

LAMPIRAN

GS_EDIT.PAS

```
Program GS_Edit;
{
    Program GS Sound Editor
    Dibuat untuk tugas Penulisan Ilmiah pada semester 6
    Judul : Memanfaatkan MIDI SysEx untuk menyunting
           parameter suara piranti MIDI Roland GS

           >>>>Programmed By<<<<<<

           AVINANTA TARIGAN
           3.02 / 50292379
           Teknik Informatika
           STMIK GUNADARMA

           >>>>Pembimbing<<<<<<

           Bpk. I Made Wiryana, Ssi, Skom, Msc.

           Dikompilasi dengan menggunakan kompilator
           TURBO PASCAL 7.0
           Tanggal : 21 Mei 1995

           Membutuhkan unit SBMIDI dan WINOOP
}

Uses Crt,SBMIDI,WinOOP;

{ Deklarasi Variabel dan Konstanta }

Type
    tSetSYX          = ( SOX, Roland_ID, Model_ID,
                        RQ1, DT1, EOX );
    tToneMod         = Record
                        Nama           : string[25];
                        ADM, ADL,
                        VMin, VMax     : byte;
                        end;

Const
    cSyxMsg          : array[tSetSYX] of byte =
        ($F0, $41, $42, $11, $12, $F7);
    cSyxMSB          = $40;
    cTry              = 2;
    cMaxTone         = $7F;
    cDefChannel      = 1;
    cDevice_ID       = $10;
    cGSFileName      = 'GS_TONE.RLD';
    cFileExt         = '.PRM';
    cGSHeader        = 'GS TONE LIST @AVI';
    cChanMax         = 16;
```

Avinanta Tarigan

```

cMDFMax          = 8;
ToneMDF          : array[1..cMDFMax] of tToneMod =

    ( ( Nama : 'Vibrato Rate' ;
      ADM  : $10 ; ADL  : $30 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'Vibrato Depth' ;
      ADM  : $10 ; ADL  : $31 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'TVF Cutoff Freq.' ;
      ADM  : $10 ; ADL  : $32 ;
      VMin : $0E ; VMax : $50 ),

      ( Nama : 'TVF Resonance' ;
      ADM  : $10 ; ADL  : $33 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'TVA+TVF Env. Attack' ;
      ADM  : $10 ; ADL  : $34 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'TVA+TVF Env. Decay' ;
      ADM  : $10 ; ADL  : $35 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'TVA+TVF Env. Release' ;
      ADM  : $10 ; ADL  : $36 ;
      VMin : $0E ; VMax : $72 ),

      ( Nama : 'Vibrato Delay' ;
      ADM  : $10 ; ADL  : $37 ;
      VMin : $0E ; VMax : $72 ) );

Type
tDataTone      = Record
    TNum : byte;
    Modi : array[1..cMDFMax] of byte;
end;

pDataTone      = ^tDataTones;
tDataTones     = array[1..cChanMax] of tDataTone;
tGSFToneName   = text;
pGSToneName    = ^tGSToneName;
tGSToneName    = array[1..cMaxTone] of string[28];

Var
GSToneName     : pGSToneName;
FileToneName   : tGSFToneName;
FileParam      : file of tDataTone;
DataTones     : pDataTone;
DefChannel,
Device_ID,I    : byte;
WinAtas,Winbesar,
WinPesan,WinDialog : tAVIWIN;

```

Avinanta Tarigan

```
{ Deklarasi fungsi dan prosedur }

Procedure RefreshDalam;
{ Prosedur untuk merefresh layar }
Begin
  BlankVir('°', $71);
  WinAtas.Refresh;
  WinBesar.Refresh;
  Refreshall;
end;

Procedure Beep;
Begin
  Sound(700);
  Delay(100);
  NoSound;
end;

Procedure KeyOn(Chan, Note, Vel : byte);
{ Procedure untuk menyalakan salah satu kunci
  pada channel tertentu MIDI Device
  Chan : Channel, Note : nomor kunci, Vel : tekanan }
Begin
  Chan:=(Chan-1) or $90;
  TulisMIDI(Chan);
  TulisMIDI(Note);
  TulisMIDI(Vel);
end;

Procedure KeyOff(Chan, Note, Vel : byte);
{ Procedure untuk mematikan salah satu kunci
  pada channel tertentu MIDI Device
  Chan : Channel, Note : nomor kunci, Vel : tekanan }
Begin
  Chan:=(Chan-1) or $80;
  TulisMIDI(Chan);
  TulisMIDI(Note);
  TulisMIDI(Vel);
end;

Procedure ProgramChange(Chan, PrgNum : byte);
{ Procedure untuk mengganti tone number pada salah
  satu channel MIDI Device
  Chan : Channel , PrgNum : nomor instrumen }
Begin
  If (Chan<=$F) and (PrgNum<=$7F) then
  Begin
    TulisMIDI($C0+Chan-1);
    TulisMIDI(PrgNum-1);
  end;
end;

Procedure PlayKey(Chan, Key, Vel : byte; Durasi : word);
{ Procedure untuk menyuarakan sample / contoh suara
```

