

MIX.PAS

```
{*****}
{                                     M I X . P A S                                     }
{*-----*}
Task           : Tool for changing mixer settings. Current
                settings are preserved.
{*-----*}
Author          : Michael Tischer / Bruno Jennrich
Developed on   : 03/20/1994
Last update    : 04/06/1995
{*****}
{$X+}          { Function result optional }
Uses SBUTIL,MIXUTIL,DSPUTIL,IRQUTIL,ARGS,DOS;

Const
MIX_NAME = 'MIX';                                { For output in Help screen }

{- Help variables -----}
var
  ParStr       : String;      { For evaluation of parameter strings }
  Argumente    : NArgStrings;
  iNumStrings,
  iIdx         : Integer;

{*****}
{ Print_Mix3Settings : Output current status of CT1345. }
{*****}
Procedure Print_Mix3Settings;

Begin
  Write( ' ADC Filter ' );
  if mix3_GetADCFilter then Writeln( 'on' )
```

```

                else Writeln( 'off' );

Write( ' DAC Filter ' );
if mix3_GetDACFilter then Writeln( 'on' )
                        else Writeln( 'off' );

Write( ' Low pass filter: ' );
if mix3_GetADDACLowPass then Writeln( '8.8 kHz' )
                        else Writeln( '3.2 kHz' );

Write( ' DAC Stereo ' );
if mix3_GetDACStereo then Writeln( 'on' )
                        else Writeln( 'off' );

Write( ' SOURCE (DSP input): ' );
case mix3_GetADCSource of
    CD:   Writeln( 'CD' );
    LINE: Writeln( 'LINE' );
    MIC:  Writeln( 'MIC' );
End;

Writeln( #10'Volumes:' );
Writeln( ' MASTER: ', mix3_GetVolume( MASTER_L ), ' ',
        mix3_GetVolume( MASTER_R ) );
Writeln( ' MIC   : ', mix3_GetVolume( MIC ) );
Writeln( ' CD    : ', mix3_GetVolume( CD_L ), ' ',
        mix3_GetVolume( CD_R ) );
Writeln( ' LINE  : ', mix3_GetVolume( LINE_L ), ' ',
        mix3_GetVolume( LINE_R ) );
Writeln( ' VOICE : ', mix3_GetVolume( VOICE_L ), ' ',
        mix3_GetVolume( VOICE_R ) );
Writeln( ' MIDI  : ', mix3_GetVolume( MIDI_L ), ' ',

```

[illegible]

```

if mix4_GetADCSourcer( CD_R ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetADCSourcer( LINE_L ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetADCSourcer( LINE_R ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetADCSourcer( MIDI_L ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetADCSourcer( MIDI_R ) <> 0 then Writeln( 'X ' )
                                else Writeln( '- ' );

Writeln( 'OUT-Settings:          MIC   CD   LINE' );
Write ( ' ' );
IF mix4_GetOUTSource( MIC ) <> 0 then Write( ' X ' )
                                else Write( ' - ' );
if mix4_GetOUTSource( CD_L ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetOUTSource( CD_R ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetOUTSource( LINE_L ) <> 0 then Write( 'X ' )
                                else Write( '- ' );
if mix4_GetOUTSource( LINE_R ) <> 0 then Writeln( 'X ' )
                                else Writeln( '- ' );

Writeln( ' ADC-Gain ', mix4_GetADCGain( CH_LEFT ), ' ',
                                mix4_GetADCGain( CH_RIGHT ) );
Writeln( ' OUT-Gain ', mix4_GetOUTGain( CH_LEFT ), ' ',
                                mix4_GetOUTGain( CH_RIGHT ) );
Write( ' Automatic Gain Control ' );
if mix4_GetAGC then Writeln( 'on' ) else Writeln( 'off' );

Writeln( ' Treble: ', mix4_GetTreble( CH_LEFT ), ' ',

