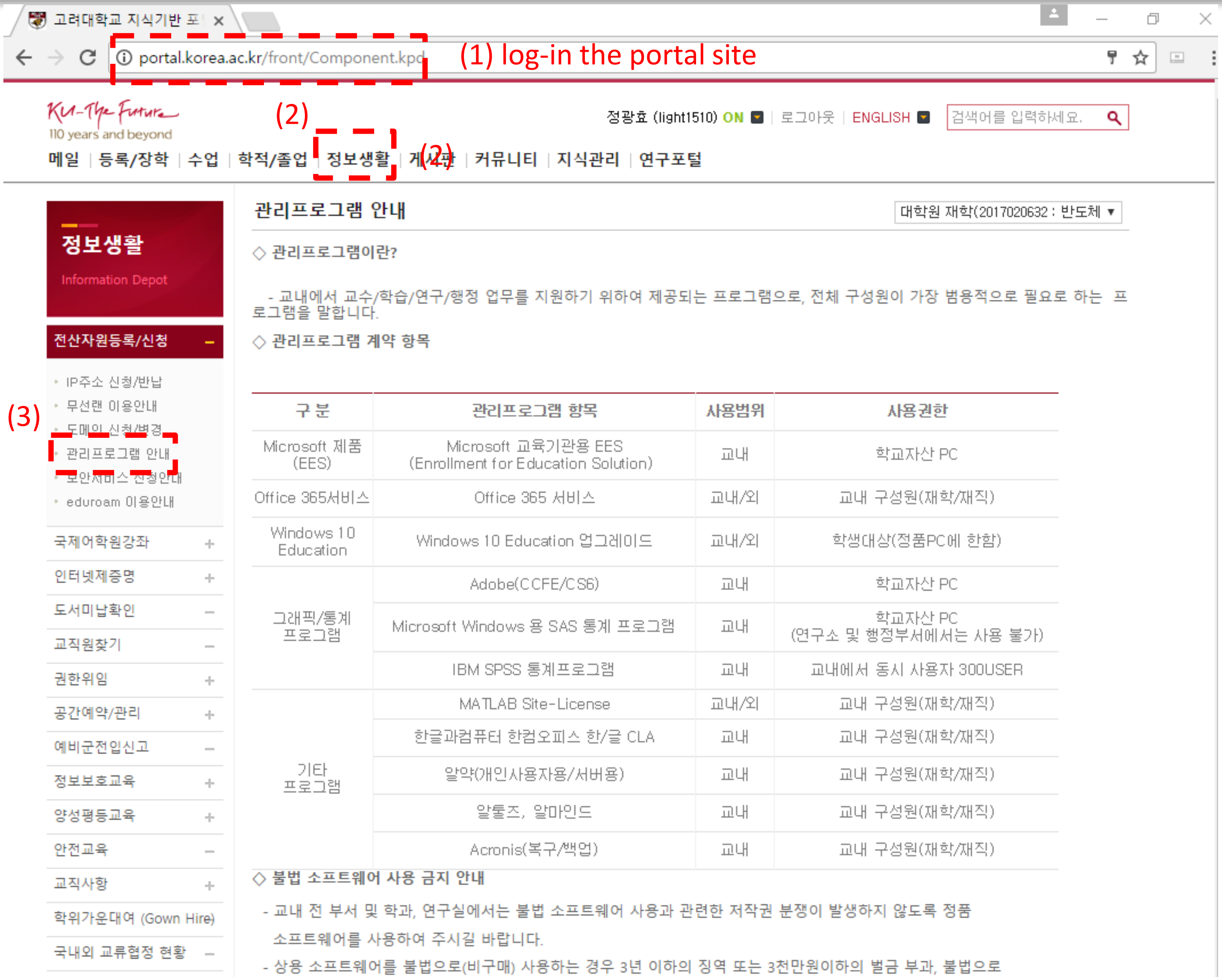
Project #2

2D DCT Design in JPEG Image Compression

- Extra material -

How to install MATLAB



(1) log-in the portal site

(2)

(2)

(3)

관리프로그램 안내

◇ 관리프로그램이란?

- 교내에서 교수/학습/연구/행정 업무를 지원하기 위하여 제공되는 프로그램으로, 전체 구성원이 가장 범용적으로 필요로 하는 프로그램을 말합니다.

◇ 관리프로그램 계약 항목

구 분	관리프로그램 항목	사용범위	사용권한
Microsoft 제품 (EES)	Microsoft 교육기관용 EES (Enrollment for Education Solution)	교내	학교자산 PC
Office 365서비스	Office 365 서비스	교내/외	교내 구성원(재학/재직)
Windows 10 Education	Windows 10 Education 업그레이드	교내/외	학생대상(정품PC에 한함)
그래픽/통계 프로그램	Adobe(CCFE/CS6)	교내	학교자산 PC
	Microsoft Windows 용 SAS 통계 프로그램	교내	학교자산 PC (연구소 및 행정부서에서는 사용 불가)
	IBM SPSS 통계프로그램	교내	교내에서 동시 사용자 300USER
기타 프로그램	MATLAB Site-License	교내/외	교내 구성원(재학/재직)
	한글과컴퓨터 한컴오피스 한/글 CLA	교내	교내 구성원(재학/재직)
	알약(개인사용자용/서버용)	교내	교내 구성원(재학/재직)
	알툴즈, 알마인드	교내	교내 구성원(재학/재직)
	Acronis(복구/백업)	교내	교내 구성원(재학/재직)

◇ 불법 소프트웨어 사용 금지 안내

- 교내 전 부서 및 학과, 연구실에서는 불법 소프트웨어 사용과 관련한 저작권 분쟁이 발생하지 않도록 정품 소프트웨어를 사용하여 주시길 바랍니다.
- 상용 소프트웨어를 불법으로(비구매) 사용하는 경우 3년 이하의 징역 또는 3천만원이하의 벌금 부과, 불법으로