Avinanta Tarigan

```
    pada key dan channel tertentu MIDI Device  }
Begin
    KeyOn(Chan,Key,Vel);
    Delay(Durasi);
    KeyOff(Chan,Key,Vel);
end;

Procedure PlayOktav(Chan,BaseKey,Vel : byte;Durasi : word);
{ Procedure untuk menyuarakan sample / contoh suara
  dalam satu oktaf  }
Var
    Tambah          : byte;
Const
    Oktav           : array[1..8] of byte =
                    (0,2,4,5,7,9,11,12);
Begin
    If (Chan < cChanMax+1) and (Vel<=$7F) then
        Begin
            For Tambah:=1 to 8 do
                Begin
                    KeyOn(Chan,BaseKey+Oktav[Tambah],Vel);
                    Delay(Durasi*8);
                    KeyOff(Chan,BaseKey+Oktav[Tambah],Vel);
                    Delay(Durasi);
                    If Keypressed then exit;
                end;
            Delay(Durasi*5);
            For Tambah:=8 downto 1 do
                Begin
                    KeyOn(Chan,BaseKey+Oktav[Tambah],Vel);
                    Delay(Durasi*8);
                    KeyOff(Chan,BaseKey+Oktav[Tambah],Vel);
                    Delay(Durasi);
                    If Keypressed then exit;
                end;
            end;
        end;
    end;
end;

Procedure KirimSYXDataSet( AD_MSBM, AD_MSBL,
                           AD_LSB, Data : byte);
{ Procedure untuk mengirim Sys-Ex Data set pada
  MIDI Device
  AD_MSBM,AD_MSBL,AD_LSB : alamat AMDT, Data : data }
Var SUM : byte;
Begin
    SUM := AD_MSBM + AD_MSBL + AD_LSB + Data;
    SUM := $100 - SUM;
    SUM := SUM and $7F;
    TulisMIDI(cSyxMsg[SOX]);
    TulisMIDI(cSyxMsg[Roland_ID]);
    TulisMIDI(Device_ID);
    TulisMIDI(cSyxMsg[Model_ID]);
    TulisMIDI(cSyxMsg[DT1]);
    TulisMIDI(AD_MSBM);
```

Avinanta Tarigan

```
TulisMIDI(AD_MSBL);
TulisMIDI(AD_LSB);
TulisMIDI(Data);
TulisMIDI(SUM);
TulisMIDI(cSyxMsg[EOX]);
end;
```

```
Procedure KirimSYXRequest( AD_MSBM, AD_MSBL, AD_LSB,
                           Size_LSB : byte);
{ Procedure untuk mengirim Sys-Ex Request Data pada
  MIDI Device
  AD_MSBM, AD_MSBL, AD_LSB : address AMDT,
  Size_LSB : LSB Size }
Var SUM : byte;
Begin
  SUM := AD_MSBM + AD_MSBL + AD_LSB + Size_LSB;
  SUM := $100 - SUM;
  SUM := SUM and $7F;
  TulisMIDI(cSyxMsg[SOX]);
  TulisMIDI(cSyxMsg[Roland_ID]);
  TulisMIDI(Device_ID);
  TulisMIDI(cSyxMsg[Model_ID]);
  TulisMIDI(cSyxMsg[RQ1]);
  TulisMIDI(AD_MSBM);
  TulisMIDI(AD_MSBL);
  TulisMIDI(AD_LSB);
  TulisMIDI(0);
  TulisMIDI(0);
  TulisMIDI(Size_LSB);
  TulisMIDI(SUM);
  TulisMIDI(cSyxMsg[EOX]);
end;
```

```
Function AmbilParameter(ChanNN,MDFNN : byte):boolean;
{ Prosedur untuk mengambil parameter berdasarkan MDFNN
  yang aktif di ChanNN
  ChanNN : Kanal, MDFNN : Nomor MDF }
Var C,SUM,Try,UEox : byte;
Begin
  Try:=0;
  Repeat
    If ChanNN > 10 then dec(ChanNN);
    C:=cSyxMSB+ToneMDF[MDFNN].ADM+
      ChanNN+ ToneMDF[MDFNN].ADL;
    KirimSyxRequest(cSyxMSB, ToneMDF[MDFNN].ADM +
      ChanNN, ToneMDF[MDFNN].ADL, 1);
    AktifkanIntr;
    UEox:=0;
    Repeat
      Delay(1);
      Inc(UEox);
    Until (BacaBuffer(11)=cSyxMsg[EOX]) or (UEox>100) ;
    NonAktifkanIntr;
    SUM:=C+BacaBuffer(9)+BacaBuffer(10);
    If ChanNN >= 10 then inc(ChanNN);
```

