LAMPIRAN

LAMPIRAN

LISTING PROGRAM DAN DATA SUARA

<< SPEECH.DPR >>

```
program Speech;
uses
  Forms,
  main in 'main.pas' {Form1},
  Fourier in 'fourier.pas',
  windowing in 'windowing.pas',
  lpc in 'lpc.pas',
  backpro in 'backpro.pas',
  paramtrain in 'paramtrain.pas' {Parameter};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TParameter, Parameter);
  Application.Run;
end.
```

<< MAIN.PAS >>

```
unit main;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  ComCtrls, StdCtrls, ExtCtrls, Buttons, TeEngine, Series,
TeeProcs, Chart,
  Menus, IniFiles, MsAcm, MMSystem, Fourier,math;

var
  Form1: TForm1;
  panjang,sequen:integer;
  ulang:byte;

    //--------------data preprocessing sinyal --------------------
    cep,potong       :Tdatabobot;

    //---------  data untuk backpropagation ----------------

    iterasi,iunit,jdata,pandata : integer;
    alpha,eror_mak          : double;
    hunit                       : array of integer;
    masuk,wbobot                : Tdatabobot;
    vbobot                      : array of Tdatabobot;
    identitas                   : array of string;
```

```
implementation
uses paramtrain,windowing,lpc,backpro;

{$R *.DFM}

const
    SectionWindow       = 'RecDemo';
    KeyTop              = 'Top';
    KeyLeft             = 'Left';
    KeyWidth            = 'Width';
    KeyHeight           = 'Height';

procedure tform1.frontEnd;
var a,awal,akhir:integer;
    temp:T1dimensi;
begin
 awal:=0;
 akhir:=0;
 for a:=0 to high(realdata) do
  if abs(realdata[a])>=confe*maksval then
   begin
    awal:=a;
    break;
   end;
 for a:=high(realdata) downto 0 do
  if abs(realdata[a])>=confe*maksval then
   begin
    akhir:=a;
    break;
   end;
 setlength(temp,akhir-awal+1);
 for a:=awal to akhir do
  temp[a-awal]:=realdata[a];
 setlength(realdata,0);
 setlength(realdata,akhir-awal+1);
 for a:=0 to high(temp) do
  realdata[a]:=temp[a];
end;

procedure Tform1.PengolahanSinyal(sinyal:array of double);
var p,i,j           : integer;
    win,aut         : array of double;
    frekdata        : array of double; //data dalam domain frek
    realtime,imgtime: Tdatabobot; // data terframing domain time
    realfrek,imgfrek: Tdatabobot; // data terframing domain frek
    jumframe        : integer;
begin
 jumframe:=FrameCount(UkuranFrame,UkuranFrame div
3,high(sinyal)+1);

 setlength(realtime,jumframe);
 setlength(imgtime,jumframe);
 setlength(realfrek,jumframe);
 setlength(imgfrek,jumframe);
 setlength(cep,jumframe);
```

```
pre_emphasis(0.94,sinyal);
setlength(realtime,jumframe);
for i:=0 to jumframe-1 do
 begin
  setlength(realtime[i],UkuranFrame);
  setlength(imgtime[i],UkuranFrame);
  setlength(realfrek[i],UkuranFrame);
  setlength(imgfrek[i],UkuranFrame);
 end;

framing(UkuranFrame,UkuranFrame div 3,sinyal,realtime);

setlength(win,UkuranFrame);
win_sinyal(0,hanning,win);

for i:=0 to jumframe-1 do
 for j:=0 to UkuranFrame-1 do
  realtime[i,j]:=realtime[i,j]*win[j];

for i:=0 to jumframe-1 do
 fft(UkuranFrame,realtime[i],imgtime[i],realfrek[i],imgfrek[i]);

for i:=0 to jumframe-1 do
 for j:=0 to UkuranFrame-1 do

realfrek[i,j]:=log10(sqrt(sqr(realfrek[i,j])+sqr(imgfrek[i,j])));

for i:=0 to jumframe-1 do
 setlength(realfrek[i],length(realfrek[i]) div 2);

// p:=MakeOrder(frek);
p:=15;
setlength(aut,p+1);
for i:=0 to jumframe-1 do
 begin
  setlength(cep[i],p+1);
  LPCAnalisis(realfrek[i],length(realfrek[i]),p,aut);
  lpc2cepstral(p,p,aut,cep[i]);
  weightingcepstral(p,cep[i]);
 end;
normalisasi;
end;

procedure tform1.normalisasi;
var a,b:integer;
    kecil,besar:double;
begin
 kecil:=100;
 besar:=-100;
 for a:=0 to high(cep) do
  for b:=0 to high(cep[a]) do
   begin
    kecil:=min(kecil,cep[a,b]);
    besar:=max(besar,cep[a,b]);
   end;
 if kecil<0 then
```

```
  for a:=0 to high(cep) do
   for b:=0 to high(cep[a]) do
    cep[a,b]:=cep[a,b]-kecil;
 for a:=0 to high(cep) do
  for b:=0 to high(cep[a]) do
   cep[a,b]:=cep[a,b]/(besar-kecil+1);
end;

function tform1.DoTrain:integer;
var
    output,out_in,eror_k        : T1dimensi;
    hiden,hiden_in,eror_j       : Tdatabobot;
    target                      : Tdatabobot;
    loop,num,jhiden             : integer;
    i,j                         : integer;
    kenal                       : integer;
    sum,sumall                  : double;
begin
 jhiden:=length(hunit);
 InitTrain(hiden,hiden_in,eror_j,output,out_in,eror_k);
 InitTarget(target);
 for loop:=1 to iterasi do
  begin
   kenal:=0;
   sumall:=0;
   application.ProcessMessages;
   for num:=0 to high(masuk) do
    begin
     //---------feedforrward process---------//
     LayerIn(masuk[num],hiden_in[0],vbobot[0]);
      FungsiAktivasi(hiden_in[0],hiden[0]);
     if high(hunit)>0 then
      for i:=0 to high(hunit)-1 do
       begin
        LayerIn(hiden[i],hiden_in[i+1],vbobot[i+1]);
        FungsiAktivasi(hiden_in[i+1],hiden[i+1]);
       end;
     LayerIn(hiden[jhiden-1],out_in,wbobot);
     FungsiAktivasi(out_in,output);
     //----------------chek for reached target----------
     if loop mod 10=0 then
      begin
       sum:=0;
       for i:=1 to high(output) do
        sum:=sum+abs(output[i]-target[num,i]);
       sum:=sum/jdata;
       if sum<eror_mak then
        inc(kenal);
       sumall:=sumall+sum;
       statusbar1.Panels[1].Text:=' Data identify
'+inttostr(kenal)+' from '+inttostr(jdata)+ ' at
'+inttostr(loop)+' epoch';
       if num=high(masuk) then statusbar1.Panels[2].Text:=' Error
= '+floattostr(sumall/jdata);
       if kenal=jdata then
```

```
        if CekBobotAll(hiden_in,hiden,out_in,output,target,jhiden)
then
         begin
          result:=2;
          exit;
         end;
      end;
     //------backforward process----------//
     CalculateOutputEror(target[num],output,out_in,eror_k);
     CalculateHidenEror(eror_k,hiden_in[jhiden-
1],wbobot,eror_j[jhiden-1]);
     if high(hunit)>0 then
      for i:=high(hunit) downto 1 do
       CalculateHidenEror(eror_j[i],hiden_in[i-
1],vbobot[i],eror_j[i-1]);

     //---------update bobot------------//
     UpdateBobot(alpha,eror_k,hiden[jhiden-1],wbobot);
     UpdateBobot(alpha,eror_j[0],masuk[num],vbobot[0]);
     if high(hunit)>0 then
      for i:=high(hunit) downto 1 do
       UpdateBobot(alpha,eror_j[i],hiden[i-1],vbobot[i]);
    end;
  end;
  result:=1;//-------normaly exit
end;

function GetDecision(output:t1dimensi;target:Tdatabobot):integer ;
var i,j          :integer;
    sum,min_e    :double;
begin
 min_e:=1000;
 result:=0;
 for i:=0 to high(target) do
  begin
   sum:=0;
   for j:=1 to high(target[i]) do
    sum:=sum+abs(output[j]-target[i,j]);
   if min_e>sum then
    begin
     result:=i;
     min_e:=sum;
    end;
  end;
end;

function tform1.Kenali(var isiData:T1dimensi):integer;
var output,out_in         : T1dimensi;
    hiden,hiden_in        : Tdatabobot;
    target                : Tdatabobot;
    i,j,k,jhiden,curr     : integer;
begin
 curr:=length(isiData);
 if curr<>pandata then
  begin
   stretchsuara(isidata);
```

```
 end;
setlength(masuk,1);
setlength(cep,0);
PengolahanSinyal(IsiData);
setlength(masuk[0],iunit);
masuk[0,0]:=1;
k:=1;
for i:=0 to high(cep) do
  for j:=0 to high(cep[i]) do
    begin
     masuk[0,k]:=cep[i,j];
     inc(k);
    end;
InitRead(hiden_in,hiden,out_in,output);
InitTarget(target);
jhiden:=length(hunit);
LayerIn(masuk[0],hiden_in[0],vbobot[0]);
FungsiAktivasi(hiden_in[0],hiden[0]);
if high(hunit)>0 then
 for i:=0 to high(hunit)-1 do
  begin
   LayerIn(hiden[i],hiden_in[i+1],vbobot[i+1]);
   FungsiAktivasi(hiden_in[i+1],hiden[i+1]);
  end;
LayerIn(hiden[jhiden-1],out_in,wbobot);
FungsiAktivasi(out_in,output);
i:=GetDecision(output,target);
result:=i;
end;
```

<< FOURIER.PAS >>

```
{$N+,E+}    (* Allows code to use type 'double' and run on any iX86
machine *)
{$R-}       (* Turn off range checking...we violate array bounds
rules *)

unit Fourier;
interface

procedure fft (
    NumSamples:   word;   { must be a positive integer power of 2
}
    var  RealIn:   array of double;
    var  ImagIn:   array of double;
    var  RealOut:  array of double;
    var  ImagOut:  array of double );


implementation

function IsPowerOfTwo ( x: word ): boolean;
var   i, y:  word;
begin
    y := 2;
    for i := 1 to 15 do begin
```

```
        if x = y then begin
            IsPowerOfTwo := TRUE;
            exit;
        end;
        y := y SHL 1;
    end;
    IsPowerOfTwo := FALSE;
end;

procedure FourierTransform (
    AngleNumerator:  double;
    NumSamples:   word;
    var  RealIn:   array of double;
    var  ImagIn:   array of double;
    var  RealOut:  array of double;
    var  ImagOut:  array of double );
var
    NumBits, i, j, k, n, BlockSize, BlockEnd: word;
    delta_angle, delta_ar: double;
    alpha, beta: double;
    tr, ti, ar, ai: double;
begin
    if not IsPowerOfTwo(NumSamples) or (NumSamples<2) then begin
        write ( 'Error in procedure Fourier:  NumSamples=',
NumSamples );
        writeln ( ' is not a positive integer power of 2.' );
        halt;
    end;

    NumBits := NumberOfBitsNeeded (NumSamples);
    for i := 0 to NumSamples-1 do begin
        j := ReverseBits ( i, NumBits );
        RealOut[j] := RealIn[i];
        ImagOut[j] := ImagIn[i];
    end;

    BlockEnd := 1;
    BlockSize := 2;
    while BlockSize <= NumSamples do begin
        delta_angle := AngleNumerator / BlockSize;
        alpha := sin ( 0.5 * delta_angle );
        alpha := 2.0 * alpha * alpha;
        beta := sin ( delta_angle );

        i := 0;
        while i < NumSamples do begin
            ar := 1.0;     (* cos(0) *)
            ai := 0.0;     (* sin(0) *)

            j := i;
            for n := 0 to BlockEnd-1 do begin
                k := j + BlockEnd;
                tr := ar*RealOut[k] - ai*ImagOut[k];
                ti := ar*ImagOut[k] + ai*RealOut[k];
                RealOut[k] := RealOut[j] - tr;
                ImagOut[k] := ImagOut[j] - ti;
```

```
                RealOut[j] := RealOut[j] + tr;
                ImagOut[j] := ImagOut[j] + ti;
                delta_ar := alpha*ar + beta*ai;
                ai := ai - (alpha*ai - beta*ar);
                ar := ar - delta_ar;
                INC(j);
            end;

            i := i + BlockSize;
        end;

        BlockEnd := BlockSize;
        BlockSize := BlockSize SHL 1;
    end;
end;

procedure fft (
    NumSamples:   word;
    var  RealIn:   array of double;
    var  ImagIn:   array of double;
    var  RealOut:  array of double;
    var  ImagOut:  array of double );
begin
    FourierTransform ( 2*PI, NumSamples, RealIn, ImagIn, RealOut,
ImagOut );
end;

end.
(*--- end of file fourier.pas ---*)
```

<< WINDOWING.PAS >>

```
{$N+,E+}    (* Allows code to use type 'double' and run on any iX86
machine *)
{$R-}       (* Turn off range checking...we violate array bounds
rules *)

unit windowing;

interface

uses math,main;
function FrameCount(n,m,panjang:integer):integer;
procedure framing(n,m:integer;sinyal:array of double;var
hasil:Tdatabobot);
procedure pre_emphasis(koefisien:double;var sinyal:array of
double);
procedure win_sinyal(nflg:integer;kode:twindow;var win:array of
double);

const M_2PI:double=2 * 3.14159265358979323846;
implementation

function FrameCount(n,m,panjang:integer):integer;
var a,jum:integer;
begin
 a:=0;jum:=0;
```

```
 repeat
  inc(jum);
  inc(a,n-m);
 until a>panjang;
 result:=jum;
end;

procedure framing(n,m:integer;sinyal:array of double;var
hasil:Tdatabobot);
var panjang:integer;
    posisi :integer;
    a,b    :integer;
begin
 panjang:=high(sinyal)+1;
 posisi:=0;
 b:=0;
 repeat
  for a:=0 to n-1 do
   begin
    if posisi+a>=panjang then
     hasil[b,a]:=0
    else
     hasil[b,a]:=sinyal[posisi+a];
   end;
  inc(posisi,n-m);
  inc(b);
 until posisi>panjang;
end;

{
 prosedur pre emphasis
 koefisien -> nilai dari penguatan berkisar antara 0.9 - 1
 sinyal -> sinyal yang akan dilakukan proses pre emhasis
}

procedure pre_emphasis(koefisien:double;var sinyal:array of
double);
var temp:array of double;
    a   :integer;
begin
 setlength(temp,high(sinyal));
 for a:=1 to high(sinyal) do
  temp[a]:=sinyal[a]-koefisien*sinyal[a-1];
 for a:=1 to high(sinyal) do
  sinyal[a]:=temp[a];
end;

procedure hanning_win(var win:array of double);
var arg:double;
    a  :integer;
    panjang:integer;
begin
  panjang:=high(win);
  arg:= M_2PI /panjang;
  for a:=0 to panjang do
   win[a]:= 0.5 * (1 - cos(a * arg));
```

```
end;

{
 prosedur windowing sinyal
 panjang -> panjang dari daerah yang akan di window
 nflg -> normalisasi flag
  0 -> tidak dinormalisasi
  1 -> normalisasi oleh power
  2 -> normalisasi oleh magnitude
 kode -> kode tipe window yang dipakai
 win -> koefisien hasil windowing;
}
procedure win_sinyal(nflg:integer;kode:twindow;var win:array of
double);
var a:integer;
    g:double;
    panjang:integer;
begin
 g:=1;
 panjang:=high(win);
 for a:=0 to panjang do
  win[a]:=0;
  hanning:hanning_win(win);
case nflg of
  0:g:=1;
  1:begin
     g:=0;
     for a:=0 to panjang do
      g:=g+sqr(win[a]);
     g:=sqrt(g);
    end;
  2:begin
     g:=0;
     for a:=0 to panjang do
      g:=g+win[a];
    end;
 end;
 for a:=0 to panjang do
  win[a]:=win[a]/g;
end;
end.
```

