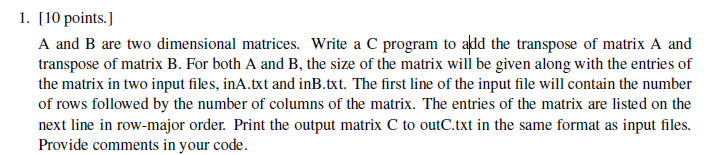
Khanh Nguyen

UIN 525000335

ECEN 449-503

Dr.Khatri

**Homework 2**



**Answer:**

**C code:**

#include <stdio.h>

#include <stdlib.h>

void add(float A[][10], float B[][10], float C[][10], int r, int c)

{// This function adds two matrixes

int i, j;// i-index for row, j-index for columns

for(i=0; i<r; i++)

{

for(j=0; j<c; j++)

{

C[i][j] = A[i][j] + B[i][j];

}

}

}

void transpose(float arr[][10], float temp[10][10], int r, int c)

{// This function transposes matrix

int i, j;// i-index for row, j-index for columns

for(i=0; i<c; i++)

{

for(j=0; j<r; j++)

{

temp[i][j] = arr[j][i];// rows will become columns and vice versa

}

}

}

void print(float arr[][10], int r, int c)

{//This function prints the array to the console, just for easy checking

int i, j;// i-index for row, j-index for columns

for(i=0; i<r; i++)

{

for(j=0; j<c; j++)

{

printf(" %.2f ", arr[i][j]);

}

printf("\n");

}

}

void load(char fileName[], float arr[][10], int \*r, int \*c)

{//This function loads data from txt file

int i, j; // i-index for row, j-index for columns

FILE \*fp;

fp = fopen(fileName,"r");

fscanf(fp, "%d %d", r, c); //read first line containing rows and columns

for(i=0; i<\*r; i++){

for(j=0; j<\*c; j++)

{ //load numbers into an array from second line

fscanf(fp, "%f", &(arr[i][j]));

}

}

fclose(fp);// cloase file

}

void write(float arr[][10], int r, int c)

{// This function writes result to file

int i, j;// i-index for row, j-index for columns

FILE \*fp;

fp = fopen("outC.txt", "w");

fprintf(fp, "%d %d\n", r, c); //write first line containing rows and columns

for(i=0; i<r; i++)

{

for(j=0; j<c; j++)

{ //write the result array to file second line

fprintf(fp, "%.2f ", arr[i][j]);

}

}

fclose(fp); // close file

}

int main()

{

int rA, cA, rB, cB, rC, cC;

float A[10][10], B[10][10], C[10][10];

load("inA.txt", A, &rA, &cA);

load("inB.txt", B, &rB, &cB);

rC=rA;// set dimensions for matrix C

cC=cA;

float transposed\_A[10][10];

float transposed\_B[10][10];

printf("\n Array A: \n");

print(A, rA, cA);

printf("\n Array B: \n");

print(B, rB, cB);

if( (rA != rB) || (cA != cB))// check if dimensions matched

{

printf("\n Dimensions not matched!\n");

return -1;

}

transpose(A ,transposed\_A , rA, cA);

transpose(B ,transposed\_B , rB, cB);

add(transposed\_A, transposed\_B, C, cC, rC);

printf("\n Array C: \n");

print(C, cC, rC);

write(C, cC, rC);

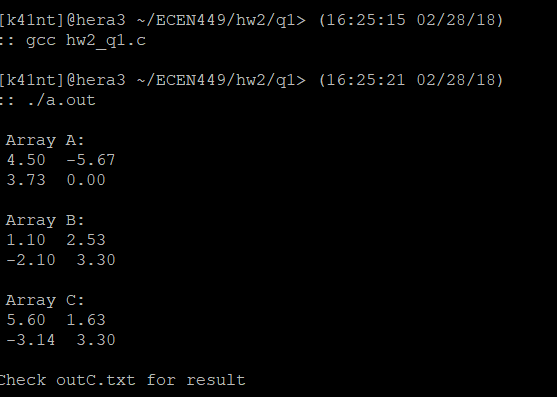
printf("\nCheck outC.txt for result\n");

return 0;