(4) Apply for a license and install Matlab

How to use given files

1) Tiff Image files

image_in_1.tif

image_in_2.tif

image_in_3.tif

image_in_4.tif

image_in_5.tif

image_in_6.tif

image_in_7.tif

image_in_8.tif

2) Two version of JPEG matlab files

JPEG_CODEC_FULL.m

JPEG_CODEC_Verilog_IO.m

JPEC_CODEC_FULL.m

(1) Full process of the JPEG (include DCT)

(2) Quantized coefficient verifying code

(3) Convert tiff image(.tif) to text vector(.txt) code

(4) Calculate PSNR

JPEC_CODEC_Verilog_IO.m

(1) Read text file(Verilog output)

(2) Process of after 2D-DCT

(3) Calculate PSNR

3) Five function matlab files

func_Bin2Dec_mag

func_DCT_Coefficient_quant

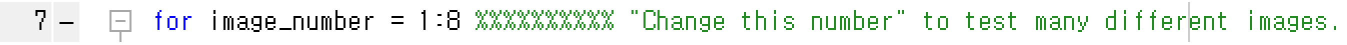
func_DCTquant

func_DCTquant_trunc

func_Dec2Bin_mag

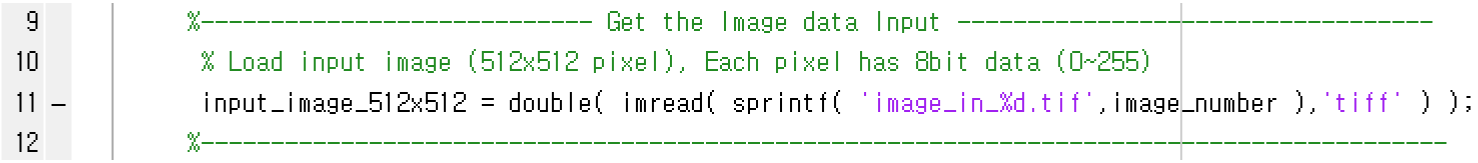
Just locate in the same folder

(1) JPEC_CODEC_FULL.m



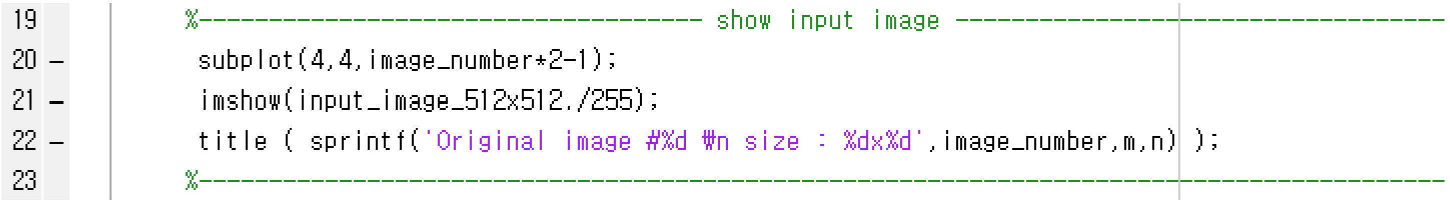
```
for image_number = 1:8 %%%%%%%%%% "Change this number" to test many different images.
```

“1:8” means that you want to test the **all of the 8 image**.
If you want to test one of the images, edit the numbers like **‘1:2’, ‘1:1’ or ‘2:2’**
(Image 1&2) (only Image 1) (only Image 2)



```
    %---------------------------- Get the Image data Input -----------------------------------
     % Load input image (512x512 pixel), Each pixel has 8bit data (0~255)
     input_image_512x512 = double( imread( sprintf( 'image_in_%d.tif',image_number ),'tiff' ) );
    %------------------------------------------------------------------------------------------
```

