CE – Programmierung

GPS – ein Fall für den PDA

Das Global Positioning System (GPS) ist reizvoll, um es mit dem PDA in Verbindung zu bringen. Gerade die Mobilität, die mit den kleinen Rechnern erreicht wird, wird durch die Nutzung des Satellitensignals erhöht. Die Datenübertragung zwischen GPS – Empfänger und PDA ist kinderleicht und kann daher genutzt werden.

Vor einigen Jahren schon hatten wir in der Redaktion einen GPS – Empfänger (GPS-Maus) mit seriellem Anschluss und einem PS2 – Stecker, der das +5V – Signal vom Laptop lieferte. Dazu gehörte eine Routenplanersoftware, die GPS – fähig war. Nichts Besonderes mehr in der heutigen Zeit.

Irgendwie war der Stecker abgequetscht und das Kabel war damit untauglich. 8 Strippen hingen in der Luft und keiner traute sich mehr, eine Reparatur vorzunehmen.

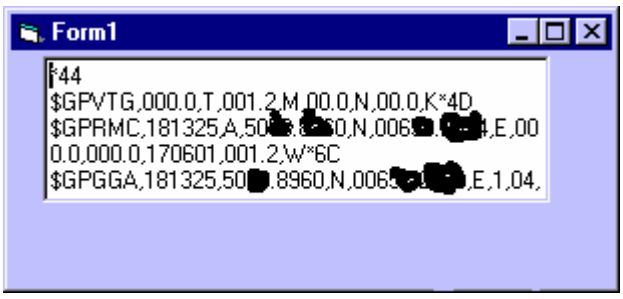
Letzte Woche kam mir das Kabel mit der GPS-Maus wieder in die Quere. Ich dachte mir: Ist doch schade, dass das Teil unbenutzt hier herumliegt. Ich nahm das Kabel mit nach Hause und fummelte das Gerät am Wochenende wieder zurecht.

Letztendlich war es ganz einfach, wenn man ein bisschen Ahnung von der Hardware und der seriellen Schnittstelle hat.

Geholfen hat hier vor allem, dass man sich bei Garmin (Hersteller) an die übliche Farbgebung von plus (rot) und minus (schwarz) gehalten hat. Mit (blau) wäre ich auch noch zurechtgekommen.

Den dritten Pin (Txd) habe ich dann per try and error probiert. Irgendwo mussten Daten herauskommen, wenn das Gerät noch intakt ist.

Zuvor hatte ich mir ein kleines Terminalprogramm in VB6 geschrieben, das im Fenster irgendwelche Erfolge anzeigen sollte. So sieht das Programm aus. Auch hier musste ich die echten Koordinaten schwärzen.



Der Code für dieses kleine Programm ist auch ganz einfach.

```
Option explicit
Private Sub Form_Load()
MSComm1.CommPort = 1
MSComm1.Settings = "4800,n,8,1"
```

```
MSComm1.RThreshold = 1
MSComm1.PortOpen = True
End Sub

Private Sub MSComm1_OnComm()
Text1 = Text1 + MSComm1.Input
If Len(Text1) > 210 Then Text1 = ""
End Sub
```

Ich dachte zuerst mich erinnern zu können, dass die Datenübertragung mit 2400 bits per sec. Stattfinden würde. Als ich dann die ersten Zeichen im Fenster erkannte, waren sie fürchterlich unsinnig.

Als ich dann auf 4800 umstellte, machte die Anzeige einigermaßen Sinn. So langsam kam jetzt auch die Erinnerung zurück, dass der Empfänger verschiedene Protokolle liefert.

Dieser Garmin-Empfänger liefert die folgenden Zeilen:

(ich habe die Koordinaten etwas geändert, damit nicht ganz Eifrige später bei mir im Garten stehen. Wenn Sie diese (geänderten) Koordinaten nachsehen, so werden Sie entdecken, dass ich während meiner Versuche mitten im Rhein gestanden haben muß.)

Was bedeuten nun die im Protokoll dargestellten Informationen?

Da GPS natürlich genormt ist, findet man im Internet schnell die Erklärungen.
Der untere (fette) Ausdruck bedeutet:

Der Ausdruck **$** leitet immer ein Telegramm ein. **GP** steht für das Gerät, hier ein GPS – Empfänger. **CGA** gehört, wie unten angezeigt, zu dem Begriff **Global Positioning System Fix Data**. Die weiteren Inhalte sind jeweils mit Kommata getrennt.

$GPGGA,144931,5054.9257,N,00659.2189,E,0,00,,,M,,M,,*5

```
GGA - Global Positioning System Fix Data
 Time, Position and fix related data for a GPS receiver.

                                                    11
        1         2       3 4        5 6 7  8   9  10 |  12 13  14   15
        |         |       | |        | | |  |   |   | |   | |   |    |
 $--GGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh

 Field Number:
  1) Universal Time Coordinated (UTC)
  2) Latitude
  3) N or S (North or South)
  4) Longitude
  5) E or W (East or West)
  6) GPS Quality Indicator,
     0 - fix not available,
     1 - GPS fix,
     2 - Differential GPS fix
  7) Number of satellites in view, 00 - 12
  8) Horizontal Dilution of precision
  9) Antenna Altitude above/below mean-sea-level (geoid)
 10) Units of antenna altitude, meters
 11) Geoidal separation, the difference between the WGS-84 earth
     ellipsoid and mean-sea-level (geoid), "-" means mean-sea-level
     below ellipsoid
 12) Units of geoidal separation, meters
```

```
 13) Age of differential GPS data, time in seconds since last SC104
     type 1 or 9 update, null field when DGPS is not used
 14) Differential reference station ID, 0000-1023
 15) Checksum
```

Wenn man meinen Empfangsstring, den $GPCGA z.B. nach der obigen Definition auflöst, so kann man erkennen:

Empfangszeit: **144931 =** 14:49:31 Uhr (natürlich **UTC** (Universal Time Coordinated (Weltzeit in Greenwich), wir haben dann eine Stunde (Winter oder Normalzeit) oder 2 Stunden (Sommerzeit) Vorlauf.
Die Koordinaten für die nördliche Position: **5054.9257 =** 50 Grad, 54 Minuten, (9257\100) * 0,6 Sekunden (Mit Sekunden arbeiten die meisten Navigationssysteme). Der letztere mathematische Ausdruck rechnet aus max. 10000 in max. 60 Sekunden um.

