the window//
clear
//clears already existing variables//
x=bin2dec('1110101')
//x is the first number//
//binary to decimal conversion//
y=bin2dec('1001')
//y is the second number w/c is to be divided//
z=x/y
//division//
```

```
a=dec2bin(z)
//decimal to binary conversion//
disp('the division of given numbers results in:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.21 decimal to octal conversion

```
//example 2.21//
clc
//clears the screen//
clear
//clears already existing variables//
x=247
//decimal to octal conversion//
a=dec2oct(x)
disp('the octal conversion of given no is:')
disp(a)
//answer in octal form//
```

Scilab code Exa 2.22 convert octal to binary

```
//example 2.22//
clc
//clears the screen//
clear
//clears already existing variables//
//octal to binary conversion//
y=oct2dec('736')
```

```
//octal to decimal conversion//
a=dec2bin(y)
//decimal to binary conversion//
disp('binary conversion of given no is:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.23 binary to octal

```
//example 2.23//
clc
//clears the screen//
clear
//clears already existing variables//
//binary to octal conversion//
y=bin2dec('1001110')
//binary to decimal conversion//
a=dec2oct(y)
//decimal to octal conversion//
disp('octal representation of given no is :')
disp(a)
//answer in octal form//
```

Scilab code Exa 2.26 addition of octal numbers

```
//example 2.26//
//addition of octal numbers//
clc
//clears the screen//
```

```
clear
//clears already existing variables//
x=oct2dec('23')
//octal to decimal conversion//
y=oct2dec('67')
z=x+y
//addition//
a=dec2oct(z)
//decimal to octal conversion//
b=dec2bin(z)
//decimal to binary conversion//
disp('addition of given no is octal form is')
disp(a)
//answer in octal form//
disp('addition of given numbers in binary form is:')
disp(b)
//answer in binary form//
```

Scilab code Exa 2.27.a addition of octal numbers

```
//example 2.27 (a)//
//subtraction of octal numbers//
clc
//clears the screen//
clear
//clears already existing variables//
x=oct2dec('53')
//octal to decimal conversion//
y=oct2dec('37')
z=x-y
//subtraction//
a=dec2oct(z)
//decimal to octal conversion//
```

```
b=dec2bin(z)
//decimal to binary conversion//
disp('result of subtraction of given numbers in
    octal form is:')
disp(a)
//answer in octal form//
disp('result of subtraction of given numbers in
    binary form is:')
disp(b)
//answer in binary form//
```

Scilab code Exa 2.27.b addition of OCTAL numbers

```
//example 2.27 (b)//
clc
//clears the screen//
clear
//clears already existing variables//
x=oct2dec('26')
//octal to decimal conversion//
y=oct2dec('75')
z=x-y
//subtraction//
t=z*(-1)
t1=bitcmp(t,8)
//1's complement//
t2=t1+1
//2's compliment//
a=dec2bin(t2)
//decimal to binary conversion//
disp('answer in 2''s compliment form:')
disp(a)
//answer in 2's complement form//
```

Scilab code Exa 2.30 convert hexadecimal to binary

```
//example 2.30//
//hexadecimal to binary conversion//
clc
//clears the screen//
clear
//clears already existing variables//
x=hex2dec('29FA')
//hexadecimal to decimal conversion//
a=dec2bin(x)
//decimal to binary conversion//
disp('conversion of hexadecimal given no to its
   binary form is:')
disp(a)
//answer in binary form//
```

Scilab code Exa 2.31 binary to hexadecimal number

```
//example 2.31//
//conversion of binary to hexadecimal//
clc
//clears the screen//
clear
//clears already existing variables//
x=bin2dec('10100110101111')
//binary to decimal conversion//
```

```
a=dec2hex(x)
//decimal to hexadecimal conversion//
disp('conversion of given binary number to its
   hexadecimal form is:')
disp(a)
//answer in hexadecimal form//
```

Scilab code Exa 2.33 hexadecimal to octal number

```
//example 2.33//
//conversion hexadecimal number to octal number//
clc
//clears the screen//
clear
//clears already existing variables//
x=hex2dec('A72E')
//hexadecimal to decimal conversion//
a=dec2oct(x)
//decimal to octal conversion//
disp('conversion of given hexadecimal no to its
   octal form results in :')
disp(a)
//answer in octal form//
```

Scilab code Exa 2.35 addition of hexadecimal numbers

```
//example 2.35//
//addition of hexadecimal number//
clc
```

```
//clears the screen//
clear
//clears already existing variables//
x=hex2dec('7F')
//hexadecimal to decimal conversion//
y=hex2dec('BA')
z=x+y
//addition//
a=dec2hex(z)
//decimal to hexadecimal conversion//
disp('addition of given hexadecimal numbers results
   in :')
disp(a)
//answer in hexadecimal form//
```

Scilab code Exa 2.36.a subtraction of hexadecimal numbers