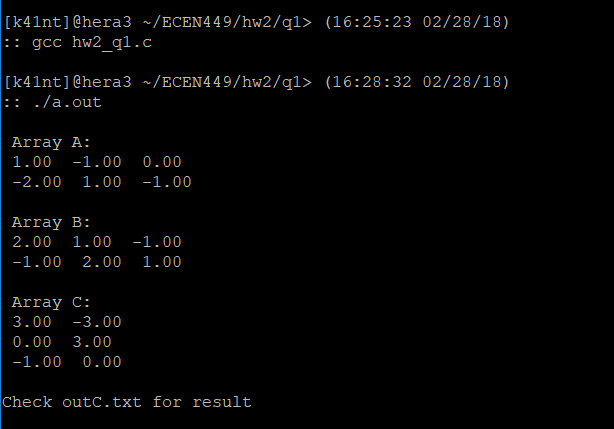
}

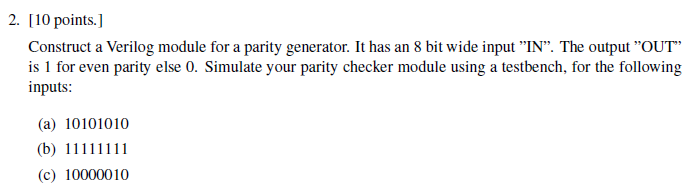
**Result: (It’s just for easy checking when it’s printed on the console, please check the outC.txt file for output)**

**Pair a:**



**Pair b:**





**Answer:**

**Verilog code:**

module hw2\_q2(Out, In);

input [7:0] In;

output reg Out;

integer i = 0;

integer cnt = 0;

always@(In)

begin

for(i = 0; i < 8; i =i+1)

begin

if(In[i] == 1)//count the number of 1

cnt = cnt + 1;

end

if((cnt % 2) == 1) //If the number of 1s is odd, parity is 1

Out = 1;

else

Out = 0;//parity is 0 if the 1's are even

end

endmodule

**Test bench:**

module hw2\_q2\_tb();

reg [7:0] In;

wire Out;

hw2\_q2 uut(.In(In),

.Out(Out)

);

initial

begin

In = 8'b10101010;//test 1

#25 In = 8'b11111111;//test 2

#25 In = 8'b10000010;//test 3

#25 $stop; // stop after third test

end

initial

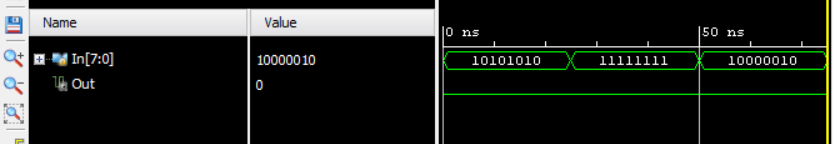
begin

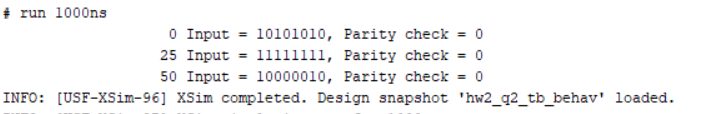
$monitor($time," Input = %b, Parity check = %d", In, Out);

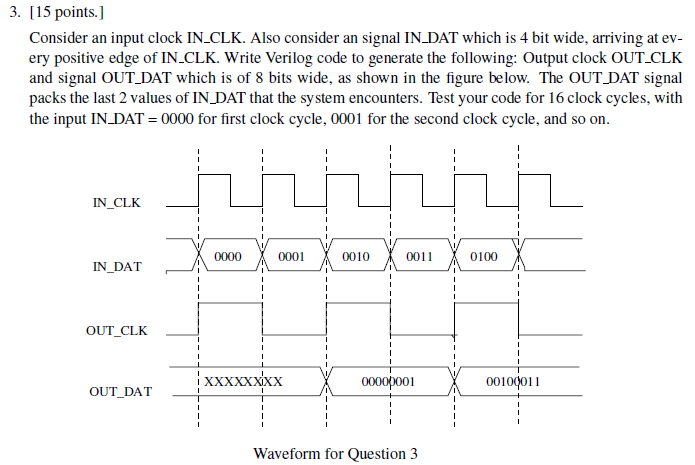
end

endmodule

**Results:**







**Answer:**

**Verilog code:**

`timescale 1ns / 1ps

`default\_nettype none

module hw2\_q3(CLK, IN, CLK\_OUT, OUT);

input wire CLK;

input wire [3:0] IN;

output reg [7:0] OUT;

output reg [3:0] CLK\_OUT;

initial CLK\_OUT = 0;

initial OUT = 8'bXXXXXXX;

always@(posedge CLK)//CLK\_OUT changes at pos edge

begin

CLK\_OUT <= ~CLK\_OUT;

end

always@(posedge CLK\_OUT)//OUT changes at CLK\_OUT

begin

#100 OUT<={IN-4'b0010,IN-4'b0001};

end

endmodule // hw2\_q3

**Test bench:**

`timescale 1ns / 1ps

`default\_nettype none

module hw2\_q3\_tb;