Die Koordinaten für die östliche Position: **00659.2189** = 6 Grad, 59 Minuten, (2189\100) * 0.6 Sekunden.

In den nächsten Abteilungen sieht es derzeit mau aus. Es wird keine Empfangsqualität angezeigt und kein Satellit empfangen. Das soll sich aber noch ändern .

Das nächste Telegramm wird mit **RMC** eingeleitet. Die Beschreibung lautet: **Recommended Minimum Navigation**. Es genügt also, dieses Telegramm auszulesen, um über die Koordinaten Bescheid zu wissen. Nicht angezeigt wird hier die Anzahl der sichtbaren Satelliten.

$GPRMC,144931,V,5054.9257,N,00659.2189,E,000.0,000.0,170601,001.2,W*7

```
RMC - Recommended Minimum Navigation Information
                                                      12
        1          2 3       4 5        6 7   8   9    10  11|
        |          | |       | |        | |   |   |    |   | |
 $--RMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,xxxx,x.x,a*hh

 Field Number:
  1) UTC Time
  2) Status, V = Navigation receiver warning
  3) Latitude
  4) N or S
  5) Longitude
  6) E or W
  7) Speed over ground, knots
  8) Track made good, degrees true
  9) Date, ddmmyy
 10) Magnetic Variation, degrees
 11) E or W
 12) Checksum
```

Bleibt noch das letzte Telegramm, das von meinem Garmin geliefert wird. Es beginnt mit dem Kürzel **VTG** für **Track mode and Ground speed**.

$GPVTG,,T,,M,,N,,K*4

```
VTG - Track made good and Ground speed

        1   2 3   4 5   6 7   8 9
        |   | |   | |   | |   | |
 $--VTG,x.x,T,x.x,M,x.x,N,x.x,K*hh
```

```
Field Number:
 1) Track Degrees
 2) T = True
 3) Track Degrees
 4) M = Magnetic
 5) Speed Knots
 6) N = Knots
 7) Speed Kilometers Per Hour
 8) K = Kilometers Per Hour
 9) Checksum
```

Dieses Telegramm wird mir sicherlich hier am Arbeitsplatz immer leere Werte liefern, da hier Geschwindigkeiten und Richtungswinkel gemessen werden und der Empfänger fest auf der Balkonbrüstung liegt.

(Ich habe an meinem Arbeitsplatz das Problem, dass ich eine große überdachte Terrasse habe und nur einen kleinen Schlitz zum offenen Himmel, Der Rest wird von Bäumen verdeckt. Daher muss ich warten, bis wieder einmal ein Satellit vorbeischaut.).

Das untere Bild zeigt den Empfang von 4 Satelliten an. Es ist mittlerweile schon nach 20 Uhr. Leider muss ich auch hier einige Teile schwärzen.

In der Zwischenzeit war ich einmal kurz weg und hatte das Gerät im Auto installiert. Deshalb kann ich hier schon von Erfolgen berichten.

Die Anzeige der Geschwindigkeit geht wunderbar. Die angezeigte Geschwindigkeit in km/h scheint plausibel zu sein – sie stimmte mit dem Tacho des Autos gut überein (mit der üblichen 10% Abweichung). Wer hier genauer anzeigt, kann ich noch nicht sagen. Ich schätze aber, dass das GPS – System hier genauer arbeitet.

PS. Nachdem dies mit der Geschwindigkeitsmessung so gut geklappt hat, werde ich mir demnächst meinen kleinen Roller, der in Garmisch steht, mit dem GPS – Empfänger und einem PDA ausstatten.

Für diesen Roller gibt es offensichtlich keine Ersatzteile mehr (bei Tchibo damals für 3 Mille gekauft).

Seit einiger Zeit ist die Tachowelle defekt und alle Versuche, sie zu erneuern versagten. Jetzt kommt da eben ein PDA mit GPS als Tacho dran – ja, wenn es doch anders nicht geht.

Vielleicht programmiere ich mir einen schönen analogen Tacho, der dann beim Aufsteigen zum Eibsee (wo er kaum hoch kommt) mit elektronischen Tricks die doppelte Geschwindigkeit anzeigt.

In der Zwischenzeit ist das Programm gewachsen (siehe Screenshots weiter unten). Die Winkelmessung kam hinzu, die ausgesprochen gut arbeitet. Im ersten Datenfeld des VTG - Telegramms wird die aktuelle Fortbewegungsrichtung (zwischen zwei Telegrammen) angezeigt. 360 oder 0 bedeutet NORD, 90 Grad ist OST, 180 Grad ist SÜD und schließlich sind 270 Grad WEST.

Das nächste Datenfeld im VTG – Telegramm zeigt den magnetischen Winkel an. Dieses Feld habe ich nicht ausgewertet. Es unterscheidet sich bei uns um ca. 1.2 Grad.

Hinzu kam die grafische Darstellung des Trackwinkels.