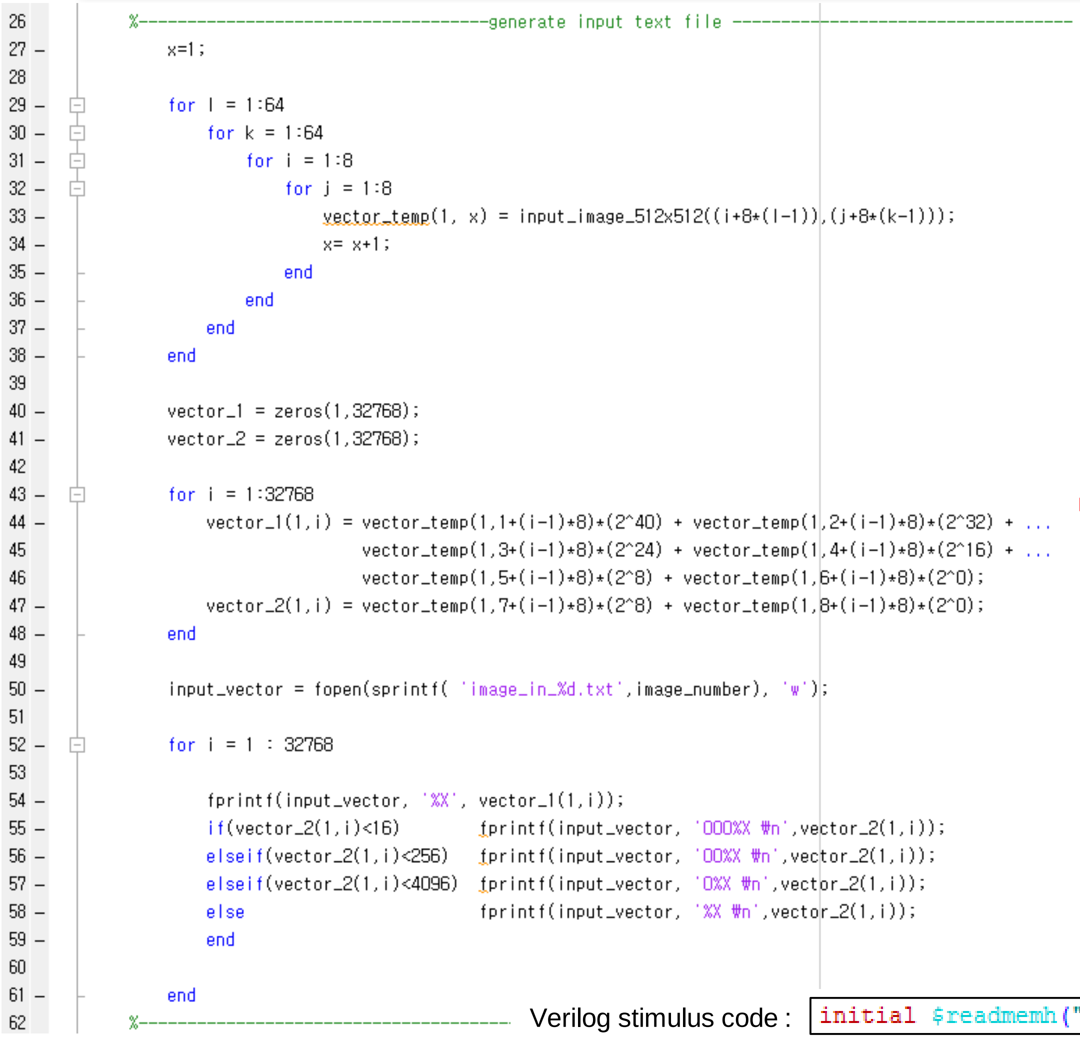
“ image_in_%d.tif ” file is loaded, ‘%d’ is replaced with ‘image_number’



```
    %------------------------------------ show input image -----------------------------------
     subplot(4,4,image_number*2-1);
     imshow(input_image_512x512./255);
     title ( sprintf('Original image #%d ₩n size : %dx%d',image_number,m,n) );
    %------------------------------------------------------------------------------------------
```

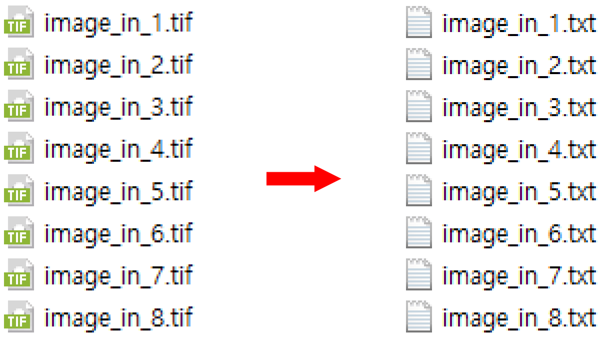
You can show original image on the matlab program

(1) JPEC_CODEC_FULL.m



```
%------------------------------------generate input text file ----------------------------------
    x=1;

    for l = 1:64
        for k = 1:64
            for i = 1:8
                for j = 1:8
                    vector_temp(1, x) = input_image_512x512((i+8*(l-1)),(j+8*(k-1)));
                    x= x+1;
                end
            end
        end
    end

    vector_1 = zeros(1,32768);
    vector_2 = zeros(1,32768);

    for i = 1:32768
        vector_1(1,i) = vector_temp(1,1+(i-1)*8)*(2^40) + vector_temp(1,2+(i-1)*8)*(2^32) + ...
                        vector_temp(1,3+(i-1)*8)*(2^24) + vector_temp(1,4+(i-1)*8)*(2^16) + ...
                        vector_temp(1,5+(i-1)*8)*(2^8) + vector_temp(1,6+(i-1)*8)*(2^0);
        vector_2(1,i) = vector_temp(1,7+(i-1)*8)*(2^8) + vector_temp(1,8+(i-1)*8)*(2^0);
    end

    input_vector = fopen(sprintf( 'image_in_%d.txt',image_number), 'w');

    for i = 1 : 32768

        fprintf(input_vector, '%X', vector_1(1,i));
        if(vector_2(1,i)<16)        fprintf(input_vector, '000%X ₩n',vector_2(1,i));
        elseif(vector_2(1,i)<256)   fprintf(input_vector, '00%X ₩n',vector_2(1,i));
        elseif(vector_2(1,i)<4096)  fprintf(input_vector, '0%X ₩n',vector_2(1,i));
        else                        fprintf(input_vector, '%X ₩n',vector_2(1,i));
        end

    end
%------------------------------------
```

For verilog



image_in_1.tif → image_in_1.txt
image_in_2.tif → image_in_2.txt
image_in_3.tif → image_in_3.txt
image_in_4.tif → image_in_4.txt
image_in_5.tif → image_in_5.txt
image_in_6.tif → image_in_6.txt
image_in_7.tif → image_in_7.txt
image_in_8.tif → image_in_8.txt

Input image data are rearranged, and it generate the **input image text file** for Verilog coded hardware.