```
//example 2.36(a)//
//subtraction of hexadecimal number//
clc
//clears the screen//
clear
//clears already existing variables//
x=hex2dec('3F')
//hexadecimal to deciimal conversion//
y=hex2dec('5C')
z=x-y
//subtraction//
t=z*-1
t1=dec2hex(t)
//answer in hexadecimal form(modulus)//
t2=bitcmp(t,8)
//complement//
```

```
t3=t2+1
//2's complement//
a=dec2bin(t3)
//answer in 2's complement form
disp('result of subtraction in 2''s compliment form
   is:')
disp(a)
```

Scilab code Exa 2.36.b subtraction of hexadecimal numbers

```
//example 2.36(b)//
//subtraction of hexadecimal numbers//
clc
//clears the screen//
clear
//clears already existing variables//
x=hex2dec('C0')
//hexadecimal to decimal conversion//
y=hex2dec('7A')
z=x-y
//subtraction//
a=dec2hex(z)
//decimal to hexadecimal conversion//
b=dec2bin(z)
//decimal to binary conversion//
disp('answer in hexadecimal form is:')
disp(a)
//answer in hexadecimal form//
disp('answer in binary form is:')
disp(b)
//answer in binary number//
```

Chapter 3

Semiconductor devices switching mode operation

Scilab code Exa 3.3 response of transistor inverter

```
//example 3.3//
clc
//clears the screen//
clear
//clears already existing varibales//
a=10;
//input voltage (in volts)//
b=.7;
//transistor voltage(saturation voltage)//
c=5;
//resistor b/w input voltage and the transistor//
d=10;
//input voltage from collector side//
e=0.1;
//transistor voltage(saturation voltage from
    collector side)//
f=2;
```

```
//resistor in kilo-ohm//
g=30;
h=-10;
//input voltage from emitter side//
I=(a-b)/c;
//base current of transistor from given figure//
disp('the base current of given circuit is (in mA):'
   )
disp(I)
//base current is in mA//
K=(d-e)/f
//collector current of transistor from given figure
   //
disp('the collector current of given circuit is (in
   mA):')
disp(K)
//collector current in mA(saturation current)//
L=K/g
disp('base current required for the transistor to be
    in saturation is (in mA):')
disp(L)
//current in mA//
M=(h-b)/c
disp('the base current is (in mA):')
disp(M)
//base current in mA//
```

Scilab code Exa 3.4.a output voltage of JFET

```
//example 3.4(a)//
clc
//clears the screen//
clear
```

```
//clears already existing variables//
disp('when the input voltage = −5V, the JFET is
   opening at point A, where I(D)=0 and V(0)=V(DD)
   =20V')
disp('this corresponds to the switch in OFF state')
```

Scilab code Exa 3.4.b output voltage of JFET

```
//example 3.4(b)//
clc
//clears the screen//
clear
//clears the command window//
disp('when V(i)=0V, the JFET is operating at point B
   , where I(D)=3.8mA and V(o)=1V')
disp('this corresponds to the switch in ON stage')
//the answers have been taken directly from the
   figure//
```

Scilab code Exa 3.5.a output characateristics of a MOSFET

```
//example 3.5(a)//
clc
//clears the window//
clear
//clears already existing variables//
disp('when V(i)=0, the transistor is cutoff because
   the voltage between the gate and the source is
```

```
below the threshold voltage. Correspondingly the
   output voltage V(0)=5V(point (N) as in figure)')
//answer according to the cuts of load line//
```

Scilab code Exa 3.5.b output voltage for MOSFET

```
//example 3.5(b)//
clc
//clears the window//
clear
//clears already existing variables//
disp('when V(i)=5V, the transistor is operating at
   point M and V(o)=0')
disp('this corresponds to ON stage!')
```

Scilab code Exa 3.7.a output voltage for identical set of transistors

```
//example 3.7(a)//
clc
//clears the screen//
clear
//clears existing variables//
disp('when V(i)=0, the transistor T(1) is operating
   at point B')
t=5;
//input voltage as given in question//
x=0;
V=t-x;
//output voltage in volts//
```

```
disp('here V(o)(in volts)=')
disp(V)
```

Scilab code Exa 3.7.b output voltage in identical transistors for input voltage 5V

```
//example 3.7(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('when V(i)=5V, the transistor T(1) is operating
    at point C')
V=0;
disp('output voltage in volts=')
disp(V)
//all the outcomes are as per the diagram//
```

Chapter 4

Digital Logic Families

Scilab code Exa 4.1.a.i calculate fan out when all inputs are high

