```
Option Explicit
Public Function LL2BearingDistance(p_Origin As String, p_Target As String, p_Var As Single, Optional p
_ReturnType As Integer) As String
    Dim l_OriginLon As Single
    Dim l_OriginLat As Single
    Dim l_TargetLon As Single
    Dim l_TargetLat As Single
    Dim l_Adjacent As Single
    Dim l_Opposite As Single
    Dim l_Hypotenuse As Single
    Dim l_UnconvertedHeading As Single
    Dim l_TrueBearing As Single
    Dim l_Loxodrome As Single
    Dim l_AdjHyp As Double
    Dim l_origin As String
    Dim l_target As String

    l_target = Trim(p_Target)
    If left(l_target, 1) <> "-" Then
        l_target = " " & l_target
    End If

    l_origin = Trim(p_Origin)
    If left(l_origin, 1) <> "-" Then
        l_origin = " " & l_origin
    End If

    l_OriginLat = CSng(left(l_origin, 10))
    'de  bug.print "------------"
    'de  bug.print p_Origin

    'de  bug.print "OriginLat:" & l_OriginLat

    l_OriginLon = CSng(right(l_origin, 11))

    l_OriginLon = AdjLon(l_OriginLon)


    'de  bug.print "OriginLon:" & l_OriginLon
    'de  bug.print p_Target
    'de  bug.print "p_Target:" & Left(p_Target, 10)
    'If IsNumeric(l_target) = False Then
    '  l_target = l_origin
    'End If

    l_TargetLat = CSng(left(l_target, 10))
    'de  bug.print "TargetLat:" & l_TargetLat

    l_TargetLon = CSng(right(l_target, 11))
    l_TargetLon = AdjLon(l_TargetLon)
    'de  bug.print "COS:" & Cos((l_OriginLat + l_TargetLat) / 2 * 3.14159265 / 180)


    l_Adjacent = Cos((l_OriginLat + l_TargetLat) / 2 * 3.14159265 / 180) * (l_OriginLon - l_TargetLon)
 * -60
    'de  bug.print "Adjacent:" & l_Adjacent

    l_Opposite = (l_TargetLat - l_OriginLat) * 60
    'de  bug.print "Opposite:" & l_Opposite

    l_Hypotenuse = Sqr(l_Adjacent ^ 2 + l_Opposite ^ 2)
    'If l_Hypotenuse <= 500 Then
    'de  bug.print "Hypotenuse:" & l_Hypotenuse
    'End If

    If l_Hypotenuse = 0 Then
        l_AdjHyp = 0
    Else
        l_AdjHyp = CSng(l_Adjacent / l_Hypotenuse)
    End If
    'de  bug.print "AdjHyp1:" & l_AdjHyp

    If l_AdjHyp = 1 Then
```

```
        l_AdjHyp = 0.999999999999999
    End If
    'de  bug.print "AdjHyp2:" & l_AdjHyp

    If l_AdjHyp = -1 Then
        l_AdjHyp = -0.999999999999999
    End If
    'de  bug.print "AdjHyp3:" & l_AdjHyp

    l_UnconvertedHeading = (Atn(-l_AdjHyp / Sqr(-l_AdjHyp * l_AdjHyp + 1)) + 2 * Atn(1)) * 180 / 3.141
59265
    'de  bug.print "UnconvertedHeading:" & l_UnconvertedHeading
    'MODIFIED 6/26/05 to remove goto

    If (Sgn(l_Opposite)) = -1 Then
        l_TrueBearing = 90 + l_UnconvertedHeading
        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If
        If l_Hypotenuse > 9999.9 Then
        LL2BearingDistance = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000.0")
        Else
        LL2BearingDistance = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "0000.00")
        End If
        'LL2BearingDistance = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000.0")
    Else
        'l_TrueBearing = IIf(Sgn(l_Adjacent) = -1, 450 - l_UnconvertedHeading, 90 - l_UnconvertedHeadi
ng)
        If Sgn(l_Adjacent) = -1 Then
            l_TrueBearing = 450 - l_UnconvertedHeading
        Else
            l_TrueBearing = 90 - l_UnconvertedHeading
        End If

        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If

        If l_Loxodrome > 360 Then
            l_Loxodrome = l_Loxodrome - 360
        End If
        Select Case p_ReturnType
        Case 0
        If l_Hypotenuse > 9999.9 Then
        LL2BearingDistance = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000.0")
        Else
        LL2BearingDistance = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "0000.00")
        End If
        Case 1
         If l_Hypotenuse > 9999.9 Then
        LL2BearingDistance = Format(l_Loxodrome, "000") & "/" & Format(CInt(l_Hypotenuse), "00000")
        Else
        LL2BearingDistance = Format(l_Loxodrome, "000") & "/" & Format(CInt(l_Hypotenuse), "000")
        End If
        End Select

    End If
End Function
Public Function AdjLon(p_lon As Single) As Single
If p_lon > 0 Then
AdjLon = p_lon - 360
Else
AdjLon = p_lon
End If
End Function
Public Function STARSPotanOffsetProcessing(p_Origin As String, p_Target As String, p_Var As Single, p_
offsetX As Single, p_OffsetY As Single, p_format As Integer) As String
    Dim l_OriginLon As Single