<< LPC.PAS >>

```
{$N+,E+}    (* Allows code to use type 'double' and run on any iX86
machine *)
{$R-}       (* Turn off range checking...we violate array bounds
rules *)

unit lpc;

interface

uses math,main;
```

```
function LPCAnalisis(sinyal:array of
double;framelength,p:integer;var a:array of double):integer;
function MakeOrder(BandWith:integer):integer;
procedure lpc2cepstral(p1,p2:integer;a:array of double;var c:array
of double);
procedure weightingcepstral(p:integer;var c:array of double);


const M_PI:double=3.14159265358979323846;

implementation

function MakeOrder(BandWith:integer):integer;
begin
 result:=2*(BandWith div 1000+1);
end;

//----------------------------------------//
{
 fungsi autokorelasi untuk meminimalisasi mse dari LPC
 p -> order dari prediksi
 r -> koefisien autokorelasi
 frame_length -> panjang frame
 sinyal -> data sinyal
 hasil procedure adalah :
   koefisien autokorelasi
}
//----------------------------------------//

procedure autocorelation(sinyal:array of
double;frame_length,p:integer;var r:array of double);
var a,b:integer;
    temp :double;
begin
 for a:=0 to p do
  begin
   temp:=0;
   for b:=0 to frame_length-1-a do
    temp:=temp+sinyal[b]*sinyal[b+a];
   r[a]:=temp;
  end;
end;

//----------------------------------------//
{
 fungsi untuk mencari koefisien prediksi dari LPC
 r -> koefisien autokorelasi
 p -> order dari prediksi
 eps -> singular check
 kp -> koefisien prediksi
 hasil fungsi adalah :
 0 -> normaly completed
 1 -> abnormaly completed
 2 -> unstable LPC
}
//----------------------------------------//
```

```
function CariKoefisienPrediksi(r:array of
double;p:integer;eps:double;var kp:array of double):integer;
var rmd,mue :double;
    a,b,flag:integer;
    c       :array of double;

begin
 flag:=0;
 setlength(c,p+1);
 if eps<0.0 then  eps:=1.0e-6;
 rmd :=r[0];
 kp[0]:=0;
 for a:=1 to p do
  begin
   mue:= -r[a];
   for b:=1 to a-1 do
    mue:=mue - c[b] * r[a - b];
   mue:= mue / rmd;
   for b:=1 to a-1 do
    kp[b]:= c[b] + mue * c[a - b];
   kp[a]:=mue;
   rmd:=(1.0 - mue * mue) * rmd;
   if rmd<0 then
    rmd:=-rmd;
   if rmd<=eps then
    begin
     result:=1;
     exit;
    end;
   if mue<0 then
    mue:=-mue;
   if mue>=1 then
    flag:=2;
   for b:= 0 to a do
    c[b]:=kp[b];
  end;
  kp[0]:=sqrt(rmd);
  result:=flag;
end;


//----------------------------------------//
{
 prosedur untuk menganalisa LPC
 frame_length -> panjang frame
 sinyal       -> data sinyal
 p            -> order dari lpc
 a            -> koefisien dari lpc
 hasilnya adalah apakah lpc kita stabil atau nggak
}
//----------------------------------------//

function LPCAnalisis(sinyal:array of
double;framelength,p:integer;var a:array of double):integer;
var r       :array of double;
    flag,b,c:integer;
```

```
    temp    :double;
    sinpred :array of double;
begin
 setlength(r,p+1);
 setlength(sinpred,framelength);
 autocorelation(sinyal,framelength,p,r);
 flag:=CariKoefisienPrediksi(r,p,-1,a);
 for b:=1 to framelength-1 do
  begin
   temp:=0;
   for c:=1 to p do
    if b-c>=0 then
      temp:=temp+sinyal[b-c]*a[c];
   sinpred[b]:=temp;
  end;
 result:=flag;
end;

//-----------------------------------------//
{
 prosedur untuk mencari koefisien cepstral
 p1 -> order dari lpc
 p2 -> order dari cepstral
 a  -> koefisien dari lpc
 c  -> koefisien dari cepstral
 hasilnya adalah koefisien cepstral -> c
}
//-----------------------------------------//

procedure lpc2cepstral(p1,p2:integer;a:array of double;var c:array
of double);
var i,j,k :integer;
    temp  :double;
begin
 c[0]:=log10(a[0]);
 c[1]:=-a[1];
 for i:=2 to p2 do
  begin
   j:=i;
   if i>p1 then k:=i-p1
   else k:=1;
   temp:=0;
    repeat
     temp:=temp+k*c[k]*a[i-k];
     inc(k);
    until k>=j;
   c[i]:=-temp/i;
   if i<=p1 then c[i]:=c[i]-a[i];
  end;
end;

//-----------------------------------------//
{
 prosedur untuk pembobotan koefisien cepstral untuk mengurangi
sensitivitas
 p -> order dari cepstral
```

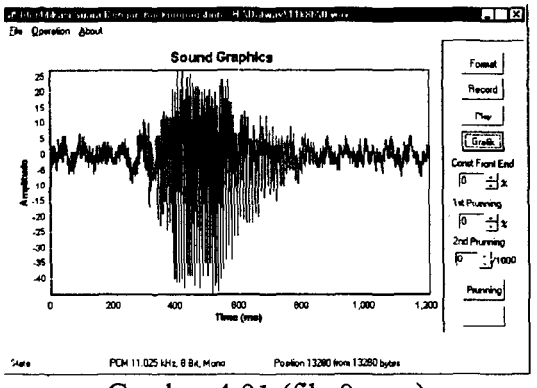
```
 c  -> koefisien dari cepstral
 hasilnya adalah koefisien cepstral yang telah di boboti -> c
}
//----------------------------------------//
procedure weightingcepstral(p:integer;var c:array of double);
var a:integer;
    w:array of double;
    arg:double;
begin
 setlength(w,p+1);
 arg:=M_PI/p;
 for a:=1 to p do
  w[a]:=1+(p/2)*sin(a*arg);
 for a:=1 to p do
  c[a]:=c[a]*w[a];
end;
end.
```

<< BACKPRO.PAS >>

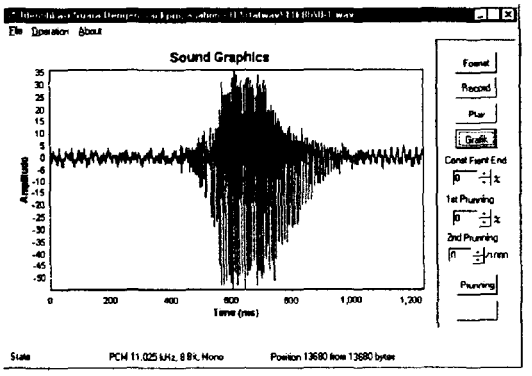
```
{$N+,E+}    (* Allows code to use type 'double' and run on any iX86
machine *)
{$R-}       (* Turn off range checking...we violate array bounds
rules *)

unit backpro;

interface
uses math,main;


implementation

procedure RandomBobot(var bobot:Tdatabobot);
var a,b:integer;
begin
 for a:=0 to high(bobot) do
  for b:=0 to high(bobot[a]) do
   bobot[a,b]:=random-0.5;
end;

procedure NguyenWidrow(var bobot:tdatabobot);
var beta:double;
    a,b :integer;
    old :double;
    UnitInput,UnitHiden:integer;
begin
 UnitInput:=high(bobot);
 UnitHiden:=high(bobot[0]);
 beta:=0.7*(power(UnitHiden,1/UnitInput));
 for a:=1 to UnitHiden do
  begin
   old:=0;
   for b:=1 to UnitInput-1 do
     old:=old+sqr(bobot[b,a]);
   old:=sqrt(old);
   for b:=1 to UnitInput-1 do
```

```
      bobot[b,a]:=beta*bobot[b,a]/old;
    bobot[0,a]:=beta*(1-2*random);
  end;
end;

function sigmoid(nilai:real):real ;
begin
 result:=1/(1+(exp(-nilai)));
end;

function TurunanSigmoid(nilai:real):real ;
begin
 result:= sigmoid(nilai)*(1-sigmoid(nilai));
end;

procedure InisialisasiBobot(var vbobot:array of Tdatabobot;var
wbobot:Tdatabobot);
var a:integer;
begin
 RandomBobot(vbobot[0]);
 NguyenWidrow(vbobot[0]);
 if high(vbobot)>0 then
  for a:=1 to high(vbobot) do
    RandomBobot(vbobot[a]);
 RandomBobot(wbobot);
end;

procedure LayerIn(prev:array of double;var
next:T1dimensi;bobot:Tdatabobot);
var a,b:integer;
begin
 for a:=1 to high(next) do
  begin
   next[a]:=bobot[0,a];
   for b:=1 to high(prev) do
    next[a]:=next[a]+bobot[b,a]*prev[b];
  end;     end;
procedure FungsiAktivasi(inp:array of double;var hasil:array of
double);
var a:integer;
begin
 for a:=1 to high(hasil) do
  hasil[a]:=sigmoid(inp[a]);
end;

procedure CalculateOutputEror(target,output,out_in:array of
double;var eror_k:array of double);
var a:integer;
begin
 for a:=1 to high(target) do
  eror_k[a]:=(target[a]-output[a])*TurunanSigmoid(out_in[a]);
end;

procedure CalculateHidenEror(eror_next,hiden_in:array of
double;bobot:Tdatabobot;var eror_j:array of double);
var a,b:integer;
```

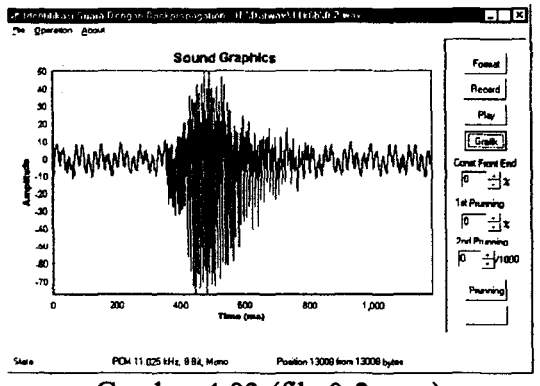
```
    eror_in:double;
begin
 for a:=1 to high(bobot) do
  begin
   eror_in:=0;
   for b:=1 to high(bobot[a]) do
    eror_in:=eror_in+eror_next[b]*bobot[a,b];
   eror_j[a]:=eror_in*TurunanSigmoid(hiden_in[a]);
  end;
end;

procedure UpdateBobot(alpha:double;eror_next,prev_data:array of
double;var bobot:Tdatabobot);
var a,b:integer;
begin
 for a:=0 to high(bobot) do
  for b:=0 to high(bobot[a]) do
    bobot[a,b]:=bobot[a,b]+alpha*eror_next[b]*prev_data[a];
end;
end.
```



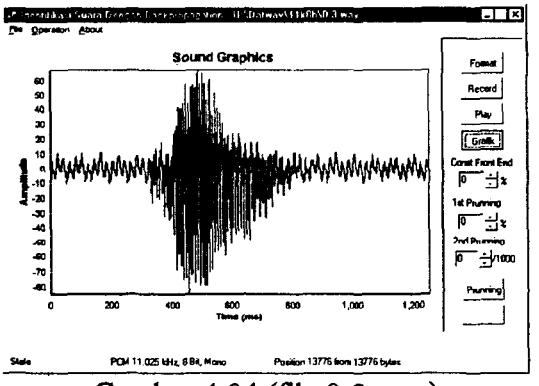
Gambar 4.01 (file 0.wav)



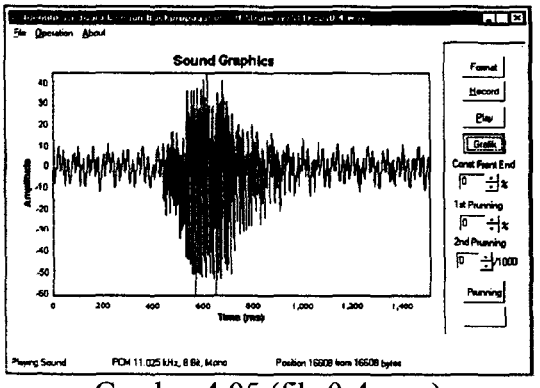
Gambar 4.02 (file 0-1.wav)



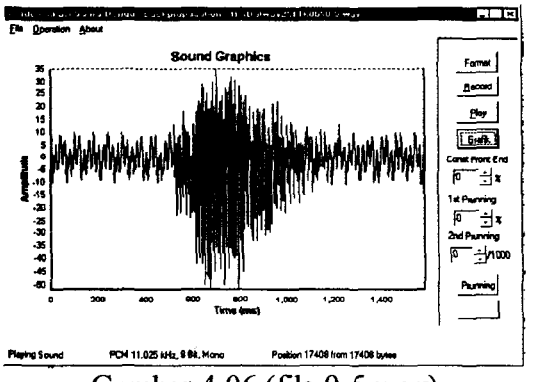
Gambar 4.03 (file 0-2.wav)



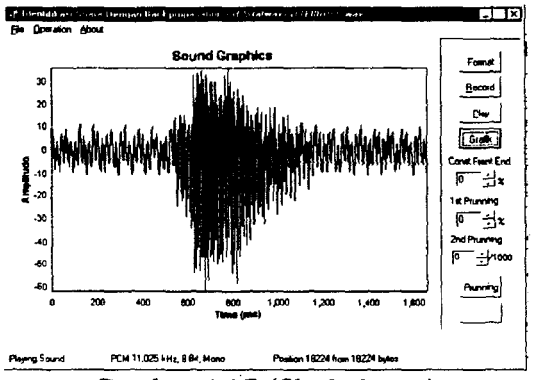
Gambar 4.04 (file 0-3.wav)