```
//example 4.1(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('as given low level, V(o)=0.2V and high level,
    V(1)=5V')
disp('case 1, when all the inputs are HIGH')
//in this case both diodes will be conducting and
    transistor will be in saturation//
a=0.7+0.7+0.8;
//V(p)//
b=5;
//high level voltage in volts//
c=5;
//resistor in Kohms//
d=0.8;
//voltage between base and emitter in volts//
e=(b-a)/c;
```

```
//I(1)//
f=d/c;
//I(2)//
i=e-f;
//writing kirchoff's current law at the base of transistor//
g=0.2;
//voltage between collector and emitter in saturation(in volts)//
h=2.2;
//resistance of collector in Kohms//
j=(b-g)/h;
//collector current without load gate connected//
disp('I(1) in mA:')
disp(e)
disp('I(2) in mA:')
disp(f)
disp('collector current(in mA):')
disp(j)
k=30;
//h(FE)//
s=k*i;
if(s>j)
disp('transistor is in saturation mode')
disp('fan out is given by I(c)<=h(FE)I(B)')
end
```

Scilab code Exa 4.1.a.ii calculate fan out for DTL NAND gate when atleast one input is LOW

```
//example 4.1.a(ii)//
clc
//clears the screen//
```

```
clear
//clears already existing variables//
//for DTL NAND gate calculate fan out//
disp('case II , if atleast one of the inputs is LOW'
   )
v=0.2+0.7;
m=0.6+0.6+0.5;
//min voltage for both diodes and transistor to be
   conducting//
disp(v)
disp('min voltage for both diodes and transistor to
   be conducting:')
disp(m)
if(v<m)
disp('transistor is in cut off mode')
disp('if the load gates are connected, the input
   diodes of the load gates are non conducting,
   which means the reverse saturation current of
   these diodes must be supplied through the
   collector resistor R(c), which will produce a
   voltage drop across R(c) and consequently the
   output voltage corresponding to HIGH state will
   be a little less than V(cc). The maximum current
   which can be supplied by the gate will depend
   upon V(OH). The fan out is determined on the
   basis of maximum current.')
end
```

Chapter 5

Combinational Logic Design

Scilab code Exa 5.1 simplification of equation

```
//exmaple 5.1(c)//
//simplification of an equation//
clc
//clears the screen//
clear
//clears already existing variables//
disp('given=> Y=(A+BC)(B+C''A)')
//given in the question//
disp('on multiplication we get')
disp('Y=AB+AC''+BC')
disp('on further simplification we get')
disp('BC+AC''')
//answer after simplification//
//as per theorem 1.19 given in the question//
```

Scilab code Exa 5.2 conversion to canonical SOP form

```
//example 5.2//
clc
//clears the screen//
clear
//clears already existing variables//
//conversion of given equation to its canonical SOP form//
disp('given=> Y=(AB+AC''+BC)')
disp('on solving')
disp('Y=AB(C+C'')+AC''(B+B'')+BC(A+A'')')
disp('Y=ABC+ABC''+ABC''+AB''C''+ABC+A''BC')
disp('Y=ABC+ABC''+AB''C''+A''BC')
//using theorem 1.5//
//result//
```

Scilab code Exa 5.3 conversion to canonical POS form

```
//example 5.3//
clc
//clears the screen//
clear
//clears already existing variables//
//conversion of given equation to its canonical POS form//
disp('given=> Y=(A+B)(A+C)(B+C'')')
disp('on solving')
disp('Y=(A+B+CC'')(A+BB''+C)(AA''+B+C'')')
disp('Y=(A+B+C)(A+B+C'')(A+C+B)(A+C+B'')(A+B+C'')(A''+B+C'')')
//using theorem 1.10//
disp('Y=(A+B+C)(A+B+C'')(A+B''+C)(A''+B+C'')')
```

```
//using theorem 1.6//
//result//
```

Chapter 6

Combinational Logic Design using MSI circuits

Scilab code Exa 6.1 boolean equation using 8 to 1 mux

```
//example 6.1//
//boolean equation using 8 to 1 mux//
clc ;
clear
a (1 ,1) =0
//taking input in this form 1 if A, 0 if A  and 2
    if no A in the term//
a (1 ,2) =1
a (1 ,3) =2
a (2 ,1) =2
a (2 ,2) =0
a (2 ,3) =0
a (3 ,1) =1
a (3 ,2) =1
a (3 ,3) =1
p =3;
for i =1:3
```

