**Carrier Sense Multiple Access with Collision Detection**

**(CSMA/CD)**

*This report is submitted for the evaluation of assignment component for the subject "Data Communication" with the subject code CS415 during the term January to May 2015.*

Submitted by

**1MS13CS068 N HARSHAD REDDY**

**1MS13CS076 P KESAVA PAVAN KUMAR**

**1MS13CS078 PIYUSH**

**1MS13CS083 RAHUL KUMAR**

Faculty In charge:

Prof. Sanjeetha .R

Assistant Professor

Dept. of CSE

M.S.Ramaiah Institute of Technology

****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**M.S.RAMAIAH INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

[www.msrit.edu](http://www.msrit.edu)

# January to May 2014

**TABLE OF CONTENTS**

Page No.

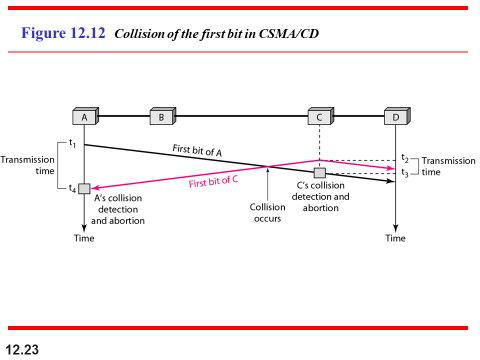
I List of Figures and Flowcharts 3

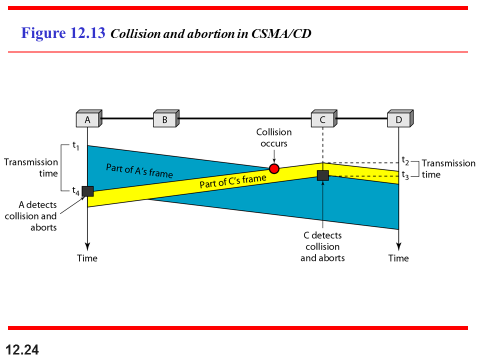
II Abstract 5

III Acknowledgement 6

|  |  |  |
| --- | --- | --- |
| 1 | Introduction | 7 |
| 2 | Hardware/Software Requirements | 10 |
| 3 | Code | 11 |
| 4 | Screenshots | 26 |
| 5 | References | 34 |

**LIST OF FIGURES AND FLOWCHARTS**





The CSMA method does not specify the procedure following a collision. Carrier sense

multiple access with collision detection (CSMA/CD) augments the algorithm to handle

the collision.

In this method, a station monitors the medium after it sends a frame to see if the

transmission was successful. If so, the station is finished. If, however, there is a collision,

the frame is sent again. first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In above Figure stations A and C are involved in the collision.

At time *t* 1, station A has executed its persistence procedure and starts sending the

bits of its frame. At time *t2,* station C has not yet sensed the first bit sent by A. Station C

executes its persistence procedure and starts sending the bits in its frame, which propagate

both to the left and to the right. The collision occurs sometime after time *t2'* Station C

detects a collision at time *t3* when it receives the first bit of A's frame. Station C immediately

(or after a short time, but we assume immediately) aborts transmission. Station A

detects collision at time *t4* when it receives the first bit of C's frame; it also immediately

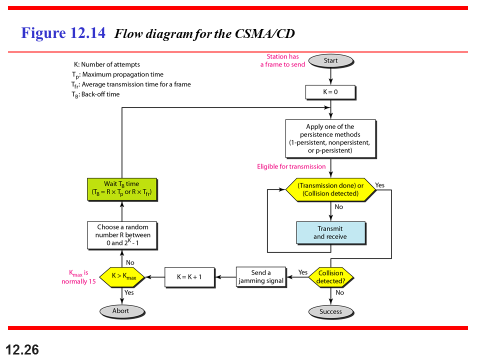
aborts transmission. Looking at the figure, we see that A transmits for the duration *t4* –t1*;*

C transmits for the duration *t3* - *t2'* Later we show that, for the protocol to work, the

length of any frame divided by the bit rate in this protocol must be more than either of

t hese durations. At time *t4,* the transmission of *A’s* frame, though incomplete, is aborted;

At time *t3,* the transmission of B's frame, though incomplete, is aborted.



***Procedure***

The flow diagram for *CSMA\CD* in Figure 12.14. It is similar to the

one for the ALOHA protocol, but there are differences.

The first difference is the addition of the persistence process. We need to sense the

channel before we start sending the frame by using one of the persistence processes we

discussed previously (nonpersistent, I-persistent, or p-persistent). The corresponding

box can be replaced by one of the persistence processes shown in Figure 12.11.