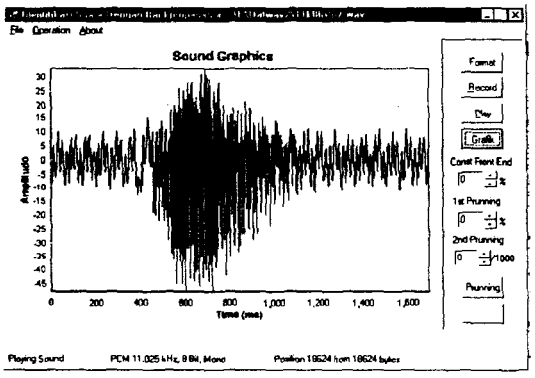
Gambar 4.05 (file 0-4.wav)



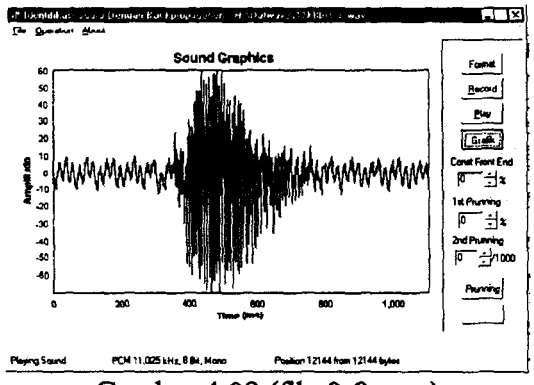
Gambar 4.06 (file 0-5.wav)



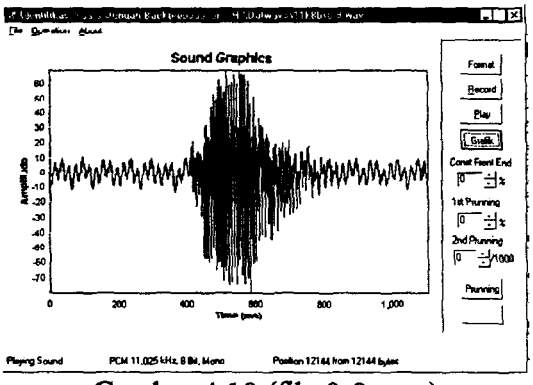
Gambar 4.07 (file 0-6.wav)



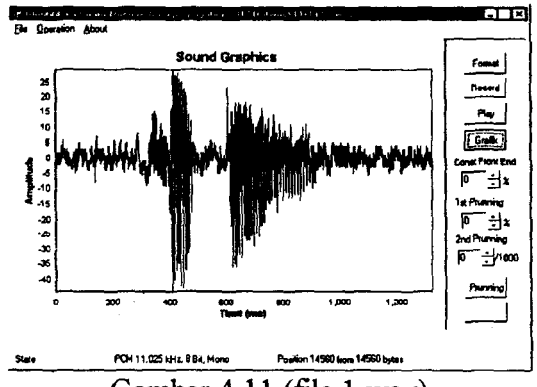
Gambar 4.08 (file 0-7.wav)



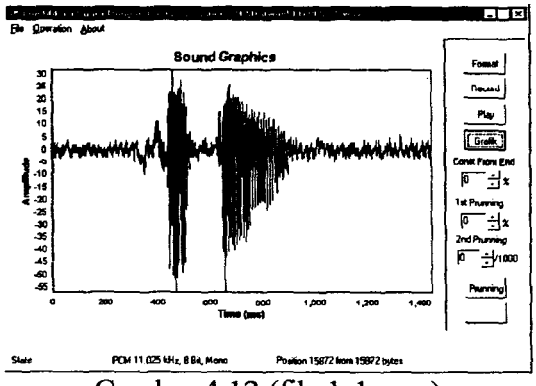
Gambar 4.09 (file 0-8.wav)



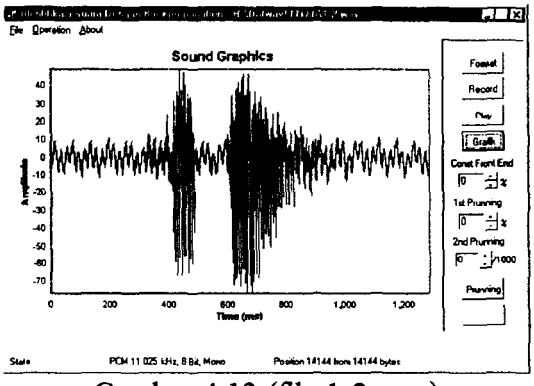
Gambar 4.10 (file 0-9.wav)



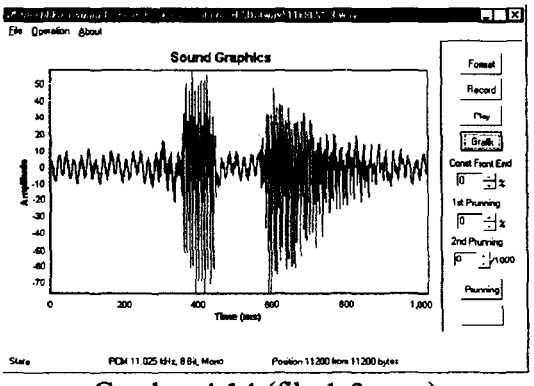
Gambar 4.11 (file 1.wav)



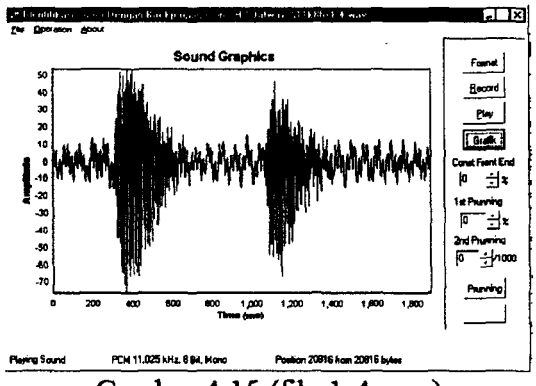
Gambar 4.12 (file 1-1.wav)



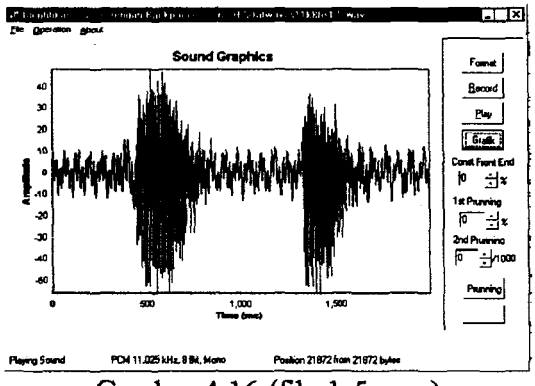
Gambar 4.13 (file 1-2.wav)



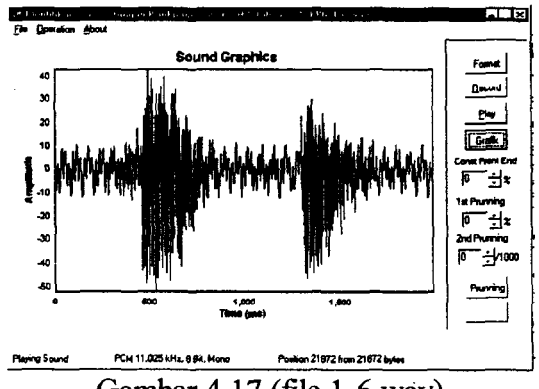
Gambar 4.14 (file 1-3.wav)



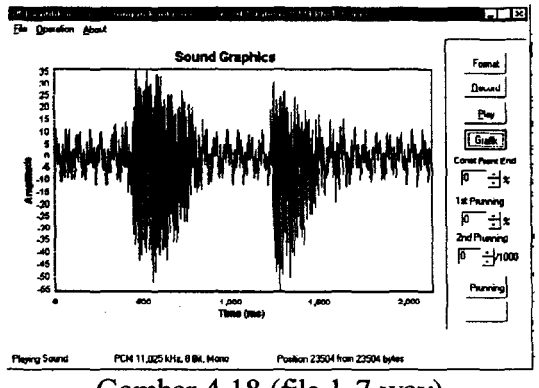
Gambar 4.15 (file 1-4.wav)



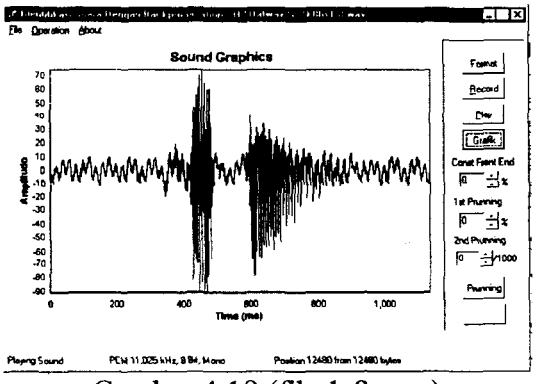
Gambar 4.16 (file 1-5.wav)



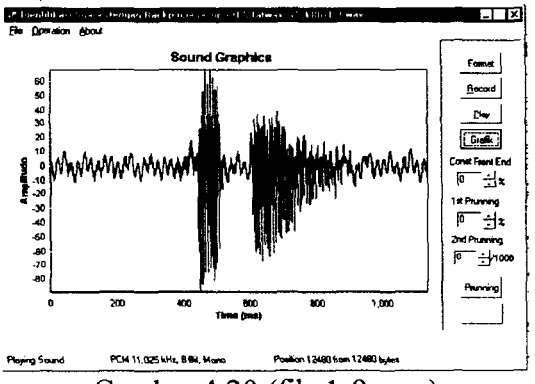
Gambar 4.17 (file 1-6.wav)



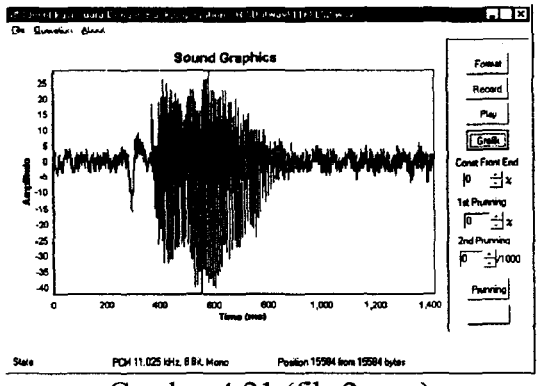
Gambar 4.18 (file 1-7.wav)



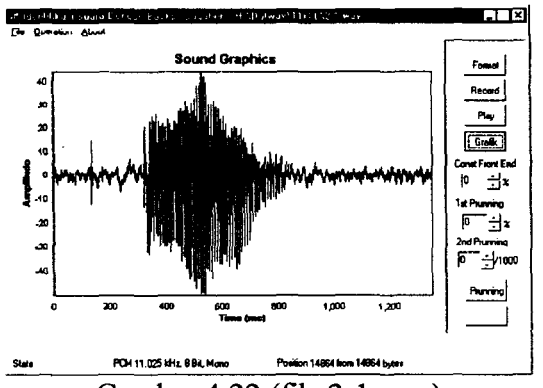
Gambar 4.19 (file 1-8.wav)



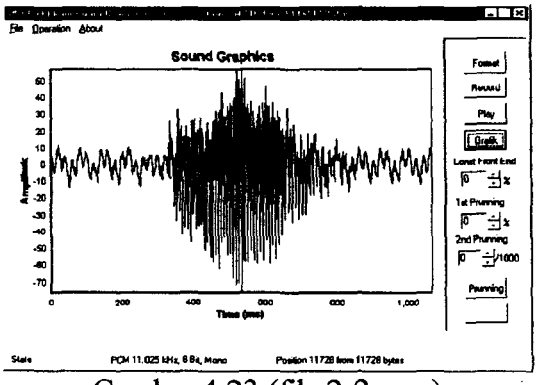
Gambar 4.20 (file 1-9.wav)



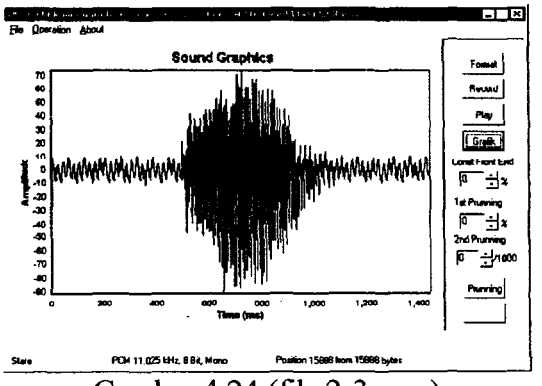
Gambar 4.21 (file 2.wav)



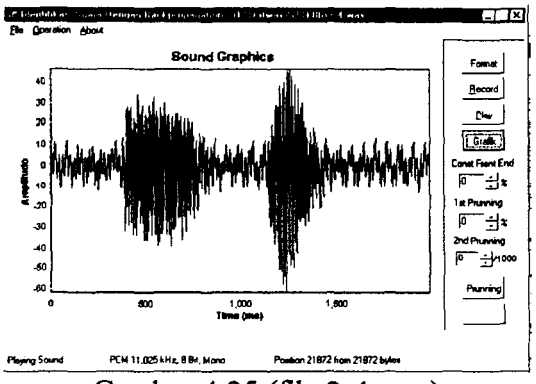
Gambar 4.22 (file 2-1.wav)



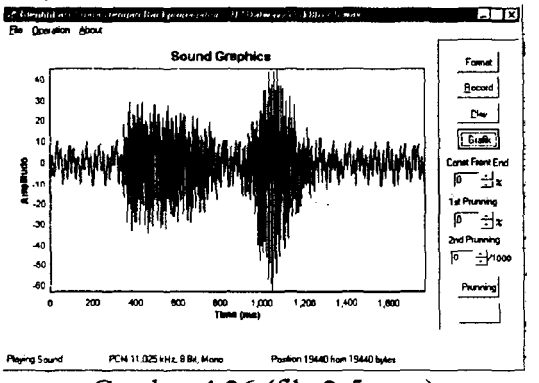
Gambar 4.23 (file 2-2.wav)



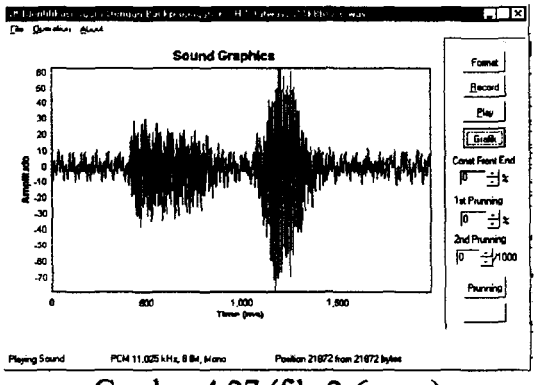
Gambar 4.24 (file 2-3.wav)