```

```

        mix4_GetTreble( CH_RIGHT ) );
Writeln( ' Bass:      ', mix4_GetBass( CH_LEFT ), ' ',
        mix4_GetBass( CH_RIGHT ) );

Writeln( 'Volumes:' );
Writeln( ' MASTER    : ', mix4_GetVolume( MASTER_L ), ' dB ',
        mix4_GetVolume( MASTER_R ), ' dB' );
Writeln( ' MIC       : ', mix4_GetVolume( MIC ) );
Writeln( ' CD        : ', mix4_GetVolume( CD_L ), ' ',
        mix4_GetVolume( CD_R ) );
Writeln( ' LINE      : ', mix4_GetVolume( LINE_L ), ' dB ',
        mix4_GetVolume( LINE_R ), ' dB' );
Writeln( ' VOICE     : ', mix4_GetVolume( VOICE_L ), ' dB ',
        mix4_GetVolume( VOICE_R ), ' dB' );
Writeln( ' MIDI      : ', mix4_GetVolume( MIDI_L ), ' dB ',
        mix4_GetVolume( MIDI_R ), ' dB' );
Writeln( ' PCSPEAKER : ', mix4_GetVolume( PCSPEAKER ) );

End;

{ ***** }
{ Process_Mix3 : Processing command line parameters to set }
{ mixer (DSP3.xx). }
{ ***** }
Procedure Process_Mix3;

var iDACFilter,
    iLowPass,
    iADCFilter : Boolean;
    iADCSource : Integer;
    iVol       : Array[0..1] of Integer;
    onoff      : NArgStrings;

```

```

Begin
  onoff[0] := 'OFF';
  onoff[1] := 'ON';

  if ParamCount = 0 then
    Begin
      Writeln( 'Call: ', MIX_NAME, ' [ -ADCFilter:ON/OFF]' );
      Writeln( '[ -DACFilter:ON/OFF][ -LowPass:3.2kHz/8.8kHz]' );
      Writeln( '[ -MIC:0-255][ -CD:0-255[,0-255]][ -LINE:0-255[,0-255]]' );
      Writeln( '[ -VOICE:0-255[,0-255]][ -MIDI:0-255[,0-255]]' );
      Writeln( '[ -MASTER:0-255[,0-255]][ -SOURCE:CD/LINE/MIC]' );
      Writeln( '[ /r ] = Rest mixer' );
      Writeln( '[ /q ] = No output (Quiet)' );
      Exit;
    End;

    iADCFilter := mix3_GetADCFilter; { Change ADC filter setting }
    if GetArg( '-ADCFilter:', _string, @ParStr, 1 ) = 1 then
      Begin
        iIdx := FindString( onoff, ParStr, 2 ) - 1;
        if iIdx >= 0 then
          iADCFilter := Boolean ( iIdx )
        else
          Writeln( 'Invalid ADCFilter setting [ON or OFF]' );
        End;
      mix3_SetADCFilter( iADCFilter );

      { Change DAC filter setting: }
      iDACFilter := mix3_GetDACFilter;
      if GetArg( '-DACFilter:', _string, @ParStr, 1 ) = 1 then
        Begin

```

```

    iIdx := FindString( onoff, ParStr, 2 ) - 1;
    if iIdx >= 0 then iDACFilter := Boolean ( iIdx ) else
        Writeln( 'Invalid DACFilter setting [ON or OFF]' );
End;
mix3_SetDACFilter( iDACFilter );

{ Change low pass filter setting: }
iLowPass := mix3_GetADDACLowPass;
if GetArg( '-LOWPASS:', _string, @ParStr, 1 ) = 1 then
Begin
    if Up( ParStr ) = '3.2KHZ' then iLowPass := FALSE
    else
        if Up( ParStr ) = '8.8KHZ' then iLowPass := TRUE
        else Writeln( 'Invalid low pass setting [3.2kHz or 8.8kHz]' );
End;
mix3_SetADDACLowPass( iLowPass );

{ Change microphone volume: }
iVol[L] := mix3_GetVolume( MIC );
GetArg( '-MIC:', _int, @iVol[L], 1 );
mix3_SetVolume( MIC, iVol[L], 0 );

{ Change CD volume: }
iVol[L] := mix3_GetVolume( CD_L );
iVol[R] := mix3_GetVolume( CD_R );
if GetArg( '-CD:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix3_SetVolume( CD, iVol[L], iVol[R] );

{ Change LINE volume: }
iVol[L] := mix3_GetVolume( LINE_L );
iVol[R] := mix3_GetVolume( LINE_R );
if GetArg( '-LINE:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];