Ein wenig Probleme hatte ich (habe ich eigentlich immer noch) mit der Höhenmessung. Hier ist die Anzeige mancher Werte problematisch. Nach der Definition des VTG – Protokolls ist im GGA – Telegramm die Höhe der Antenne enthalten. Um dieses überhaupt berechnen zu können, benötigt man mindestens 4 Satelliten. Da die Erde keine exakte Kugel ist, sondern ein Ellipsoid, der ein wenig verformt ist, muss man bei dem Höhewert mit einem Korrekturfaktor arbeiten. Dieser Faktor wird im Protokoll als nächstes Datenfeld mitgeliefert. Benutzt man beide gemessenen Höhenwerte (Höhe – Korrektur), so kommen oft plausible Werte zu Tage. Oft aber sind die Werte recht unsinnig oder aber recht ungenau.

Zum Beispiel:

Köln liegt laut Atlas auf 50 Meter ü. NN. Ich schätze einmal, dass man den Bahnhofplatz für die Höhenbestimmung genommen hat. Die Domplatte soll 56 m hoch liegen. In meinem Büro müssten dann ca. 56+10 (3. Stock) = 66 Meter gemessen werden. Bei mir zu Hause dürfte ich so um die 52 Meter messen.

Im Büro habe ich schon Minuswerte gehabt; die Maximalwerte waren über 200 Meter. Dieses ist natürlich eine sehr große Spanne, die man sich nicht erklären kann. Zu Hause misst er meistens um die 80 Meter Höhe ohne Korrektur.

Aktueller Wert: 83.3 Korrektur 47.3 macht 83.3 – 47.3 = 36 Meter.

Bei einem Anruf im Landesvermessungsamt in Bonn wurde mir bestätigt, dass die Höhenmessung problematisch sein kann.

Man sprach hier davon, dass man bei uns in Deutschland in der Regel 40 Meter vom gemessenen Höhenwert abzieht. Sicherlich dürfte der Korrekturwert genauer sein, denn er misst den Unterschied zwischen WGS 84 – Modell und Meereshöhe (siehe GGA - Protokoll oben). Ich muss einfach noch ein paar Messungen machen, um zu sehen, wie der Empfänger sich wann und wo und wie verhält.

Synchronisieren mit GPS

Wenn man die exakte Zeit geliefert bekommt, so sollte man dies nutzen, um die Systemzeit des PDA's automatisch zu stellen. Dieses ist nicht ganz so einfach, wie es auf den ersten Blick aussieht. Windows CE hat nicht die Fähigkeit, Datum und Zeit direkt zu beeinflussen. Es ist halt ein abgespecktes Betriebssystem. In VB6 kann man ja time="01:02:54" oder date="01-01-2001" nutzen, um die Rechnerzeit einzustellen.

Bei CE müssen wir eine API bemühen : Setsystemtime. Leider arbeitet diese API mit einer Type- Struktur, was bei CE nicht implementiert ist. Ein Umweg kann begangen werden: die nötige Struktur muss bei CE als String übergeben werden. Mehr dazu in meinem Artikel zu den UDT's (User Defined Types).

Im Einsatz:

Hier drei Screenshots einer echten Fahrt. Die Route ist auf der Rheinuferstraße in Köln (von der Innenstadt Richtung Süden). Man kann einiges erkennen:

Zum einen werden die Koordinaten genau angezeigt. Je nach Geschmack lässt sich die Koordinatenanzeige auf Sekundenangabe (wie hier gezeigt inkl. Kommastelle) oder auf die systemeigene Anzeige des Sekundenwertes in 10000 er Einheiten. 60 Sekunden = 10000 stellen..

Beim Start an der Ampel zeigt die Geschwindigkeit 00.0 an.

Danach wird beschleunigt und eine Minute später werden 58.8 km/h angezeigt. (ich glaube hier gibt es eine Geschwindigkeitsbegrenzung von 50 km/h)

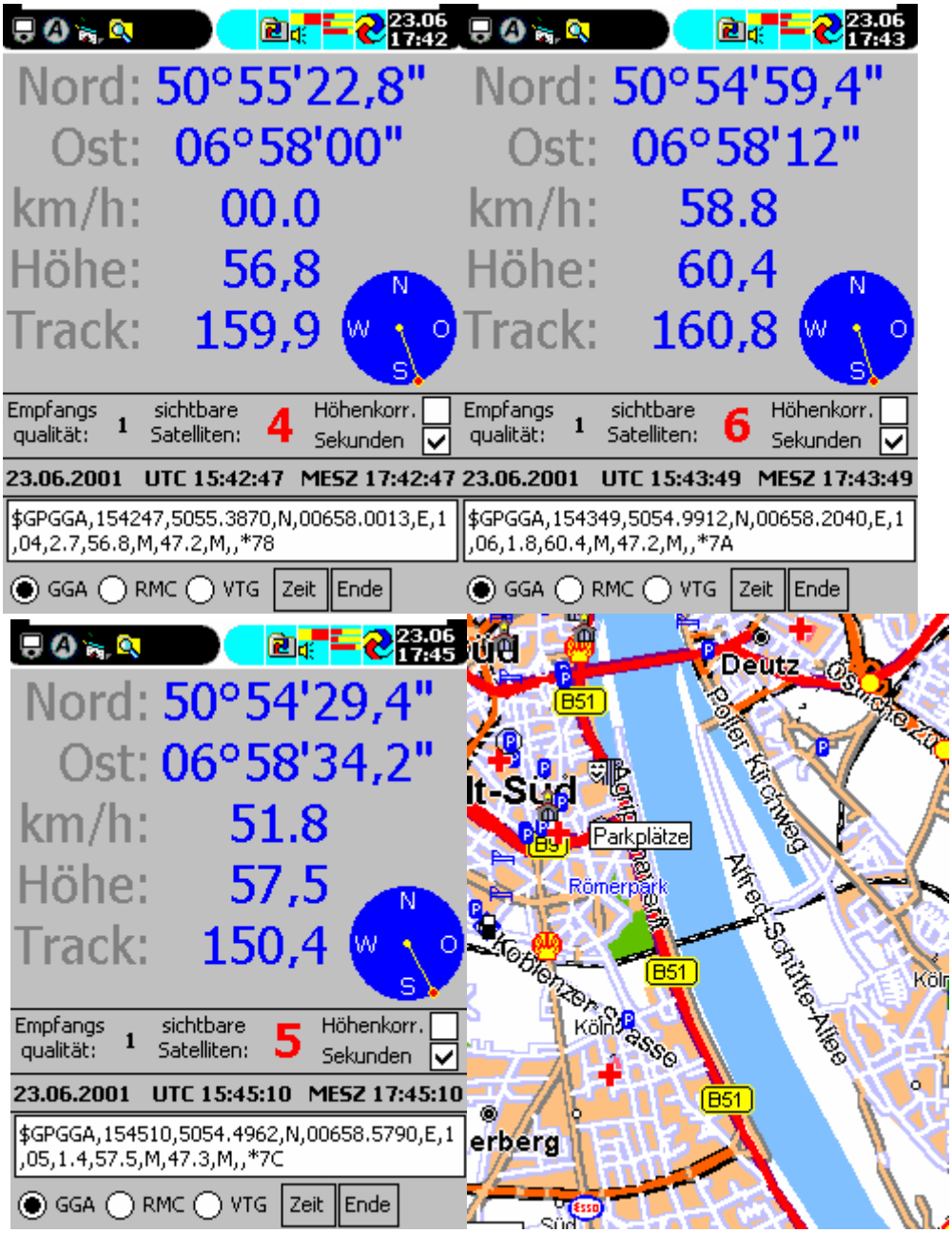
Das Problem der Höhe ist hier erkenntlich – obwohl die Abweichungen sich im Rahmen halten.