Avinanta Tarigan

```
        If SUM = $0 then
            DataTones^[ChanNN].Modi[MDFNN]:=BacaBuffer(9);
            SBAda;
            Delay(20);
            Inc(Try);
        Until (Try>=cTry) or (SUM = $0) or (UEox>30);
        If (Try>=cTry) or (UEox>100)
            then AmbilParameter:=False
            else AmbilParameter:=True;
    end;

Procedure AmbilParamChan(Chan : byte);
{ Prosedur untuk mengambil semua parameter suara
  yang aktif di channel Chan }
Var G : byte;
Begin
    For G:=1 to cMDFMax do
        If Not AmbilParameter(Chan,G) then
            Begin
                Beep;
                WinPesan.WCenter(2,$4F,'Data MIDI Error ...');
                Delay(2000);
                RefreshDalam;
                Exit;
            end;
    end;
end;

Procedure ResetGS;
{ Prosedur untuk mereset piranti MIDI Roland GS
  ke standar GS }
Begin
    Delay(20);
    KirimSyxDataSet($40,00,$7F,00);
    Delay(20);
end;

Procedure KirimParam(Chan, MDFNum : byte);
{ Prosedur untuk mengirim parameter suara nomor
  MDFNum di channel Chan }
Begin
    If Chan < 10 then
        KirimSyxDataSet(cSyxMSB,ToneMDF[MDFNum].ADM
            +Chan,ToneMDF[MDFNum].ADL,
            DataTones^[Chan].Modi[MDFNum])
    else
        KirimSyxDataSet(cSyxMSB,ToneMDF[MDFNum].ADM
            +Chan-1,ToneMDF[MDFNum].ADL,
            DataTones^[Chan].Modi[MDFNum])
    end;
end;

Procedure KirimSemuaParam;
{ Prosedur untuk mengirim semua parameter suara }
Var I,J : byte;
Begin
    For I:=1 to cChanMax do
        For J:=1 to cMDFMax do
```

Avinanta Tarigan

```
        If I<>10 then
            KirimParam(I,J);
end;

Procedure AmbilNamaTone;
{ Prosedur untuk mengambil nama tone dari file
  GS_TONE.RLD ke dalam variabel dinamik GSToneName }
Var I : byte;
    S : String;
Begin
    Assign(FileToneName,cGSFileName);
    {$I-}
    Reset(FileToneName);
    {$I+}
    If IOResult<>0 then
        Begin
            Writeln('Error : File ',cGSFileName,
                ' tidak ada ...');
            close(FileToneName);
            halt;
        end
    else
        Begin
            Readln(FileToneName,S);
            I:=0;
            If S=cGSHeader then
                Begin
                    New(GSToneName);
                    While Not EOF(FileToneName) do
                        Begin
                            Inc(I);
                            Readln(FileToneName,GSToneName^[I]);
                        end;
                    end
                else
                    Begin
                        Writeln('Error : File ',cGSFileName,
                            ' tidak benar ....');
                        Close(FileToneName);
                        Halt;
                    end;
                end;
            end;
            Close(FileToneName);
end;

Procedure InitAllParam;
{ Prosedur untuk menginisial semua parameter
  ke nilai default }
Var F,G : byte;
Begin
    For F:=1 to cChanMax do
        Begin
            DataTones^[F].Tnum:=1;
            For G:=1 to cMDFMax do
                DataTones^[F].Modi[G]:=$40;
            end;
        end;
    end;
end;
```

Avinanta Tarigan

```
        end;
end;

Procedure ProsedurToEnter;
{ Prosedur untuk menginisial program yang terdiri dari
  membentuk data dinamis, auto detect Sound Blaster,
  mengambil nama tone, menseset Default Channel, Default
  Parameter Default Device ID, dan mengambil parameter
  channel 1 dari piranti MIDI }
Begin
  If Not SBAda then
    Begin
      Writeln('Error : Sound Blaster Tidak Terpasang ...');
      Halt;
    end;
  AmbilNamaTone;
  DefChannel:=cDefChannel;
  Device_ID:=cDevice_ID;
  New(DataTones);
  InitAllParam;
  RefreshSimpan;
  Delay(1000);
  ProgramChange(DefChannel,1);
end;

Procedure ProsedurToExit;
{ Prosedur untuk menghapus data dinamis pada memory
  dan semua window sebelum program selesai }
Begin
  WinBesar.Tutup;
  WinAtas.Tutup;
  WinPesan.Tutup;
  WinDialog.Tutup;
  Dispose(GSToneName);
  Dispose(DataTones);
end;

Procedure InitAllWindows;
{ Prosedur untuk menginisial Window yang dipakai
  dalam program utama }
Var I: byte;
Begin
  WinAtas.Init(5,2,70,5,$11,$4F,'');
  WinBesar.Init(19,9,43,15,$71,$4E,'PROGRAM UTAMA PENYUNTING');
  WinPesan.Init(20,12,40,4,$4F,$0F,'Pesan Program');
  WinDialog.Init(20,8,40,6,$5F,$70,'File Dialog');
  RefreshDalam;
  WinAtas.WString(8,1,$1F,'Program Penyunting Parameter Suara'+
    ' Instrumen Roland GS');
  WinAtas.WString(11,2,$1D,'Dengan Memanfaatkan MIDI System '+
    'Exclusive Message');
  WinAtas.WString(18,3,$13,'Programmed By Avinanta Tarigan 1995');
  WinBesar.WString(3,2,$78,'Tone : ');
  WinBesar.WLine(10,2,$2F,28,GSToneName^[1]);
  WinBesar.WString(3,3,$78,'Channel : ');
  WinBesar.WString(3,4,$78,'Parameter');
  WinBesar.WString(35,4,$78,'Nilai');
```