```

[illegible]


```

if GetArg( '/q', _none, NIL, 0 ) = 0 then
  Print_Mix3Settings;
End;

{*****}
{ Process_Mix4 : Processing command line parameters to set
  mixer (DSP4.xx).
{*****}
Procedure Process_Mix4;
var iVol,
    iTreble,
    iBass,
    iADCGain,
    iOUTGain : Array[0..1] of Integer;
    iADC,iDAC : Array[0..1] of Boolean;
    iAGC
        : Integer;
    onoff
        : NArgStrings;

Begin
  onoff[0] := 'OFF';
  onoff[1] := 'ON';

  if ParamCount = 0 then
    Begin
      Writeln( 'Guide ', MIX_NAME );
      Writeln( '[ -MIC:0-255][ -PCSPEAKER:0-255][ -CD:0-255[,0-255]]' );
      Writeln( '[ -LINE:0-255[,0-255]][ -VOICE:0-255[,0-255]]' );
      Writeln( '[ -MIDI:0-255[,0-255]][ -MASTER:0-255[,0-255]]' );
      Writeln( '[ -ADC_L:[ MIDI_L:0/1]      3  [-ADC_R:[ MIDI_L:0/1]' );
      Writeln( '          [,MIDI_R:0/1]      3  [,MIDI_R:0/1]' );
      Writeln( '          [,MIDI:0/1]      3  [,MIDI:0/1]' );
      Writeln( '          [,LINE_L:0/1]      3  [,LINE_L:0/1]' );
    End
  end

```

```

Writeln( '          [,LINE_R:0/1]      3          [,LINE_R:0/1]' );
Writeln( '          [,LINE:0/1]      3          [,LINE:0/1]' );
Writeln( '          [,CD_L:0/1]      3          [,CD_L:0/1]' );
Writeln( '          [,CD_R:0/1]      3          [,CD_R:0/1]' );
Writeln( '          [,CD:0/1]      3          [,CD:0/1]' );
Writeln( '          [,MIC:0/1]]      3          [,MIC:0/1]]' );
Writeln( '[ -OUT_L:[LINE_L:0/1][,LINE_R:0/1][,LINE:0/1]' );
Writeln( '          [,CD_L:0/1][,CD_R:0/1][,CD:0/1][,MIC:0/1]]' );
Writeln( '[ -TREBLE:0-15[,0-15]][ -BASS:0-15[,0-15]]' );
Writeln( '[ -ADCGAIN:0-3[,0-3]][ -OUTGAIN:0-3[,0-3]][-AGC:ON/OFF]' );
Writeln( ' [/r] = Reset mixer' );
Writeln( ' [/q] = No output (Quiet)',#10);
Write('Sample call: ',MIX_NAME,' -MIC:255 -CD:128,255 -LINE:255 ');
Writeln( '-ADC_L:CD_L:1,CD_R:0 -ADC_R:CD_L:0,CD_R:1\n' );
Exit;
End;

                                { Set microphone volume: }
iVol[L] := mix4_GetVolume( MIC );
GetArg( '-MIC:', _int, @iVol[L], 1 );
mix4_SetVolume( MIC, iVol[L], 0 );
                                { Set PC speaker volume: }
iVol[L] := mix4_GetVolume( PCSPEAKER );
GetArg( '-PCSPEAKER:', _int, @iVol[L], 1 );
mix4_SetVolume( PCSPEAKER, iVol[L], 0 );
                                { Set CD volume : }
iVol[L] := mix4_GetVolume( CD_L );
iVol[R] := mix4_GetVolume( CD_R );
if GetArg( '-CD:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix4_SetVolume( CD, iVol[L], iVol[R] );
                                { Set LINE volume : }
iVol[L] := mix4_GetVolume( LINE_L );