Die Korrekturkonstante (hier 47.2 m) ist ausgeschaltet, sonst wären die Werte unsinniger. Wer die Rheinuferstraße in Köln kennt, der weiß, dass hier keine nennenswerte Steigung anliegt. Sie läuft parallel zum Rheinbett. Der dritte Wert – es ging rheinaufwärts – müsste daher höher sein. Wahrscheinlich ist eine genauere Messung über GPS nicht zu erreichen. Immerhin liegt der Wert nicht bei 500 oder 1000.

Der Trackwinkel wird gut angezeigt. Die Fahrt geht fast nach Süden, wie man an dem kleinen Kartenausschnitt erkennen kann. Ich befahre die B51 nach Süden.

Der Empfang der Satelliten ändert sich zum Teil sehr schnell. Hier, bei freier Sicht nach oben, wurden 4, 5 und 6 Satelliten auf kurzer Strecke empfangen.

Das Datum und die Uhrzeit werden exakt angezeigt – auch wenn der Empfängerkeien Satelliten empfängt. Offenbar läuft intern eine Quarzuhr mit, die auch dann weiterläuft, wenn man das Gerät für einige Zeit von der Batterieversorgung trennt.



Das obenstehende Bild wird von dem Programm **gpsterminal.vb** geliefert. Es sind ein paar Programmiertipps eingebaut, die nicht alltäglich sind.