Avinanta Tarigan

```
AmbilParamChan(1);
For I:=1 to cMDFMax do
    WinBesar.WLine(3,5+I,$17,34,' '+ToneMDF[I].Nama);
For I:=1 to cMDFMax do
    WinBesar.WByte(37,5+I,$1F,DataTones^[DefChannel].Modi[I]);
WinBesar.WByte(13,3,$5F,DefChannel);
end;
```

```
Procedure Save;
{ Prosedur untuk menyimpan nilai parameter di
  file yang diinput }
Var NamaFile : SString;
    Res      : integer;
    Ch       : char;
    I        : word;
Begin
    WinDialog.WString(5,3,$5F,'Save File ');
    WinDialog.WLine(15,3,$3F,20,' NONAME');
    Textattr:=$3F;
    Window(36,11,55,11);
    Gotoxy(1,1);
    Readln(NamaFile);
    Window(1,1,79,24);
    If NamaFile='' then NamaFile:='NONAME';
    NamaFile:=Copy(NamaFile,1,8)+cFileExt;
    Assign(FileParam,NamaFile);
    {$I-}
        Reset(FileParam);
    {$I+}
    Res:=IOResult;
    If Res=0 then
        Begin
            WinPesan.WCenter(2,$4E,'File Sudah Ada, Tumpangi ?');
            Ch:=Uppcase(Readkey);
            Close(FileParam)
        end;
    If (Ch='Y') or (Res<>0) then
        Begin
            Rewrite(FileParam);
            For I:=1 to cChanMax do
                Write(FileParam,DataTones^[I]);
            Close(FileParam);
        end;
end;
```

```
Procedure Load;
{ Prosedur untuk mengambil nilai parameter di
  file yang diinput }
Var NamaFile : SString;
    Ch       : char;
    I,L      : word;
Begin
    WinDialog.WString(5,3,$5F,'Load File ');
    WinDialog.WLine(15,3,$3F,20,' NONAME');
    Textattr:=$3F;
    Window(36,11,55,11);
```