    Dim l_OriginLat As Single
    Dim l_TargetLon As Single
    Dim l_TargetLat As Single
    Dim l_Adjacent As Single
    Dim l_Opposite As Single
    Dim l_UnshiftedAdjacent As Single
    Dim l_UnshiftedOpposite As Single
    Dim l_Hypotenuse As Single
    Dim l_UnconvertedHeading As Single
    Dim l_TrueBearing As Single
    Dim l_Loxodrome As Single
    Dim l_AdjHyp As Double
    Dim l_origin As String
    Dim l_target As String
    'p_format 1 is Bearing Dist
    'p_format 2 Is XY

    l_target = Trim(p_Target)
    If left(l_target, 1) <> "-" Then
        l_target = " " & l_target
    End If

    l_origin = Trim(p_Origin)
    If left(l_origin, 1) <> "-" Then
        l_origin = " " & l_origin
    End If

    l_OriginLat = CSng(left(l_origin, 10))
    'de  bug.print "------------"
    'de  bug.print p_Origin

    'de  bug.print "OriginLat:" & l_OriginLat

    l_OriginLon = CSng(right(l_origin, 11))
    l_OriginLon = AdjLon(l_OriginLon)

    'de  bug.print "OriginLon:" & l_OriginLon
    'de  bug.print p_Target
    'de  bug.print "p_Target:" & Left(p_Target, 10)

    l_TargetLat = CSng(left(l_target, 10))
    'de  bug.print "TargetLat:" & l_TargetLat

    l_TargetLon = CSng(right(l_target, 11))
    l_TargetLon = AdjLon(l_TargetLon)
    'de  bug.print "TargetLon:" & l_TargetLon
    'p_OffsetY really needs to be one tenth the value of the nM when calculating Lat, mult by .2 doubl
es that so it can be divided when added with other values
    l_UnshiftedAdjacent = (Cos((l_OriginLat + l_TargetLat) / 2 * 3.14159265 / 180) * (l_OriginLon - l_
TargetLon) * -60)

    l_Adjacent = l_UnshiftedAdjacent - p_offsetX
    'de  bug.print "Adjacent:" & l_Adjacent
     l_UnshiftedOpposite = ((l_TargetLat - l_OriginLat) * 60)
    l_Opposite = l_UnshiftedOpposite - p_OffsetY
    'de  bug.print "Opposite:" & l_Opposite

    l_Hypotenuse = Sqr(l_Adjacent ^ 2 + l_Opposite ^ 2)
    'If l_Hypotenuse <= 500 Then
    'de  bug.print "Hypotenuse:" & l_Hypotenuse
    'End If

    If l_Hypotenuse = 0 Then
        l_AdjHyp = 0
    Else
        l_AdjHyp = CSng(l_Adjacent / l_Hypotenuse)
    End If
    'de  bug.print "AdjHyp1:" & l_AdjHyp
    If l_AdjHyp = 1 Then
        l_AdjHyp = 0.999999999999999
    End If
    'de  bug.print "AdjHyp2:" & l_AdjHyp

    If l_AdjHyp = -1 Then
        l_AdjHyp = -0.999999999999999
    End If
    'de  bug.print "AdjHyp3:" & l_AdjHyp

    l_UnconvertedHeading = (Atn(-l_AdjHyp / Sqr(-l_AdjHyp * l_AdjHyp + 1)) + 2 * Atn(1)) * 180 / 3.141
59265
    'de  bug.print "UnconvertedHeading:" & l_UnconvertedHeading
    'MODIFIED 6/26/05 to remove goto

    If (Sgn(l_Opposite)) = -1 Then
        l_TrueBearing = 90 + l_UnconvertedHeading
        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If

        'STARSPotanOffsetProcessing = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000
.0")
        'If p_format = 1 Then
        Select Case p_format
        Case 1
        STARSPotanOffsetProcessing = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000.
0")
       Case 2
        '26 Feb 16 188-156 use XYH values instead of RAH for much improved starting precision
        STARSPotanOffsetProcessing = Format(l_Adjacent, "000.000") & "/" & Format(l_Opposite, "000.000
")
        Case 3
       'STARSPotanOffsetProcessing = "START POINT Range/Azm:" & Format(l_Loxodrome, "000") & "/" & For
mat(l_Hypotenuse, "000") & "  X/Y: " & Format(l_Adjacent, "0.00") & "/" & Format(l_Opposite, "0.00")
        STARSPotanOffsetProcessing = " R/A:" & Format(l_Loxodrome, "000") & "/" & Format(l_Hypotenuse,
 "000") & vbCrLf & " X/Y: " & Format(l_Adjacent, "0.00") & "/" & Format(l_Opposite, "0.00")
        Case 4
       STARSPotanOffsetProcessing = " X/Y: " & Format(l_Adjacent, "0.00") & "/" & Format(l_Opposite, "
0.00")

         End Select
    Else
        'l_TrueBearing = IIf(Sgn(l_Adjacent) = -1, 450 - l_UnconvertedHeading, 90 - l_UnconvertedHeadi
ng)
        If Sgn(l_Adjacent) = -1 Then
            l_TrueBearing = 450 - l_UnconvertedHeading
        Else
            l_TrueBearing = 90 - l_UnconvertedHeading
        End If

        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If

        If l_Loxodrome > 360 Then
            l_Loxodrome = l_Loxodrome - 360
        End If
        Select Case p_format
        Case 1

        STARSPotanOffsetProcessing = Format(l_Loxodrome, "000.0") & "/" & Format(l_Hypotenuse, "00000.