```
//finding them in terms here//
      coun =0;
      for j =1:3
      if a (i , j ) ==2 then
      coun = coun +1
      end
  end
  if coun == 2 then
        p = p +3
  else if coun ==1 then
      p = p +1
  end
end
end
n =4;
for m =4: p
      for l =1:3
              a (m , l ) =0;
      end
end
for i = 1: p
      for j =1:3
              if a (i , j ) ==2 then
                     for k =1:3

a (n , k ) = a (i , k )
end
a (i , j ) = 0;
a (n , j ) =1;
n = n +1;
end
end
end
for h =1: p
f ( h ) =0
c =2;
for m =1:3
//finding equivlent//
```

```
end
end
disp ('the min terms are decimal values for the
    minterms f ( h ) = f ( h ) + a (h , m ) *(2^ c )
    ;c =c -1; : ')
//displaying the min terms//
disp('ABC')
disp(a)
l =1
o (1 , l ) = f ( l ) ;
//removing the repetations in minterms//
for i =2: p
       q =0;
             for b =1: l
             if o (1 , b ) == f ( i ) then
                     q =89 ;
             end
       end
       if q ==0 then
             o (1 , l +1) = f ( i ) ;
             l = l +1;
             q =0;
       end
end
disp ('The following data lines are to be given
        1          and remaining should be given
        0') ;
//displaying the decimal equivlent of minterms//
disp(o) ;
disp('For a 4   1  mux, we should give D0= C       , D1
    =       1        , D2 = C'' and D3 = C with A and
    B as data selector inputs');
```

Scilab code Exa 6.4.a subtraction

```
//exmple 6 .4(a)//
clc
//clears the screen//
clear
//clears already existing variables//
a =0;
b =0;
q =0;
//bb=input( Enter the first no (in decimal) :
    ) ;
//aa=input(  Enter the number from which first no
    has to subtracted :      ) ;
aa =9;
bb =5;
while ( aa >0)
//converting the inputs into binary numbers//
       x = modulo ( aa ,2) ;
       a = a + (10^ q ) * x ;
       aa = aa /2;
       aa = floor ( aa ) ;
       q = q +1;
end
nn = a
q =0;
while ( bb >0)
       x = modulo ( bb ,2) ;
       b = b + (10^ q ) * x ;
     bb = bb /2;

bb = floor ( bb ) ;
q = q +1;
end
printf ( ' \n The binary equivalent of first no is
    %f\n\n ' ,b ) ;
printf ( 'The binary equivalent of second no is %f\n
    \n' ,a ) ;
```

```
for i =1:40
      a1 ( i ) = modulo (a ,10) ;
      a = a /10;
      a = round ( a ) ;
      b1 ( i ) = modulo (b ,10) ;
      b = b /10;
      b = round ( b ) ;
end
bro (1) =0;
for i =1:40
     c1 ( i ) = a1 ( i ) - b1 ( i ) - bro ( i ) ;
     if c1 ( i ) == -1 then
            bro ( i +1) = 1;
            c1 ( i ) =1;
     elseif c1 ( i ) == -2 then
              bro ( i +1) = 1;
            c1 ( i ) =0;
     else
            bro ( i +1) =0;
     end
   end
   re =0;
   format ( 'v' ,18) ;
   for i =1:40
         re = re +( c1 ( i ) *(10^( i -1) ) )
   end
   printf ( ' The difference of given two numbers is
       %f\n\n ' , re ) ;
q =1;
b =0;
f =0;
a = re ;
while (a >0)
//converting the binary result to decimal then to
   hexadecimal//
 r = modulo (a ,10) ;
 b (1 , q ) = r ;
 a = a /10;
```

```
 a = floor ( a ) ;
 q = q +1;
end
for m =1: q -1
       c =m -1
      f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ( ' Difference in decimal notation is %d\n\n
   ' ,f ) ;
printf ( ' Difference in hexadecimal notation is %s
   \n ' , hex ) ;
```

Scilab code Exa 6.4.b subtraction

```
//exmple 6 .4(b)//
clc
//clears the screen//
clear
//clears already existing variables//
a =0;
b =0;
q =0;
//bb=input(  Enter the first no (in decimal) :
   ) ;
//aa=input(   Enter the number from which first no
   has to subtracted :      ) ;
aa =8;
bb =1;
while ( aa >0)
//converting the inputs into binary numbers//
       x = modulo ( aa ,2) ;
       a = a + (10^ q ) * x ;
```

```
        aa = aa /2;
        aa = floor ( aa ) ;
        q = q +1;
end
nn = a
q =0;
while ( bb >0)
        x = modulo ( bb ,2) ;
        b = b + (10^ q ) * x ;
      bb = bb /2;

bb = floor ( bb ) ;
q = q +1;
end
printf ( ' \n The binary equivalent of first no is %f\n\n ' ,b ) ;
printf ( 'The binary equivalent of second no is %f\n\n' ,a ) ;
for i =1:40
       a1 ( i ) = modulo (a ,10) ;
       a = a /10;
       a = round ( a ) ;
       b1 ( i ) = modulo (b ,10) ;
       b = b /10;
       b = round ( b ) ;
end
bro (1) =0;
for i =1:40
      c1 ( i ) = a1 ( i ) - b1 ( i ) - bro ( i ) ;
      if c1 ( i ) == -1 then
             bro ( i +1) = 1;
             c1 ( i ) =1;
      elseif c1 ( i ) == -2 then
               bro ( i +1) = 1;
             c1 ( i ) =0;
      else
             bro ( i +1) =0;
      end
```