//Inputs

reg CLK;

reg [3:0] IN;

//Outputs

wire [7:0] OUT;

wire CLK\_OUT;

//Instantiate the Unit Under Test (UUT)

hw2\_q3 uut(

.CLK(CLK),

.IN(IN),

.CLK\_OUT(CLK\_OUT),

.OUT(OUT)

);

//generate 20MHz clock signal

always

#25 CLK = ~CLK; //since 2MHz is 50ns per cycle

//=>#25 half cycle

initial begin

CLK=0;

#24.999 IN=4'b0000; //first input has to start before #25

#50 IN=4'b0001;

#50 IN=4'b0010;

#50 IN=4'b0011;

#50 IN=4'b0100;

#50 IN=4'b0101;

#50 IN=4'b0110;

#50 IN=4'b0111;

#50 IN=4'b1000;

#50 IN=4'b1001;

#50 IN=4'b1010;

#50 IN=4'b1011;

#50 IN=4'b1100;

#50 IN=4'b1101;

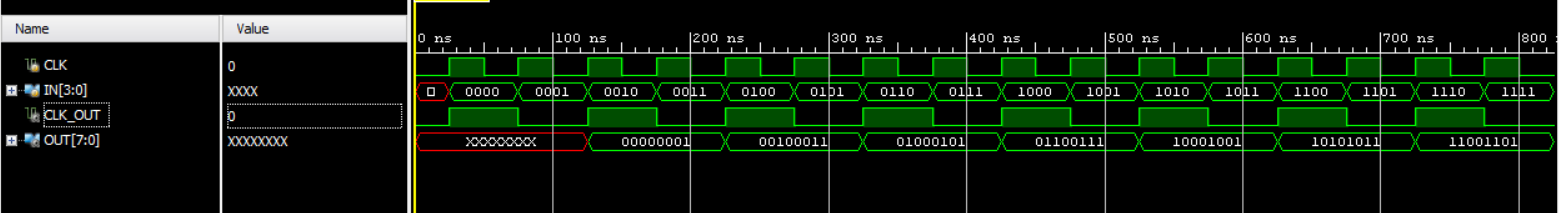
#50 IN=4'b1110;

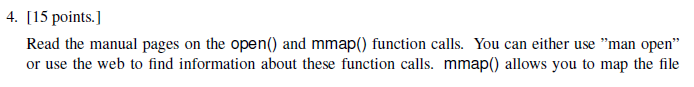
#50 IN=4'b1111;

end

endmodule // hw2\_q3\_tb

**Result:**





**Answer:**

**C code:**

#include <stdio.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/mman.h>

int main( int argc, char \*argv[] )

{

char msg[]="Hello hola how are you";

int i;

int msg\_size = sizeof(msg);

int file\_end;

// Open a file for writing,create if not exist

int fd = open( "Output\_q4.txt", O\_RDWR | O\_CREAT | O\_TRUNC , S\_IRUSR | S\_IWUSR | S\_IRGRP | S\_IROTH | S\_IWGRP | S\_IWOTH);

if (fd == -1) {

perror("Error: file cannot be opened!");

exit(0);

}

// write to Null

file\_end = lseek(fd, msg\_size - 2 , SEEK\_SET);

write( fd , "", 1);

// mapping file

void \*map = mmap(0, msg\_size, PROT\_READ | PROT\_WRITE, MAP\_SHARED, fd, 0);

if (map == MAP\_FAILED) {

close(fd);

perror("Error: cannot mmap to file");

exit(0);

}

// write message to file

strcpy( map , msg);

printf("Message written to file successfully!\n");

// free mmapped memory

if (munmap(map, msg\_size) == -1) {

perror("Error: cannot ummapped memory!");

}

close(fd);

return 0;

}

**Result: Please check Output\_q4.txt for result**