0")
       Case 2
        '26 Feb 16 188-156 use XYH values instead of RAH for much improved starting precision
        STARSPotanOffsetProcessing = Format(l_Adjacent, "000.00") & "/" & Format(l_Opposite, "000.00")
        Case 3
        STARSPotanOffsetProcessing = " R/A:" & Format(l_Loxodrome, "000") & "/" & Format(l_Hypotenuse,
 "000") & vbCrLf & " X/Y: " & Format(l_Adjacent, "0.00") & "/" & Format(l_Opposite, "0.00")
        Case 4
        STARSPotanOffsetProcessing = " X/Y: " & Format(l_Adjacent, "0.00") & "/" & Format(l_Opposite,
"0.00")

        End Select

    End If
End Function
Public Function CheckLLFormat(p_LatLonInput As String) As String
    Dim l_LatDeg As Integer
    Dim l_latMin As Integer
    Dim l_LonDeg As Integer
    Dim l_lonMin As Integer
    Dim l_latSec As Single
    Dim l_lonSec As Single
    Dim l_lat As Single
    Dim l_Lon As Single
    Dim l_LatString As String
    Dim l_LonString As String

    'de  bug.print "BGN Test: " & Len(p_LatLonInput)

    If Len(p_LatLonInput) > 36 Or Len(p_LatLonInput) < 19 Then
        CheckLLFormat = "error"
    Else
        Select Case Len(p_LatLonInput)
            Case 24 'ARTS Site Adaptation Format
                '        10        20
                '123456789012345678901234567890
                '33 34 25.2   086 45 24.2   Terminal LL
                '35 04 18.2  106 52 11.0    Enroute LL

                '34:05:03.4N 117:29:02.2W
                If IsNumeric(left(p_LatLonInput, 2)) = False Then
                    l_lat = 99
                Else
                    l_LatDeg = CInt(left(p_LatLonInput, 2))
                End If

                If l_LatDeg < 0 Or l_LatDeg > 90 Then
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 4, 2)) = False Then
                    l_lat = 99
                Else
                    l_latMin = CInt(Mid(p_LatLonInput, 4, 2))
                End If

                If l_latMin < 0 Or l_latMin > 59 Then
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 7, 4)) = False Then
                    l_lat = 99
                Else
                    l_latSec = CInt(Mid(p_LatLonInput, 7, 4))
                End If

                If l_latSec < 0 Or l_latSec > 59.9 Then
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 13, 3)) = False Then
                    l_Lon = 361
                Else
                    l_LonDeg = CInt(Mid(p_LatLonInput, 13, 3))
                End If
                If l_LonDeg < 0 Or l_LonDeg > 179 Then
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 17, 2)) = False Then
                    l_Lon = 361
                Else
                    l_lonMin = CInt(Mid(p_LatLonInput, 17, 2))
                End If

                If l_lonMin < 0 Or l_lonMin > 59 Then
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 20, 4)) = False Then
                    l_Lon = 361
                Else
                    l_lonSec = CInt(Mid(p_LatLonInput, 20, 4))
                End If

                If l_lonSec < 0 Or l_lonMin > 59 Then
                    l_Lon = 361
                End If
               'de bug.print Mid(p_LatLonInput, 1, 2)
                If (CInt(Mid(p_LatLonInput, 1, 2))) < 0 Then
                    l_lat = 99
                End If

                'Previously commented IIF, not part of PB redo
                'lat = IIf((CInt(Mid(p_LatLonInput, 1, 2))) <= 90, CInt(Mid(p_LatLonInput, 1, 2)), 99)
                l_lat = l_LatDeg + (l_latMin / 60) + (l_latSec / 3600)
                l_Lon = l_LonDeg + (l_lonMin / 60) + (l_lonSec / 3600)

                If UCase(Mid(p_LatLonInput, 11, 1)) = "S" Then
                    l_LatDeg = l_LatDeg * -1
                End If

                If UCase(right(p_LatLonInput, 1)) = "W" Then
                    l_Lon = l_Lon * -1
                End If

            Case 29 'PC Format  for ProController /ASRC programs
                'de  bug.print "made it to PC"
                'N034.11.22.333 W112.11.22.333 Latitude
                If IsNumeric(Mid(p_LatLonInput, 2, 3)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 2, 3))) < 0 Then
                    l_lat = 99
                End If

                If CInt(Mid(p_LatLonInput, 2, 3)) <= 90 Then
                    l_lat = CInt(Mid(p_LatLonInput, 2, 3))
                Else
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 6, 2)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 6, 2))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CInt(Mid(p_LatLonInput, 6, 2))) < 60, l_Lat + (CSng(Mid(p_LatLonInput, 6
, 2)) / 60), 99)
                If CInt(Mid(p_LatLonInput, 6, 2)) < 60 Then
                    l_lat = l_lat + (CSng(Mid(p_LatLonInput, 6, 2)) / 60)
                Else
                    l_lat = 99
                End If
                If IsNumeric(Mid(p_LatLonInput, 9, 6)) = False Then
                    l_lat = 99
                End If

                If (CSng(Mid(p_LatLonInput, 9, 6))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CSng(Mid(p_LatLonInput, 9, 6))) <= 60, l_Lat + (CSng(Mid(p_LatLonInput,
9, 6)) / 3600), 99)
                If CSng(Mid(p_LatLonInput, 9, 6)) <= 60 Then
                    l_lat = l_lat + (CSng(Mid(p_LatLonInput, 9, 6)) / 3600)
                Else
                    l_lat = 99
                End If

                'l_Lat = IIf(UCase(left(p_LatLonInput, 1)) = "S", l_Lat * -1, l_Lat)
                If UCase(left(p_LatLonInput, 1)) = "S" Then
                    l_lat = l_lat * -1
                End If

                'Lat = IIf(Left(p_LatLonInput, 1) = "s", l_Lat * -1, l_Lat)
                'N034.11.22.333 W112.11.22.333 Longitude
                If IsNumeric(Mid(p_LatLonInput, 17, 3)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 17, 3))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 17, 3))) <= 180, CInt(Mid(p_LatLonInput, 17, 3))
, 181)
                If CInt(Mid(p_LatLonInput, 17, 3)) <= 180 Then
                    l_Lon = CInt(Mid(p_LatLonInput, 17, 3))
                Else
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 21, 2)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 21, 2))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 21, 2))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
21, 2)) / 60), 181)
                If CInt(Mid(p_LatLonInput, 21, 2)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 21, 2)) / 60)
                Else
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 24, 6)) = False Then
                    l_Lon = 361
                End If

                If (CSng(Mid(p_LatLonInput, 24, 6))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CSng(Mid(p_LatLonInput, 24, 6))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
24, 6)) / 3600), 181)
                If CSng(Mid(p_LatLonInput, 24, 6)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 24, 6)) / 3600)
                Else
                    l_Lon = 361
                End If

                'l_Lon = IIf(UCase(Mid(p_LatLonInput, 16, 1)) = "W", l_Lon * -1, l_Lon)
                If UCase(Mid(p_LatLonInput, 16, 1)) = "W" Then
                    l_Lon = l_Lon * -1
                End If
            Case 35 'GP format
                '        10        20        30
                '12345678901234567890123456789012345
                'GP 32 42 53.7811  117 43 35.9598  !
