```
% % Mfile name
%   mtl_aae_sim_float2decusf.m

% Version:
%   Matlab R2007a

% Revised:
%   August 28, 2008

% Authors:
%   Luke Snyder, Dr. Autar Kaw
%   University of South Florida
%   kaw@eng.usf.edu
%   Website: http://numericalmethods.eng.usf.edu

% Purpose
%   To illustrate the concept of the conversion of a decimal number to
%   binary floating point representation.

% Keywords
%   Decimal to binary conversion
%   Fixed point register

% Clearing all data, variable names, and files from any other source and
% clearing the command window after each successive run of the program.
clc
clear all

% Inputs:
%    This is the only place in the program where the user makes the changes
%    based on their wishes.
    % The generalized formula for floating point format is given by:
    % y = sigma x m x 10^e.

    % Enter number to be converted to a floating point binary number:
    dec_num = 5.8946;

    % Enter the total number of bits to be used: (1st bit will be used for
    % the sign of the number, 2nd bit will be used for the sign of the
    % exponent.)
    tot_bits = 9;

    % Enter number of bits for mantissa (m).
    mant_bits = 4;


% ************************************************************************

disp(sprintf('\nConcepts of Conversion of Base 10 Number to Base 2 '))
disp(sprintf('Floating Point Binary Representation'))
disp(sprintf('\nUniversity of South Florida'))
disp(sprintf('United States of America'))
disp(sprintf('kaw@eng.usf.edu'))
disp(sprintf('Website: http://numericalmethods.eng.usf.edu'))
```

```
disp(sprintf('\nNOTE: This worksheet illustrates the use of Matlab to convert'))
disp('a base-10 number to a floating point binary representation.')

disp(sprintf('\n*************************Introduction*************************'))
disp(sprintf('\nThe following worksheet illustrates how to convert a base-10 real '))
disp(sprintf('number to a base-2 floating point binary representation using '))
disp(sprintf('loops and various conditional statements. The user inputs a '))
disp(sprintf('base-10 number in any format, the total number of bits, and the '))
disp(sprintf('number of bits for the mantissa in the Input section of the '))
disp(sprintf('program. The program will then convert the base-10 number into '))
disp(sprintf('a floating point binary representation by first converting the given '))
disp(sprintf('number into a number of the format 1.xxx * 2^m where m '))
disp(sprintf('represents the exponent of the number.'))


disp(sprintf('\n\n*************************Input Data*****************************'))
disp(sprintf('\nBase-10 number to convert to floating point binary representation, dec_num↙
= %g',dec_num));
disp(sprintf('\nTotal number of bits to be used, tot_bits = %g',tot_bits));
disp(sprintf('\nNumber of bits used for the sign of the number = 1'));
disp(sprintf('\nNumber of bits used for the sign of the exponent = 1'));
disp(sprintf('\nNumber of bits to be used in mantissa, mant_bits = %g',mant_bits));

% The number of bits to be used for the exponent will be the total number
% of bits minus the number of bits used for  the mantissa, plus two more
% bits which are used for the sign of the decimal number and the sign of
% the exponent.
exp_bits = tot_bits - (mant_bits + 2);
disp(sprintf('\nNumber of bits used for the exponent, exp_bits = %g',exp_bits));

disp(sprintf('\n\n*************************Procedure******************************'))



% To find the sign of the decimal number (sigma), and the sign of the exponent, we
% test to see if the number is negative and if it is less than 1.
if dec_num > 0
    sigma = '0';
    numsign = '+';
else
    sigma = '1';
    numsign = '-';
end

if abs(dec_num) < 1 && abs(dec_num) > 0
    exp_sign = '1';
    esign = '-';
else
    exp_sign = '0';
    esign = '+';
end

% Calculating the maximum possible value given the number of bits as
% specified by the user.
```

```
% Calculating the maximum value for the exponent.
expsum = 0;

for i=0:1:exp_bits-1
    expsum = expsum + 1*2^i;
end

% Calculating the maximum value for the mantissa
mant_sum = 1;

for i=1:1:mant_bits
    mant_sum = mant_sum + 1*2^(-i);
end

% The maximum value will be the mantissa sum multiplied by 2^(expsum).
maxval = mant_sum * 2^(expsum);
disp(sprintf('\nThe magnitude of the maximum number that can be represented in'));
disp(sprintf('floating point representation given the number of bits is, maxval = %g',↙
maxval));

% The minimum value that can be represented will be the mantissa sum
% multiplied by 2^(-expsum)
minval = mant_sum * 2^(-expsum);
disp(sprintf('\nThe magnitude of the minimum number that can be represented in'));
disp(sprintf('floating point representation given the number of bits is, minval = %g',↙
minval));



% If the maximum value that can be produced given the user specified number
% of bits is smaller than the number to be represented, then more bits are
% needed.  Similarly, if the minimum value that can be produced given the
% user specified number of bits is larger than the number to be
% represented, a change in bits is needed.
dec_num = abs(dec_num);

if  maxval < dec_num
    disp(sprintf('\nSince |%g| > %g, The number of bits specified is not sufficient',↙
dec_num,maxval));
    disp(sprintf('to represent this number in floating point representation.'));
    disp(sprintf('Either specify fewer mantissa bits, or more total bits for'))
    disp(sprintf('the worksheet.'))