Avinanta Tarigan

```
Gotoxy(1,1);
Readln>NamaFile);
Window(1,1,79,24);
If>NamaFile='' then>NamaFile:='NONAME';
>NamaFile:=Copy>NamaFile,1,8)+cFileExt;
Assign(FileParam,>NamaFile);
{$I-}
    Reset(FileParam);
{$I+}
If IOResult<>0 then
    Begin
        WinPesana.WCenter(2,$4E,'File '+>NamaFile+ ' tidak ada');
        Readln;
    end
else
    Begin
        For I:=1 to cChanMax do
            Read(FileParam>DataTones^[I]);
        Close(FileParam);
        For I:=1 to cMDFMax do
            Begin
                WinBesara.WByte(37,5+I,$1F>DataTones^
                    [DefChannel].Modi[I]);
            end;
        WinBesara.WLine(10,2,$2F,28,GSToneName^[1]);
        KirimSemuaParam;
    end;
end;

Procedure Editor;
{ Prosedur untuk mendeteksi penekanan tombol dalam
  program utama dan menjalankannya apabila sesuai }

Var ASCII,Scode,
    MDFN,
    YPOS,I      : byte;
Begin
MDFN:=1;
Repeat
    RefreshDalam;
    LineXY(22,14+MDFN,36,$3F);
    asm
        mov ah,0
        int 16h
        mov ASCII,al
        mov Scode,ah
    end;
    If ASCII=0 then
        Case Scode of
            $51 : Begin
                If>DataTones^[DefChannel].TNum<=$7F then
                    Begin
                        Inc>DataTones^[DefChannel].TNum);
                        WinBesara.WLine(10,2,$2F,28,GSToneName^
                            [DataTones^[DefChannel].TNum]);
                    end;
                end;
        end;
    end;
```

```

ProgramChange(DefChannel,DataTones^[DefChannel].TNum);
    End;
    end;
$49 : Begin
    If DataTones^[DefChannel].TNum>1 then
        Begin
            Dec(DataTones^[DefChannel].TNum);
            WinBesar.WLine(10,2,$2F,28,GSToneName^
                [DataTones^[DefChannel].TNum]);
        End;
    End;

ProgramChange(DefChannel,DataTones^[DefChannel].TNum);
    End;
    end;
$48 : Begin
    If MDFN>1 then
        Begin
            Dec(MDFN);
        End;
    End;
$50 : Begin
    If MDFN<cMDFMax then
        Begin
            Inc(MDFN);
        End;
    End;
$4D : Begin
    If DefChannel<$10 then
        Begin
            if DefChannel+1 = 10 then Inc(DefChannel);
            Inc(DefChannel);
            AmbilParamChan(DefChannel);
        End;
    End;

ProgramChange(DefChannel,DataTones^[DefChannel].TNum);
    WinBesar.WByte(13,3,$5F,DefChannel);
    WinBesar.WLine(10,2,$2F,28,GSToneName^
        [DataTones^[DefChannel].TNum]);
    For I:=1 to cMDFMax do
        Begin
            WinBesar.WByte(37,5+I,$1F,DataTones^
                [DefChannel].Modi[I]);
        End;
    End;
$4B : Begin
    If DefChannel>$1 then
        Begin
            if DefChannel-1 = 10 then Dec(DefChannel);
            Dec(DefChannel);
            AmbilParamChan(DefChannel);
        End;
    End;

ProgramChange(DefChannel,DataTones^[DefChannel].TNum);
    WinBesar.WByte(13,3,$5F,DefChannel);
    WinBesar.WLine(10,2,$2F,28,GSToneName^
        [DataTones^[DefChannel].TNum]);
    For I:=1 to cMDFMax do
        Begin
            WinBesar.WByte(37,5+I,$1F,DataTones^

```

```

                                [DefChannel].Modi[I]);
                                end;
                                end;
                                end;
$1F : Begin
    Save;
    end;
$26 : Begin
    Load;
    end;
$22 : Begin
    Beep;
    WinPesan.WCenter(2,$4F,'Reseting GS ....');
    ResetGS;
    InitAllParam;
    Delay(1000);
    For I:=1 to cMDFMax do
        WinBesar.WByte(37,5+I,$1F,
            DataTones^[DefChannel].Modi[I]);
        WinBesar.WLine(10,2,$2F,28,GSToneName^[1]);
    end;
$3B : Begin
    PlayOktav(DefChannel,40,100,60);
    end;
$3C : Begin
    PlayOktav(DefChannel,55,100,60);
    end;
$3D : Begin
    PlayOktav(DefChannel,70,100,60);
    end;

end
else
Case ASCII of
    $2B : Begin
        If DataTones^[DefChannel].Modi[MDFN] <
            ToneMDF[MDFN].VMax then
            Begin
                Inc(DataTones^[DefChannel].Modi[MDFN]);
                KirimParam(DefChannel,MDFN);
                WinBesar.WByte(37,5+MDFN,$1F,
DataTones^[DefChannel].Modi[MDFN]);
            end;
        end;
    $2D : Begin
        If DataTones^[DefChannel].Modi[MDFN] >
            ToneMDF[MDFN].VMin then
            Begin
                Dec(DataTones^[DefChannel].Modi[MDFN]);
                KirimParam(DefChannel,MDFN);
                WinBesar.WByte(37,5+MDFN,$1F,
DataTones^[DefChannel].Modi[MDFN]);
            end;
        end;
    32 : Begin
        PlayKey(DefChannel,60,127,400);

```

Avinanta Tarigan

```
                end;  
            end;  
Until (SCode=$1) and (ASCII=$1B);  
end;  
  
Begin  
    ProsedurToEnter;  
    InitAllWindows;  
    Editor;  
    ProsedurToExit;  
end.
```

