Reference

[1] E. Yablonovitch, **Phys. Rev. Lett. 58,2059-2062, 1987**

[2] S. John, **Phys. Rev. Lett. 58, 2486-2488,1987**

[3] J.D. Joannopoulos, R.D.Meade and J.N. Winn, **Photonic Crystals-Molding the Flow of Light, 1995.**

[4] T.F. Krauss, and R.M. De La Rue, Prog. **Quantum Electron. 23, 51-96,1999**

[5] Kazuaki Sakoda, **Optical Properties of Photonic Crystal, 2001**

[6] Steven G. Johnson, John D. Joannopoulos, **Photonic crystals: the road from theory to practice,2002**

[7] Costas M. Soukoulis, **Photonic crystals and light localization in the 21st century: Proceedings of the NATO Advanced Study Institute on Photonic Crystals and Light Localization, 2000**

[8] S. John and R. Rangarjan: **Phys. Rev. Lett. B, Vol 38, 10101 (1988)**

[9] M. Plihal and A. A. Maradudin: **Phys. Rev. B, Vol 44, 8569 (Oct, 1991)**

[10] Karlheinz Bierwirth: **IEEE Trans. Microwave Theory and Tech., Vol 34, p1104 (Nov,1986)**

[11] Dennis M.Sullivan: **Electromagnetic Simulation Using The FDTD Method. IEEE press, New York, 2001**

[12] Yee, K.S., *IEEE Trans. Antennas and Propagation*, Vol. 14, 1966, pp. 302-307

[13] A. Taflove and S.C. Hagness, *Computation Electrodynamics: The Finite-Difference Time-Domain Method*. Boston．London: Artech House, 2000.

[14] K.S. Kunz and R.J. Luebbers, *The Finite Difference Time Domain Method for Electromagnetics*. Boca Raton, FL; CRC Press, 1993.

[15] Ke-Yuan Chen, **Analysis and Simulation of 1-D and 2-D Periodical Photonic Crystal, 2002**

[16] V. Lehmann, The Electrochemical Society, Vol.140, No.10 (Oct, 1993)