                'GP 32 43 03.9056  117 42 23.0333  !

                l_LatDeg = Mid(p_LatLonInput, 4, 2)
                l_latMin = Mid(p_LatLonInput, 7, 2)
                l_latSec = Mid(p_LatLonInput, 10, 7)
                l_LonDeg = Mid(p_LatLonInput, 19, 3)
                l_lonMin = Mid(p_LatLonInput, 23, 2)
                l_lonSec = Mid(p_LatLonInput, 26, 7)
                l_lat = l_LatDeg + (l_latMin / 60) + (l_latSec / 3600)
                l_Lon = l_LonDeg + (l_lonMin / 60) + (l_lonSec / 3600)
                l_Lon = l_Lon * -1
            Case 36 'GP format east  Longitude
                '        10        20        30
                '12345678901234567890123456789012345
                'GP 32 42 53.7811  117 43 35.9598  !
                'GP 32 43 03.9056  117 42 23.0333  !
                'GP 32 43 03.9056  -033 42 23.0333  !

                l_LatDeg = Mid(p_LatLonInput, 4, 2)
                l_latMin = Mid(p_LatLonInput, 7, 2)
                l_latSec = Mid(p_LatLonInput, 10, 7)
                l_LonDeg = Mid(p_LatLonInput, 19, 4)
                l_lonMin = Mid(p_LatLonInput, 24, 2)
                l_lonSec = Mid(p_LatLonInput, 27, 7)
                l_lat = l_LatDeg + (l_latMin / 60) + (l_latSec / 3600)
                l_Lon = l_LonDeg - (l_lonMin / 60) - (l_lonSec / 3600)
                l_Lon = l_Lon * -1
            Case 26 ' DMS Format
                'de  bug.print "made it to DMS"
                'N34 22 33.22 W116 22 33.11 Latitude

                If IsNumeric(Mid(p_LatLonInput, 2, 2)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 2, 2))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CInt(Mid(p_LatLonInput, 2, 2))) <= 90, CInt(Mid(p_LatLonInput, 2, 2)), 9
9)
                If CInt(Mid(p_LatLonInput, 2, 2)) <= 90 Then
                    l_lat = CInt(Mid(p_LatLonInput, 2, 2))
                Else
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 5, 2)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 5, 2))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CInt(Mid(p_LatLonInput, 5, 2))) < 60, l_Lat + (CSng(Mid(p_LatLonInput, 5
, 2)) / 60), 99)
                If CInt(Mid(p_LatLonInput, 5, 2)) < 60 Then
                    l_lat = l_lat + (CSng(Mid(p_LatLonInput, 5, 2)) / 60)
                Else
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 8, 5)) = False Then
                    l_lat = 99
                End If
                If (CSng(Mid(p_LatLonInput, 8, 5))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CSng(Mid(p_LatLonInput, 8, 5))) < 60, l_Lat + (CSng(Mid(p_LatLonInput, 8
, 5)) / 3600), 99)
                If CSng(Mid(p_LatLonInput, 8, 5)) < 60 Then
                    l_lat = l_lat + CSng(Mid(p_LatLonInput, 8, 5)) / 3600
                Else
                    l_lat = 99
                End If

                'l_Lat = IIf(left(p_LatLonInput, 1) = "S", l_Lat * -1, l_Lat)
                If left(p_LatLonInput, 1) = "S" Then
                    l_lat = l_lat * -1
                End If

                'l_Lat = IIf(left(p_LatLonInput, 1) = "s", l_Lat * -1, l_Lat)
                If left(p_LatLonInput, 1) = "s" Then
                    l_lat = l_lat * -1
                End If

                'N34 11 22.33 W112 11 22.33 Longitude
                If IsNumeric(Mid(p_LatLonInput, 15, 3)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 15, 3))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 15, 3))) <= 180, CInt(Mid(p_LatLonInput, 15, 3))
, 181)
                If CInt(Mid(p_LatLonInput, 15, 3)) <= 180 Then
                    l_Lon = CInt(Mid(p_LatLonInput, 15, 3))
                Else
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 19, 2)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 19, 2))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 19, 2))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
19, 2)) / 60), 181)
                If CInt(Mid(p_LatLonInput, 19, 2)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 19, 2)) / 60)
                Else
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 22, 5)) = False Then
                    l_Lon = 361
                End If

                If (CSng(Mid(p_LatLonInput, 22, 5))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CSng(Mid(p_LatLonInput, 22, 5))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
22, 5)) / 3600), 181)
                If CSng(Mid(p_LatLonInput, 22, 5)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 22, 5)) / 3600)
                Else
                    l_Lon = 361
                End If

                'l_Lon = IIf(UCase(Mid(p_LatLonInput, 14, 1)) = "W", l_Lon * -1, l_Lon)
                If UCase(Mid(p_LatLonInput, 14, 1)) = "W" Then
                    l_Lon = l_Lon * -1
                End If
            Case 21 'Decimal Format
                'de  bug.print "made it to Dec"
                ' 37.123456-118.123456 Latitude
'                If IsNumeric(Mid(p_LatLonInput, 2, 2)) = False Then
'                    l_Lat = 99
'                End If
'
'                If (CInt(Mid(p_LatLonInput, 2, 2))) < 0 Then
'                    l_Lat = 99
'                End If
'
'                If CInt(Mid(p_LatLonInput, 2, 2)) > 90 Then
'                    l_Lat = 99
'                End If
'
'                If IsNumeric(Mid(p_LatLonInput, 5, 6)) = False Then
'                    l_Lat = 99
'                End If

                l_lat = CSng(Mid(p_LatLonInput, 2, 9))