```

```

iVol[R] := mix4_GetVolume( LINE_R );
if GetArg( '-LINE:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix4_SetVolume( LINE, iVol[L], iVol[R] );
                                { Set VOICE volume : }
iVol[L] := mix4_GetVolume( VOICE_L );
iVol[R] := mix4_GetVolume( VOICE_R );
if GetArg( '-VOICE:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix4_SetVolume( VOICE, iVol[L], iVol[R] );
                                { Set MIDI volume : }
iVol[L] := mix4_GetVolume( MIDI_L );
iVol[R] := mix4_GetVolume( MIDI_R );
if GetArg( '-MIDI:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix4_SetVolume( MIDI, iVol[L], iVol[R] );
                                { Set MASTER volume : }
iVol[L] := mix4_GetVolume( MASTER_L );
iVol[R] := mix4_GetVolume( MASTER_R );
if GetArg( '-MASTER:', _int, @iVol, 2 ) = 1 then
    iVol[R] := iVol[L];
mix4_SetVolume( MASTER, iVol[L], iVol[R] );
                                { Set input sources for left ADC : }
iNumStrings := GetArg( '-ADC_L:', _string, @Argumente, 7 ); {xxx}

iADC[L] := Boolean ( mix4_GetADCSourceL( MIDI_L ) );
iADC[R] := Boolean ( mix4_GetADCSourceL( MIDI_R ) );
if FindString(Argumente,'MIDI_L:0',iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'MIDI_R:0',iNumStrings) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'MIDI_L:1',iNumStrings) <> 0 then iADC[L]:=TRUE;
if FindString(Argumente,'MIDI_R:1',iNumStrings) <> 0 then iADC[R]:=TRUE;
if FindString( Argumente, 'MIDI:0', iNumStrings ) <> 0 then

```

```

Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
if FindString( Argumente, 'MIDI:1', iNumStrings ) <> 0 then
  Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourceL( MIDI_L, iADC[L] );
mix4_SetADCSourceL( MIDI_R, iADC[R] );

iADC[L] := Boolean ( mix4_GetADCSourceL( LINE_L ) );
iADC[R] := Boolean ( mix4_GetADCSourceL( LINE_R ) );
if FindString(Argumente,'LINE_L:0',iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'LINE_R:0',iNumStrings) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'LINE_L:1',iNumStrings) <> 0 then iADC[L]:=TRUE;
if FindString(Argumente,'LINE_R:1',iNumStrings) <> 0 then iADC[R]:=TRUE;
  if FindString( Argumente, 'LINE:0', iNumStrings ) <> 0 then
    Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
  if FindString( Argumente, 'LINE:1', iNumStrings ) <> 0 then
    Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourceL( LINE_L, iADC[L] );
mix4_SetADCSourceL( LINE_R, iADC[R] );

iADC[L] := Boolean ( mix4_GetADCSourceL( CD_L ) );
iADC[R] := Boolean ( mix4_GetADCSourceL( CD_R ) );
if FindString( Argumente,'CD_L:0',iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString( Argumente,'CD_R:0',iNumStrings) <> 0 then iADC[R]:=FALSE;
if FindString( Argumente,'CD_L:1',iNumStrings) <> 0 then iADC[L]:=TRUE;
if FindString( Argumente,'CD_R:1',iNumStrings) <> 0 then iADC[R]:=TRUE;
if FindString( Argumente,'CD:0', iNumStrings ) <> 0 then
  Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
if FindString( Argumente, 'CD:1', iNumStrings ) <> 0 then
  Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourceL( CD_L, iADC[L] );
mix4_SetADCSourceL( CD_R, iADC[R] );