```
    end
    re =0;
    format ( 'v' ,18) ;
    for i =1:40
          re = re +( c1 ( i ) *(10^( i -1) ) )
    end
    printf ( ' The difference of given two numbers is
        %f\n\n ' , re ) ;
q =1;
b =0;
f =0;
a = re ;
while (a >0)
//converting the binary result to decimal then to
   hexadecimal//
 r = modulo (a ,10) ;
 b (1 , q ) = r ;
 a = a /10;
 a = floor ( a ) ;
 q = q +1;
end
for m =1: q -1
       c =m -1
      f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ( ' Difference in decimal notation is %d\n\n
   ' ,f ) ;
printf ( ' Difference in hexadecimal notation is %s
   \n ' , hex ) ;
```

Scilab code Exa 6.5.a.i 8 bit adder

```
// exmple 6.5(a)(i)
clc ;
clear ;
// a=input ( enter the first 8 bit number : ) ;
// b=input ( enter the second 8 bit number : ) ;
  a =01100001  ;
//taking given inputs//
  b =00011101 ;
  for i =1:8
        a1 ( i ) = modulo (a ,10) ;
        a = a /10;
        a = round ( a ) ;
        b1 ( i ) = modulo (b ,10) ;
        b = b /10;
        b = round ( b ) ;
  end
  car (1) =0;
  for i =1:8
//adding both the inputs (binary addition)//
c1 ( i ) = car ( i ) + a1 ( i ) + b1 ( i ) ;
if c1 ( i ) == 2 then
car ( i +1) = 1;

    c1 ( i ) =0;
elseif c1 ( i ) ==3 then
     car ( i +1) = 1;
    c1 ( i ) =1;
else
    car ( i +1) =0;
end
end
c1 (9) = car (9) ;
re =0;
format ( 'v' ,18) ;
for i =1:9
      re = re +( c1 ( i ) *(10^( i -1) ) )
end
```

```
printf ( ' The sum of given two binary numbers is %d \n ', re ) ;
q =1;
b =0;
f =0;
a = re ;
while (a >0)
//converting the result to a hexadecimal no//
       r = modulo (a ,10) ;
       b (1 , q ) = r ;
       a = a /10;
       a = floor ( a ) ;
       q = q +1;
end
for m =1: q -1
         c =m -1;
       f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ('The sum in hexadecimal notation is %s \n' , hex ) ;
disp('the sum in decimal form is:')
disp(f)
//displaying result//
```

Scilab code Exa 6.5.a.ii 8 bit subtractor

```
// exmple 6.5.a(ii)
clc ;
clear ;
a =0;
b =0;
q =0;
```

```
   // bb=input(      Enter the first no (in decimal) :      ) ;
   // aa=input(      E n t e r t h e number from which first no has to substracted :      ) ;
aa =97;
//taking the given input//
bb =29;

while ( aa >0)
//converting the inputs into binary numbers//
      x = modulo ( aa ,2) ;
      a = a + (10^ q ) * x ;
      aa = aa /2;
      aa = floor ( aa ) ;
      q = q +1;
end
q =0;
while ( bb >0)
      x = modulo ( bb ,2) ;
      b = b + (10^ q ) * x ;
    bb = bb /2;
      bb = floor ( bb ) ;
      q = q +1;
end
printf ( '\nThe binary equivalent of first no is %f\n \n ' ,b ) ;
printf ( 'The binary equivalent of second no is %f\n\n' ,a ) ;
for i =1:40
      a1 ( i ) = modulo (a ,10) ;
      a = a /10;
      a = round ( a ) ;
      b1 ( i ) = modulo (b ,10) ;
      b = b /10;
      b = round ( b ) ;
end
bro (1) =0;
for i =1:40
```