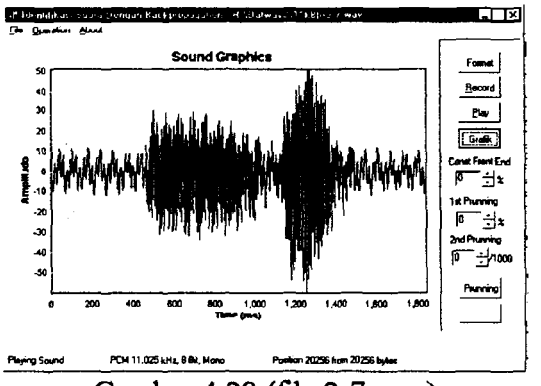
Gambar 4.25 (file 2-4.wav)



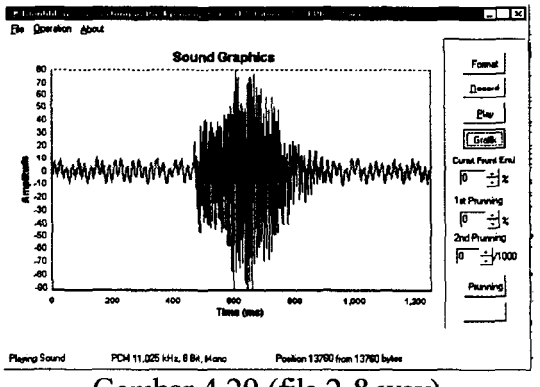
Gambar 4.26 (file 2-5.wav)



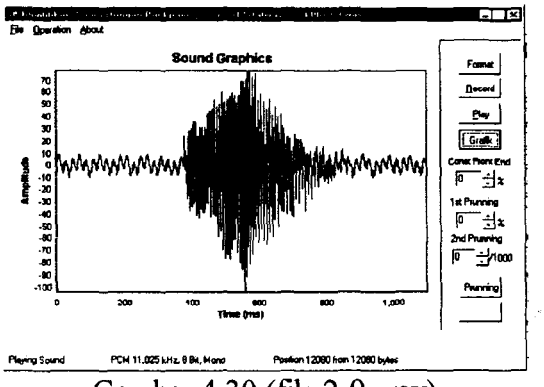
Gambar 4.27 (file 2-6.wav)



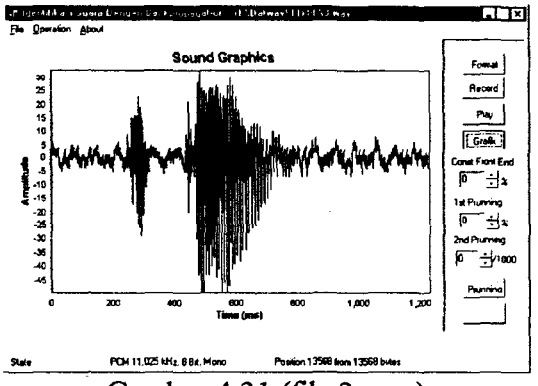
Gambar 4.28 (file 2-7.wav)



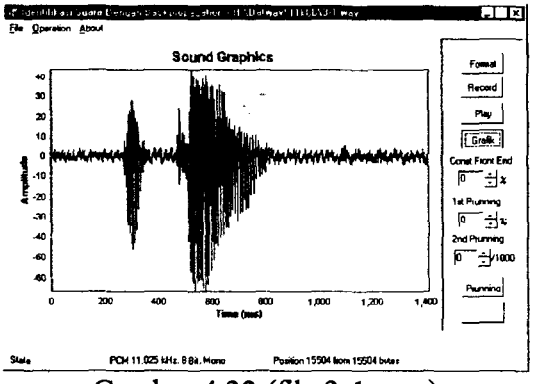
Gambar 4.29 (file 2-8.wav)



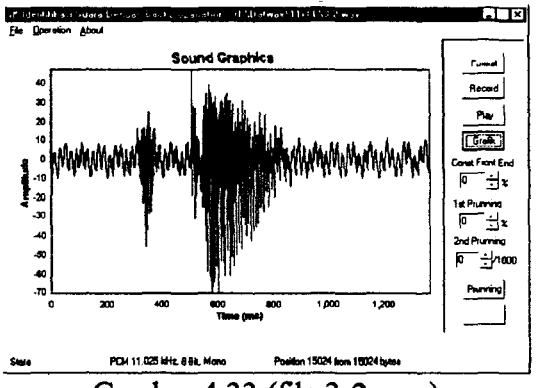
Gambar 4.30 (file 2-9.wav)



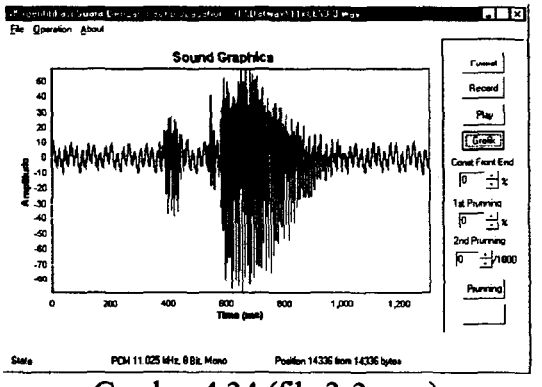
Gambar 4.31 (file 3.wav)



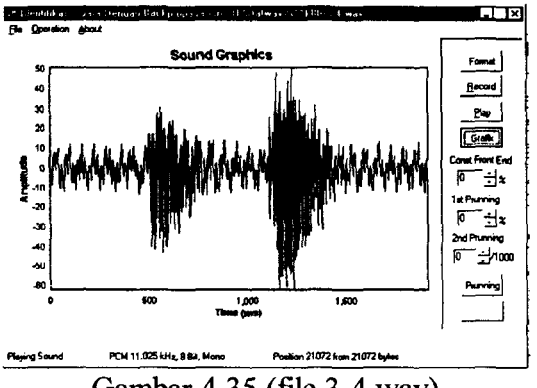
Gambar 4.32 (file 3-1.wav)



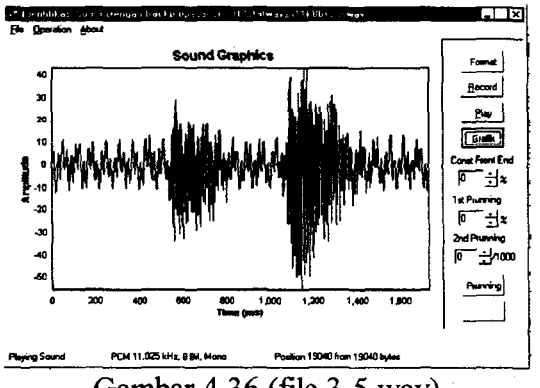
Gambar 4.33 (file 3-2.wav)



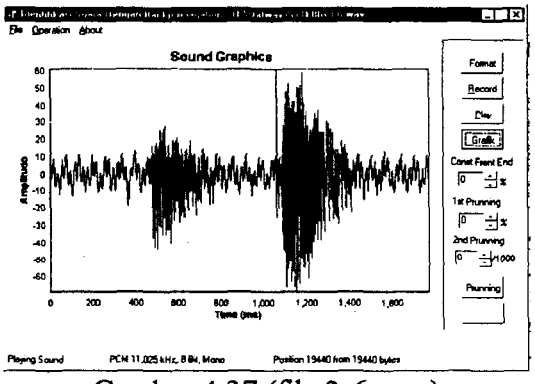
Gambar 4.34 (file 3-3.wav)



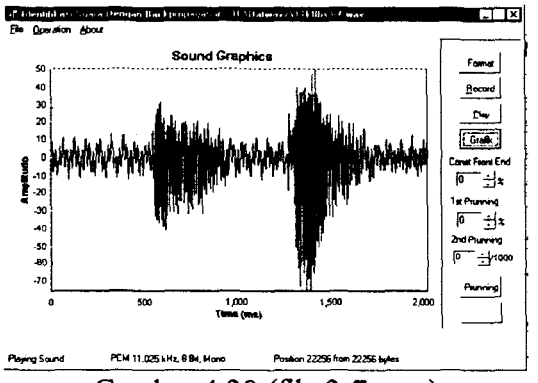
Gambar 4.35 (file 3-4.wav)



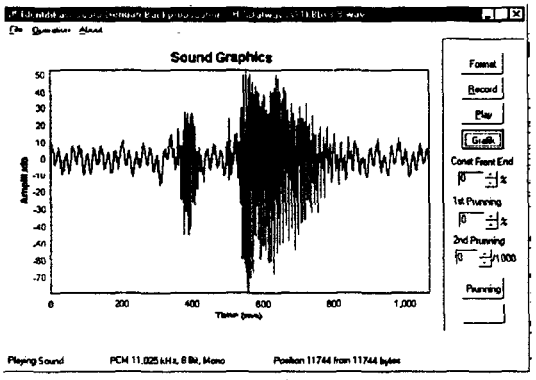
Gambar 4.36 (file 3-5.wav)



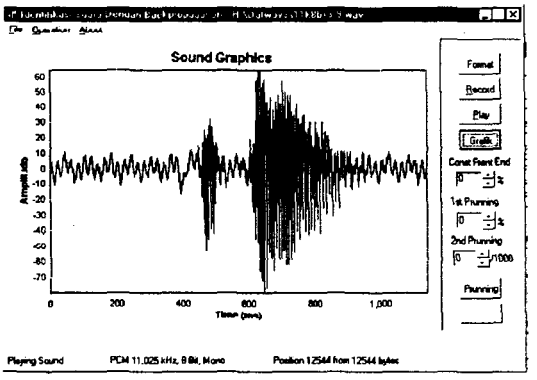
Gambar 4.37 (file 3-6.wav)



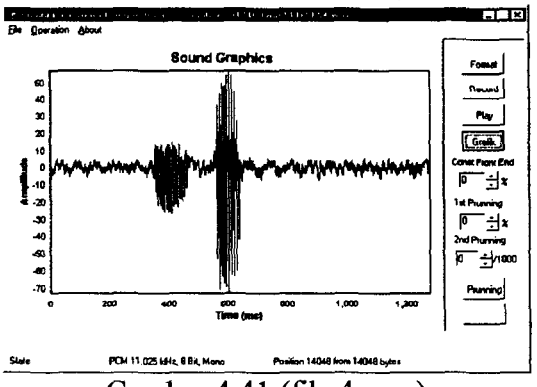
Gambar 4.38 (file 3-7.wav)



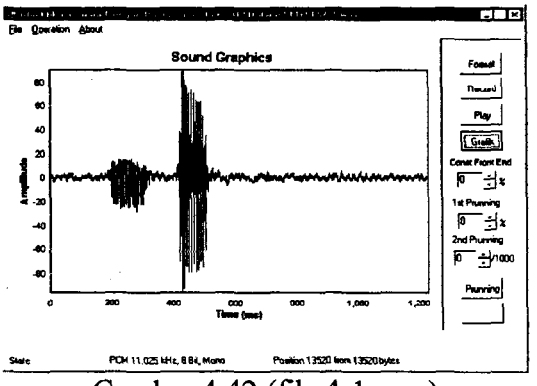
Gambar 4.39 (file 3-8.wav)



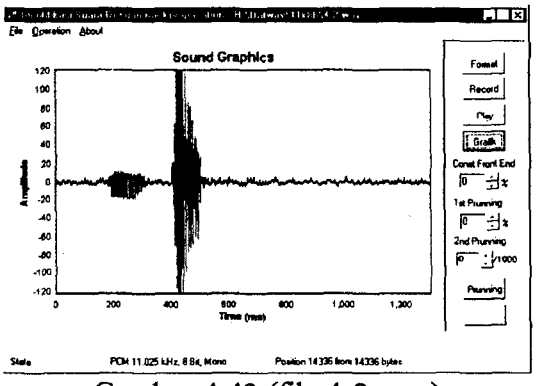
Gambar 4.40 (file 3-9.wav)



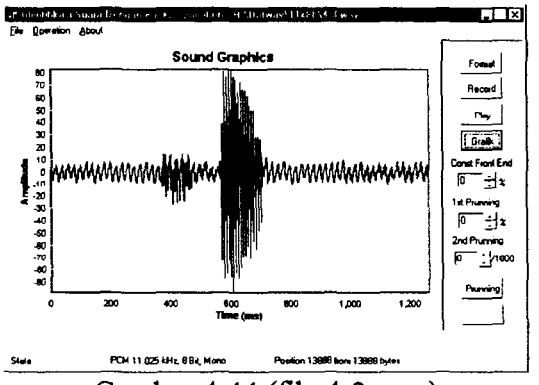
Gambar 4.41 (file 4.wav)



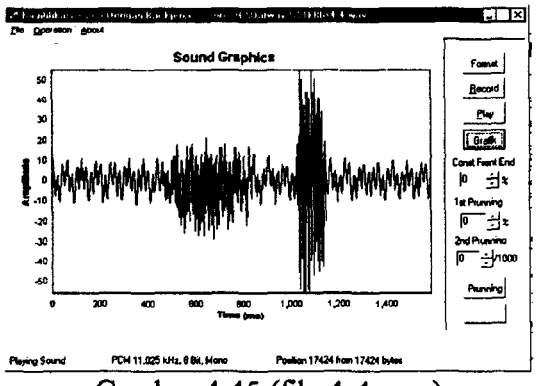
Gambar 4.42 (file 4-1.wav)



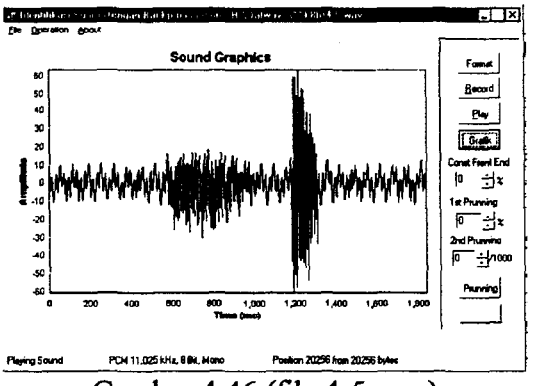
Gambar 4.43 (file 4-2.wav)



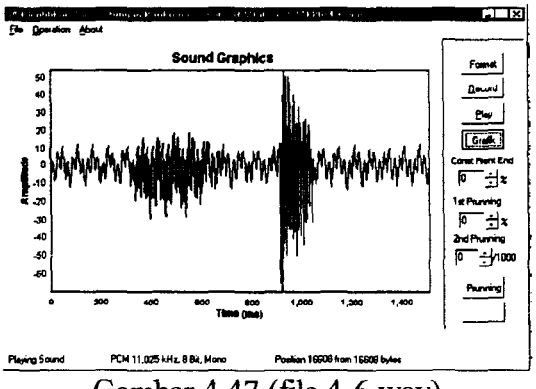
Gambar 4.44 (file 4-3.wav)