```

```

iADC[L] := Boolean ( mix4_GetADCSourceL( MIC ) );
if FindString(Argumente,'MIC:0', iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'MIC:1', iNumStrings) <> 0 then iADC[L]:=TRUE;
mix4_SetADCSourceL( MIC, iADC[L] );

        { Set input sources for right ADC : }
iNumStrings := GetArg( '-ADC_R:', _string, @Argumente, 7 );

iADC[L] := Boolean ( mix4_GetADCSourceR( MIDI_L ) );
iADC[R] := Boolean ( mix4_GetADCSourceR( MIDI_R ) );
if FindString(Argumente,'MIDI_L:0',iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'MIDI_R:0',iNumStrings) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'MIDI_L:1',iNumStrings) <> 0 then iADC[L]:=TRUE;
if FindString(Argumente,'MIDI_R:1',iNumStrings) <> 0 then iADC[R]:=TRUE;
if FindString( Argumente, 'MIDI:0', iNumStrings ) <> 0 then
    Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
if FindString( Argumente, 'MIDI:1', iNumStrings ) <> 0 then
    Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourceR( MIDI_L, iADC[L] );
mix4_SetADCSourceR( MIDI_R, iADC[R] );

iADC[L] := Boolean ( mix4_GetADCSourceR( LINE_L ) );
iADC[R] := Boolean ( mix4_GetADCSourceR( LINE_R ) );
if FindString(Argumente,'LINE_L:0',iNumStrings) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'LINE_R:0',iNumStrings) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'LINE_L:1',iNumStrings) <> 0 then iADC[L]:=TRUE;
if FindString(Argumente,'LINE_R:1',iNumStrings) <> 0 then iADC[R]:=TRUE;
if FindString( Argumente, 'LINE:0', iNumStrings ) <> 0 then
    Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
if FindString( Argumente, 'LINE:1', iNumStrings ) <> 0 then
    Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourceR( LINE_L, iADC[L] );

```

```

mix4_SetADCSourcER( LINE_R, iADC[R] );

iADC[L] := Boolean ( mix4_GetADCSourcER( CD_L ) );
iADC[R] := Boolean ( mix4_GetADCSourcER( CD_R ) );
if FindString(Argumente,'CD_L:0',iNumStrings ) <> 0 then iADC[L]:=FALSE;
if FindString(Argumente,'CD_R:0',iNumStrings ) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'CD_L:1',iNumStrings ) <> 0 then iADC[L]:=TRUE;
if FindString(Argumente,'CD_R:1',iNumStrings ) <> 0 then iADC[R]:=TRUE;
if FindString( Argumente, 'CD:0',iNumStrings ) <> 0 then
    Begin iADC[L]:=FALSE; iADC[R]:=FALSE; End;
if FindString( Argumente, 'CD:1', iNumStrings ) <> 0 then
    Begin iADC[L]:=TRUE; iADC[R]:=TRUE; End;
mix4_SetADCSourcER( CD_L, iADC[L] );
mix4_SetADCSourcER( CD_R, iADC[R] );

iADC[R] := Boolean ( mix4_GetADCSourcER( MIC ) );
if FindString(Argumente,'MIC:0', iNumStrings ) <> 0 then iADC[R]:=FALSE;
if FindString(Argumente,'MIC:1', iNumStrings ) <> 0 then iADC[R]:=TRUE;
mix4_SetADCSourcER( MIC, iADC[R] );

                                { Set output sources : }
iNumStrings := GetArg( '-OUT:', _string, @Argumente, 7 );

iDAC[L] := Boolean ( mix4_GetOUTSource( LINE_L ) );
iDAC[R] := Boolean ( mix4_GetOUTSource( LINE_R ) );
if FindString(Argumente,'LINE_L:0',iNumStrings)<> 0 then iDAC[L] := FALSE;
if FindString(Argumente,'LINE_R:0',iNumStrings)<> 0 then iDAC[R] := FALSE;
if FindString(Argumente,'LINE_L:1',iNumStrings)<> 0 then iDAC[L] := TRUE;
if FindString(Argumente,'LINE_R:1',iNumStrings)<> 0 then iDAC[R] := TRUE;
if FindString( Argumente, 'LINE:0', iNumStrings ) <> 0 then
    Begin iDAC[L] := FALSE; iDAC[R] := FALSE; End;
if FindString( Argumente, 'LINE:1', iNumStrings ) <> 0 then