The second difference is the frame transmission. In ALOHA, we first transmit the

entire frame and then wait for an acknowledgment. In *CSMA/CD,* transmission and

collision detection is a continuous process. We do not send the entire frame and then

look for a collision. The station transmits and receives continuously and simultaneously

(using two different ports). We use a loop to show that transmission is a continuous

process. We constantly monitor in order to detect one of two conditions: either transmission

is finished or a collision is detected. Either event stops transmission. When we

come out of the loop, if a collision has not been detected, it means that transmission is

complete; the entire frame is transmitted. Otherwise, a collision has occurred.

The third difference is the sending of a short jamming signal that enforces the collision

in case other stations have not yet sensed the collision.

***Energy Level***

We can say that the level of energy in a channel can have three values: zero, normal,

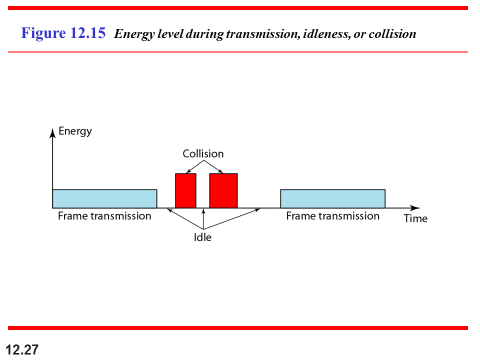
and abnormal. At the zero level, the channel is idle. At the normal level, a station has

successfully captured the channel and is sending its frame. At the abnormal level, there

is a collision and the level of the energy is twice the normal level. A station that has a

frame to send or is sending a frame needs to monitor the energy level to determine if

the channel is idle, busy, or in collision mode. Figure 12.15 shows the situation.



**ABSTRACT**

The project “Carrier sense multiple access with collision detection” is a software implementation and show collision detection between any of the station using a common channel to send and receive data through the Data Link Layer of the Internet Model. This program follows a typical approach to the detect collision, with the Sender uses non-persistent method in which, a station that has a frame to send senses the line. If the line is idle, it sends immediately .If the line is not idle, it waits for random amount of time and then senses the line again.

The program includes the usual functions of the data link layer like collision, back off time, getting data and making a frame and sending data to the receiving station.The program is designed to graphically visualize the sending data and the collision detection. This program is designed such that the user can simulate all the things that can occur while transferring a frame and once a collision is detected the station again retransmits the data using back off time.

The program is designed such that it like an interaction between the user and computer. The user has to give the required input when prompted by the program. The user has the freedom to choose from the options available whether to send frame or not. Based on the input given by the user the program runs the simulation of CSMA/CD.

**ACKNOWLEDGEMENT**

I express my sincere gratitude to Prof. Sanjeetha .R, Dept. of Computer Science and Engineering, MSRIT, for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

**Signature(s) of Students**

N HARSHAD REDDY P KESAVA PAVAN KUMAR

(1MS13CS068) (1MS13CS076)

PIYUSH RAHUL KUMAR

(1MS13CS078) (1MS13CS083)

**1. INTRODUCTION**

CSMA/CD “CARRIER SENSE MULTIPLE ACCESS WITH COLLISION DETECTION”

Is used to send data between stations using the same channel as medium. Each station in the channel senses the channel before transmitting the data that needs to be sent, using any one of the three persistent method, we use non persistent method .once the station sense the channel if the channel is free, it sends the data to concerned station, if the channel is not free the station waits for random amount of time and then again senses the channel and then retransmits.

While sending data, if a station’s data encounter collision with another station’s data. The channel is jammed and then again using random back off time the station retransmits the data. This procedure is followed till each data is sent successfully to their concerned destination station.

**2.HARDWARE/SOFTWARE REQUIREMENTS**

We have designed is a C language code Implementation of CSMA/CD, no hardware is required. Only a PC which has Windows XP or higher or any other OS which can run a GCC compiler is more than sufficient. We have created this program using Code Blocks IDE.

We have used RAND () function in our program, so to generate random back off time.

Other than this not much software is required.

**3.CODE**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<math.h>

int channel[50]={0},n,tf=5,ttemp[10],a=0,k=2,w=0;

int ds[10]={1,1,1,1,1,1,1,1,1,1},is[10]={0},js[10]={0};

int status[10]={1,1,1,1,1,1,1,1,1,1};

typedef struct

{int data[5];

int dest;

int time;}station;

void send(int sno);

int stat\_ini();

int send\_data();

void collision();

int backoff()

{int kmax=15,r,s,tb;

if(k>kmax)

{printf("station exceeded its limit\n%d:",k);

exit(0);}

else

{s=(pow(2,k)-1);

r=rand()%s;

tb=r\*tf;

k=k+1;}

return (tb);}

station stat[10];

main()