Zunächst: Um etwas mehr Platz auf dem kleinen Bildschirm zu bekommen, wird der Fullscreenmodus eingeschaltet.

```
Option Explicit
Const SW_HIDE = 0
Const SW_SHOWNORMAL = 1

Const SHFS_SHOWTASKBAR = &H1
Const SHFS_HIDETASKBAR = &H2

Const SHFS_SHOWSIPBUTTON = &H4
Const SHFS_HIDESIPBUTTON = &H8

Const SWP_SHOWWINDOW = &H40
Const SM_CXSCREEN = &H0
```

```
Const SM_CYSCREEN = &H1

Declare Function GetSystemMetrics Lib "Coredll" ( _
    ByVal nIndex As Long) As Long

Declare Function SHFullScreen Lib "aygshell" ( _
    ByVal hwndRequester As Long, _
    ByVal dwState As Long) As Boolean

Declare Function MoveWindow Lib "Coredll" ( _
    ByVal hwnd As Long, _
    ByVal x As Long, _
    ByVal y As Long, _
    ByVal nWidth As Long, _
    ByVal nHeight As Long, _
    ByVal bRepaint As Long) As Long

Declare Function SetForegroundWindow Lib "Coredll" ( _
    ByVal hwnd As Long) As Boolean

Declare Function ShowWindow Lib "Coredll" ( _
    ByVal hwnd As Long, _
    ByVal nCmdShow As Long) As Long

Declare Function FindWindow Lib "Coredll" Alias "FindWindowW" ( _
    ByVal lpClassName As String, _
    ByVal lpWindowName As String) As Long

Declare Function SetSystemTime Lib "Coredll" _
(ByVal lpstr As String) As Long

Declare Function PlaySound Lib "Coredll" Alias "PlaySoundW" _
    (ByVal lpszName As String, ByVal hModule As Long, _
    ByVal dwFlags As Long) As Long

Dim a, pi, zusatz, lpstr

Private Sub NachBinary(ByVal ein As Integer, ByVal longorint As Integer)
Dim i As Integer
    For i = 1 To longorint - 1
        lpstr = lpstr & ChrB(ein Mod 256)
        ein = ein \ 256
    Next i
       lpstr = lpstr & ChrB(ein)
End Sub

Private Sub mache_udt()
    lpstr = ""
    Call NachBinary(CInt(Mid(Label13, 7, 4)), 2) ' Jahr
    Call NachBinary(CInt(Mid(Label13, 4, 2)), 2) ' Monat
    Call NachBinary(Weekday(Now, vbMonday), 2)   ' Wochentag
    Call NachBinary(CInt(Mid(Label13, 1, 2)), 2) ' Tag

    Call NachBinary(CInt(Mid(Label15, 1, 2)), 2) ' Stunde
    Call NachBinary(CInt(Mid(Label15, 4, 2)), 2) ' Minute
    Call NachBinary(CInt(Mid(Label15, 7, 2)), 2) ' Sekunde
    Call NachBinary(0, 2)                        ' Millisekunde
End Sub

Private Sub Mache_fullscreen()
    Dim lret
    lret = FindWindow("menu_worker", "")
    If lret <> 0 Then 'window found
```

```
        ShowWindow lret, SW_HIDE
    End If

    lret = SetForegroundWindow(Me.hwnd)
    lret = MoveWindow(Me.hwnd, 0, 0, _
            GetSystemMetrics(SM_CXSCREEN), _
            GetSystemMetrics(SM_CYSCREEN) + 26, 0)

      ' verstecke das Tastatursymbol
    lret = SHFullScreen(Me.hwnd, SHFS_HIDESIPBUTTON)
End Sub

Private Sub Comm1_OnComm()
      a = a + Comm1.Input
End Sub

Private Sub Command1_Click()
      Call mache_udt
      Call SetSystemTime(lpstr)
End Sub

Private Sub Command2_Click()
      App.End
End Sub

Private Sub Form_SIPChange(bSIPVisible As Boolean)
    Call Mache_fullscreen
End Sub


Private Sub Form_Load()
      Comm1.CommPort = 1
      Comm1.Settings = "4800,n,8,1"
      Comm1.RThreshold = 1

      If Comm1.PortOpen = False Then Comm1.PortOpen = True

Call Mache_fullscreen
      Pb1.BackColor = RGB(192, 192, 192)

      pi = 4 * Atn(1)

      Pb1.FillStyle = 0

      ' muss noch genauer eingegrenzt werden Sommerzeit / Winterzeit
      If Month(Now) > 3 And Month(Now) < 11 Then
            Label16 = "MESZ"
            zusatz = 2
      Else
            Label16 = "MEZ"
            zusatz = 1
      End If
End Sub

Private Sub Timer1_Timer()
On Error Resume Next
Dim b, c, gga, rmc, vtg, art, zaehler, sek1, sek2, nord, ost, track, winkel
On Error Resume Next
b = a: a = ""
art = "$GPRMC"
rmc = Mid(b, InStr(b, art), InStr(InStr(b, art) + 1, b, vbCrLf) - InStr(b,
art))
art = "$GPGGA"
```

```
gga = Mid(b, InStr(b, art), InStr(InStr(b, art) + 1, b, vbCrLf) - InStr(b,
art))
art = "$GPVTG"
vtg = Mid(b, InStr(b, art), InStr(InStr(b, art) + 1, b, vbCrLf) - InStr(b,
art))

If vtg = "" Or gga = "" Or rmc = "" Then Exit Sub

If Option1.Value Then Text1 = gga
If Option2.Value Then Text1 = rmc
If Option3.Value Then Text1 = vtg

gga = Mid(gga, InStr(gga, ",") + 1)

' Zeit
Label15 = Mid(Mid(gga, 1, InStr(gga, ",") - 1), 1, 2) & ":" + Mid(Mid(gga,
1, InStr(gga, ",") - 1), 3, 2) + ":" + Mid(Mid(gga, 1, InStr(gga, ",") -
1), 5, 2)
Label12 = Mid(Mid(gga, 1, InStr(gga, ",") - 1), 1, 2) + zusatz & ":" +
Mid(Mid(gga, 1, InStr(gga, ",") - 1), 3, 2) + ":" + Mid(Mid(gga, 1,
InStr(gga, ",") - 1), 5, 2)

' Nord
gga = Mid(gga, InStr(gga, ",") + 1)
nord = Mid(gga, 1, 2) + "°" + Mid(gga, 3, 2) + "'"
gga = Mid(gga, InStr(gga, ".") + 1)
If Check2 = 1 Then
sek1 = (Mid(gga, 1, InStr(gga, ",") - 1) \ 100) * 0.6

If sek1 < 10 Then sek1 = "0" & sek1
    Label3 = nord & sek1 & Chr(34)
Else
    sek1 = Mid(gga, 1, InStr(gga, ",") - 1)
    Label3 = nord & sek1
End If

' Ost
gga = Mid(gga, InStr(gga, "N") + 2)
ost = Mid(gga, 2, 2) + "°" + Mid(gga, 4, 2) + "'"
gga = Mid(gga, InStr(gga, ".") + 1)

If Check2.Value = 1 Then
    sek2 = (Mid(gga, 1, InStr(gga, ",") - 1) \ 100) * 0.6
    If sek2 < 10 Then sek2 = "0" & sek2
    Label4 = ost & sek2 & Chr(34)
Else
    sek2 = Mid(gga, 1, InStr(gga, ",") - 1)
    Label4 = ost & sek2
End If

' Qualität
gga = Mid(gga, InStr(gga, "E") + 2)
Label6 = Mid(gga, 1, InStr(gga, ",") - 1)

' Satelliten
gga = Mid(gga, InStr(gga, ",") + 1)
Label8 = CInt(Mid(gga, 1, InStr(gga, ",") - 1))

' Höhe
Dim hoehe, korrektur, var1, var2
gga = Mid(gga, InStr(gga, ",") + 1)
gga = Mid(gga, InStr(gga, ",") + 1)
hoehe = Mid(gga, 1, InStr(gga, ",") - 1)
```

