```
'Messung über die Soundkarte
'Programmlisting, erstellt mit Visual Basic
'30. Januar 2005, F.-P. Zantis

Option Explicit
Option Base 1

Const wertebereich As Long = 2 ^ 16 '2-Byte-Auflösung
Const samplerate As Long = 11025 'samplewerte pro Sekunde der von der Soundkarte kommenden Daten
Const samplewerte As Long = 8192 'Anzahl Abtastwert pro Recording
Dim paketzahl As Long 'Anzahl der Pakete zu je samplewerte

Dim zeitlog As Single 'Dauer der Messung

'Array für ein Paket Rohdaten
Dim gwavedata() As Integer

'Array für die demodulierten Daten die in einem Fenster dargestellt werden
Dim signaldatademoduliert() As Long
'Array für das Fensterabbild
Dim signaldatademo() As Long

'Displaypixel festhalten für das Paketdisplay
Dim Displaypaketx As Long
Dim Displaypakety As Long

'Displaypixel festhalten für das Displaysignal
Dim Displaysignalx As Long
Dim Displaysignaly As Long

'Zähler der erfaßten Pakete (Blöcke)
Dim blockindex As Long

'globaler Zwischenspeicher
Dim gzw1 As Long

'Gleichanteil und Kalibrierung
Dim dcvalue As Long
Dim calflag As Boolean
Const calwerte As Long = samplewerte * 5 '5 Pakete für die Kalibrierung
Dim gcalvalues(calwerte)


Private Sub Form_Initialize()
    Dim lreturn As Long
    Form1.TextEingabeBlockzahl.text = 5
    lreturn = initialisierung()
    calflag = False
End Sub

Private Sub starten_Click()
    Dim lreturn As Long
    Dim lreturnstring As String
    Dim lzaehler As Long
'--------------------------------------------------------------------------------
    'Einstellungen der SoundProcessor.ocx bzw. der Soundkarte
    'erstes Soundgerät (normalerweise die Soundkarte) verwenden
    SndProc1.Device = 0
    'Vorgabe der Abtastrate
    SndProc1.SPS = samplerate
    'Festlegen der einzulesenden samplewerte
    SndProc1.Samples = samplewerte
    'Anzahl der zu verwendenden Buffer vorgeben
    'bei 5 Buffern kann man davon ausgehen, dass lückenlos aufgezeichnet wird
    'unter Umständen reichen auch 3
    SndProc1.Buffers = 5
    'Aktivieren des Subclassing (das Ende der Aufnahme stößt die nächste Aufnahme an)
    SndProc1.LoopMethod = 1
    'Pfad und Name der zu speichernden Datei festlegen
    SndProc1.filename = "C:\recording.wav"
    'Einschalten des Aufnahme-Modus
    SndProc1.Recording = True
    'Abschalten des Analyse-Modus
    SndProc1.analysis = None
    'Vorgabe der maximalen Aufnahmezeit
    SndProc1.MaxRecordingTime = CLng(Form1.TextAusgabeAufnahmezeit.text)
    'Kanal zur Soundkarte öffnen
    SndProc1.Init
'--------------------------------------------------------------------------------
```

```
    lreturn = volcontrol()

    'die Aufnahme starten
    lreturn = initialisierung()

    lreturnstring = statusoutput("Aufnahme gestartet", False)
    lreturnstring = statusoutput("Blockanzahl: " & CStr(paketzahl), True)

    zeitlog = Timer

    'Aufnahmestart
    blockindex = 0
    Call SndProc1.Start
End Sub

Private Sub SndProc1_Complete(ByVal lblockindex As Long, ByVal lBufferIndex As Long, ByVal dt As
Double)
    'wenn ein Packet von der Größe samplewerte eingelesen wurde
    'wird hier weitergemacht
    Dim lzaehler As Long
    Dim lzeitlog As Single
    Dim lreturn As Long
    Dim lreturnstring As String
    Dim lzw As Long
    Dim lpx As Long
    Dim lpy As Long

    'Abholen der Daten vom ersten Kanal in das Array gwavedata
    'lreturn enthält jetzt die Anzahl der tatsächlich übergebenen samplewerte
    'lreturn entspricht samplewerte
    lreturn = SndProc1.WaveData(gwavedata(), 1)
    blockindex = blockindex + 1

    'da die OCX noch über die angegebene Blockzahl hinaus aufnimmt, wird
    'die Aufzeichnung beendet, sobald die im blockindex angegebene Blockzahl erreicht wurde
    If lblockindex > blockindex Then
        lzeitlog = Timer - zeitlog
        lreturnstring = statusoutput(CStr(lzeitlog & " s"), False)
        Exit Sub
    End If

    lreturnstring = statusoutput("Block " & CStr(blockindex), False)

    If calflag = False Then
        'Ausgabe eines Paketes mit der Anzahl samplewerte