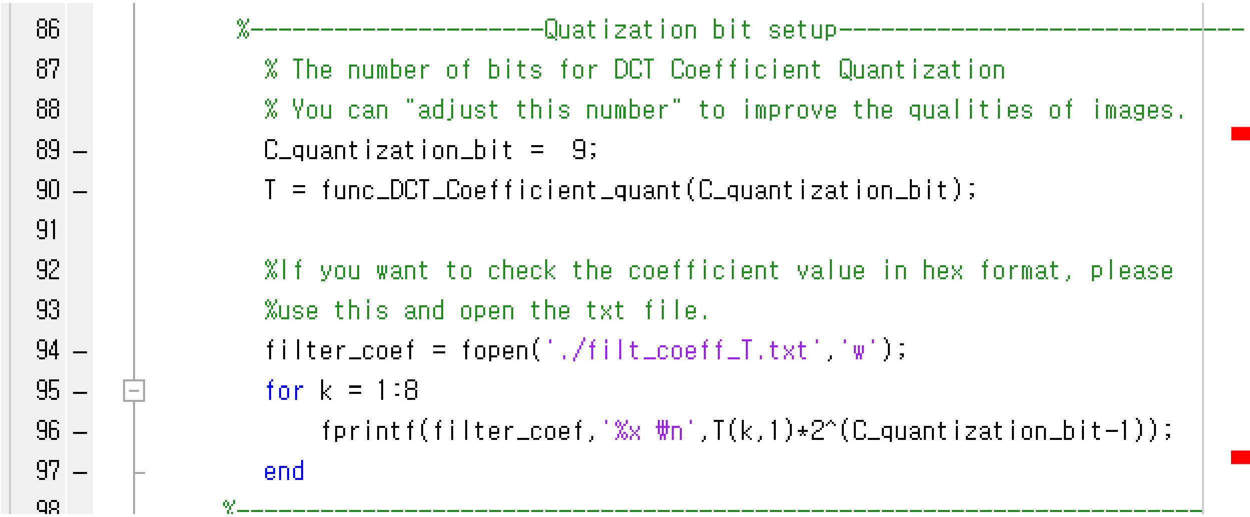
Input text file is read at the stimulus and written to the Input memory.

Text file should be locate in the same folder with Verilog files.
(Otherwise, please edit the context of stimulus.)

Verilog stimulus code :

```
initial $readmemh("image_in_1.txt", TEST.MEM_IN.SRAM_syn.SRAM32768x64.Mem);
```

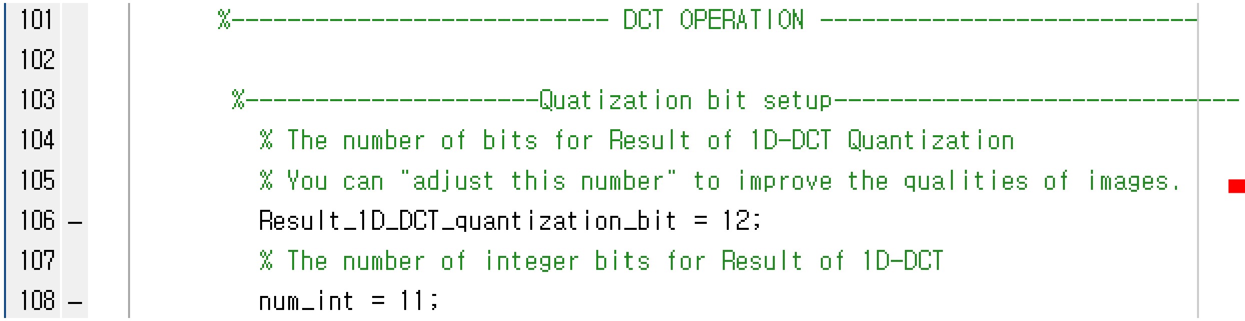
(1) JPEC_CODEC_FULL.m



```
%---------------------Quatization bit setup-----------------------------
  % The number of bits for DCT Coefficient Quantization
  % You can "adjust this number" to improve the qualities of images.
  C_quantization_bit =  9;
  T = func_DCT_Coefficient_quant(C_quantization_bit);

  %If you want to check the coefficient value in hex format, please
  %use this and open the txt file.
  filter_coef = fopen('./filt_coeff_T.txt','w');
  for k = 1:8
      fprintf(filter_coef,'%x ₩n',T(k,1)*2^(C_quantization_bit-1));
  end
%----------------------------------------------------------------------
```

Coefficient quantization bit
“func_DCT_Coefficient_quant.m” 참고
It includes sign bit
Ex) C_quantization_bit = 9
출력되는 C0 ~ C7 (9.8)