APPENDIX A

We use this program to calculate the transmission spectra of the photonic crystal. Fig. 3-4 ~ 3-6, 3-9, 3-15, 3-19, 3-20, 3-21, 3-22, and Table 3-1 ~ 3-5 are calculated from this program.

```
/***************************************************/
/*1D FDTD simulation of a pulse hitting a dielectric medium */
/***************************************************/

/*******************Initial Condition*********************/
/*Using Flux density                                      */
/*Let lattice constant a'=a*dz, b'=b*dz                   */
/*Let dt=dz/(u*dz),                                       */
/*Let normalized frequency add o.o1 for every time        */
/*******************************************************/

#include <cmath>
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <time.h>

const int PS=41;          // photonic crystal start at PS grid.
const int mc=21;          // where the source generate
const int t0=1;
clock_t Start, Finish;

void Structure(double *,int,int,int,int,int,double,double,double);
void Source(double *,double,int,int,int);
void CalculateDx(double *,double *,double,int);
void CalculateEx(double *,double *,double *,int);
void ABC(double,double,double,double);
void CalculateHy(double *,double *,double,int);
void FourierTransform (double *,double *,double *,double,double,double,double,int);
void Amplitude(double *, double *,double *,int);
void WriteData(char *,double *,int,int);
```

```
void Report(char *,int,double,double,double,int,double,double,double);
void FindPBG(char *,double *,double *,int,double,int);
void TotalTime(int);

int main()
{
        //Difine every paramater
    int m,i,j;
    int a=40;
    int TE;             // total time steps of the fourier transform of incident wave
        int FE=1000;
    int TypeSwitch;
    TE=2*t0;

    double u=1.;
    double w=1;
    double v;
    double W;
    double x;
    double Frequency1;
    double Frequency2;

    v=1/u;
    W=(1-u)/(1+u);

    char Q='y';
    char P;
    char ChangeFE='y';
        char L='n';
        char MoreTimeSteps='n';
        char *StructureDataOutput ="Structure.dat";
        char *ExDataOutput ="Ex.dat";
        char *ReportOutput ="Report.dat";
        char *SpaceDataOut ="Space.dat";
        char *FrequencyDataOut ="Frequency.dat";
        char *dBOut ="dB.dat";
        char *TransmissionDataOutput ="Trans.dat";
        char *dispersionOut ="dispersion.dat";
```

```
while(Q=='y' ||Q=='Y')
{
	int T1=0;
	int T2=0;
	int Layers=40;
	int PE;			// photonic crystal end at PE grid
	int ME;			// total grides of FDTD calculation
	int Nsteps;		// total time steps
			int elapsed_time=0;
	int count=0;
	int jj=0;

	double Ratio;
	double epsilon1=1.;
	double epsilon2=1.;

	double T=0.;			// T is the index of total times of calculation
	double TT=0.;		// TT is the total time steps of Fourier Transform

	//Define the structure of photonic crystal
	cout <<"Lattice constant is "<<a<<endl;
	cout <<"Do you want to change a?(n/y)";
	cin >> L;

	if(L=='y'){
		cout<< "Please input a:";
		cin >>a;}
	else;

	cout << "The dielectric constant of Material(1):" ;
	cin    >> epsilon1;
	cout << "The dielectric constant of Material(2):" ;
	cin    >> epsilon2;
	cout << "The ratio of b and a (b/a):" ;
	cin    >> Ratio;
	cout << "How many layers do you want to calculate?";
	cin    >> Layers;
```

```
PE=PS+Layers*a-1;
ME=PE+40;
Nsteps=(PE-mc)*Layers*60;

double *space=new double[ME];
double *sp=new double[ME];        //sp[m] is the dielectric constant of
gride m.
double *ex=new double[ME];
double *hy=new double[ME];
double *dx=new double[ME];
double *nfreq=new double[FE];             // nfreq[n] is the normalized
frequency
double *real=new double[FE];              // real[f] is the real part of the
Fourier Transform of ex
double *image=new double[FE];             // image[f] is the image part of
the Fourier Transform of ex
double *trans=new double[FE];
double *dB=new double[FE];
double *k_dispersion=new double[FE];
double exm1=0.;
double exm2=0.;
double exm3=0.;
double exm4=0.;

//Generate the array of every quantity
for(m=0;m<ME;m++){
    ex[m]=0.;
    hy[m]=0.;
    dx[m]=0.;
    space[m]=m;}

for(i=0;i<FE;i++){
    nfreq[i]=i*x+Frequency1;
    real[i]=0.;
    image[i]=0.;
    trans[i]=0.;
    dB[i]=0.;
```

```
          k_dispersion[i]=0.;}

     //define the structure of PC
     Structure(sp,a,PS,PE,ME,Layers,Ratio,epsilon1,epsilon2);
     WriteData(StructureDataOutput,sp,0,ME-1);
     WriteData(FrequencyDataOut,nfreq,0,FE-1);
     WriteData(SpaceDataOut,space,0,ME-1);

Calculate_Begin_2:

     time_t     start, finish;
     time( &start );

     cout << "Calculating..." << endl;
     cout << "Nsteps="<<Nsteps<<endl;

     //Begin calculating every quanty
     for(int n=0;n<Nsteps;n++)
     {
          T=T+1;
          //Calculate dx at every grid
          CalculateDx(dx,hy,v,ME);
          //Give the source of FDTD
          Source(dx,T,mc,t0,TE);
          //calculate ex at every grid
          CalculateEx(ex,dx,sp,ME);
          //apply the Mur's ABC
          ex[0]=exm2+W*(ex[1]-exm1);
          exm1=ex[0];
          exm2=ex[1];
          ex[ME-1]=exm4+W*(ex[ME-2]-exm3);

          exm3=ex[ME-1];
          exm4=ex[ME-2];
          //Calculate hy at every grid
          CalculateHy(hy,ex,v,ME);
          //Calculate the fourier transform of trasmissive wave
          if(ex[PE+10]!=0)
```

```
	{
		TT=TT+1.;
		FourierTransform(real,image,nfreq,ex[PE+10],TT,a,u,FE);
	}

	T1=T;
	T2=T1%10000;
	if(T2==0)
	{cout <<"T is "<<T<<" now."<<endl;}
}

//Calculate the ampiltude of incident and tramissive wave
Amplitude(trans,real,image,FE);

for(i=0;i<FE;i++)
{dB[i]=20*log10(trans[i]);}

cout<< "T=" <<T <<endl;
cout<< "TT="<<TT<<endl;
//Write the data in the files
WriteData(ExDataOutput,ex,1,ME-1);
WriteData(dBOut,dB,0,FE-1);
//Find the PBG
FindPBG(ReportOutput,trans,nfreq,a,x,FE);

WriteData(TransmissionDataOutput,trans,0,FE-1);
WriteData(dispersionOut,k_dispersion,0,FE-1);

time( &finish );
elapsed_time = difftime( finish, start );
TotalTime(elapsed_time);

cout << "Do you want to calculate for more time-steps?(n/y)";
cin   >> MoreTimeSteps;

if(MoreTimeSteps=='y' || MoreTimeSteps=='Y')
{
	cout << "Time-Steps:";
```

```
            cin    >> Nsteps;
            goto Calculate_Begin_2;
        }
        else;

        delete [] ex;
        delete [] hy;
        delete [] dx;
        delete [] sp;
        delete [] nfreq;
        delete [] real;
        delete [] image;
        delete [] trans;
        delete [] space;
    }

    return 0;
}
```