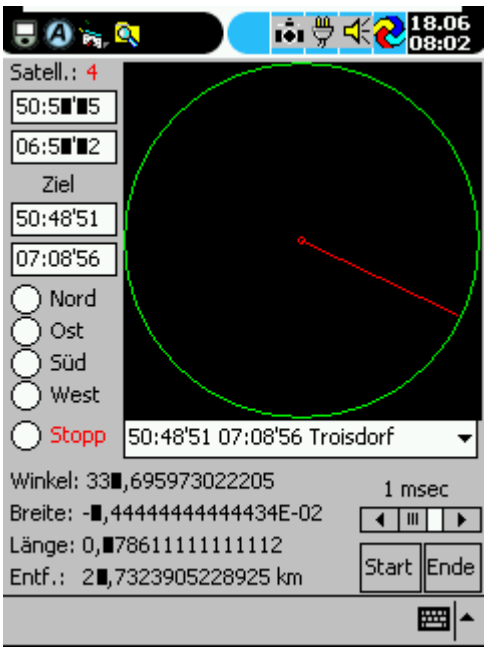
```
var1 = CSng(Mid(hoehe, 1, InStr(hoehe, ".") - 1))
var2 = CSng(Mid(hoehe, InStr(hoehe, ".") + 1)) * 0.1
hoehe = var1 + var2

' Korrektur
gga = Mid(gga, InStr(gga, ",") + 1)
gga = Mid(gga, InStr(gga, ",") + 1)
korrektur = Mid(gga, 1, InStr(gga, ",") - 1)

var1 = CSng(Mid(korrektur, 1, InStr(korrektur, ".") - 1))
var2 = CSng(Mid(korrektur, InStr(korrektur, ".") + 1)) * 0.1
korrektur = var1 + var2

If hoehe = 0 Then hoehe = ""
If korrektur = 0 Then korrektur = ""

If Label8 < 3 Then
    Label3.FontStrikethru = True
    Label4.FontStrikethru = True
    Label11.FontStrikethru = True
Else
    Label3.FontStrikethru = False
    Label4.FontStrikethru = False
    Label11.FontStrikethru = False
End If

' Satellitenempfang bewerten
If Label8 < 4 Then
    Label21.FontStrikethru = True
Else
    Label21.FontStrikethru = False
End If
If Check1.Value = 0 Then Label21 = hoehe Else Label21 = hoehe - korrektur

' Trackwinkel
nord = ""
vtg = Mid(vtg, InStr(vtg, ",") + 1)
track = Mid(vtg, 1, InStr(vtg, ",") - 1)

If track <> "" Then
    var1 = 0: var2 = 0
    var1 = CSng(Mid(track, 1, InStr(track, ".") - 1))
    var2 = CSng(Mid(track, InStr(track, ".") + 1)) * 0.1
    Label11 = var1 + var2
End If

' Geschwindigkeit
vtg = Mid(vtg, InStr(vtg, "N") + 2)
Label10 = Mid(vtg, 1, InStr(vtg, ",") - 1)

' Datum (nach 9. Komma)

Do While InStr(rmc, ",")
    zaehler = zaehler + 1
    rmc = Mid(rmc, InStr(rmc, ",") + 1)
    If zaehler = 9 Then Exit Do
Loop

Label13 = Mid(rmc, 1, 2) & "." & Mid(rmc, 3, 2) & "." & "20" & Mid(rmc, 5,
2)
```

```
' Richtung grafisch darstellen
Pb1.Cls
    Pb1.FillColor = vbBlue
    Pb1.DrawCircle 32, 32, 30, vbBlue
    Pb1.FillColor = vbYellow
    Pb1.DrawCircle 32, 32, 2, vbYellow
    winkel = (Label11 / 180 * pi) - pi / 2
    Pb1.DrawLine 32, 32, 32 + Cos(winkel) * 30, 32 + _
        Sin(winkel) * 30, vbYellow

    Pb1.FillColor = vbRed
    Pb1.DrawCircle 32 + Cos(winkel) * 30, 32 + _
        Sin(winkel) * 30, 3, vbYellow

    Pb1.FillColor = vbBlue
    Pb1.ForeColor = vbWhite

    Pb1.DrawText "N", 29, 1
    Pb1.DrawText "S", 29, 46
    Pb1.DrawText "W", 5, 24
    Pb1.DrawText "O", 51, 24
End Sub
```

Ein primitives Navigationssystem

Nachdem die Grundlagen für den GPS – Empfang jetzt bekannt sind, können wir uns an ein eigenes GPS – Navigationssystem wagen. Es bleibt natürlich ein Primitivsystem, da wir auf keine Karte zurückgreifen.

Wir kennen über den GPS – Empfänger die momentane geographische Lage. Das geographische Ziel müssen wir uns vorher aus einem GPS – Routenplaner herausholen. Im unten abgebildeten Screenshot (wo ich Teile wieder schwärzen musste, um nicht alles zurückrechnen zu können) ist das Prinzip erkennbar.



Zum einen werden 4 Satelliten empfangen. Darunter sind die aktuellen Koordinaten für Nord und Ost zu sehen. Wir wollen zum Ziel nach Troisdorf (südwestlich von Köln).

Ein bisschen Schulwissen hilft jetzt weiter. Wie war das mit dem Satz von Pythagoras? A zum Quadrat + b zum Quadrat = c zum Quadrat. Es ist klar: Zwischen Start und Ziel entsteht ein rechtwinkliges Dreieck, dessen Hypotenuse die Fahrstrecke und die Entfernung angibt. Das ist zwar eine flächige 2 D – Berechnung, doch für unser Primitivmodell ist es ausreichend genau.

Bei einer Strecke von Köln nach München macht das nur ca. 30 km Differenz aus.

Die unten aufgezeigte Methode ist nur eine Art der Darstellung. Wir sehen hier an der roten Linie, dass wir nach Süden und nach Osten fahren müssen, damit wir das Ziel auf dem grünen Kreis erreichen.

Bei dieser Darstellung bleibt der Radius immer gleich. Die Angabe, wie weit es zum Ziel ist, zeigt das unterste Label ‚Entf.:' an. Es berechnet die Luftlinie zum Ziel (Hypotenuse). Natürlich kann das auch ganz anders programmiert werden.