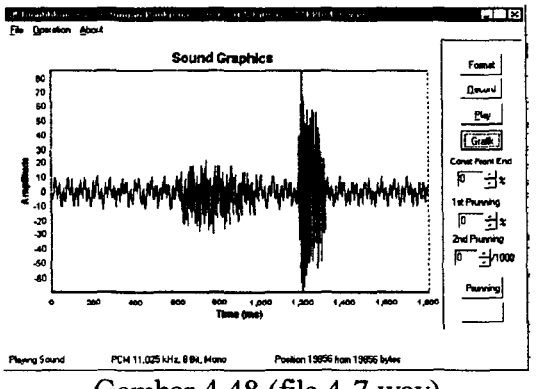
Gambar 4.45 (file 4-4.wav)



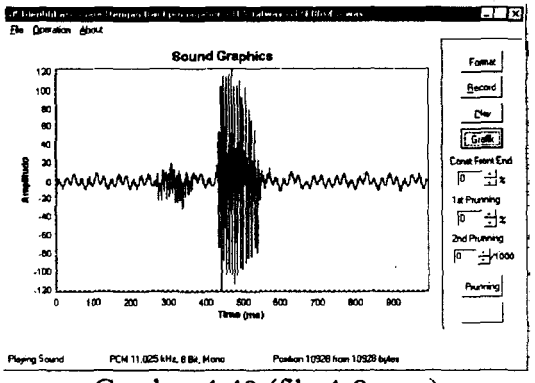
Gambar 4.46 (file 4-5.wav)



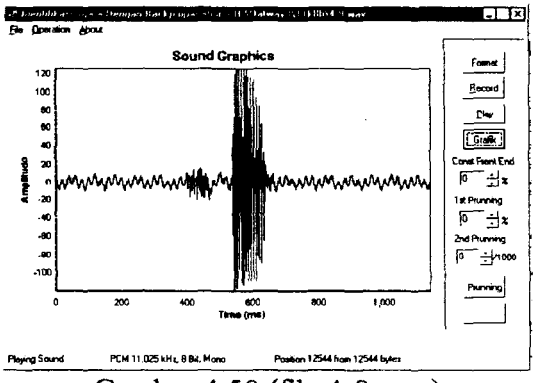
Gambar 4.47 (file 4-6.wav)



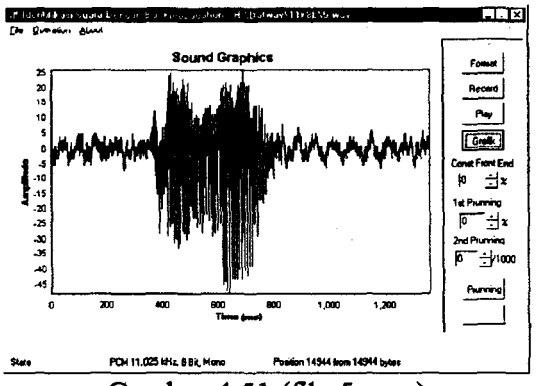
Gambar 4.48 (file 4-7.wav)



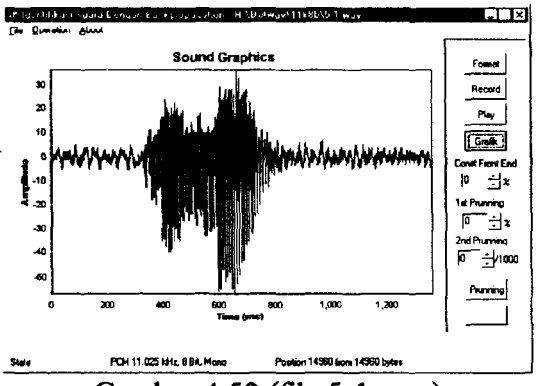
Gambar 4.49 (file 4-8.wav)



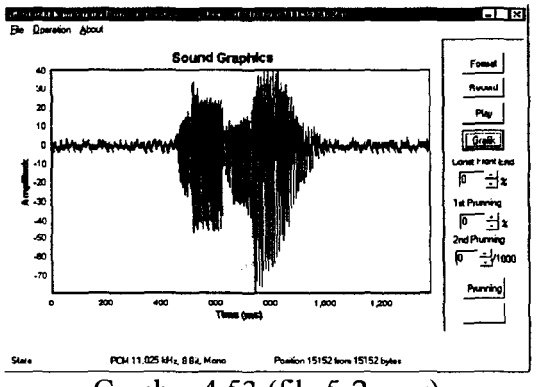
Gambar 4.50 (file 4-9.wav)



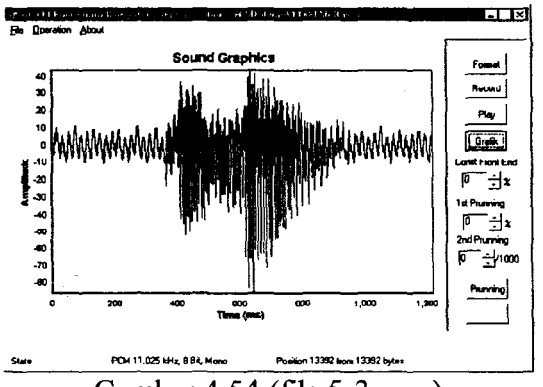
Gambar 4.51 (file 5.wav)



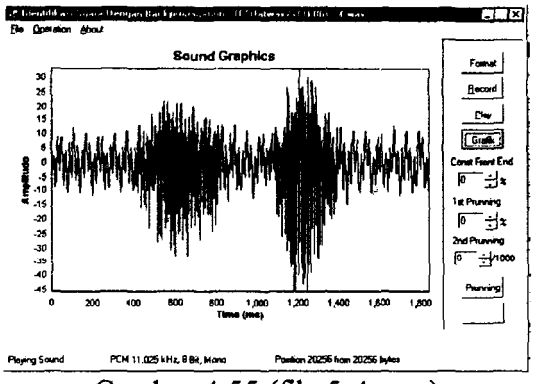
Gambar 4.52 (file 5-1.wav)



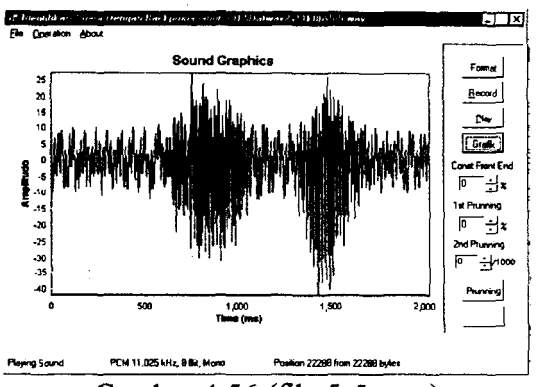
Gambar 4.53 (file 5-2.wav)



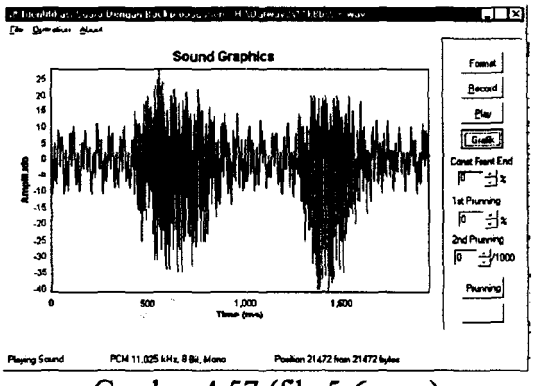
Gambar 4.54 (file 5-3.wav)



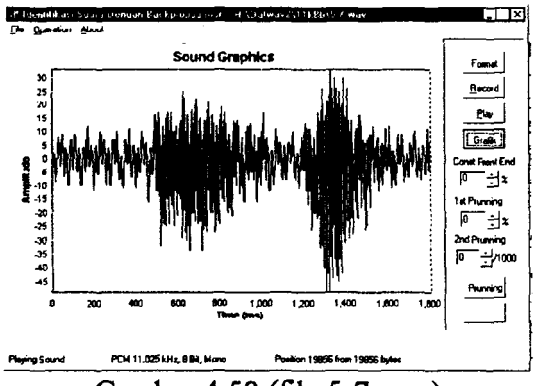
Gambar 4.55 (file 5-4.wav)



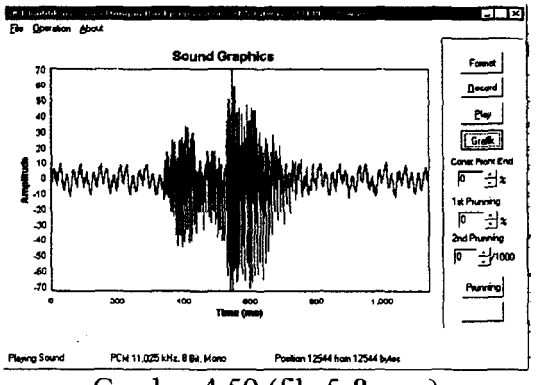
Gambar 4.56 (file 5-5.wav)



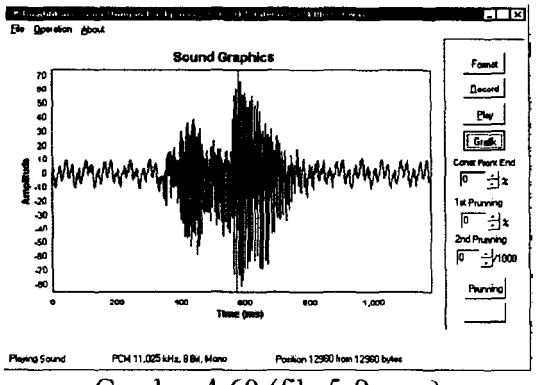
Gambar 4.57 (file 5-6.wav)



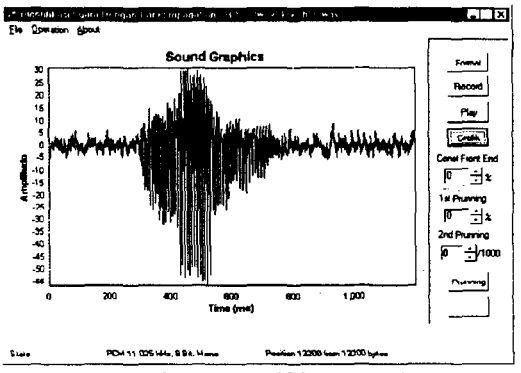
Gambar 4.58 (file 5-7.wav)



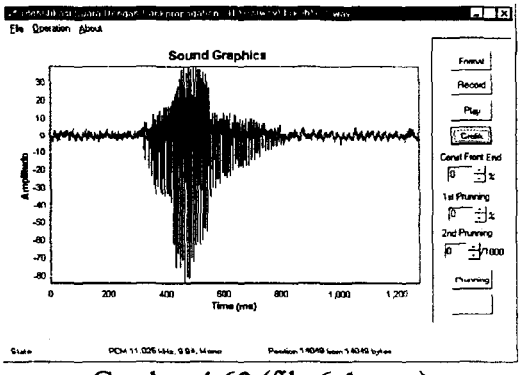
Gambar 4.59 (file 5-8.wav)



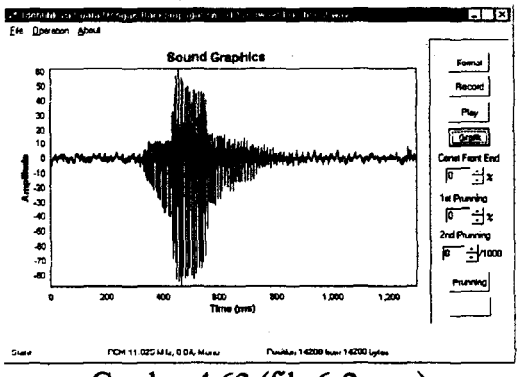
Gambar 4.60 (file 5-9.wav)



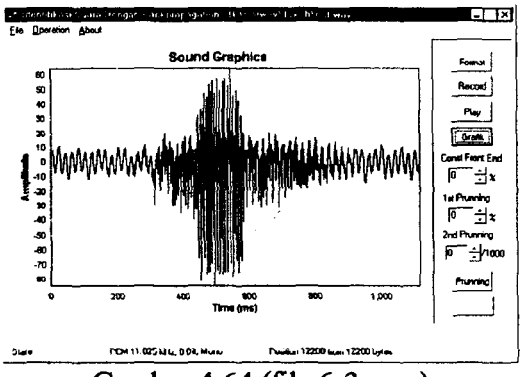
Gambar 4.61 (file 6.wav)



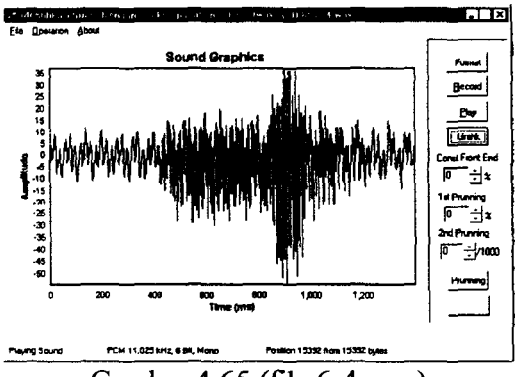
Gambar 4.62 (file 6-1.wav)



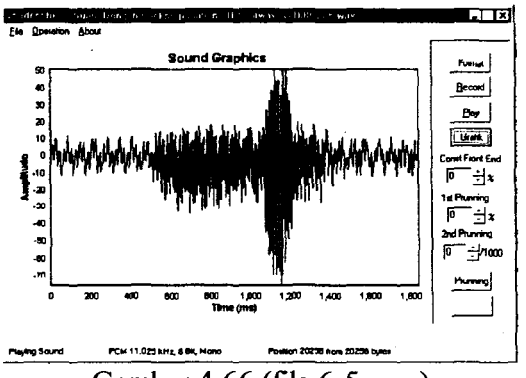
Gambar 4.63 (file 6-2.wav)



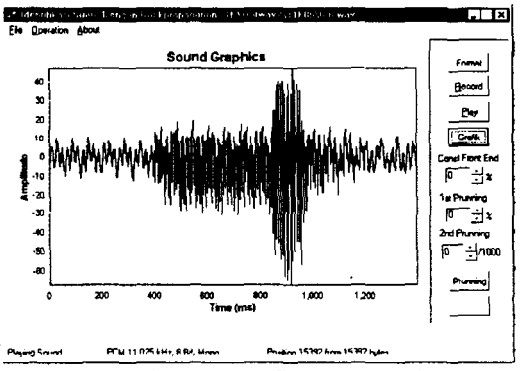
Gambar 4.64 (file 6-3.wav)