                'l_Lat = IIf(left(p_LatLonInput, 1) = "-", l_Lat * -1, l_Lat)
                If left(p_LatLonInput, 1) = "-" Then
                    l_lat = l_lat * -1
                End If

'                ' 37.123456-118.123456 Longitude
'                If IsNumeric(Mid(p_LatLonInput, 12, 3)) = False Then l_Lon = 361
'                If (CInt(Mid(p_LatLonInput, 12, 3))) < 0 Then l_Lon = 361
'                If CInt(Mid(p_LatLonInput, 12, 3)) > 180 Then l_Lon = 361
'                If IsNumeric(Mid(p_LatLonInput, 16, 6)) = False Then l_Lon = 361

                l_Lon = CSng(Mid(p_LatLonInput, 12, 10))

                'l_Lon = IIf(Mid(p_LatLonInput, 11, 1) = "-", l_Lon * -1, l_Lon)
                If Mid(p_LatLonInput, 11, 1) = "-" Then
                    l_Lon = l_Lon * -1
                End If
            Case 20
                'de  bug.print "made it to chart20"
                'N34 22.22 W116 22.11 Latitude
                If IsNumeric(Mid(p_LatLonInput, 2, 2)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 2, 2))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CInt(Mid(p_LatLonInput, 2, 2))) <= 90, CInt(Mid(p_LatLonInput, 2, 2)), 9
9)
                If CInt(Mid(p_LatLonInput, 2, 2)) <= 90 Then
                    l_lat = CInt(Mid(p_LatLonInput, 2, 2))
                Else
                    l_lat = 99
                End If

                If IsNumeric(Mid(p_LatLonInput, 5, 5)) = False Then
                    l_lat = 99
                End If

                If (CSng(Mid(p_LatLonInput, 5, 5))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CSng(Mid(p_LatLonInput, 5, 5))) < 60, l_Lat + (CSng(Mid(p_LatLonInput, 5
, 5)) / 60), 99)
                If CSng(Mid(p_LatLonInput, 5, 5)) < 60 Then
                    l_lat = l_lat + (CSng(Mid(p_LatLonInput, 5, 5)) / 60)
                Else
                    l_lat = 99
                End If

                'l_Lat = IIf(left(p_LatLonInput, 1) = "S", l_Lat * -1, l_Lat)
                If left(p_LatLonInput, 1) = "S" Then
                    l_lat = l_lat * -1
                End If

                'l_Lat = IIf(left(p_LatLonInput, 1) = "s", l_Lat * -1, l_Lat)
                If left(p_LatLonInput, 1) = "s" Then
                    l_lat = l_lat * -1
                End If

                'N34 22.22 W116 22.11 Longitude
                If IsNumeric(Mid(p_LatLonInput, 12, 3)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 12, 3))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 12, 3))) <= 180, CInt(Mid(p_LatLonInput, 12, 3))
, 181)
                If CInt(Mid(p_LatLonInput, 12, 3)) <= 180 Then
                    l_Lon = CInt(Mid(p_LatLonInput, 12, 3))
                Else
                    l_Lon = 361
                End If

                If l_Lon < 100 Then
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 16, 5)) = False Then
                    l_Lon = 361
                End If

                If (CSng(Mid(p_LatLonInput, 16, 5))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CSng(Mid(p_LatLonInput, 16, 5))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
16, 5)) / 60), 181)
                If CSng(Mid(p_LatLonInput, 16, 5)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 16, 5)) / 60)
                Else
                    l_Lon = 361
                End If

                If Mid(p_LatLonInput, 11, 1) = "w" Or Mid(p_LatLonInput, 11, 1) = "W" Then
                    l_Lon = l_Lon * -1
                End If
            Case 19 ' Chart 19 Format
                'Chart19FormatProcess:
                'de  bug.print "made it to chart 19"
                'N34 22.22 W96 22.11 Latitude
                If IsNumeric(Mid(p_LatLonInput, 2, 2)) = False Then
                    l_lat = 99
                End If

                If (CInt(Mid(p_LatLonInput, 2, 2))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CInt(Mid(p_LatLonInput, 2, 2))) <= 90, CInt(Mid(p_LatLonInput, 2, 2)), 9
9)
                If CInt(Mid(p_LatLonInput, 2, 2)) <= 90 Then
                    l_lat = CInt(Mid(p_LatLonInput, 2, 2))
                Else
                    l_lat = 99
                End If
```

```
                If IsNumeric(Mid(p_LatLonInput, 5, 5)) = False Then
                    l_lat = 99
                End If

                If (CSng(Mid(p_LatLonInput, 5, 5))) < 0 Then
                    l_lat = 99
                End If

                'l_Lat = IIf((CSng(Mid(p_LatLonInput, 5, 5))) < 60, l_Lat + (CSng(Mid(p_LatLonInput, 5
, 5)) / 60), 99)
                If CSng(Mid(p_LatLonInput, 5, 5)) < 60 Then
                    l_lat = l_lat + (CSng(Mid(p_LatLonInput, 5, 5)) / 60)
                Else
                    l_lat = 99
                End If

                'l_Lat = IIf(left(p_LatLonInput, 1) = "S", l_Lat * -1, l_Lat)
                If UCase(left(p_LatLonInput, 1)) = "S" Then
                    l_lat = l_lat * -1
                End If

                'N34 22.22 W96 22.11 Longitude
                If IsNumeric(Mid(p_LatLonInput, 12, 2)) = False Then
                    l_Lon = 361
                End If

                If (CInt(Mid(p_LatLonInput, 12, 2))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CInt(Mid(p_LatLonInput, 12, 2))) <= 99, CInt(Mid(p_LatLonInput, 12, 2)),
 181)
                If CInt(Mid(p_LatLonInput, 12, 2)) <= 99 Then
                    l_Lon = CInt(Mid(p_LatLonInput, 12, 2))
                Else