Damit man auch zu Hause (ohne bewegtem GPS-Empfänger) arbeiten kann, lassen sich die Himmelsrichtungen einstellen. Ein Timer addiert oder subtrahiert im Koordinaten-Sekundenschritt die aktuelle Position.

Das Programm **navigation.vb**

```
Option Explicit

Dim ergebnis, richtung, a

Private Sub Combo1_Click()
      Dim a
      a = Combo1.List(Combo1.ListIndex)
      Text3 = Mid(a, 1, InStr(a, " ") - 1)
      a = Mid(a, InStr(a, " ") + 1)
      Text4 = Mid(a, 1, InStr(a, " ") - 1)
End Sub

Private Sub Comm1_OnComm()
      a = a + Comm1.Input
End Sub

Private Sub Command1_Click()
      Dim breite1, laenge1, breite2, laenge2, breitdiff, langdiff
      Dim mittex, mittey, radius, sinalpha, cosalpha, alpha, winkel, pi, i

      pi = 4 * Atn(1)
      pb1.Cls

      Call eingabe_zerlegen(Text1)
      breite1 = ergebnis
      Call eingabe_zerlegen(Text2)
      laenge1 = ergebnis
      Call eingabe_zerlegen(Text3)
      breite2 = ergebnis
      Call eingabe_zerlegen(Text4)
      laenge2 = ergebnis

breitdiff = breite2 - breite1
langdiff = laenge2 - laenge1

      Label3 = breitdiff
```

```
      Label4 = langdiff

mittex = pb1.Width / 2
mittey = pb1.Height / 2

radius = Sqr(breitdiff ^ 2 + langdiff ^ 2)

sinalpha = breitdiff / radius
cosalpha = langdiff / radius

If sinalpha And cosalpha = 0 Then Exit Sub

alpha = Abs(Atn(sinalpha / cosalpha))

winkel = alpha

If breitdiff < 0 And langdiff < 0 Then winkel = pi + alpha
If breitdiff >= 0 And langdiff < 0 Then winkel = pi - alpha
If breitdiff < 0 And langdiff >= 0 Then winkel = 2 * pi - alpha

Label6 = (180 / pi) * winkel

Label11 = radius * 110 & " km"
radius = 1300

      pb1.DrawCircle mittex, mittey, pb1.Width / 2, RGB(0, 255, 0)
      pb1.DrawCircle mittex, mittey, 30, RGB(255, 0, 0)

      pb1.DrawLine mittex, mittey, mittex + Cos(winkel) * radius, _
      mittey -Sin(winkel) * radius, RGB(255, 0, 0)

End Sub

Private Sub Command2_Click()
      App.End
End Sub



Private Sub Form_Load()
      Comm1.CommPort = 1
      Comm1.Settings = "4800,n,8,1"
      Comm1.RThreshold = 1
      If Comm1.PortOpen = False Then Comm1.PortOpen = True
      msg = String(18, Chr(0))

      Combo1.AddItem "50:48'05 06:45'59 Erftstadt"
      Combo1.AddItem "50:48'51 07:08'56 Troisdorf"
      Combo1.AddItem "51:10'41 07:05'29 Solingen"
      Combo1.AddItem "51:00'05 06:48'10 Pulheim"
      Combo1.Text = "Frechen"

      pb1.Height = pb1.Width

      Timer2.Enabled = True
End Sub

Private Sub HScroll1_Change()
      Timer1.Interval = HScroll1.Value
      Label7 = HScroll1.Value & " msec"
End Sub

Private Sub HScroll1_Scroll()
```

```
      HScroll1_Change
End Sub

Private Sub Option1_Click()
      richtung = 1
      Timer1.Enabled = True
      Timer2.Enabled = False
End Sub

Private Sub Option2_Click()
      richtung = 2
      Timer1.Enabled = True
      Timer2.Enabled = False
End Sub

Private Sub Option3_Click()
      richtung = 3
      Timer1.Enabled = True
      Timer2.Enabled = False
End Sub

Private Sub Option4_Click()
      richtung = 4
      Timer1.Enabled = True
      Timer2.Enabled = False
End Sub

Private Sub Option5_Click()
      Timer1.Enabled = False
      Timer2.Enabled = True
End Sub


Private Sub eingabe_zerlegen(eingabe As String)
Dim wert1, wert2, wert3 As Integer

      wert1 = Mid(eingabe, 1, InStr(eingabe, ":") - 1)
      eingabe = Mid(eingabe, InStr(eingabe, ":") + 1)

      wert2 = Mid(eingabe, 1, InStr(eingabe, "'") - 1)
      eingabe = Mid(eingabe, InStr(eingabe, "'") + 1)

      wert3 = eingabe
      ergebnis = wert1 + wert2 / 60 + wert3 / 3600
End Sub

Private Sub Timer1_Timer()
Dim wert1, wert2, wert3 As Integer

      On Error Resume Next
      Label12.Caption = ""

Select Case richtung
      Case 1
            wert1 = Mid(Text1, 1, InStr(Text1, ":") - 1)
            Text1 = Mid(Text1, InStr(Text1, ":") + 1)
            wert2 = Mid(Text1, 1, InStr(Text1, "'") - 1)
            Text1 = Mid(Text1, InStr(Text1, "'") + 1)
            wert3 = Text1
            wert3 = wert3 + 1
        If wert3 > 59 Then
            wert3 = 0
            wert2 = wert2 + 1
```

```
        End If

        If wert2 > 59 Then
            wert2 = 0
            wert1 = wert1 + 1