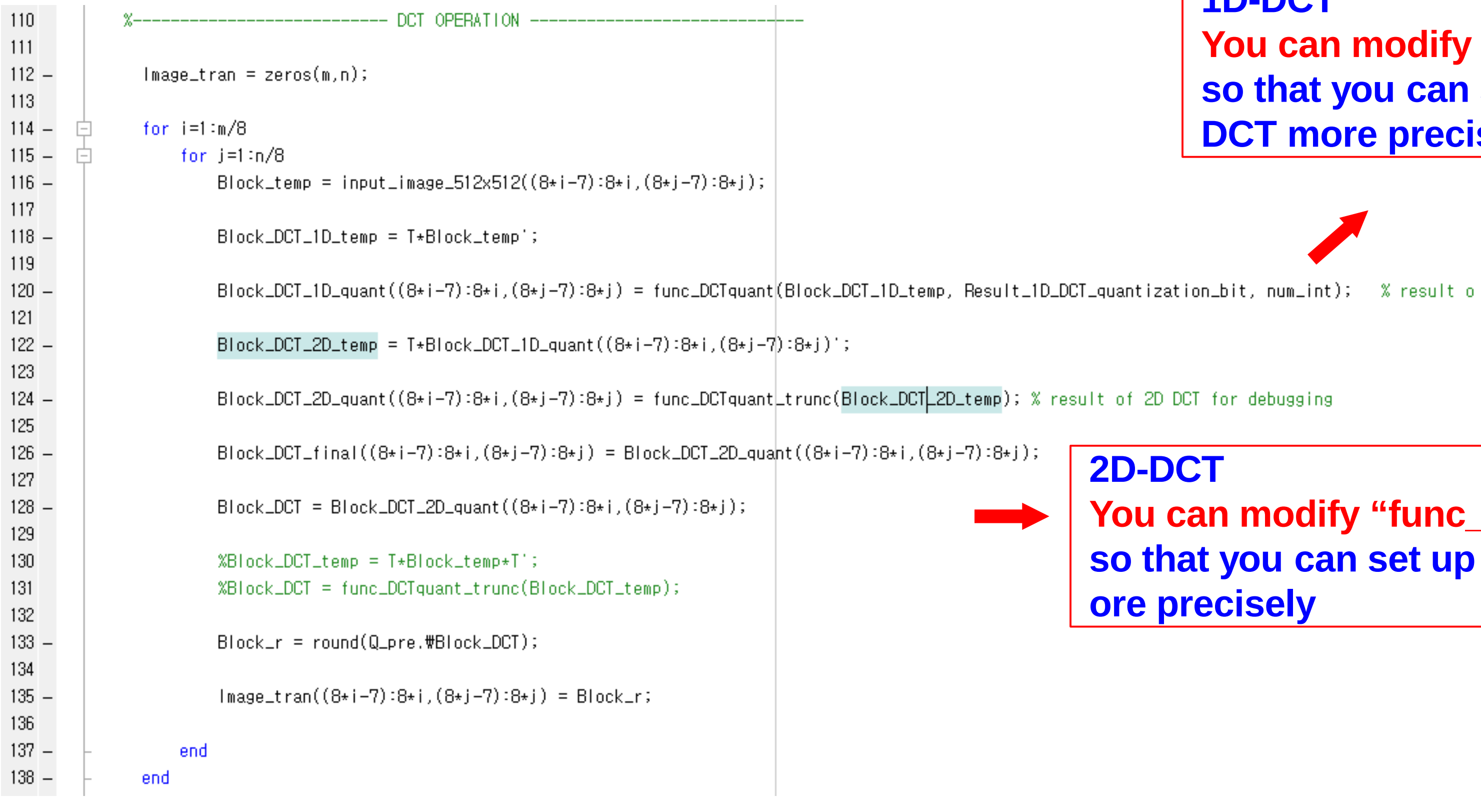
In “filt_coeff_T.txt”,
you can get results of quantization in hexa radix



```
%--------------------------- DCT OPERATION ---------------------------

 %--------------------Quatization bit setup-----------------------------
  % The number of bits for Result of 1D-DCT Quantization
  % You can "adjust this number" to improve the qualities of images.
  Result_1D_DCT_quantization_bit = 12;
  % The number of integer bits for Result of 1D-DCT
  num_int = 11;
```

Set 1D-DCT output bit width
Also set the number of integer bits for result of 1D-DCT

(1) JPEC_CODEC_FULL.m

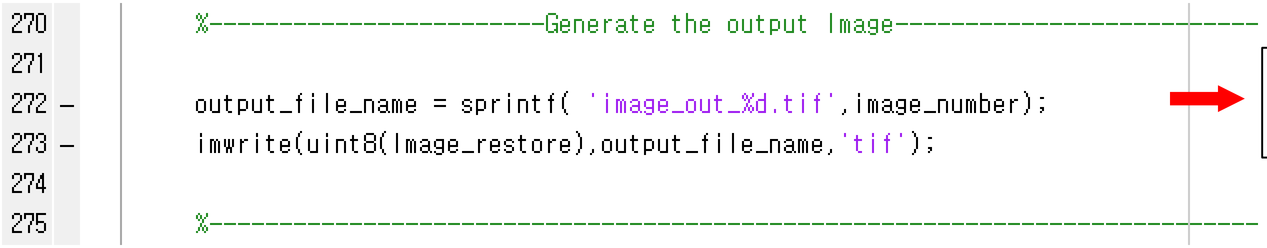


```matlab
  %---------------------------- DCT OPERATION -----------------------------

    Image_tran = zeros(m,n);

    for i=1:m/8
        for j=1:n/8
            Block_temp = input_image_512x512((8*i-7):8*i,(8*j-7):8*j);

            Block_DCT_1D_temp = T*Block_temp';

            Block_DCT_1D_quant((8*i-7):8*i,(8*j-7):8*j) = func_DCTquant(Block_DCT_1D_temp, Result_1D_DCT_quantization_bit, num_int);   % result o

            Block_DCT_2D_temp = T*Block_DCT_1D_quant((8*i-7):8*i,(8*j-7):8*j)';

            Block_DCT_2D_quant((8*i-7):8*i,(8*j-7):8*j) = func_DCTquant_trunc(Block_DCT_2D_temp); % result of 2D DCT for debugging

            Block_DCT_final((8*i-7):8*i,(8*j-7):8*j) = Block_DCT_2D_quant((8*i-7):8*i,(8*j-7):8*j);

            Block_DCT = Block_DCT_2D_quant((8*i-7):8*i,(8*j-7):8*j);

            %Block_DCT_temp = T*Block_temp*T';
            %Block_DCT = func_DCTquant_trunc(Block_DCT_temp);

            Block_r = round(Q_pre.₩Block_DCT);

            Image_tran((8*i-7):8*i,(8*j-7):8*j) = Block_r;

        end
    end
```

1D-DCT
You can modify "func_DCTquant.m" so that you can set up the result of 1D-DCT more precisely

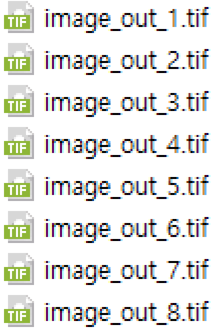
2D-DCT
You can modify "func_DCTquant_trunc.m" so that you can set up the result of 2D-DCT more precisely