APPENDIX B

We use this program to simulate the behavior of light in the photonic crystal. Fig. 3-10 ~ 3-13, 3-16 and 3-17 are calculated form this program.

```
/*******************Initial Condition*********************/
/*Using Flux density                                      */
/*Let lattice constant a'=a*dz, b'=b*dz                   */
/*Let dt=dz/(u*dz),                                       */
/*Let normalized frequency add o.o1 for every time        */
/*********************************************************/

#include <cmath>
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <time.h>

const int PS=300;          //photonic crystal start at PS grid.
const int mc=100;           //where the source generate
const int t0=10;

void Structure(double *,int,double,double,double,int,int,int);
void Source(double *,double,double,double,double,int);
void CalculateDx(double *,double *,double,int);
void CalculateEx(double *,double *,double *,int);
void CalculateHy(double *,double *,double,int);
void WriteData(char *,double *,int,int,int,int,int);

int main()
{
    //index
    int a=40;
    int PE;
    int ME;
    int CheckStart;
    int CheckEnd;
```

```
double u=2.;
double b;
double c;
double z;
double v;
double W;
double w;
double epsilon1;        // epsilon1 is the dieletric constants of material(1)
double epsilon2;        // epsilon2 is the dieletric constants of material(2)
double nfreq;
double exm1;
double exm2;
double exm3;
double exm4;

v=1/u;
W=(1-u)/(1+u);

char Q='y';
char *FileNameOut1 ="Structure.dat";
char *FileNameOut2 ="CheckStructure.dat";
char *FileNameOut3 ="Ex.dat";
char *FileNameOut4 ="Hy.dat";
char *FileNameOut5 ="CheckEx.dat";

while(Q=='y')
{
    //Define the structure of photonic crystal
    cout << "The lattice constant of PC is "<<a<<endl;
    cout << "The ratio of b and a (b/a):" ;
    cin  >> c;
    cout << "How many layers do you want to calculate?";
    cin  >> z;

    b=c*a;
    PE=PS+z*a-1;
    ME=PE+200;
```

```
cout << "The source start at gride " <<mc <<endl;
cout << "Photonic crystal start at gride " <<PS <<endl;
cout << "Photonic crystal end at gride " <<PE <<endl;
cout << "Input the gride range you want to check: StartGride ~ 
EndGride"<<endl;
cout << "StartGride:";
cin  >> CheckStart;
cout << "EndGride:";
cin  >> CheckEnd;

cout << "The dielectric constant of Material(1):" ;
cin  >> epsilon1;
cout << "The dielectric constant of Material(2):" ;
cin  >> epsilon2;
cout << "Normalized Frequency:";
cin  >> nfreq;


//Generate the array of every quantity
int Nsteps=1;
double *sp=new double[ME];
double *ex=new double[ME];
double *hy=new double[ME];
double *dx=new double[ME];
double T=0.;

//initialize the value of every quanty
for(int m=0;m<ME;m++)
{
    ex[m]=0.;
    hy[m]=0.;
    dx[m]=0.;
}

exm1=0.;
exm2=0.;
exm3=0.;
```

```
exm4=0.;

//define the structure of PC
Structure(sp,a,b,epsilon1,epsilon2,ME,PS,PE);

WriteData(FileNameOut1,sp,PS-20,PE+10,1,1,1);
WriteData(FileNameOut2,sp,CheckStart,CheckEnd,2,1,5);

//begine the calculation
while(Nsteps>0)
{
    cout <<"Nsteps:";
    cin   >>Nsteps;

    if (Nsteps==0)
    {goto Calculation_Over;}

    for(int n=0;n<Nsteps;n++)
    {
        T=T+1;
        //Calculate dx at every grid
        CalculateDx(dx,hy,v,ME);
        //Give the source of FDTD
        Source(dx,nfreq,T,a,u,mc);
        //calculate ex at every grid
        CalculateEx(ex,dx,sp,ME);
        //apply the Mur's ABC
        ex[0]=exm2+W*(ex[1]-exm1);
        exm1=ex[0];
        exm2=ex[1];
        ex[ME-1]=exm4+W*(ex[ME-2]-exm3);
        exm3=ex[ME-1];
        exm4=ex[ME-2];
        //Calculate hy at every grid
        CalculateHy(hy,ex,v,ME);
    }

    cout << "T=" <<T<<endl;
```

```
            //Write the data in the files
            WriteData(FileNameOut3,ex,1,ME-1,1,1,1);
            WriteData(FileNameOut4,hy,0,ME-2,1,1,1);
            WriteData(FileNameOut5,ex,CheckStart,CheckEnd,2,1,1);
        }

        Calculation_Over: ;

        delete ex;
        delete hy;
        delete dx;
        delete sp;

    }

    return 0;

}
```