elseif minval > dec_num
    disp(sprintf('Since %g > |%g|, the number of bits specified is',minval,dec_num));
    disp(sprintf('not sufficient to represent this number in floating point'))
    disp(sprintf('binary format. Either specify fewer mantissa bits, or more'));
    disp(sprintf('total bits for the worksheet.'))
else
    disp(sprintf('\nSince %g < |%g| < %g the base-10 number can be represented',minval,↙
dec_num,maxval));
    disp(sprintf('in floating point binary format'));

        % Conversion of the base-10 decimal number to floating point format given
```

```
        % the number of bits as specified by the user.
disp(sprintf('----------------------------------------------------------------------'));

        % We will convert the given variable "dec_num" into some
        % number 1.____ * 2^m, where m is the value for the exponent.
        % First, we will find this value "m" for the exponent using the
        % properties of logarithms.
        exp_val = floor(log(dec_num)/log(2));


        % Since the exponent value is now known, it is now possible to find
        % the fractional portion of the decimal number (1.____).
        mant_val = dec_num/2^exp_val;

        % To isolate the fractional portion, we simply subtract one from
        % the mant_val variable.
        mant_frac = mant_val - 1;

        % Using loops to approximate the mantissa up to the specified
        % number of mantissa bits.
        Mantstr = '';

        for i=1:1:mant_bits
            new_mant_frac = mant_frac * 2;

            % The value for the mantissa will be the floor of this number.
            Mant(i) = floor(new_mant_frac);

            % Using the newly approximated value to "reset" the mant_frac
            % variable for the next iteration.
            mant_frac = new_mant_frac - Mant(i);

            % Creating a character array for the mantissa.
            Mantissa(i) = num2str(Mant(i));
            Mantstr = [Mantstr,Mantissa(i),'|'];

       end

       % Approximating the exponent value based on the calculated value of
       % "m".
       exp_val = abs(exp_val);

       for i=1:1:exp_bits
           new_exp_val = floor(exp_val/2);

           % Approximating the floating point values for the exponent value
           % "m".
           Bin_exp(i) = ceil(exp_val/2) - new_exp_val;

           % Reinitializing the "old" value to become the "new" value.
           exp_val = new_exp_val;
       end

       % Using this method, the floating point values approximated for the
```

```
        % exponent are backwards from the actual representation.  This
        % section of the program reverses this and creates a character array
        % with the existing values as well.
        t = length(Bin_exp);
        Expstr = '';

        for i=1:1:t
             Binary_exp(i) = num2str(Bin_exp((t+1)-i));

             % This "Binary_exp" array is now a character array.  Now we use
             % the properties of loops to create the floating point
             % representation that will be displayed later.
             Expstr = [Expstr,Binary_exp(i),'|'];
        end

% Since the absolute value of the number was taken earlier, it may or may
% not be necessary to make the sign of the number negative again for
% displaying purposes.
if sigma == '1'
    dec_num = -1*dec_num;
end

% Concatenating all previously calculated binary components of the
% floating point binary number.
disp(sprintf('\nThe floating point representation of the base-10 number,'));
disp(sprintf('dec_num = %g, is given in floating point binary format as:',dec_num));
fprintf('\n');

% Displaying each component of the representation individually.
str1 = ['Sign of number entry = ',sigma];
disp(str1);
str = ['Sign of exponent entry = ',exp_sign];
disp(str);
str = ['Mantissa entry = ',Mantissa];
disp(str);
str = ['Exponent entry = ',Binary_exp];
disp(str);
str = ['Binary equivalent = ',numsign,'(1.',Mantissa,') * 2^(',esign,Binary_exp,')',];
disp(str)

fprintf('\n');
str1 = ['Floating Point: |',sigma,'|  |',exp_sign,'|  |',Mantstr,'  |',Expstr];
disp(str1)

end

  disp(sprintf('\n\n**************************Conclusion*****************************'))
disp(sprintf('This worksheet illustrates the use of Matlab to convert a '))
disp(sprintf('base-10 number to a floating point binary representation. Recall '))
disp(sprintf('that floating point representation is used more often than fixed '))
disp(sprintf('point representation due to two primary advantages: floating point '))
disp(sprintf('representation supports a much larger range of values while '))
disp(sprintf('maintaining a relative error of similar magnitude for all numbers.'))
```

```
disp(sprintf('\n\n**************************Refrences*****************************'))
disp('See: <a href = "http://numericalmethods.eng.usf.↙
edu/mws/gen/01aae/mws_gen_aae_txt_floatingpoint.pdf">Floating Point Representation</a>')

disp(sprintf('\n\nLegal Notice: The copyright for this application is owned'))
disp(sprintf('by the author(s). Neither MathWorks nor the author(s)'))
disp(sprintf('are responsible for any errors contained within and are '))
disp(sprintf('not liable for any damages resulting from the use of this'))
disp(sprintf('material. This application is intended for non-commercial,'))
disp(sprintf('non-profit use only. Contact the author for permission if'))
disp(sprintf('you wish to use this application in for-profit activities.'))
```