                    l_Lon = 361
                End If

                If IsNumeric(Mid(p_LatLonInput, 15, 5)) = False Then
                    l_Lon = 361
                End If

                If (CSng(Mid(p_LatLonInput, 15, 5))) < 0 Then
                    l_Lon = 361
                End If

                'l_Lon = IIf((CSng(Mid(p_LatLonInput, 15, 5))) < 60, l_Lon + (CSng(Mid(p_LatLonInput,
15, 5)) / 60), 181)
                If CSng(Mid(p_LatLonInput, 15, 5)) < 60 Then
                    l_Lon = l_Lon + (CSng(Mid(p_LatLonInput, 15, 5)) / 60)
                Else
                    l_Lon = 361
                End If

                'l_Lon = IIf(Mid(p_LatLonInput, 11, 1) = "W", l_Lon * -1, l_Lon)
                If UCase$(Mid(p_LatLonInput, 11, 1)) = "W" Then
                    l_Lon = l_Lon * -1
                End If
        End Select

        If Abs(l_lat) > 90 Then
            CheckLLFormat = "error"
        ElseIf Abs(l_Lon) > 360 Then
            CheckLLFormat = "error"
        Else
            l_LatString = Format(l_lat, "00.000000")

            'l_LatString = IIf(left(l_LatString, 1) = "-", l_LatString, " " & l_LatString)
            If left(l_LatString, 1) = "-" Then
                'do nothing
            Else
                l_LatString = " " & l_LatString
            End If
            l_LonString = Format(l_Lon, "000.000000")

            'l_LonString = IIf(left(l_LonString, 1) = "-", l_LonString, " " & l_LonString)
            If left(l_LonString, 1) = "-" Then
                    'do nothing
            Else
                l_LonString = " " & l_LonString
            End If

            CheckLLFormat = l_LatString & l_LonString
        End If
    End If
End Function


Public Function BearingDistCheck(p_BearingDistance As String) As String
    Dim l_Slant As Integer
'    Dim Count As Integer
    Dim l_DistanceString As String
    Dim l_BearingString As String
    Dim l_Bearing As Single
    Dim l_distance As Single
    'de  bug.print "Got here"

    l_Slant = InStr(1, p_BearingDistance, "/")

    If l_Slant = Len(p_BearingDistance) Then
        BearingDistCheck = "error"
        Exit Function
    End If

    l_BearingString = left(p_BearingDistance, l_Slant - 1)
    l_DistanceString = right(p_BearingDistance, Len(p_BearingDistance) - l_Slant)

    If IsNumeric(l_BearingString) Then
        l_Bearing = CSng(l_BearingString)
    Else
        BearingDistCheck = "error"
        Exit Function
    End If

    If IsNumeric(l_distance) Then
        l_distance = CSng(l_DistanceString)
    Else
        BearingDistCheck = "error"
        Exit Function
    End If

    If l_Bearing < 0 Or l_Bearing > 360 Then
        BearingDistCheck = "error"
        Exit Function
    End If

    If l_distance < 0 Or l_distance > 5000 Then
        BearingDistCheck = "error"
        Exit Function
    End If

    BearingDistCheck = Format(l_Bearing, "000.00") & "/" & Format(l_distance, "000.00")
End Function
Public Function FindLLBearingDist(ByVal p_LatLon As String, p_BrgDist As String, _
                                  p_LLFormat As Integer, p_MV As Single) As String
    Dim l_XString As String
    Dim l_YString As String
    Dim l_lat As Single
    Dim l_Lon As Single
    Dim l_RawBearing As Single
    Dim l_distance As Single
    Dim l_ConvertedBearing As Single
    Dim l_XAxis As Single
    Dim l_YAxis As Single
    Dim l_NewLat As Single
    Dim l_NewLon As Single
    Dim l_SignLat As String
    Dim l_SignLon As String
    Dim l_LatNS As String
    Dim l_LonEW As String
    Dim l_DegLat As Integer
    Dim l_MinLat As Integer
    Dim l_SecLat As Single
    Dim l_DegLon As Integer
    Dim l_MinLon As Integer
    Dim l_SecLon As Single
    Dim l_SlantPos As Integer
   p_BrgDist = Replace(p_BrgDist, "Z", "")
    l_SlantPos = InStr(p_BrgDist, "/")

    'if RZ value is used, and shows up without slant correct that here
    If l_SlantPos = 0 Then
        If InStr(p_BrgDist, "Z") > 0 Then
            p_BrgDist = right(p_BrgDist, 3) & "/" & Mid(p_BrgDist, 2, 4)
            l_SlantPos = 4
        End If
    End If
    p_LatLon = Trim(p_LatLon)
    If left(p_LatLon, 1) <> "-" Then p_LatLon = " " & p_LatLon

    l_lat = CSng(left(p_LatLon, 10))
    l_Lon = CSng(right(p_LatLon, 11))
    l_Lon = AdjLon(l_Lon)

    'l_RawBearing = CSng(left(p_BrgDist, 5))
    l_RawBearing = CSng(left(p_BrgDist, l_SlantPos - 1))

    'l_Distance = CSng(Mid(p_BrgDist, 8, 6))
    l_distance = CSng(right(p_BrgDist, Len(p_BrgDist) - l_SlantPos))

    'l_ConvertedBearing = IIf(l_RawBearing - p_MV < 270, 90 - (l_RawBearing - p_MV), 450 - (l_RawBeari
ng - p_MV))
    If l_RawBearing - p_MV < 270 Then
        l_ConvertedBearing = 90 - (l_RawBearing - p_MV)
    Else
        l_ConvertedBearing = 450 - (l_RawBearing - p_MV)
    End If

    l_YAxis = Sin(l_ConvertedBearing * 3.14159 / 180) * l_distance
    l_XAxis = Cos(l_ConvertedBearing * 3.14159 / 180) * l_distance
    l_NewLat = l_YAxis / 60 + l_lat
    l_NewLon = (l_XAxis / Cos(l_NewLat * 3.14159 / 180)) / 60 + l_Lon
    'need to deconstruct here and return to proper hemisphere
    If l_NewLon < -180 Then l_NewLon = l_NewLon + 360

    Select Case p_LLFormat
        Case 1
            'l_SignLat = IIf(Sgn(l_NewLat) = 1, " ", "-")
            If Sgn(l_NewLat) = 1 Then
                l_SignLat = " "
            Else
                l_SignLat = "-"
            End If

            'l_SignLon = IIf(Sgn(l_NewLon) = 1, " ", "-")