```
      c1 ( i ) = a1 ( i ) - b1 ( i ) - bro ( i ) ;
//finding the difference of the given inputs//
      if c1 ( i ) == -1 then
             bro ( i +1) = 1;
             c1 ( i ) =1;
      elseif c1 ( i ) == -2 then
               bro ( i +1) = 1;
             c1 ( i ) =0;

else
bro ( i +1) =0;
end
end
re =0;
format ( 'v' ,18) ;
for i =1:40
       re = re +( c1 ( i ) *(10^( i -1) ) )
end
printf ( ' The diference of given two numbers is %f\n\n ' , re ) ;
q =1;
b =0;
f =0;
a = re ;
while (a >0)
       r = modulo (a ,10) ;
       b (1 , q ) = r ;
       a = a /10;
       a = floor ( a ) ;
       q = q +1;
end
for m =1: q -1
         c =m -1
       f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ( ' Difference in decimal notation is %d\n\n ' ,f ) ;
```

```
//displaying the results//
printf ( ' Difference in hexadecimal notation is %s
   \n ' , hex ) ;
```

Scilab code Exa 6.5.b.i 8 bit adder

```
// exmple 6.5(b)(i)
clc ;
clear ;
// a=input ( enter the first 8 bit number :   ) ;
// b=input ( enter the second 8 bit number :     )
    ;
   a =00011000  ;
//taking given inputs//
   b =00111010 ;
   for i =1:8
         a1 ( i ) = modulo (a ,10) ;
         a = a /10;
         a = round ( a ) ;
         b1 ( i ) = modulo (b ,10) ;
         b = b /10;
         b = round ( b ) ;
   end
   car (1) =0;
   for i =1:8
//adding both the inputs (binary addition)//
c1 ( i ) = car ( i ) + a1 ( i ) + b1 ( i ) ;
if c1 ( i ) == 2 then
car ( i +1) = 1;

     c1 ( i ) =0;
elseif c1 ( i ) ==3 then
      car ( i +1) = 1;
```

```
      c1 ( i ) =1;
else
      car ( i +1) =0;
end
end
c1 (9) = car (9) ;
re =0;
format ( 'v' ,18) ;
for i =1:9
       re = re +( c1 ( i ) *(10^( i -1) ) )
end
printf ( ' The sum of given two binary numbers is %d
   \n ', re ) ;
q =1;
b =0;
f =0;
a = re ;
while (a >0)
//converting the result to a hexadecimal no//
       r = modulo (a ,10) ;
       b (1 , q ) = r ;
       a = a /10;
       a = floor ( a ) ;
       q = q +1;
end
for m =1: q -1
         c =m -1;
       f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ('The sum in hexadecimal notation is %s \n' ,
   hex ) ;
disp('the sum in decimal form is:')
disp(f)
//displaying result//
```

Scilab code Exa 6.5.b.ii 8 bit subtractor

```
// exmple 6.5.b(ii)
clc ;
clear ;
a =0;
b =0;
q =0;
   // bb=input(    Enter the first no (in decimal) :    ) ;
   // aa=input(    E n t e r t h e number from which first no has to substracted :    ) ;
aa =58;
//taking the given input//
bb =24;

while ( aa >0)
//converting the inputs into binary numbers//
      x = modulo ( aa ,2) ;
      a = a + (10^ q ) * x ;
      aa = aa /2;
      aa = floor ( aa ) ;
      q = q +1;
end
q =0;
while ( bb >0)
      x = modulo ( bb ,2) ;
      b = b + (10^ q ) * x ;
    bb = bb /2;
      bb = floor ( bb ) ;
      q = q +1;
end
```

```
printf ( '\nThe binary equivalent of first no is %f\n \n ' ,b ) ;
printf ( 'The binary equivalent of second no is %f\n\n' ,a ) ;
for i =1:40
      a1 ( i ) = modulo (a ,10) ;
      a = a /10;
      a = round ( a ) ;
      b1 ( i ) = modulo (b ,10) ;
      b = b /10;
      b = round ( b ) ;
end
bro (1) =0;
for i =1:40
      c1 ( i ) = a1 ( i ) - b1 ( i ) - bro ( i ) ;
//finding the difference of the given inputs//
      if c1 ( i ) == -1 then
             bro ( i +1) = 1;
             c1 ( i ) =1;
      elseif c1 ( i ) == -2 then
               bro ( i +1) = 1;
             c1 ( i ) =0;

else
bro ( i +1) =0;
end
end
re =0;
format ( 'v' ,18) ;
for i =1:40
       re = re +( c1 ( i ) *(10^( i -1) ) )
end
printf ( ' The diference of given two numbers is %f\n\n ' , re ) ;
q =1;
b =0;
f =0;
a = re ;
```

```
while (a >0)
        r = modulo (a ,10) ;
        b (1 , q ) = r ;
        a = a /10;
        a = floor ( a ) ;
        q = q +1;
end
for m =1: q -1
          c =m -1
        f = f + b (1 , m ) *(2^ c ) ;
end
hex = dec2hex ( f ) ;
printf ( ' Difference in decimal notation is %d\n\n' ,f ) ;
//displaying the results//
printf ( ' Difference in hexadecimal notation is %s\n ' , hex ) ;
```

Chapter 7

Flip flops

Scilab code Exa 7.1 clocked SR flip flop