(1) JPEC_CODEC_FULL.m

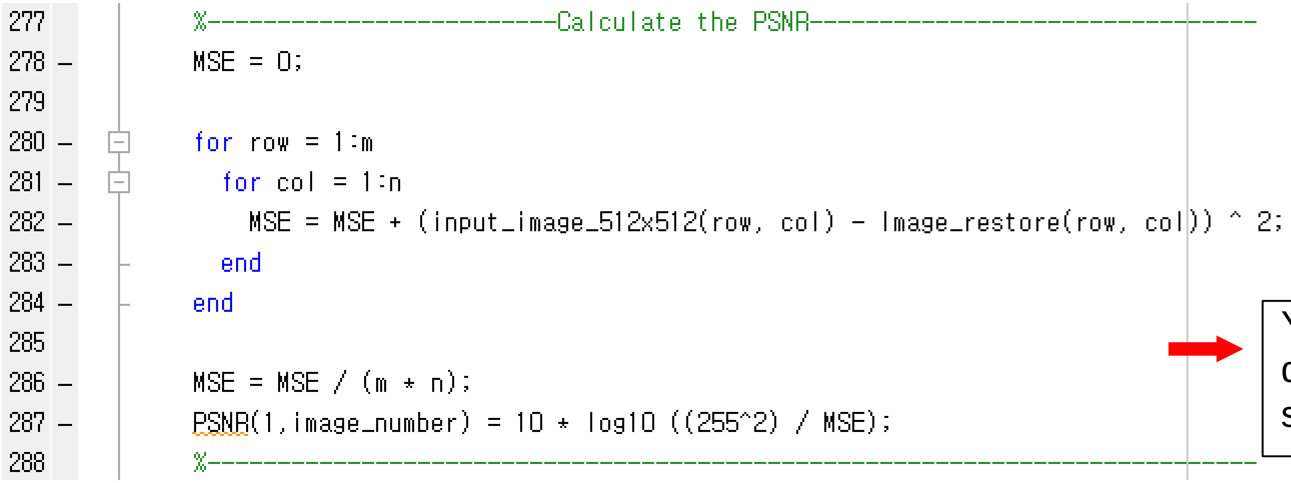


```
%------------------------Generate the output Image--------------------------

output_file_name = sprintf( 'image_out_%d.tif',image_number);
imwrite(uint8(Image_restore),output_file_name,'tif');

%---------------------------------------------------------------------------
```

Genarate output image
You can verify restored images .

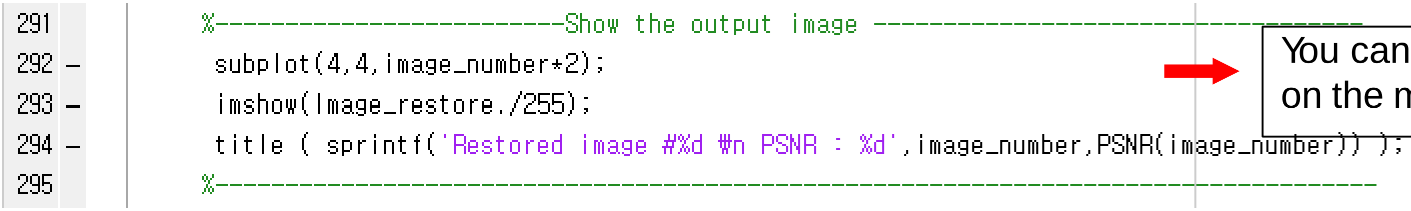


image_out_1.tif
image_out_2.tif
image_out_3.tif
image_out_4.tif
image_out_5.tif
image_out_6.tif
image_out_7.tif
image_out_8.tif



```
%-------------------------Calculate the PSNR--------------------------------
MSE = 0;

for row = 1:m
  for col = 1:n
    MSE = MSE + (input_image_512x512(row, col) - Image_restore(row, col)) ^ 2;
  end
end

MSE = MSE / (m * n);
PSNR(1,image_number) = 10 * log10 ((255^2) / MSE);
%---------------------------------------------------------------------------
```

You can show PSNR which is decided by your quantization setups.

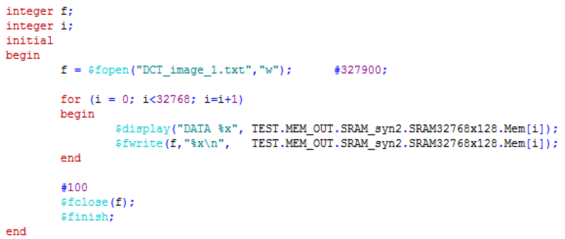


```
%------------------------Show the output image ------------------------------------
 subplot(4,4,image_number*2);
 imshow(Image_restore./255);
 title ( sprintf('Restored image #%d ₩n PSNR : %d',image_number,PSNR(image_number)) );
%----------------------------------------------------------------------------------
```

You can show restored image on the matlab program

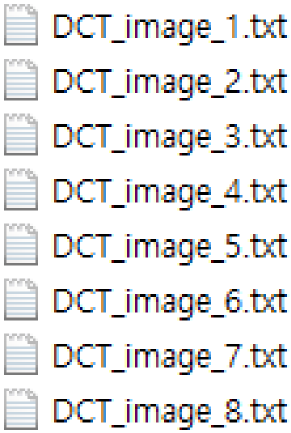
(2) JPEC_CODEC_Verilog_IO.m

Verilog output data
(2D-DCT transformed image)



```
integer f;
integer i;
initial
begin
        f = $fopen("DCT_image_1.txt","w");      #327900;

        for (i = 0; i<32768; i=i+1)
        begin
                $display("DATA %x", TEST.MEM_OUT.SRAM_syn2.SRAM32768x128.Mem[i]);
                $fwrite(f,"%x\n",   TEST.MEM_OUT.SRAM_syn2.SRAM32768x128.Mem[i]);
        end

        #100
        $fclose(f);
        $finish;
end
```