            If Sgn(l_NewLon) = 1 Then
                l_SignLon = " "
            Else
                l_SignLon = "-"
            End If

            FindLLBearingDist = l_SignLat & Format(Abs(l_NewLat), "00.000000") & l_SignLon & Format(Ab
s(l_NewLon), "000.000000")
        Case 2
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = l_LatNS & Format(l_DegLat, "000") & "." & Format(l_MinLat, "00") & _
            "." & Format(l_SecLat, "00.000") & " " & l_LonEW & Format(l_DegLon, "000") & "." & Format(
l_MinLon, "00") & _
            "." & Format(l_SecLon, "00.000")
        Case 3
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = l_LatNS & Format(l_DegLat, "00") & " " & Format(l_MinLat, "00") & _
            " " & Format(l_SecLat, "00.00") & " " & l_LonEW & Format(l_DegLon, "000") & " " & _
            Format(l_MinLon, "00") & " " & Format(l_SecLon, "00.00")
        Case 4
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = CSng((Abs(l_NewLat) - l_DegLat) * 60)
            'l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 6
0
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = CSng((Abs(l_NewLon) - l_DegLon) * 60)
            'l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 6
0
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = l_LatNS & Format(l_DegLat, "00") & " " & Format(l_MinLat, "00.00") & _
            " " & l_LonEW & Format(l_DegLon, "#00") & " " & Format(l_MinLon, "00.00")
        Case 5 'outputs x y format
            l_XString = Format(l_XAxis, "000.00")
            If Len(l_XString) = 6 Then
                l_XString = " " & l_XString
            End If

            l_YString = Format(l_YAxis, "000.00")
            If Len(l_YString) = 6 Then
                l_YString = " " & l_YString
            End If

            'returns x and y
            FindLLBearingDist = l_XString & "/" & l_YString
        Case 6 'format for .i86 site adaptation ND file list
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            If l_SecLat = 60 Then
                l_SecLat = 0
                l_MinLat = l_MinLat + 1
            End If

            If l_MinLat = 60 Then
                l_MinLat = 0
                l_DegLat = l_DegLat + 1
            End If

            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If l_SecLon = 60 Then
                l_SecLon = 0
                l_MinLon = l_MinLon + 1
            End If

            If l_MinLon = 60 Then
                l_MinLon = 0
                l_DegLon = l_DegLon + 1
            End If

            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = Format(l_DegLat, "00") & ":" & Format(l_MinLat, "00") & ":" & Format(l
_SecLat, "00.0") & l_LatNS & " " & Format(l_DegLon, "000") & ":" & Format(l_MinLon, "00") & ":" & Form
at(l_SecLon, "00.0") & l_LonEW
        Case 7 'aircraft xmove,ymove
            l_XString = Format(l_XAxis, "000.000")
            If Len(l_XString) = 7 Then
                l_XString = " " & l_XString
            End If

            l_YString = Format(l_YAxis, "000.000")
            If Len(l_YString) = 7 Then
                l_YString = " " & l_YString
            End If

            'returns x and y
            FindLLBearingDist = l_XString & "/" & l_YString
        Case 8 'outputs x y format   'this is more precise than 5, testing on rwy display
            l_XString = Format(l_XAxis, "000.000")
            If Len(l_XString) = 7 Then
                l_XString = " " & l_XString
            End If

            l_YString = Format(l_YAxis, "000.000")
            If Len(l_YString) = 7 Then
                l_YString = " " & l_YString
            End If

            'returns x and y
            FindLLBearingDist = l_XString & "/" & l_YString
        Case 9 '034-03-09.00 117-35-39.60  for .ini map output
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            If l_SecLat = 60 Then
                l_SecLat = 0
                l_MinLat = l_MinLat + 1
            End If
            If l_MinLat = 60 Then
                l_MinLat = 0
                l_DegLat = l_DegLat + 1
            End If

            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If l_SecLon = 60 Then
                l_SecLon = 0
                l_MinLon = l_MinLon + 1
            End If

            If l_MinLon = 60 Then
                l_MinLon = 0
                l_DegLon = l_DegLon + 1
            End If

            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            '034-03-09.00 117-35-39.60  for .ini map output
            FindLLBearingDist = Format(l_DegLat, "000") & "-" & Format(l_MinLat, "00") & "-" & Format(
l_SecLat, "00.00") & l_LatNS & " " & Format(l_DegLon, "000") & "-" & Format(l_MinLon, "00") & "-" & Fo
rmat(l_SecLon, "00.00") & l_LonEW
        Case 10
            'l_SignLat = IIf(Sgn(l_NewLat) = 1, " ", "-")
            If Sgn(l_NewLat) = 1 Then
                l_SignLat = " "
            Else
                l_SignLat = "-"
            End If

            'l_SignLon = IIf(Sgn(l_NewLon) = 1, " ", "-")
            If Sgn(l_NewLon) = 1 Then
                l_SignLon = " "
            Else
                l_SignLon = "-"
            End If

            FindLLBearingDist = l_SignLat & Format(Abs(l_NewLat), "00.000000") & ", " & l_SignLon & Fo
rmat(Abs(l_NewLon), "000.000000")
      Case 11  'output for STARSSimrunway '360612,N,0864117,W
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = Int((((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60))