        Form1.Displaypaket.Cls
        lreturn = mittelliniedisplaypaket()
        lreturn = mittelliniedisplaysignal()

        For lzaehler = 1 To samplewerte Step 1

            'Trägeranzeige
            lpy = CLng(gwavedata(lzaehler - 1))
            lpy = py(Displaypakety, lpy) 'Umrechnung auf die Displayhöhe
            lpx = Displaypaketx / samplewerte * lzaehler 'Umrechnung auf die Displayweite
            Form1.Displaypaket.PSet (lpx, lpy)

            'Demodulation
            'nur die positive Halbschwingung betrachten und dabei die Grenzen des Arrays
berücksichtigen
            If (gwavedata(lzaehler - 1) > 0) And (lzaehler > 1) And (lzaehler < samplewerte - 1)
Then
                'wenn der aktuelle Wert größer ist als der nächste und größer ist als der vorherige:
dann speichern
                If (gwavedata(lzaehler - 1) > gwavedata(lzaehler)) And (gwavedata(lzaehler - 1) >
gwavedata(lzaehler - 2)) Then
                    'nur wenn die Bedingung erfüllt ist, wird gzw1 geändert
                    'ansonsten wird der Wert von gzw1 beibehalten
                    gzw1 = CLng(gwavedata(lzaehler - 1)) - dcvalue
                End If
            End If

            'Einschreiben der Werte in ein Array
            signaldatademoduliert(lzaehler) = gzw1

            lpx = Displaysignalx / (samplewerte * paketzahl) * lzaehler 'Umrechnung auf die
Displayweite
```

```
            lpx = lpx + (Displaysignalx / paketzahl * (blockindex - 1)) 'Darstellung aller Blöcke im
Display
            lpy = py(Displaysignaly, gzw1)

            If lzaehler > 1 Then
                Form1.Displaysignal.Line -(lpx, lpy)
            Else
                Form1.Displaysignal.PSet (lpx, lpy)
            End If
        Next lzaehler

    Else
        For lzaehler = 1 To samplewerte Step 1
            'Demodulation
            'nur die positive Halbschwingung betrachten und dabei die Grenzen des Arrays
berücksichtigen
            If (gwavedata(lzaehler - 1) > 0) And (lzaehler > 1) And (lzaehler < samplewerte - 1)
Then
                'wenn der aktuelle Wert größer ist als der nächste und größer ist als der vorherige:
dann speichern
                If (gwavedata(lzaehler - 1) > gwavedata(lzaehler)) And (gwavedata(lzaehler - 1) >
gwavedata(lzaehler - 2)) Then
                    'nur wenn die Bedingung erfüllt ist, wird gzw1 geändert
                    'ansonsten wird der Wert von gzw1 beibehalten
                    gzw1 = CLng(gwavedata(lzaehler - 1))
                End If
            End If
            lzw = lzaehler + (samplewerte * (blockindex - 1))
            If lzw > UBound(gcalvalues) Then
            Else
                gcalvalues(lzw) = gzw1
            End If
        Next lzaehler
    End If
    If blockindex >= CInt(Form1.TextEingabeBlockzahl.text) Then
        Call stoppen_Click
    End If
End Sub

Private Sub stoppen_Click()
    Dim lreturnstring As String
    'Aufnahme stoppen und ocx Rücksetzen
    SndProc1.Recording = False
    SndProc1.Reset
    lreturnstring = statusoutput("Recording gestoppt", False)
End Sub

Private Sub SndProc1_DoneRecordig()
    Dim lcounter As Long
    Dim lsum As Long
    Dim lreturn2 As String

    Call stoppen_Click
    If calflag = False Then
        lreturn2 = statusoutput("Aufnahme komplett", False)
    Else
        calflag = False
        lsum = 0
        For lcounter = 1 To calwerte Step 1
            lsum = gcalvalues(lcounter) + lsum
        Next lcounter
        dcvalue = lsum / calwerte
        Form1.Textausgaben.text = vbCrLf & "Kalibrierung beendet" & vbCrLf & "DC-Value: " &
CStr(dcvalue) & vbCrLf & Form1.Textausgaben.text
    End If

End Sub

Private Sub UserForm_Terminate()
    SndProc1.Reset
End Sub

Private Sub cal_Click()
    Dim lreturn As Integer
    Dim lreturnstring As String

    'Einstellungen der SoundProcessor.ocx bzw. der Soundkarte
    'erstes Soundgerät (normalerweise die Soundkarte) verwenden
    SndProc1.Device = 0
```

```
    'Vorgabe der Abtastrate
    SndProc1.SPS = samplerate