Text file

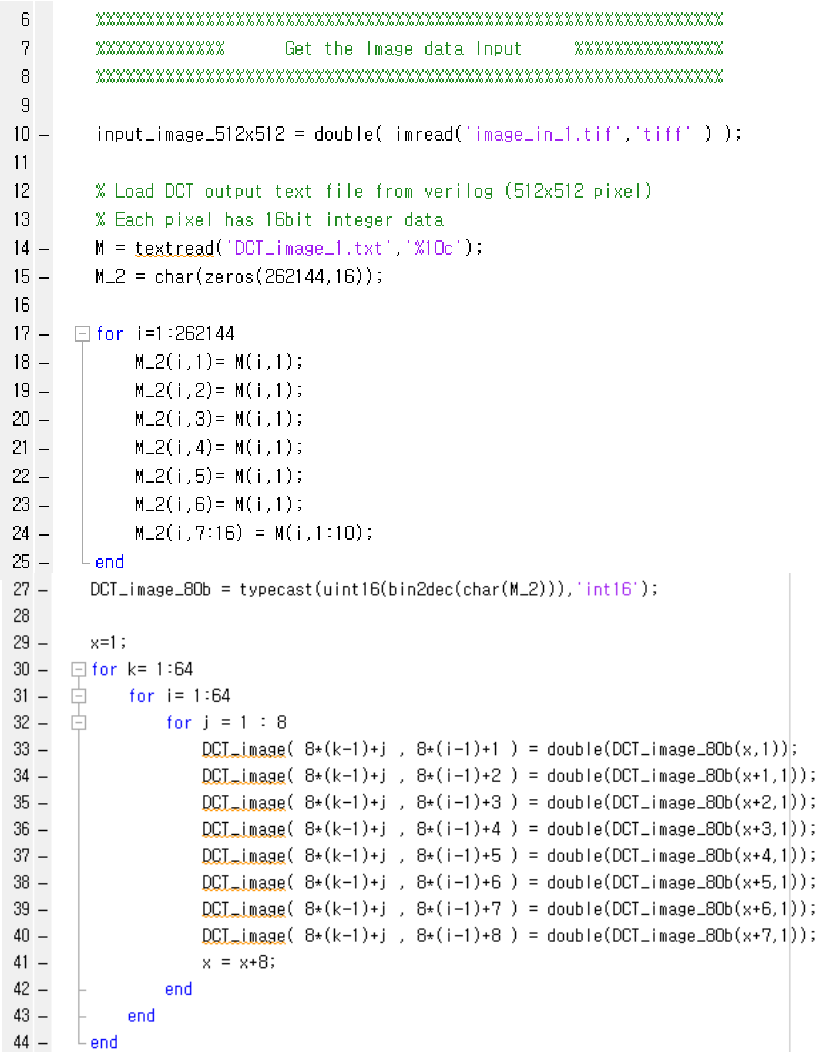


DCT_image_1.txt
DCT_image_2.txt
DCT_image_3.txt
DCT_image_4.txt
DCT_image_5.txt
DCT_image_6.txt
DCT_image_7.txt
DCT_image_8.txt

Matlab file

JPEG_CODEC_Verilog_IO.m

(2) JPEC_CODEC_Verilog_IO.m



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%      Get the Image data Input     %%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

input_image_512x512 = double( imread('image_in_1.tif','tiff' ) );

% Load DCT output text file from verilog (512x512 pixel)
% Each pixel has 16bit integer data
M = textread('DCT_image_1.txt','%10c');
M_2 = char(zeros(262144,16));

for i=1:262144
    M_2(i,1)= M(i,1);
    M_2(i,2)= M(i,1);
    M_2(i,3)= M(i,1);
    M_2(i,4)= M(i,1);
    M_2(i,5)= M(i,1);
    M_2(i,6)= M(i,1);
    M_2(i,7:16) = M(i,1:10);
end
DCT_image_80b = typecast(uint16(bin2dec(char(M_2))),'int16');