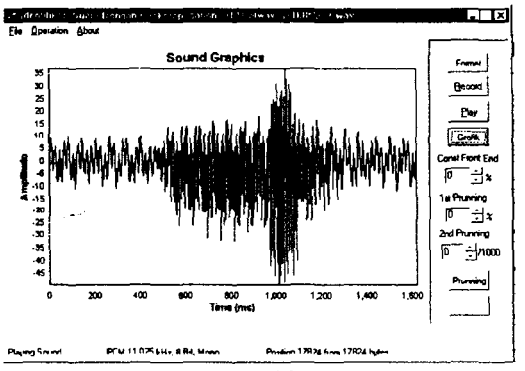
Gambar 4.65 (file 6-4.wav)



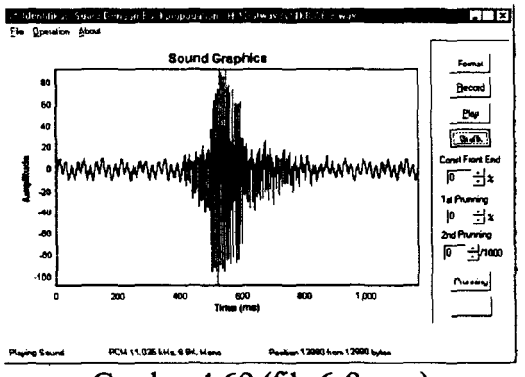
Gambar 4.66 (file 6-5.wav)



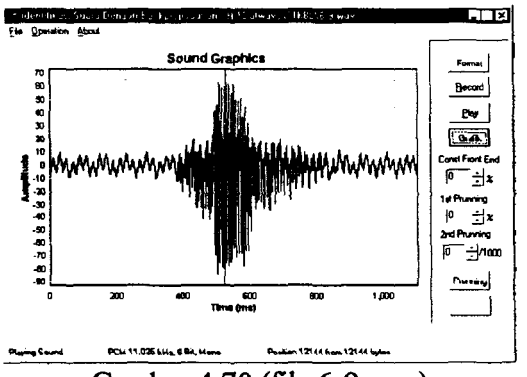
Gambar 4.67 (file 6-6.wav)



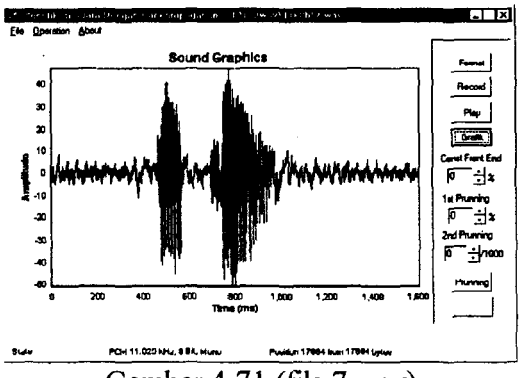
Gambar 4.68 (file 6-7.wav)



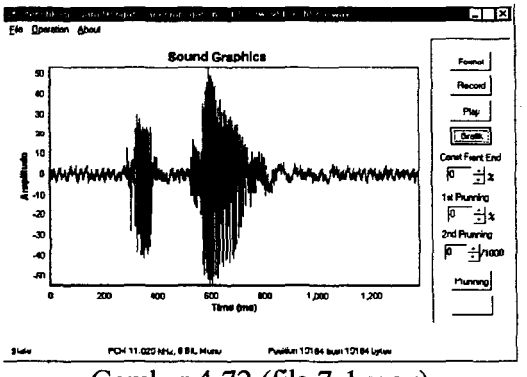
Gambar 4.69 (file 6-8.wav)



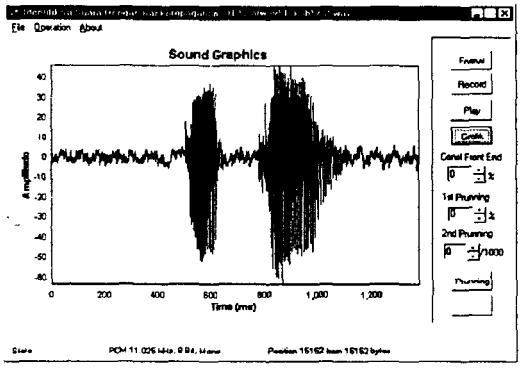
Gambar 4.70 (file 6-9.wav)



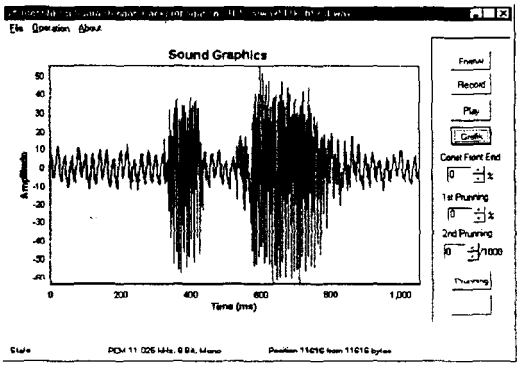
Gambar 4.71 (file 7.wav)



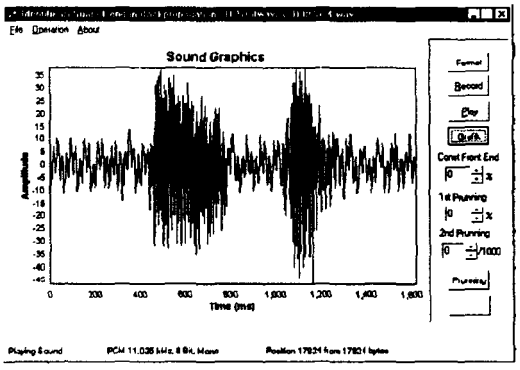
Gambar 4.72 (file 7-1.wav)



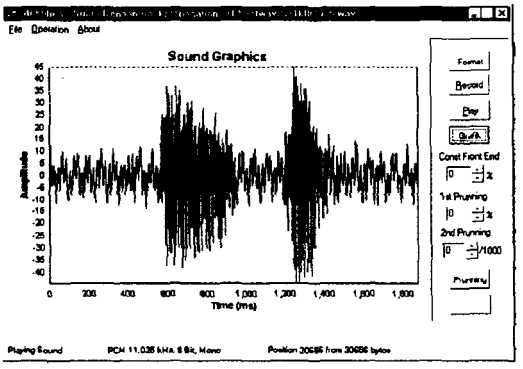
Gambar 4.73 (file 7-2.wav)



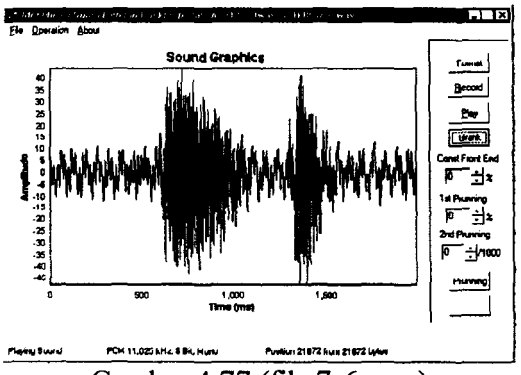
Gambar 4.74 (file 7-3.wav)



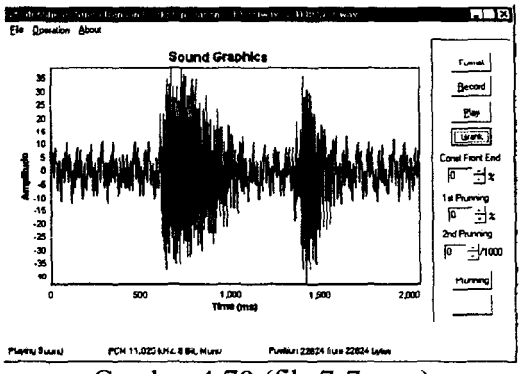
Gambar 4.75 (file 7-4.wav)



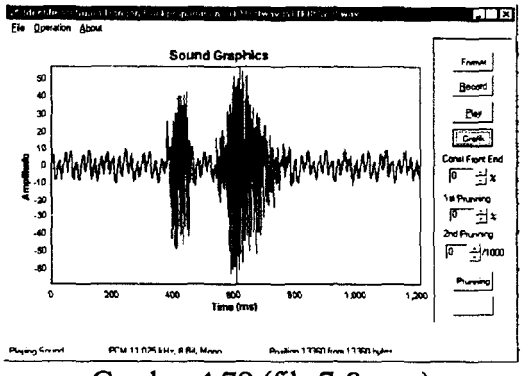
Gambar 4.76 (file 7-5.wav)



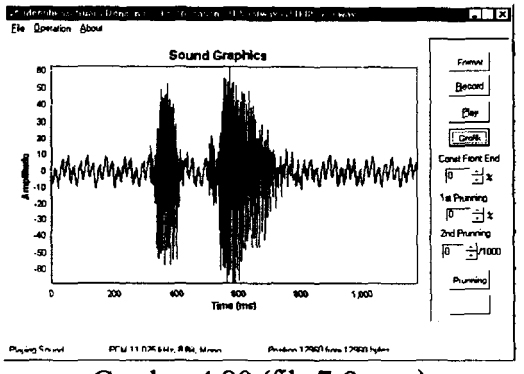
Gambar 4.77 (file 7-6.wav)



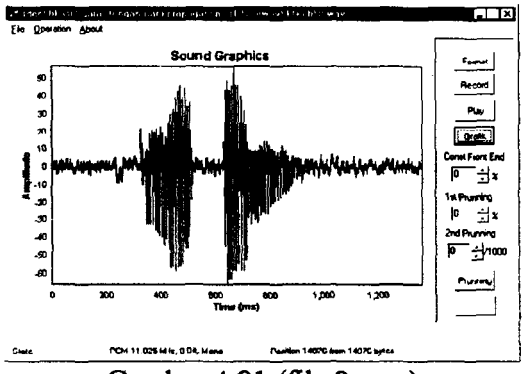
Gambar 4.78 (file 7-7.wav)



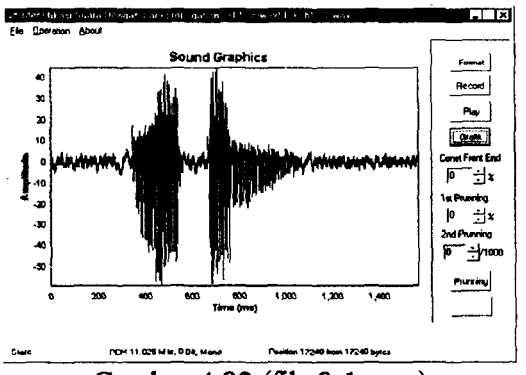
Gambar 4.79 (file 7-8.wav)



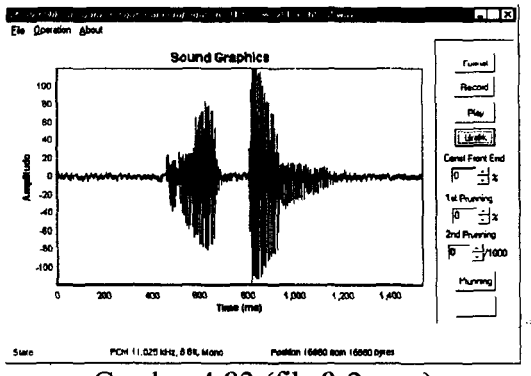
Gambar 4.80 (file 7-9.wav)



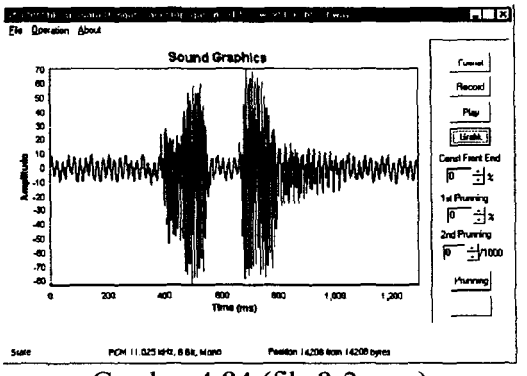
Gambar 4.81 (file 8.wav)



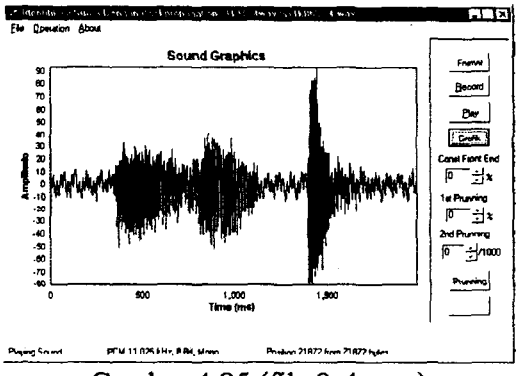
Gambar 4.82 (file 8-1.wav)



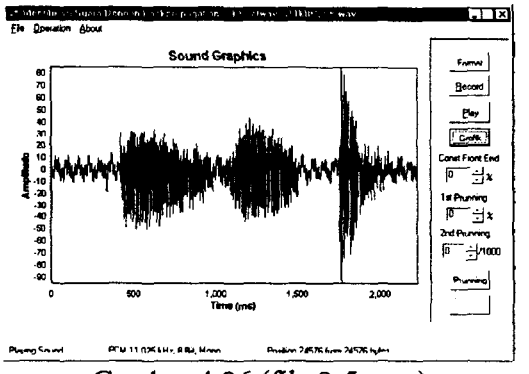
Gambar 4.83 (file 8-2.wav)



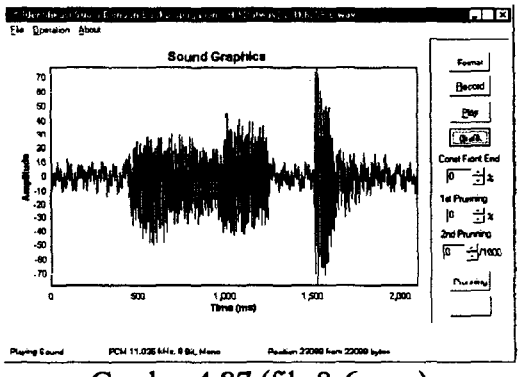
Gambar 4.84 (file 8-3.wav)



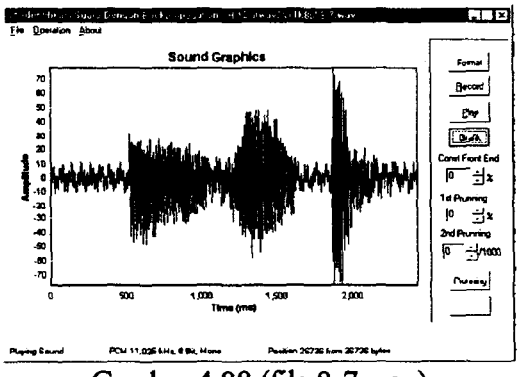
Gambar 4.85 (file 8-4.wav)