SBMIDI.PAS

```
Unit SBMIDI;

{
  Unit SBMIDI digunakan untuk interfacing
  dengan Sound Blaster MIDI Interface.
  Programmed by :

      AVINANTA TARIGAN
      50292379 / Teknik Informatika
      STMIK GUNADARMA JAKARTA

  Dikompilasi dengan kompiler TURBO PASCAL 7.0

  Tanggal : 1 Mei 1995
}

Interface

Uses Dos;

Function BacaBuffer(POS : word) : byte;
{ Fungsi untuk mendapatkan nilai data dalam buffer
  pada posisi tertentu }

Procedure TulisSB(Cmd : Byte);
{ Prosedur Untuk Menulis Perintah / Data
  Ke DSP Sound Blaster }

Procedure TulisMIDI(Data : Byte);
{ Prosedur Untuk Menulis Data Ke MIDI Port
  Sound Blaster Dengan UART Mode }

Procedure AktifkanIntr;
{ Prosedur Untuk mengaktifkan Interrupt MIDI
  input Sound Blaster }

Procedure NonAktifkanIntr;
{ Prosedur Untuk mengnonaktifkan Interrupt MIDI
  input Sound Blaster }

Function SBAda : Boolean;
{ Fungsi untuk mengenali keberadaan SB sekaligus
  mereset dan mendapatkan alamat port-nya }

Implementation

Const
  SBIntVec          = $0F;
  MaxBuffSize      = $F;
```

```

Var
  SBAddrBase,
  POSBuf           : word;
  OldSBIntVec,
  pExit           : pointer;
  DataBuffer      : array[1..MaxBuffSize] of byte;
  Dump            : byte;

Function BacaBuffer;
Var Data : byte;
Begin
  Data:=$FF;
  If (POS<=MaxBuffSize) And (POS<=POSBuf)
    then Data:=DataBuffer[POS];
  BacaBuffer:=Data;
end;

Procedure Tunggu(F : Word);assembler;
Asm
  mov cx,F
  @@Loop :
  Loop @@Loop
end;

Function SBAda;
Var Lamal,Lama2 : Word;
  Kode : boolean;
Begin
  SBAddrBase := $220;
  Kode := False;
  Repeat
    Port[SBAddrBase+$6]:=$1;
    Tunggu(30);
    Port[SBAddrBase+$6]:=$0;
    Lama2:=0;
    Repeat
      Lamal:=0;
      Repeat
        Inc(Lamal);
      Until (Port[SBAddrBase+$E] and $80=$80)
        Or (Lamal>100);
      Inc(Lama2);
    Until (Port[SBAddrBase+$A] = $0AA)
      Or (Lama2>100);
    If (Lama2<=100) or
      (SBAddrBase=$240) then Kode:=True;
    If (Not Kode) And (SBAddrBase=$220) then
      SBAddrBase:=$240;
  Until Kode;
  If Lama2>100 then SBAda:=False
  Else SBAda:=True;
end;

Procedure TulisSB;
Begin

```

```

Repeat
  Until Port[SBAddrBase+$C] And $80<>$80;
  Port[SBAddrBase+$C]:=Cmd;
end;

Procedure TulisMIDI;
Begin
  TulisSB($38);
  TulisSB(Data);
end;

Procedure UARTIntr;interrupt;
{ Prosedur yang diaktifkan saat
  interrupt UART Sound Blaster terjadi }
Begin
  Dump:=Port[SBAddrBase+$A];
  If Dump<>$FE then
    Begin
      Inc(POSBuf);
      DataBuffer[POSBuf]:=Dump;
    end;
  Dump:=Port[SBAddrBase+$E];
  Port[$20]:=$20;
end;

Procedure AktifkanIntr;
Begin
  POSBuf := 0;
  TulisSB($35);
  Port[$21]:=Port[$21] and (Not (1 shl 7));
  Port[$20]:=$20;
  Dump:=Port[SBAddrBase+$E];
end;

Procedure NonAktifkanIntr;
Begin
  Port[$21]:=Port[$21] and (Not (0 shl 7));
end;

Procedure ExitPros;
Begin
  SetIntVec(SBIntVec,OldSBIntVec);
  ExitProc:=pExit;
end;

Begin
  pExit := ExitProc;
  ExitProc:=@ExitPros;
  GetIntVec(SBIntVec,OldSBIntVec);
  SetIntVec(SBIntVec,@UARTIntr);
end.

```

WINOOP.PAS

```
Unit WINOOP;
{
    Unit WINOOP
    Window-Text-Based User Interface
    Dibuat untuk melengkapi program
    GS_EDIT.PAS

    >>>>Programmed By<<<<

    Avinanta Tarigan
    3.02 / 50292379
    Teknik Informatika
    STMIK GUNADARMA

    Kompilator : TURBO PASCAL 7.0

    Tanggal    : 20 Mei 1995
}

Interface
Type
    tPixel      = record
                    Data,Att : byte;
                end;
    pLayar      = ^tLayar;
    tLayar      = array[1..25,1..80] of tPixel;

Type
    tAviWin = Object
                X,Y,W,H          : byte;
                Att,AttJudul     : byte;
                Judul            : String;
                Buffer            : pLayar;
                InitDone         : boolean;
    Procedure Init(XX,YY,WW,HH,AT,ATJ : byte;Juduls : string);
    Procedure Tutup;
    Procedure WByte(BBX,BBY,BBAtt,BBData:byte);
    Procedure TulisChar(BX,BY,BAtt : byte; BData : char);
    Procedure TulisString(SX,SY,SAtt : byte; SData : string);
    Procedure WChar(CX,CY,CAtt : byte;CData : char);
    Procedure WString(SX,SY,SAtt : byte; SData : String);
    Procedure WLine(SSX,SSY,SSAtt,SSL : byte;SSData : String);
    Procedure WCenter(DSSY,DSSAtt:byte;DSSData : string);
    Procedure Refresh;
    Procedure Move(MX,MY : byte);
end;
```