x=1;
for k= 1:64
    for i= 1:64
        for j = 1 : 8
            DCT_image( 8*(k-1)+j , 8*(i-1)+1 ) = double(DCT_image_80b(x,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+2 ) = double(DCT_image_80b(x+1,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+3 ) = double(DCT_image_80b(x+2,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+4 ) = double(DCT_image_80b(x+3,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+5 ) = double(DCT_image_80b(x+4,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+6 ) = double(DCT_image_80b(x+5,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+7 ) = double(DCT_image_80b(x+6,1));
            DCT_image( 8*(k-1)+j , 8*(i-1)+8 ) = double(DCT_image_80b(x+7,1));
            x = x+8;
        end
    end
end
```

Original image is loaded for PSNR calculation.

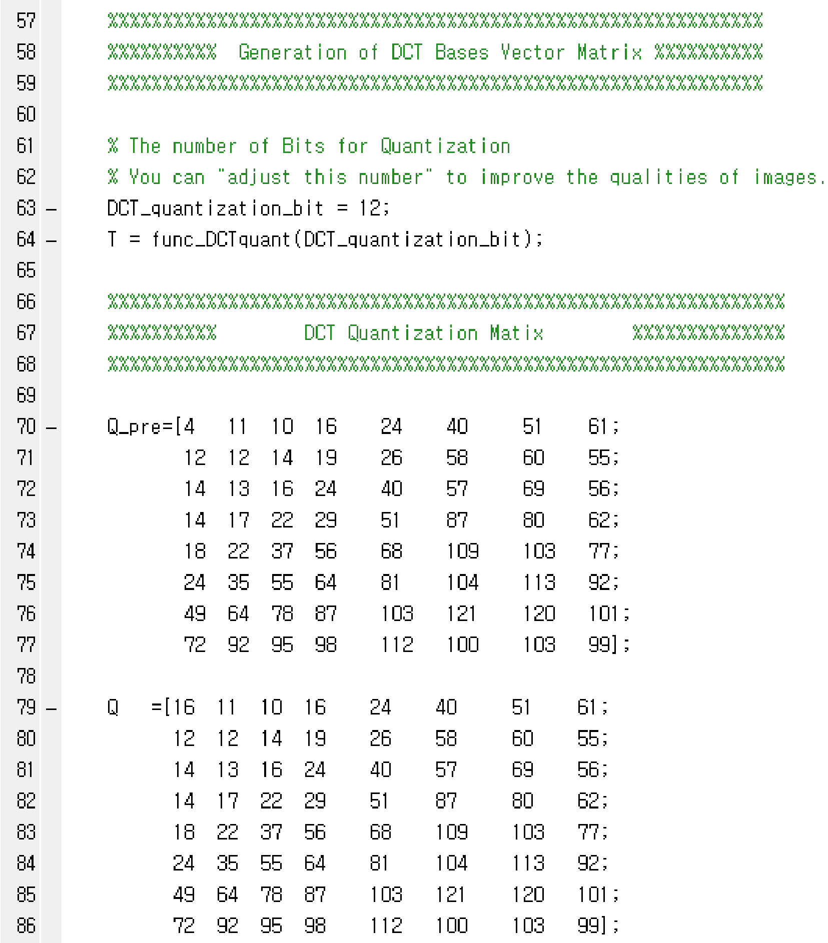
DCT output text file is loaded.
This is output file of Verilog coding which consist of 80bit binary sequences.

Sign extend from 10bit to 16bit

Convert binary to signed integer

Reshape the data matrix

(2) JPEC_CODEC_Verilog_IO.m



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%  Generation of DCT Bases Vector Matrix %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% The number of Bits for Quantization
% You can "adjust this number" to improve the qualities of images.
DCT_quantization_bit = 12;
T = func_DCTquant(DCT_quantization_bit);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%        DCT Quantization Matix        %%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Q_pre=[4   11  10  16    24    40     51    61;
       12  12  14  19    26    58     60    55;
       14  13  16  24    40    57     69    56;
       14  17  22  29    51    87     80    62;
       18  22  37  56    68    109    103   77;
       24  35  55  64    81    104    113   92;
       49  64  78  87    103   121    120   101;
       72  92  95  98    112   100    103   99];

Q    =[16  11  10  16    24    40     51    61;
       12  12  14  19    26    58     60    55;
       14  13  16  24    40    57     69    56;
       14  17  22  29    51    87     80    62;
       18  22  37  56    68    109    103   77;
       24  35  55  64    81    104    113   92;
       49  64  78  87    103   121    120   101;
       72  92  95  98    112   100    103   99];
```

Set the quantization bit corresponds to your hardware architecture.

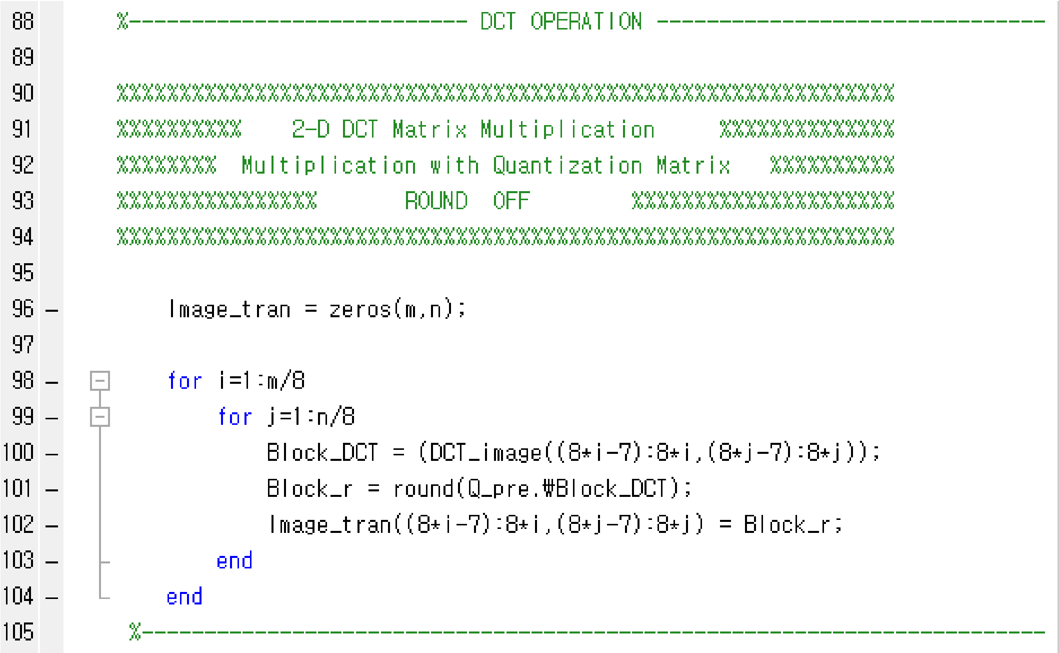
It is used for restoring the image(IDCT)

There are two kinds of quantization matrix.

A difference between two matrix is just first element.

We might shift first 2D-DCT output to left 2 bits. So, we should divide first element by ‘4’ instead of ‘16’

(2) JPEC_CODEC_Verilog_IO.m



```
%--------------------------- DCT OPERATION -------------------------------

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%    2-D DCT Matrix Multiplication     %%%%%%%%%%%%%%
%%%%%%%%  Multiplication with Quantization Matrix   %%%%%%%%%%
%%%%%%%%%%%%%%%%       ROUND   OFF        %%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Image_tran = zeros(m,n);

    for i=1:m/8
        for j=1:n/8
            Block_DCT = (DCT_image((8*i-7):8*i,(8*j-7):8*j));
            Block_r = round(Q_pre.₩Block_DCT);
            Image_tran((8*i-7):8*i,(8*j-7):8*j) = Block_r;
        end
    end
 %-----------------------------------------------------------------------
```

Read 8 by 8 block of output image. Divide each elements by ‘Q_pre’ and restore the 2D-DCT image by collecting the blocks