```
//example 7.1//
clc
//clears the screen//
clear
//clears the command window//
e=input('Enter the enable i/p level (1or0) :');
//accepting the input of enable//
r=input('enter the R i/p level(1or0):');
//accepting the inputs from the user//
s=input('enter the S i/p level(1or0):');
//accepting the input S from the user//
qn=input('Enter the previous output value (1or0):');
//accepting the old input from the user//
flag=0;
if e==0 then
//calculating the output//
op=qn ;
elseif(s==0 & r ==0) then
op=qn ;
```

```
elseif ( s ==1& r ==1) then
disp('The inputs are illegal')
flag=1;
else
op=s;
end
if(flag==0)
disp('output(Qn+1)=')
disp(op)
end

//displaying the output//
```

Scilab code Exa 7.2 convert SR flip flop to JK flip flop

```
//example 7.2//
clc
//clears the screen//
clear
//clears the command window//
disp('for SR flip flop Q(n+1)=S+R''Q(n)')
disp('thus with S=JQ'' & R = KQ we get circuit which
    behaves like JK flip flop')
```

Chapter 8

Sequential Logic Design

Scilab code Exa 8.2 maximum frequency

```
//example 8.2//
clc
//clears the screen//
clear
//clears the command window//
s=4;
//s=stage of ripple counter//
d=50;
//delay of flip-flop in nano sec//
p=30;
//pulse width in nano secs//
f=1000/(s*d+p);
disp('maximum frequency (in MHz) is')
disp(f)
```

Chapter 9

Timing Circuits

Scilab code Exa 9.3 Schmitt trigger

```
//example 9.3(a)//
clc
clear
V=0.1*13/100.1;
X=0.1*(-13)/100.1;
disp('V(UT)=')
disp(V)
disp('V(LT)=')
disp(X)
```

Chapter 10

A to D and D to A Converters

Scilab code Exa 10.1 find analog output of 4 bit D to A converter

```
//example 10(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('digital input to analog output of 4 bit is as
   follows')
for(i=0:15)
x=dec2bin(i);
disp(x)
disp(i);
i=i+1;
end
//displays the result//
```

Scilab code Exa 10.2.a 4 bit unipolar D to A converter

```
//example 10.2(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('digital input to analog output of 4 bit is as
   follows')
for(i=0:15)
x=dec2bin(i);
//conversion of decimal to binary//
disp(x)
//binary form of the number//
disp(i);
//decimal form of the number//
i=i+1;
end
//displays the result//
```

Scilab code Exa 10.2.b 4 bit unipolar D to A converter after adjusting the offset voltage

```
//example 10.2(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('digital input to analog output of 4 bit is as
   follows')
for(i=0:15)
x=dec2bin(i);
//decimal to binary conversion//
```

```
disp(x)
//binary number//
j=i-8;
disp(j);
//analog number//
i=i+1;
end
//displays the result//
```

Scilab code Exa 10.2.c 4 bit unipolar D to A converter after complimenting MSB

```
//example 10.2(c)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('digital input to analog output of 4 bit as per
    given condition is as follows')
for(i=0:7)
x=dec2bin(i);
//decimal to binary conversion//
disp(x)
//displays binary or say digital form//
disp(i)
//displays analog form//
i=i+1;
end
for(i=8:15)
x=dec2bin(i);
//conversion//
disp(x)
f=i-8;
```

```
y=2*f;
t=y-i;
disp(t)
//displays analog form//
i=i+1
end
//displays the result//
```

Scilab code Exa 10.3 D to A converter in ones complement form

```
//example 10.3//
clc
//clears the window//
clear
//clears already existing variables//
disp('since the 1''s compliment representations of
   the positive numbers +0 to +7 are same as the
   representations of the unipolar binary numbers,
   no offset voltage is required for these inputs.')
disp('For the negative numbers 1111 to 1000, the
   output analog voltage is to be offset by -15V.
   This can be achieved by operating a switch with
   MSB of input to introduce proper value of Voff.')
//answer//
```

Scilab code Exa 10.4 2 decade BCD D to A converter

```
//example 10.4//
//design a 2 decade BCD D/A converter//
```

```
clc
//clears the screen//
clear
//clears already existing variables//
disp('the circuit of given figure can be used for
    BCD D/A converter. The binary inputs
    corresponding to LSB are applied to b3,b2,b1,b0
    and those corresponding to the next digit at b7,
    b6,b5,b4. the value of r is chosen so as to make
    the input current of OP-AMP corresponding to LSD
    as 1/10th of that of current due to MSD, and is
    given by')
disp('((V(R)*(8/7*R))/(R*(r+8*R/7)+r*8/7*R))=V(R)
    /(10*R)')
disp('r=4.8R')
```

Scilab code Exa 10.5 determine the quantization interval