```

```

Begin iDAC[L] := TRUE; iDAC[R] := TRUE; End;
mix4_SetOUTSource( LINE_L, iDAC[L] );
mix4_SetOUTSource( LINE_R, iDAC[R] );

iDAC[L] := Boolean ( mix4_GetOUTSource( CD_L ) );
iDAC[R] := Boolean ( mix4_GetOUTSource( CD_R ) );
if FindString(Argumente,'CD:L:0',iNumStrings) <> 0 then iDAC[L] := FALSE;
if FindString(Argumente,'CD:R:0',iNumStrings) <> 0 then iDAC[R] := FALSE;
if FindString(Argumente,'CD:L:1',iNumStrings) <> 0 then iDAC[L] := TRUE;
if FindString(Argumente,'CD:R:1',iNumStrings) <> 0 then iDAC[R] := TRUE;
if FindString( Argumente, 'CD:0', iNumStrings ) <> 0 then
    Begin iDAC[L] := FALSE; iDAC[R] := FALSE; End;
if FindString( Argumente, 'CD:1', iNumStrings ) <> 0 then
    Begin iDAC[L] := FALSE; iDAC[R] := TRUE; End;
mix4_SetOUTSource( CD_L, iDAC[L] );
mix4_SetOUTSource( CD_R, iDAC[R] );

iDAC[L] := Boolean ( mix4_GetOUTSource( MIC ) );
if FindString(Argumente,'MIC:0',iNumStrings) <> 0 then iDAC[L] := FALSE;
if FindString(Argumente,'MIC:1',iNumStrings) <> 0 then iDAC[L] := TRUE;
mix4_SetOUTSource( MIC, iDAC[L] );

{ Set treble: }

iTreble[L] := mix4_GetTreble( CH_LEFT );
iTreble[R] := mix4_GetTreble( CH_RIGHT );
if GetArg( '-TREBLE:', _int, @iTreble, 2 ) = 1 then
    iTreble[R] := iTreble[L];
mix4_SetTreble( iTreble[L], iTreble[R] );

{ Set bass: }

iBass[L] := mix4_GetBass( CH_LEFT );
iBass[R] := mix4_GetBass( CH_RIGHT );
if GetArg( '-BASS:', _int, @iBass, 2 ) = 1 then
    iBass[R] := iBass[L];

```

```

mix4_SetBass( iBass[L], iBass[R] );

                                { Set input preamplifier: }
iADCGain[L] := mix4_GetADCGain( CH_LEFT );
iADCGain[R] := mix4_GetADCGain( CH_RIGHT );
if GetArg( '-ADCGAIN:', _int, @iADCGain, 2 ) = 1 then
    iADCGain[R] := iADCGain[L];
mix4_SetADCGain( iADCGain[L], iADCGain[R] );

                                { Set output preamplifier: }
iOUTGain[L] := mix4_GetOUTGain( CH_LEFT );
iOUTGain[R] := mix4_GetOUTGain( CH_RIGHT );
if GetArg( '-OUTGAIN:', _int, @iOUTGain, 2 ) = 1 then
    iOUTGain[R] := iOUTGain[L];
mix4_SetOUTGain( iOUTGain[L], iOUTGain[R] );

                                { Set Automatic Gain Control: }
iAGC := integer(Mix4_GetAGC);
if GetArg( '-AGC:', _string, @ParStr, 1 ) = 1 then
Begin
    iIdx := FindString( onoff, ParStr, 2 ) - 1;
    if iIdx >= 0 then iAGC := iIdx else
        Writeln( 'Invalid AGC setting [ON or OFF]' );
End;
mix4_SetAGC( iAGC );

                                { Output current mixer setting? }
if GetArg( '/q', _none, NIL, 0 ) = 0 then
    Print_Mix4Settings;
End;

{ ***** }
{ --          M A I N    P R O G R A M          -- }
{ ***** }

```



```

var SBB : SBBASE;

Begin
  if sb_GetEnviron( SBB, getenv( 'BLASTER' ) ) <> NO_ERROR then
    Begin
      Writeln( 'BLASTER environment variable not available' );
      Halt( 0 );
    End;

    dsp_SetBase( SBB, TRUE ); { Get DSP version }
    mix_SetBase( SBB, Boolean ( GetArg( '/r', _none, NIL, 0 ) ) );

    if( SBB.uDspVersion < $0400 ) then
      Process_Mix3
    else
      Process_Mix4;
End.

```