Avinanta Tarigan

```
Procedure RefreshAll;
Procedure BlankVir(C:char;Att:byte);
Procedure RefreshSimpan;
Procedure Refreshlayar;
Procedure LineXY(X,Y,Lens,Att : byte);

Implementation

Var VirLayar,
    Simpan    : ^tLayar;
    Layar     : tLayar absolute $B800:0000;
    ProcExit  : Pointer;
    PSLX,PSLY : byte;

Procedure ProcExits;
Begin
    Layar:=Simpan^;
    FreeMem(Simpan,SizeOf(Simpan^));
    FreeMem(VirLayar,SizeOf(VirLayar^));
    asm
        pusha
        mov ah,2h
        mov bh,0h
        mov dl, PSLX
        mov dh, PSLY
        int 10h
        popa
    end;
    ExitProc:=ProcExit;
end;

Procedure BlankVir;
Var X,Y : byte;
Begin
    For X:=1 to 80 do
        For Y:=1 to 25 do
            Begin
                VirLayar^[Y,X].Data:=Ord(C);
                VirLayar^[Y,X].Att:=Att;
            end;
        end;
    end;

Procedure RefreshSimpan;
Begin
    VirLayar^:=Simpan^;
end;

Procedure Refreshlayar;
Begin
    VirLayar^:=Layar;
end;

Procedure RefreshAll;
Begin
    Layar:=VirLayar^;
end;

Procedure LineXY;
```

Avinanta Tarigan

```
Var I : byte;
Begin
  If (X+Lens<=80) and (Y<=25) then
    For I:=X to X+Lens do
      Layar[Y,I].att:=Att
    end;
Function ChkAllPOS(X,Y,W,H : byte):boolean;
Begin
  If (X<=80) and (Y<=25) and (X>=1) and (Y>=1) then
    ChkAllPOS:=True else ChkAllPOS:=False;
  end;

Procedure tAwiWin.Init;
Var FX,FY : byte;
Begin
  If ChkAllPOS(XX,YY,WW,HH) and Not InitDone then
    Begin
      X:=XX;Y:=YY;
      W:=WW;H:=HH;
      Att:=At;Judul:=Juduls;
      AttJudul:=AtJ;
      GetMem(Buffer,4000);
      InitDone:=True;
      For FX:=1 to W do
        For FY:=1 to H do
          TulisChar(FX,FY,Att,' ');
        If Judul<>' ' then
          Begin
            For FX:=1 to W do
              TulisChar(FX,1,AttJudul,' ');
            TulisChar(2,1,AttJudul and $F0,#254);
            TulisString((W-Length(Judul)) DIV 2+1,1,
              AttJudul,Judul);
            end;
          end;
        end;
    end;

Procedure tAwiWin.Tutup;
Begin
  If InitDone then
    Begin
      FreeMem(Buffer,W*H);
    end;
  end;

Procedure tAwiWin.TulisChar;
Begin
  Buffer^[BY,BX].Data:=Ord(BData);
  Buffer^[BY,BX].Att:=Ord(BAtt);
  end;

Procedure tAwiWin.TulisString;
Var I : byte;
Begin
  If (Length(SData) < (W-SX-1)) and (SY<H) then
    For I:=1 to Length(SData) do
      TulisChar(SX+I-1,SY,SAtt,SData[I]);
    end;
  end;
```