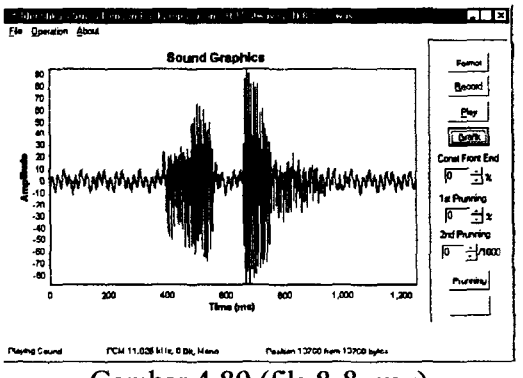
Gambar 4.86 (file 8-5.wav)



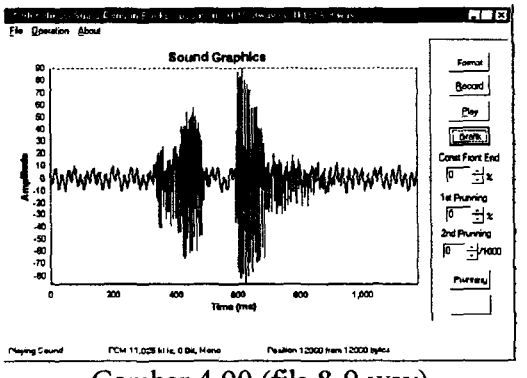
Gambar 4.87 (file 8-6.wav)



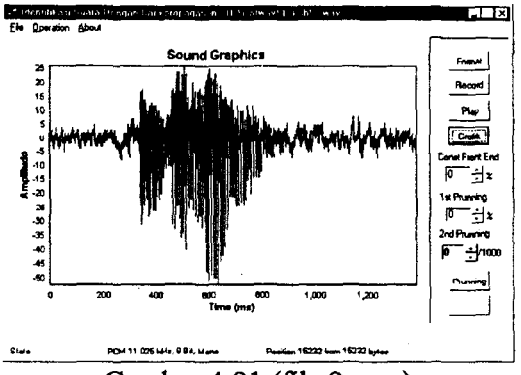
Gambar 4.88 (file 8-7.wav)



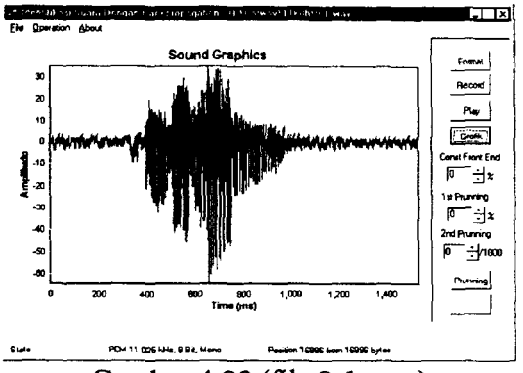
Gambar 4.89 (file 8-8.wav)



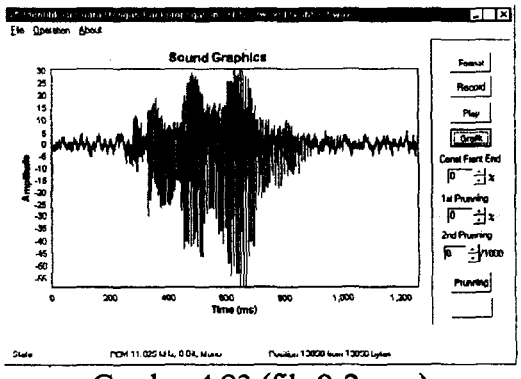
Gambar 4.90 (file 8-9.wav)



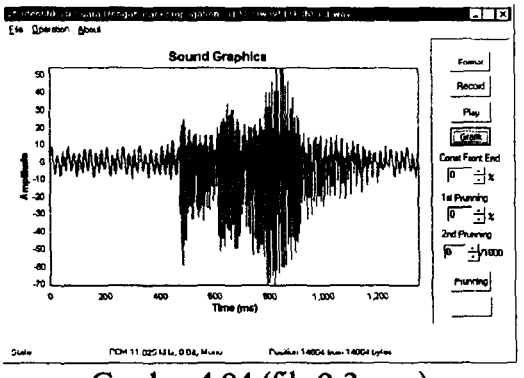
Gambar 4.91 (file 9.wav)



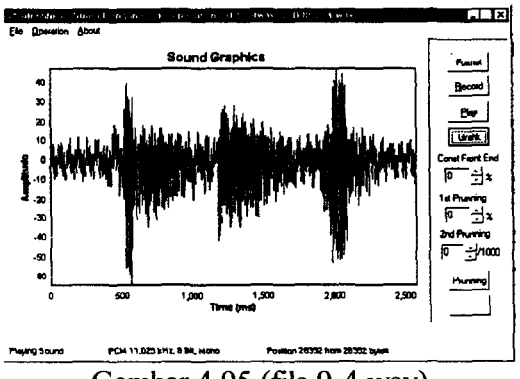
Gambar 4.92 (file 9-1.wav)



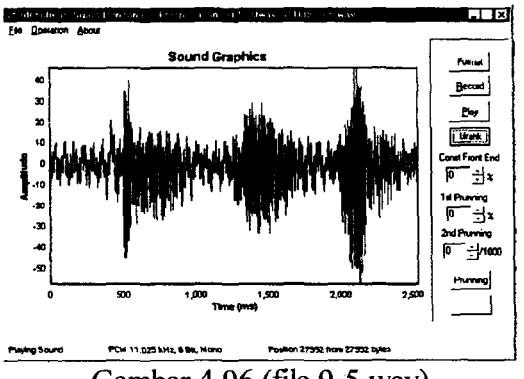
Gambar 4.93 (file 9-2.wav)



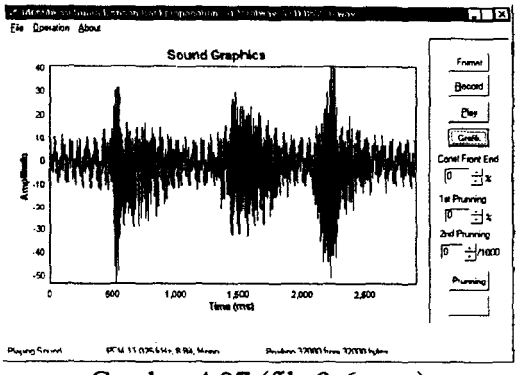
Gambar 4.94 (file 9-3.wav)



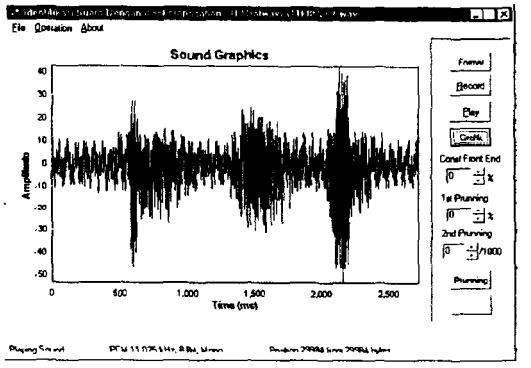
Gambar 4.95 (file 9-4.wav)



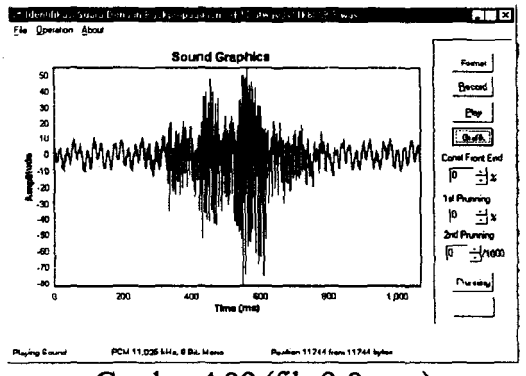
Gambar 4.96 (file 9-5.wav)



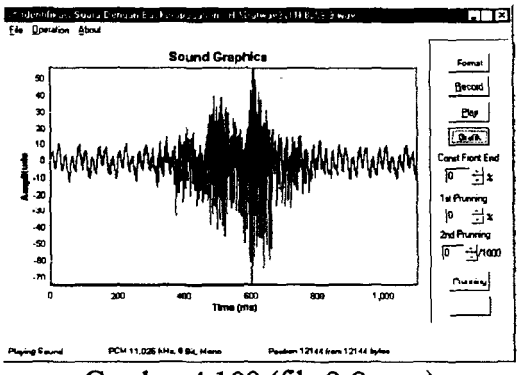
Gambar 4.97 (file 9-6.wav)



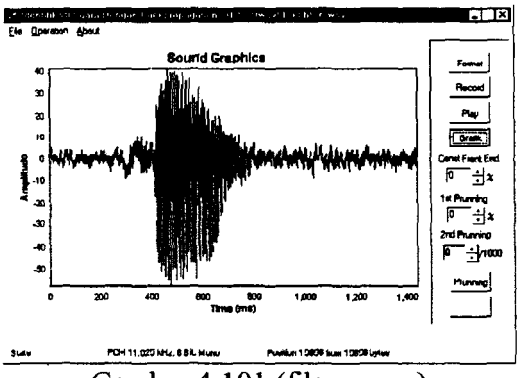
Gambar 4.98 (file 9-7.wav)



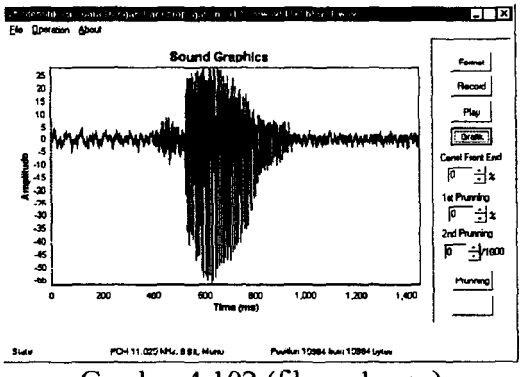
Gambar 4.99 (file 9-8.wav)



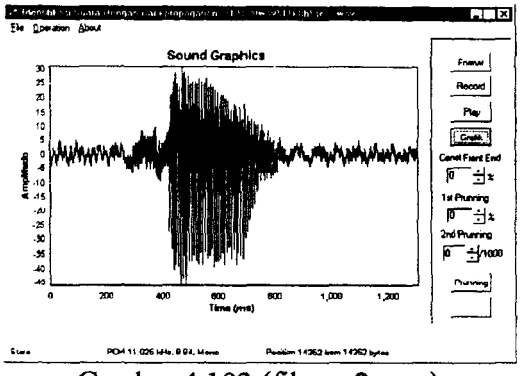
Gambar 4.100 (file 9-9.wav)



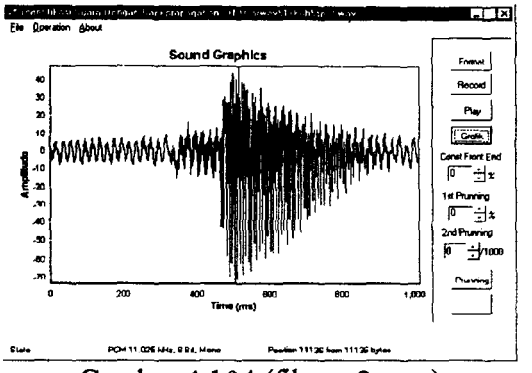
Gambar 4.101 (file se.wav)



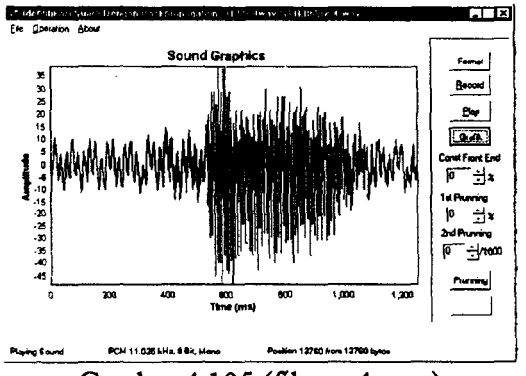
Gambar 4.102 (file se-1.wav)



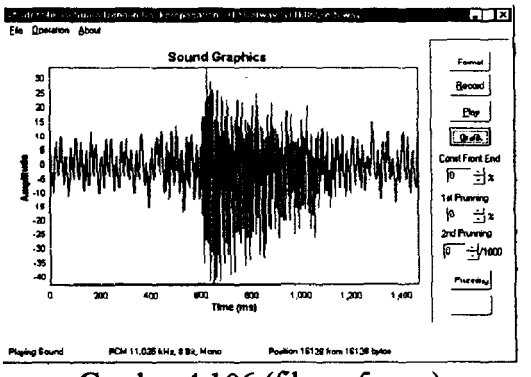
Gambar 4.103 (file se-2.wav)



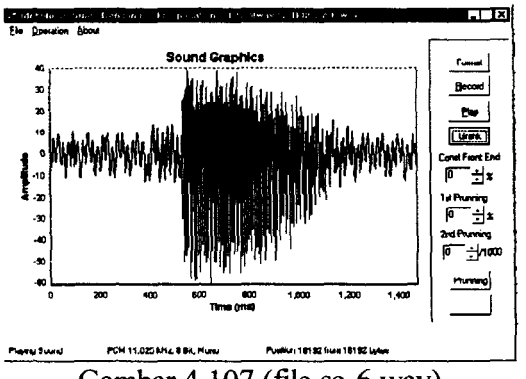
Gambar 4.104 (file se-3.wav)



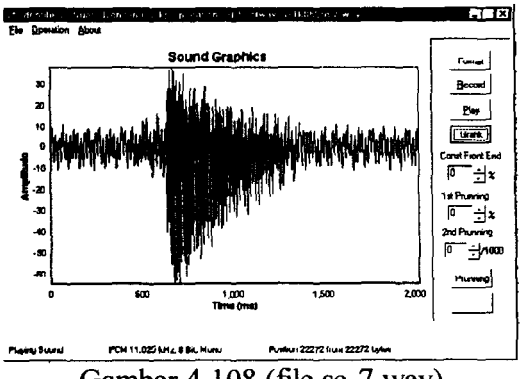
Gambar 4.105 (file se-4.wav)



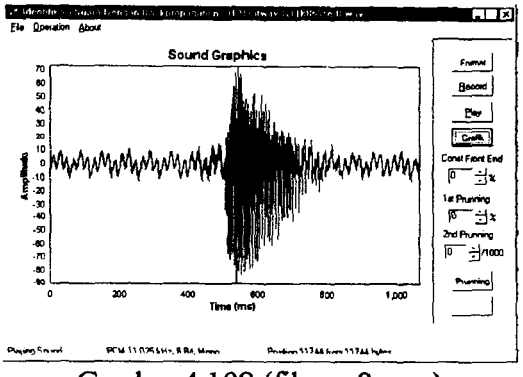
Gambar 4.106 (file se-5.wav)



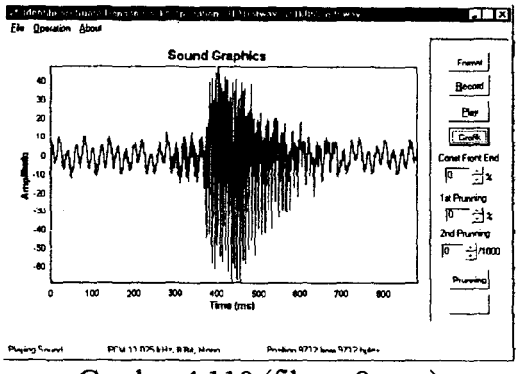
Gambar 4.107 (file se-6.wav)