    'Festlegen der einzulesenden samplewerte
    SndProc1.Samples = samplewerte
    'Anzahl der zu verwendenden Buffer vorgeben
    'bei 5 Buffern kann man davon ausgehen, dass lückenlos aufgezeichnet wird
    'unter Umständen reichen auch 3
    SndProc1.Buffers = 5
    'Aktivieren des Subclassing (das Ende der Aufnahme stößt die nächste Aufnahme an)
    SndProc1.LoopMethod = 1
    'Pfad und Name der zu speichernden Datei festlegen
    SndProc1.filename = "C:\recording.wav"
    'Einschalten des Aufnahme-Modus
    SndProc1.Recording = True
    'Abschalten des Analyse-Modus
    SndProc1.analysis = None
    'Vorgabe der maximalen Aufnahmezeit
    SndProc1.MaxRecordingTime = calwerte / samplerate
    'Kanal zur Soundkarte öffnen
    SndProc1.Init

    lreturnstring = statusoutput("bitte warten", True)
    lreturnstring = statusoutput("Kalibrierung gestartet", False)

    lreturn = volcontrol()

    'die Aufnahme starten
    blockindex = 0
    calflag = True
    Call SndProc1.Start

End Sub

Private Function py(pymax As Long, y As Long) As Long
    'zu zeichnender Punkt y berechnen
    py = pymax / 2 - pymax / wertebereich * y
End Function

Private Function mittelliniedisplaypaket() As Integer
    Form1.Displaypaket.Line (1, Displaypakety / 2)-(Displaypaketx, Displaypakety / 2), RGB(150, 150,
255)
End Function

Private Function mittelliniedisplaysignal() As Integer
      Form1.Displaysignal.Line (1, Displaysignaly / 2)-(Displaysignalx, Displaysignaly / 2),
RGB(150, 150, 255)
End Function

Private Function initialisierung() As Long
    Dim lreturn As Integer

    'Form anzeigen
    Form1.Show

    'Schreibfarbe und -stärke in den Displays festlegen
    Form1.Displaypaket.BackColor = RGB(255, 255, 255)
    Form1.Displaypaket.DrawWidth = 1
    Form1.Displaysignal.BackColor = RGB(255, 255, 255)
    Form1.Displaysignal.DrawWidth = 1

    'Displaydimensionierung festhalten
    Displaypaketx = Form1.Displaypaket.ScaleWidth
    Displaypakety = Form1.Displaypaket.ScaleHeight
    Displaysignalx = Form1.Displaysignal.ScaleWidth
    Displaysignaly = Form1.Displaysignal.ScaleHeight

    If Form1.TextEingabeBlockzahl.text = "" Then
        Form1.TextEingabeBlockzahl.text = 5
    End If
    paketzahl = CInt(Form1.TextEingabeBlockzahl.text)

    Form1.TextAusgabeAufnahmezeit.Locked = True
    Form1.TextAusgabeAufnahmezeit.text = CStr(CSng(samplewerte / samplerate * paketzahl))

    'Dimensionierung der Ausgabearrays
    ReDim signaldatademo(Displaysignalx)
    ReDim signaldatademoduliert(samplewerte * paketzahl)

    'lreturn = volcontrol()
```

```
End Function

Private Sub TextEingabeBlockzahl_Change()
    Dim lret As Long
    lret = initialisierung()
End Sub

Private Function volcontrol() As Integer
    'Einstellen des Vol-Control
    Dim lappid As Long
    Dim lreturn As Long
    Dim lreturn2 As String
    Const duration As Integer = 1 's Wartezeit für die Kontrolle des Lin-Input

    lappid = Shell("sndvol32", vbNormalFocus)
    AppActivate lappid
    SendKeys String:="%", Wait:=True
    SendKeys String:="OE", Wait:=True
    SendKeys String:="%A", Wait:=True
    SendKeys String:="{ENTER}", Wait:=True

    lreturn2 = statusoutput("wegen volcontrol", False)
    lreturn2 = statusoutput(CStr(duration) & "s warten", False)

    lreturn = warteetwas(duration)
    AppActivate lappid
    SendKeys String:="{TAB}", Wait:=True
    SendKeys String:="{UP 5}", Wait:=True
    SendKeys String:="{DOWN 5}", Wait:=True
    SendKeys String:="%", Wait:=True
    SendKeys String:="OB", Wait:=True
End Function

Private Function warteetwas(timelimit As Integer) As Integer
     Dim starttime As Single
     starttime = Timer
     Do Until starttime + timelimit < Timer
        DoEvents
     Loop
End Function
Private Function statusoutput(text As String, clearflag As Boolean) As String
    If clearflag = False Then
        Form1.Textausgaben.text = text & vbCrLf & Form1.Textausgaben.text & vbCrLf
    Else
        Form1.Textausgaben.text = text & vbCrLf
    End If
End Function
```