Avinanta Tarigan

```
end;

Procedure tAwiWin.WChar;
Begin
  If (CX+1<W-1) and (CY+1<W-1) then
    TulisChar(CX+1,CY+1,CAtt,CData);
  Refresh;
  Refreshall;
end;

Procedure tAwiWin.WString;
Begin
  TulisString(SX+1,SY+1,SAtt,SData);
  Refresh;
  RefreshAll;
end;

Procedure tAwiWin.Refresh;
Var I,K : byte;
Begin If InitDone then
Begin
  For I:=0 to W-1 do
    For K:=0 to H-1 do
      If (I+X<=80) and (K+Y<=25) then
        Begin
          VirLayar^[K+Y,I+X].Data:=Buffer^[K+1,I+1].Data;
          VirLayar^[K+Y,I+X].Att:=Buffer^[K+1,I+1].Att;
        end;
      For I:=1 to W do
        If (Y+H<=25) and (I+X+1<=80) then
          VirLayar^[Y+H,I+X+1].Att:=VirLayar^[Y+H,I+X+1].Att and $07;
        For I:=1 to H do
          If (I+Y<=25) and (X+W<=80) then
            Begin
              VirLayar^[I+Y,X+W].Att:=VirLayar^[I+Y,X+W].Att and $07;
            end;
          For I:=1 to H do
            If (I+Y<=25) and (X+W+1<=80) then
              Begin
                VirLayar^[I+Y,X+W+1].Att:=VirLayar^[I+Y,X+W+1].Att and $07;
              end;
            end;
          end;
        end;
      end;
    end;
  end;
end;

Procedure tAwiWin.Move;
Begin
  If CHKAllPOS(MX,MY,W,H) then
    Begin
      X:=MX;Y:=MY;
      Refresh;
    end;
end;

end;

Procedure tAwiWin.WByte;
Var S : String;
Begin
```

Avinanta Tarigan

```
Str(BBData:3,S);
WString(BBX,BBY,BBAtt,S);
end;

Procedure tAwiWin.WLine;
Var I : Byte;
Begin
  If SSL+SSX<=W then
    For I:=1 to SSL-Length(SSData) do
      SSData:=SSData+' ';
    WString(SSX,SSY,SSAtt,SSData);
  end;

Procedure tAwiWin.WCenter;
Var I,SX:byte;
Begin
  If Length(DSSData)+4<W then
    Begin
      SX:=(W-3-Length(DSSData)) DIV 2;
      For I:=1 to SX do
        DSSData:=' '+DSSData;
      For I:=1 to SX-1 do
        DSSData:=DSSData+' ';
      TulisString(2,DSSY+1,DSSAtt,DSSData);
      Refresh;
      RefreshAll;
    end;
  end;

Begin
  asm
    pusha
    mov ah,3h
    mov bh,0h
    int 10h
    mov PSLY,DH
    mov PSLX,DL
    mov ah,01
    mov cl,3
    mov cx,02020h
    int 10h
    popa
  end;
  GetMem(Simpan,SizeOf(Simpan^));
  GetMem(VirLayar,SizeOf(VirLayar^));
  Simpan^:=Layar;
  ProcExit:=ExitProc;
  ExitProc:=@ProcExits;
end.
```

GS_TONE.RLD

GS TONE LIST @AVI

Acoustic Piano 1
Acoustic Piano 2
Acoustic Piano 3
Honky-Tonk Piano
Elect. Piano 1
Elect. Piano 2
Harpsichord
Clavinet
Celesta
Glockenspiel
Music Box
Vibraphone
Marimba
Xylophone
Tubular Bells
Dulcimer
Hammond Organ
Percussive Organ
Rock Organ
Church Organ
Reed Organ
Accordion
Harmonica
Tango Accordion
Acoustic Nylon Guitar
Acoustic Steel Guitar
Electric Jazz Guitar
Electric Clean Guitar
Electric Muted Guitar
Overdriven Guitar
Distortion Guitar
Guitar Harmonics
Acoustic Bass
Electric Bass Fingered
Electric Bass Picked
Fretless Bass
Slap Bass 1
Slap Bass 2
Synth Bass 1
Synth Bass 2
Violin
Viola
Cello

Contrabass
Tremolo Strings
Pizzicato Strings
Orchestral Harp
Timpani
String Ensemble 1
String Ensemble 2
Synth Strings 1
Synth Strings 2
Choir Aahs
Voice Oohs
Synth Voice
Orchestral Hit
Trumpet
Trombone
Tuba
Muted Trumpet
French Horn
Brass Section
Synth Brass 1
Synth Brass 2
Soprano Sax
Alto Sax
Tenor Sax
Baritone Sax
Oboe
English Horn
Bassoon
Clarinet
Piccolo
Flute
Recorder
Pan Flute
Bottle Blow
Shakuhachi
Whistle
Ocarina
Square Wave Lead
Wave Lead
Calliope Lead
Chiff Lead
Charang
Solo Synth Voice
Bright Saw Wave

Avinanta Tarigan

Brass and Lead
Fantasia Pad
Warm Pad
Poly Synth Pad
Space Voice Pad
Bowed Glass Pad
Metal Pad
Halo Pad
Sweep Pad
Ice Rain
Sound Track
Crystal
Atmosphere
Brightness
Goblin
Echo Drops
Star Theme
Sitar
Banjo
Shamisen
Koto

Kalimba
Bagpipe
Fiddle
Shanai
Tinkle Bells
Agogo
Steel Drums
Woodblock
Taiko Drum
Melodic Drum
Synth Drum
Reverse Cymbal
Guitar Fret Noise
Breath Noise
Seashore
Bird Tweet
Telephone Ring
Helicopter
Applause
Gunshot