```
//example 10.5//
clc
//clears the window//
clear
//clears already existing variables//
disp('the digital value 000 should be assigned to
    the analog voltage interval 0V +-S/2, Since in 2'
    's compliment representation, there is one more
    negative number than the number of positive
    numbers, the analog voltage from -V to +V should
    be divided in seven intervals, each of size S=2V
    /7, and one digital value is to be assigned to
    each interval. The extra digit output 100 can be
    used to represent the interval -V to -9V/7')
```

Chapter 11

Semiconductor Memories

Scilab code Exa 11.1 Binary address of each location of size 16 words

```
//example 11.1//
clc
//clears the screen//
clear
//clears already existing variables//
for (i=0:15)
disp('word number to binary address is as follows:')
disp(i)
//displays the word number//
t=dec2bin(i);
//converts it into memory address//
disp(t)
//displays binary address//
i=i+1
end
```

Scilab code Exa 11.2.a maximum rate at which data can be stored

```
//example 11.2(a)//
//maximum rate at which data can be stored//
clc
//clears the screen//
clear
//clears already existing variables//
disp('the maximum rate at which data can be stored
   is:')
t=200*(10^-9);
//write cycle time//
r=1/t;
//maximum rate//
disp(r)
disp('words/sec')
```

Scilab code Exa 11.2.b maximum rate at which data can be read

```
//example 11.2(b)//
//maximum rate at which data can be read//
clc
//clears the screen//
clear
//clears already existing variables//
disp('the maximum rate at which data can be read is:
   ')
t=200*(10^-9);
```

```
//read cycle time//
r=1/t;
//maximum rate//
disp(r)
disp('words/sec')
```

Scilab code Exa 11.6.a data output

```
//example 11.6(a)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('since Y(5)=0, the memory location 5 is
   selected for read out, i.e.')
disp('D1D0=01')
disp('the memory contents do not change')
//given A1A0=00, W''=1, Y=11011111//
```

Scilab code Exa 11.6.b data output

```
//example 11.6(b)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('since Y5=Y4=0, the memory locations 4 and 5
   are selected for readout. The output is obtained
   by ORing the contents of these locations, i.e.')
```

```
disp('D1D0=11')
disp('The memory contents do not change')
//given A1A0=00, W''=1, Y=11001111//
```

Scilab code Exa 11.6.c data output

```
//example 11.6(c)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('since Y(3), write operation is performed at
   memory location 3. The input data is stored in
   this location and also appears at the output')
disp('D1D0=00')
disp('contents of memory location 3=00')
//given A1A0=00, W''=0, Y=11110111//
```

Scilab code Exa 11.6.d data output

```
//example 11.6(d)//
clc
//clears the screen//
clear
//clears already existing variables//
disp('In this case, the memory locations 3 and 5 are
     selected for writing since Y3=Y5=0. The contents
     of these locations will become 10 and')
disp('D1D0=10')
```

```
//given A1A0=00, W''=0, Y=11010111//
//I1I0=10//
```

Scilab code Exa 11.7.a output and change in memory location

```
//example 11.7//
clc
//clears the screen//
clear
//clears already existing variables//
//given A1A0=11, I1I0=01//
disp('the association operation is performed with
   keyword 01. The memory locations 1, 5 and 7 match
    the keyword giving out logic 0 at the
   corresponding Y outputs. Therefore,')
disp('Y=01011101')
```

Chapter 12

Programmable Logic Design

Scilab code Exa 12.1 find the product term

```
//example 12.1//
//find the product term//
clc
//clears the screen//
clear
//clears already existing variables//
disp('For an open link, the input to AND gate is
   logic 1, whereas for a closed link the
   corresponding input to the AND gate is same as
   the voltage applied at that input, therefore,')
disp('P(o)=I(o)I(2)''I(3)''I(6)')
```

Chapter 14

Computer Aided Design of Digital Systems

Scilab code Exa 14.2 entity construction for EXOR circuit

```
//example 14.2//
//entity construction for EXOR circuit//
clc
//clears the screen//
clear
//clears already existing variables//
disp('Let the name of entity be Circuit_Fig. It has
   two input ports A and B and one output port Y.
   The entity declaration for this circuit will be')
disp(' ENTITY Circuit_Fig IS')
disp('PORT(A,B : IN BIT; OUT BIT);')
disp('END Circuit_Fig;')
disp('From this entity declaration, we observe that
   although this circuit consists of AND, OR and NOT
    gates, the circuit itself is an entity and the
   entity declaration gives no information about the
    structure or behaviour of the circuit')
```