{

int i,j,ch;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\n\tCARRIER SENSE MULTIPLE ACCESS WITH COLLISION DETECTION\t\n");

printf("\t\t\t\tCSMA|CD\t\t\t\t\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("Enter the number of stations :");

scanf("%d",&n);

for(i=1;i<=n;i++)

{printf("\nEnter 1, If station%d want to transmit signal :",i);

scanf("%d",&ch);

stat[i].time=0;

if(ch==1)

{

printf("\nEnter time of sending signal of station%d :",i);

scanf("%d",&stat[i].time);

a++;

status[i]=0;

l:printf("\nEnter destination of station%d :",i);

scanf("%d",&stat[i].dest);

if(stat[i].dest==i||stat[i].dest<1||stat[i].dest>n)

{printf("Wrong destination,Try again\n");

goto l;}

printf("\nEnter the data(3-Bit data) :");

for(j=1;j<=3;j++)

{scanf("%d",&stat[i].data[j]);}

}

}

for(i=1;i<=n;i++)

{ttemp[i]=stat[i].time;}

send\_data();}

void collision(int sno)

{int i,tb;

for(i=1;i<=n;i++)

{if(status[i]!=1)

{stat[i].time=ttemp[i];

is[i]=0,js[i]=0,ds[i]=1;}}

for(i=1;i<=(n\*5);i++)

channel[i]=0;

printf("jamming signal sent\n");

tb=backoff();

stat[sno].time=stat[sno].time+tb;

printf("station backoff by:%d",tb);

send\_data();

}

int stat\_ini() //to slect the least time

{int x,i,min;

for(i=1;i<=n;i++)

{if(stat[i].time!=0)

{min=i;

break;}}

x=stat[min].time;

for(i=1;i<=n;i++)

{if(x>stat[i].time&&stat[i].time!=0)

x=stat[i].time;}

printf("Time:%d\n",x);

return x;}

int send\_data() //to the state which sends data

{int tmin=stat\_ini(),t,i,tb;

for(t=tmin;;t++)

{for(i=1;i<=n;i++)

{if(t==stat[i].time)

{//printf("\nstation:%d",i);

if(ds[i]==1&&channel[i\*5-4]!=0) //for non-persistent

{tb=backoff();

printf("channel not free backing off by:%dsec",tb);

stat[i].time=stat[i].time+tb;}

else

{

send(i);

stat[i].time++;}

}}

printf("\n");

for(i=1;i<=(n\*5);i++)

printf("%d",channel[i]);

getch();

}}

void send(int sno)

{

int databit=0;

if(ds[sno]<=3)

databit=stat[sno].data[ds[sno]];

printf("\nDatabit=%d\nsno:%d\n",databit,sno);

if(is[sno]==0 && js[sno]==0) //sending for first time

{is[sno]=js[sno]=sno\*5-4;

channel[sno\*5-4]=databit;}

else

{

if(channel[is[sno]+1]!=0 || channel[js[sno]-1]!=0)

{

printf("\nCollision!!\n");

getch();

collision(sno);

}

else

{if(ds[sno]<=3)

{

int x;

if(sno!=n)

{ if(is[sno]<=(n\*5)-1||stat[sno].dest!=n)

{is[sno]++;}

for(x=is[sno];x>sno\*5-4;x--)

channel[x]=channel[x-1];

}

if(sno!=1)

{if(js[sno]>=2||stat[sno].dest==1)

{js[sno]--;}

for(x=js[sno];x<sno\*5-4;x++)

channel[x]=channel[x+1];

}

channel[sno\*5-4]=databit;

}

else

{

int x;

if(sno!=n)

{

if(is[sno]<=(n\*5)-1||stat[sno].dest!=n)

is[sno]++;

x=is[sno];

int j;

for(j=1;j<=3;x--)

{channel[x]=channel[x-1];j++;}

if(x!=(sno\*5-4)||x==1)

{channel[x]=0;}

}

if(sno!=1)

{

if(js[sno]>=2||stat[sno].dest==1)

js[sno]--;

x=js[sno];

int j;

for(j=1;j<=3;x++)

{channel[x]=channel[x+1];

j++;}

channel[x]=0;

}

if(channel[stat[sno].dest\*5-3]==stat[sno].data[3])

{{printf("\n Data sending succesfull.\n");

status[sno]=1;

w++;

getch();}

if(a==w)