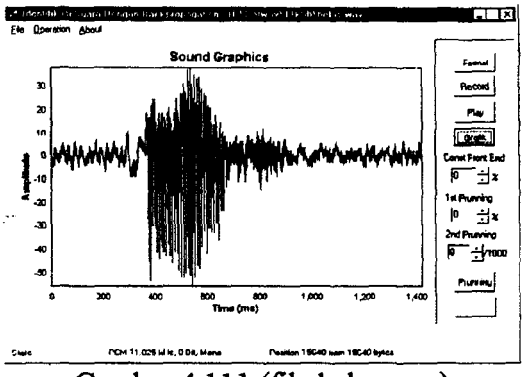
Gambar 4.108 (file se-7.wav)



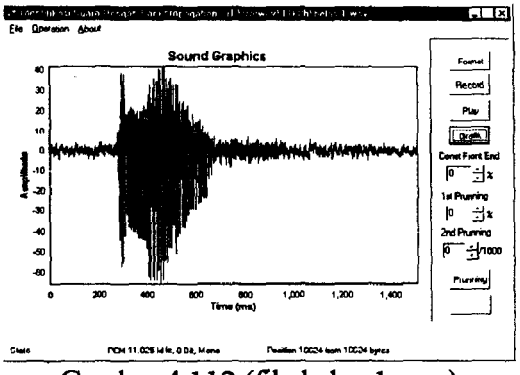
Gambar 4.109 (file se-8.wav)



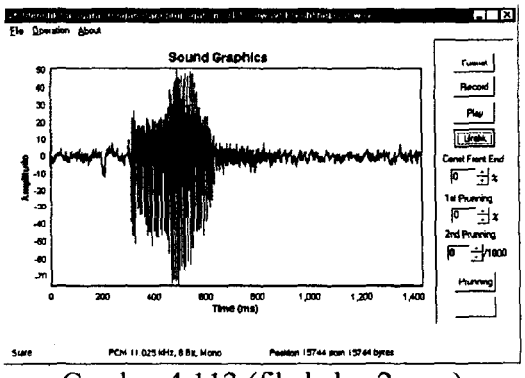
Gambar 4.110 (file se-9.wav)



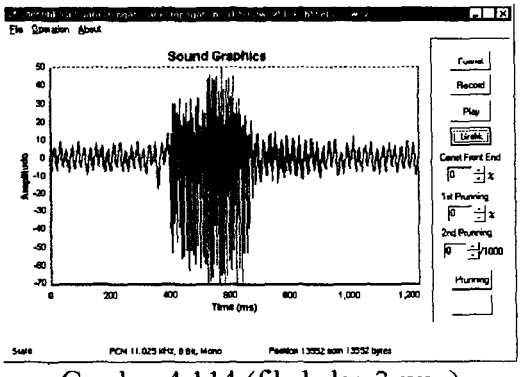
Gambar 4.111 (file belas.wav)



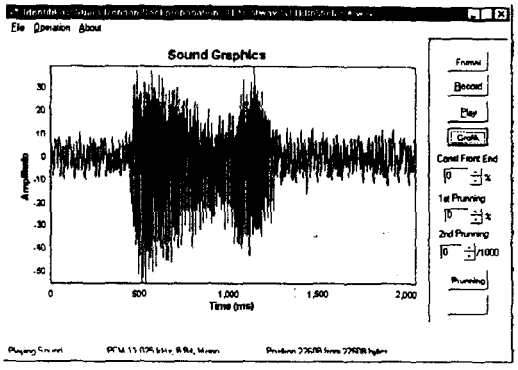
Gambar 4.112 (file belas-1.wav)



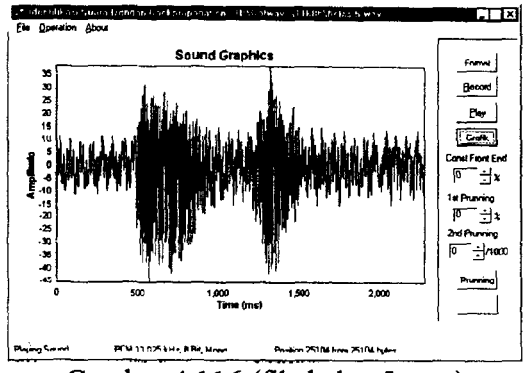
Gambar 4.113 (file belas-2.wav)



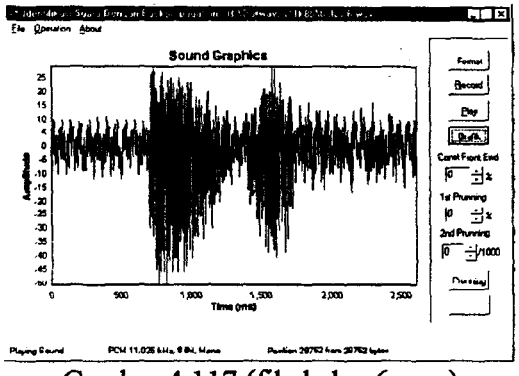
Gambar 4.114 (file belas-3.wav)



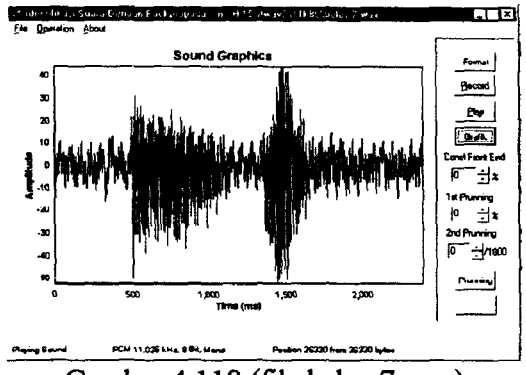
Gambar 4.115 (file belas-4.wav)



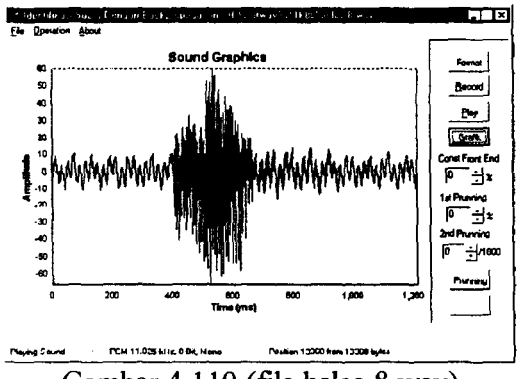
Gambar 4.116 (file belas-5.wav)



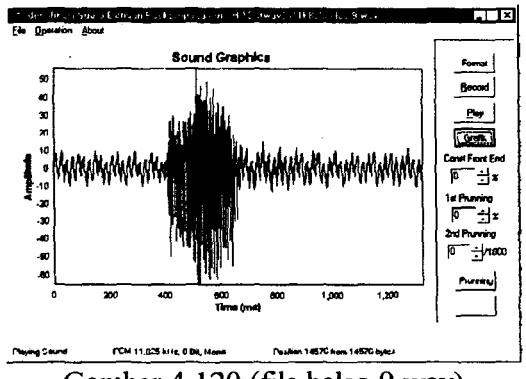
Gambar 4.117 (file belas-6.wav)



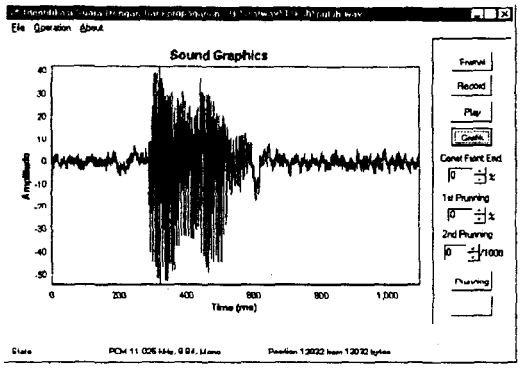
Gambar 4.118 (file belas-7.wav)



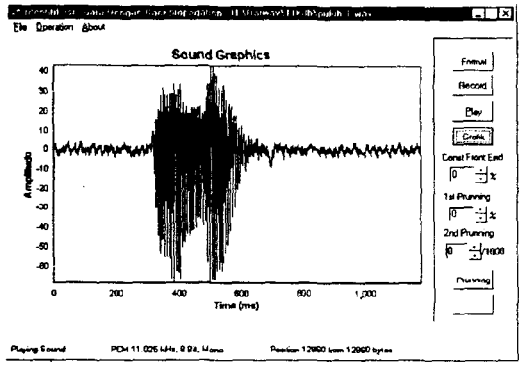
Gambar 4.119 (file belas-8.wav)



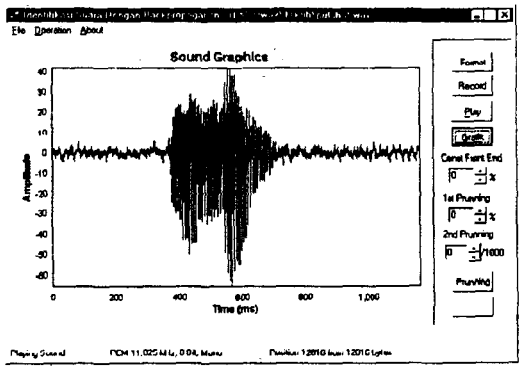
Gambar 4.120 (file belas-9.wav)



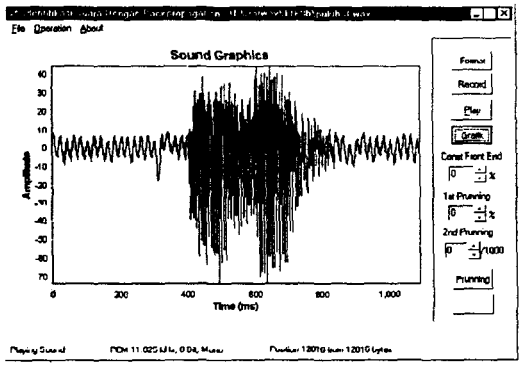
Gambar 4.121 (file puluh.wav)



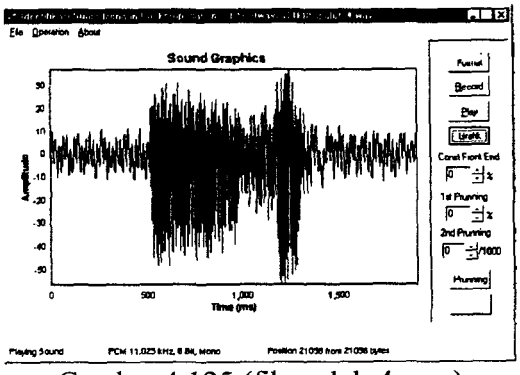
Gambar 4.122 (file puluh-1.wav)



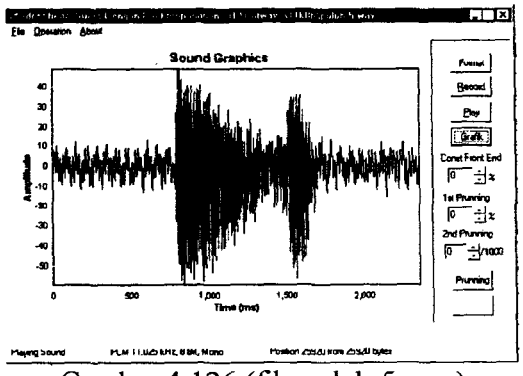
Gambar 4.123 (file puluh-2.wav)



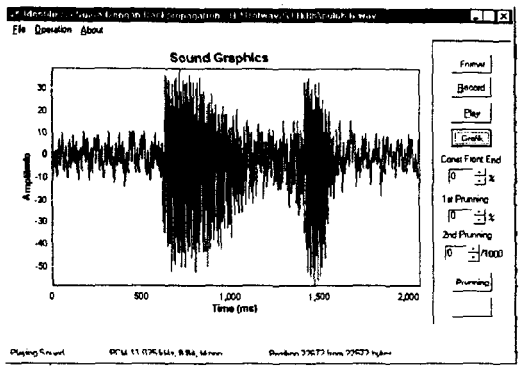
Gambar 4.124 (file puluh-3.wav)



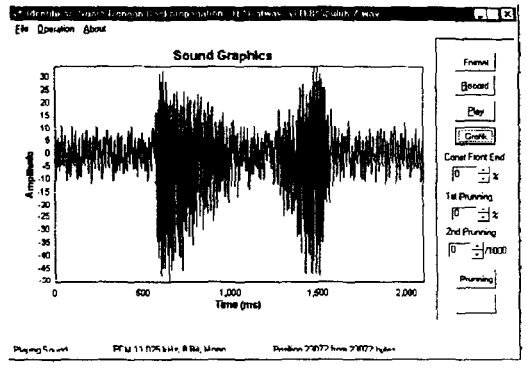
Gambar 4.125 (file puluh-4.wav)



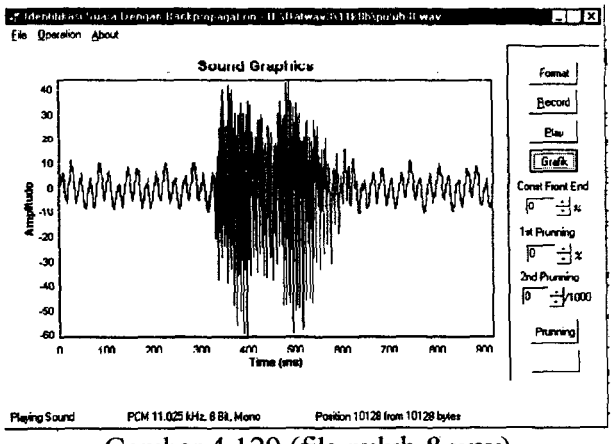
Gambar 4.126 (file puluh-5.wav)



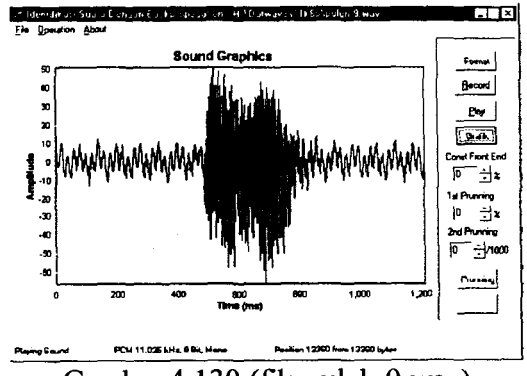
Gambar 4.127 (file puluh-6.wav)



Gambar 4.128 (file puluh-7.wav)



Gambar 4.129 (file puluh-8.wav)



Gambar 4.130 (file puluh-9.wav)