        End If

      If wert1 < 10 Then wert1 = "0" & wert1
      If wert2 < 10 Then wert2 = "0" & wert2
      If wert3 < 10 Then wert3 = "0" & wert3

      If Len(wert1) > 2 Then wert1 = Mid(wert1, 2, 2)
      If Len(wert2) > 2 Then wert2 = Mid(wert2, 2, 2)
      If Len(wert3) > 2 Then wert3 = Mid(wert3, 2, 2)

      Text1 = wert1 & ":" & wert2 & "'" & wert3
      Command1_Click

Case 2

    wert1 = Mid(Text2, 1, InStr(Text2, ":") - 1)
    Text2 = Mid(Text2, InStr(Text2, ":") + 1)
    wert2 = Mid(Text2, 1, InStr(Text2, "'") - 1)
    Text2 = Mid(Text2, InStr(Text2, "'") + 1)
    wert3 = Text2

    wert3 = wert3 + 1
        If wert3 > 59 Then
            wert3 = 0
            wert2 = wert2 + 1
        End If

        If wert2 > 59 Then
            wert2 = 0
            wert1 = wert1 + 1
        End If

    If wert1 < 10 Then wert1 = "0" & wert1
    If wert2 < 10 Then wert2 = "0" & wert2
    If wert3 < 10 Then wert3 = "0" & wert3

    If Len(wert1) > 2 Then wert1 = Mid(wert1, 2, 2)
    If Len(wert2) > 2 Then wert2 = Mid(wert2, 2, 2)
    If Len(wert3) > 2 Then wert3 = Mid(wert3, 2, 2)

    Text2 = wert1 & ":" & wert2 & "'" & wert3
    Command1_Click

Case 3
    wert1 = Mid(Text1, 1, InStr(Text1, ":") - 1)
    Text1 = Mid(Text1, InStr(Text1, ":") + 1)
    wert2 = Mid(Text1, 1, InStr(Text1, "'") - 1)
    Text1 = Mid(Text1, InStr(Text1, "'") + 1)
    wert3 = Text1

    wert3 = wert3 - 1

        If wert3 < 0 Then
            wert3 = 59
            wert2 = wert2 - 1
        End If

        If wert2 < 0 Then
```

```
            wert2 = 59
            wert1 = wert1 - 1
        End If

    If wert1 < 10 Then wert1 = "0" & wert1
    If wert2 < 10 Then wert2 = "0" & wert2
    If wert3 < 10 Then wert3 = "0" & wert3

    If Len(wert1) > 2 Then wert1 = Mid(wert1, 2, 2)
    If Len(wert2) > 2 Then wert2 = Mid(wert2, 2, 2)
    If Len(wert3) > 2 Then wert3 = Mid(wert3, 2, 2)

    Text1 = wert1 & ":" & wert2 & "'" & wert3
    Command1_Click


Case 4

    wert1 = Mid(Text2, 1, InStr(Text2, ":") - 1)
    Text2 = Mid(Text2, InStr(Text2, ":") + 1)
    wert2 = Mid(Text2, 1, InStr(Text2, "'") - 1)
    Text2 = Mid(Text2, InStr(Text2, "'") + 1)
    wert3 = Text2

    wert3 = wert3 - 1
        If wert3 < 0 Then
            wert3 = 59
            wert2 = wert2 - 1
        End If

        If wert2 < 0 Then
            wert2 = 59
            wert1 = wert1 - 1
         End If

    If wert1 < 10 Then wert1 = "0" & wert1
    If wert2 < 10 Then wert2 = "0" & wert2
    If wert3 < 10 Then wert3 = "0" & wert3

    If Len(wert1) > 2 Then wert1 = Mid(wert1, 2, 2)
    If Len(wert2) > 2 Then wert2 = Mid(wert2, 2, 2)
    If Len(wert3) > 2 Then wert3 = Mid(wert3, 2, 2)

    Text2 = wert1 & ":" & wert2 & "'" & wert3
    Command1_Click
End Select

End Sub

Private Sub Timer2_Timer()
On Error Resume Next
Dim b, nord, ost, art, gga

      b = a: a = ""
      ' Lese das GGA - Telegramm
      art = "$GPGGA"

gga = Mid(b, InStr(b, art), InStr(InStr(b, art) + 1, b, vbCrLf) - 1 -
InStr(b, art))

If gga = "" Then Exit Sub

      gga = Mid(gga, InStr(gga, ",") + 1)
```

```
        gga = Mid(gga, InStr(gga, ",") + 1)

        ' Lese die Nordposition
        nord = Mid(gga, 1, 2) + ":"
        nord = nord + Mid(gga, 3, 2) + "'"
        gga = Mid(gga, InStr(gga, ".") + 1)
        b = (Mid(gga, 1, InStr(gga, ",") - 1) \ 100) * 0.6 \ 1
        If b < 10 Then b = "0" & b
        nord = nord & b

Text1 = nord
        ' Lese die Ostposition
        gga = Mid(gga, InStr(gga, "N") + 2)
        ost = Mid(gga, 2, 2) + ":"
        ost = ost + Mid(gga, 4, 2) + "'"
        gga = Mid(gga, InStr(gga, ".") + 1)
        b = (Mid(gga, 1, InStr(gga, ",") - 1) \ 100) * 0.6 \ 1
        If b < 10 Then b = "0" & b
        ost = ost & b

Text2 = ost
        ' Lese die Anzahl der Satelliten
        gga = Mid(gga, InStr(gga, "E") + 2)
        gga = Mid(gga, InStr(gga, ",") + 1)

Label12 = CInt(Mid(gga, 1, InStr(gga, ",") - 1))
End Sub
```

Zum Schluß noch ein Screenshot, der das Navigationsergebnis zeigt. Wir sind gerade noch 1.2 km von Troisdorf entfernt. Wir müssen weiter nach Osten fahren. Den exakten Breitengrad haben wir erreicht, wie man an dem Winkel 0 ablesen kann.