{printf("All signals sent successfully\n");

exit(0);}}

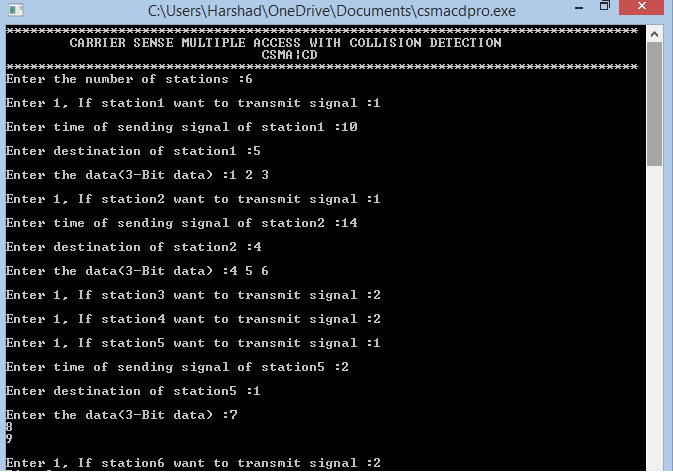
channel[sno\*5-4]=databit;}}

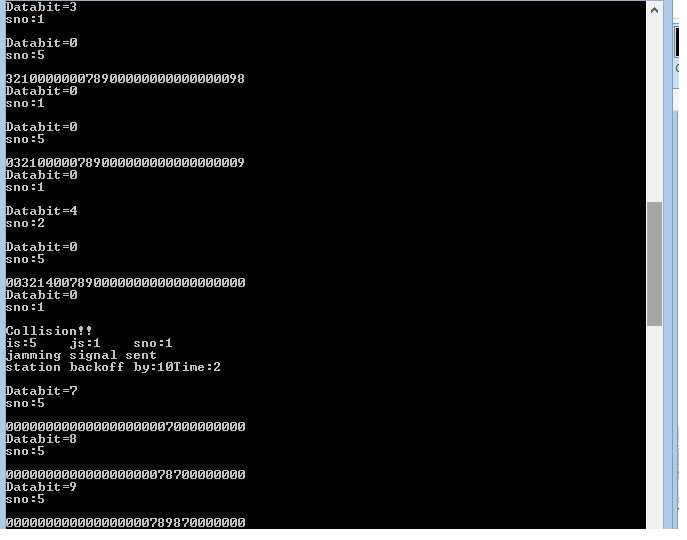
}

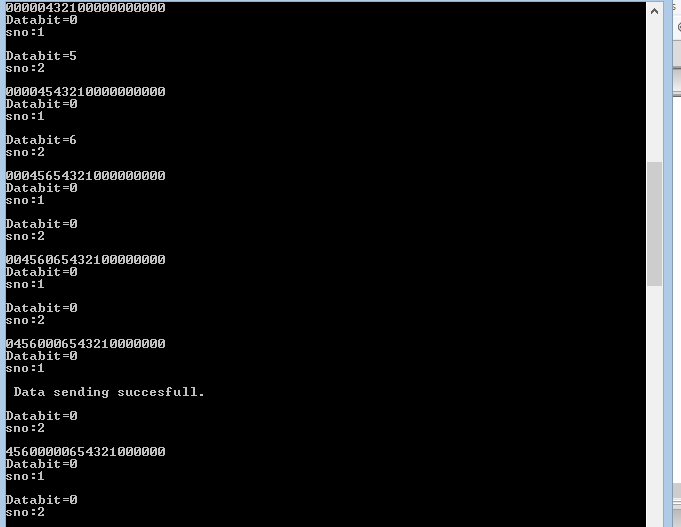
ds[sno]++;

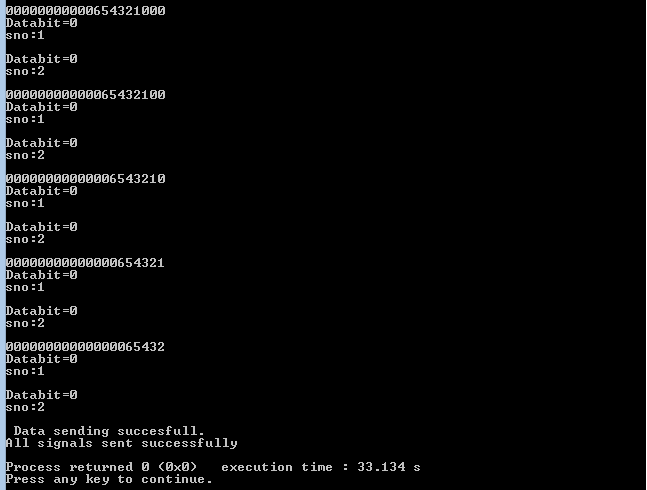
}

**SCREENSHOTS:-**

****







**5.REFERENCES**

**TEXT BOOKS:**

1.) DATA COMMUNICATIONS AND NETWORKING, Fourth Edition, Behrouz A Forouzan, McGraw Hill Education.

2.)DATA AND COMPUTER COMMUNICATIONS, Eighth Edition, William Stallings.

**WEBSITES:**

1.)[www.google.com](http://www.google.com)

2.)[www.wikipedia.org](http://www.wikipedia.org)