* 60)
            If l_SecLat = 60 Then
                l_SecLat = 0
                l_MinLat = l_MinLat + 1
            End If

            If l_MinLat = 60 Then
                l_MinLat = 0
                l_DegLat = l_DegLat + 1
            End If

            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = Int((((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60))
* 60)
            If l_SecLon = 60 Then
                l_SecLon = 0
                l_MinLon = l_MinLon + 1
            End If
            If l_MinLon = 60 Then
                l_MinLon = 0
                l_DegLon = l_DegLon + 1
            End If

            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If
            '360612,N,0864117,W,

            FindLLBearingDist = Format(l_DegLat, "00") & Format(l_MinLat, "00") & Format(l_SecLat, "00
") & "," & _
            l_LatNS & "," & Format(l_DegLon, "000") & Format(l_MinLon, "00") & Format(l_SecLon, "00")
& "," & l_LonEW
Case 12 'format for .ST files no N/S  or E/W
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = Int(((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) *
 60
            If l_SecLat = 60 Then
                l_SecLat = 0
                l_MinLat = l_MinLat + 1
            End If

            If l_MinLat = 60 Then
                l_MinLat = 0
                l_DegLat = l_DegLat + 1
            End If

            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = Int(((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) *
 60
            If l_SecLon = 60 Then
                l_SecLon = 0
                l_MinLon = l_MinLon + 1
            End If

            If l_MinLon = 60 Then
                l_MinLon = 0
                l_DegLon = l_DegLon + 1
            End If

            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = Format(l_DegLat, "00") & " " & Format(l_MinLat, "00") & " " & Format(l
_SecLat, "00") & " " & Format(l_DegLon, "00") & " " & Format(l_MinLon, "00") & " " & Format(l_SecLon,
"00")
    Case 13 'format for .ST files WITH N/S   E/W
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)

            l_SecLat = Int((((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60))
* 60)
            If l_SecLat = 60 Then
                l_SecLat = 0
                l_MinLat = l_MinLat + 1
            End If
            If l_MinLat >= 60 Then
                l_MinLat = l_MinLat - 60
                l_DegLat = l_DegLat + 1
            End If

            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = Int((((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60))
* 60)
            If l_SecLon = 60 Then
                l_SecLon = 0
                l_MinLon = l_MinLon + 1
            End If

            If l_MinLon >= 60 Then
                l_MinLon = l_MinLon - 60
                l_DegLon = l_DegLon + 1
            End If

            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = l_LatNS & Format(l_DegLat, "00") & " " & Format(l_MinLat, "00") & " "
& Format(l_SecLat, "00") & " " & l_LonEW & Format(l_DegLon, "000") & " " & Format(l_MinLon, "00") & "
" & Format(l_SecLon, "00")


'This was temporarily used to generate LL without
'FindLLBearingDist = Format(l_DegLat, "00") & Format(l_MinLat, "00") & Format(l_SecLat, "00") & "   "
& Format(l_DegLon, "000") & Format(l_MinLon, "00") & Format(l_SecLon, "00")
    Case 14
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = Format(l_DegLat, "00") & Format(l_MinLat, "00") & Format(l_SecLat, "00
") & l_LatNS & Format(l_DegLon, "000") & Format(l_MinLon, "00") & Format(l_SecLon, "00") & l_LonEW
    Case 15
           'special case for output to .csv for TARGETS

           'N,34,46.13,W,119,21.82,N,34,46,7.53,W,119,21,49.45,
            l_LatNS = "N"
            l_LonEW = "W"
            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = l_LatNS & "," & Format(l_DegLat, "00") & "," & Format(l_MinLat + (l_Se
cLat / 60), "00.00") & "," & l_LonEW & "," & _
            Format(l_DegLon, "000") & "," & Format(l_MinLon + (l_SecLon / 60), "00.00") & "," & _
             l_LatNS & "," & Format(l_DegLat, "00") & "," & Format(l_MinLat, "00") & "," & Format(l_Se
cLat, "00.00") & "," & l_LonEW & "," & _
             Format(l_DegLon, "000") & "," & Format(l_MinLon, "00") & "," & Format(l_SecLon, "00.00")
& ","
  Case 16 'outputs x y format   'in screen pixels converted for scale
            l_XString = Format(l_XAxis * g_ScaleFactor, "0000.000")
            If Len(l_XString) = 8 Then
                l_XString = " " & l_XString
            End If

            l_YString = Format(l_YAxis * g_ScaleFactor, "0000.000")
            If Len(l_YString) = 8 Then
                l_YString = " " & l_YString
            End If

            'returns x and y
            FindLLBearingDist = l_XString & "/" & l_YString
    Case 17 'outputs x y format higher precision
            l_XString = Format(l_XAxis, "000.000")
            If Len(l_XString) = 7 Then
                l_XString = " " & l_XString
            End If

            l_YString = Format(l_YAxis, "000.000")
            If Len(l_YString) = 7 Then
                l_YString = " " & l_YString
            End If

            'returns x and y
            FindLLBearingDist = l_XString & "/" & l_YString
    Case 18 'output to GPDat Format from decimal

            l_DegLat = Int(Abs(l_NewLat))
            l_MinLat = Int((Abs(l_NewLat) - l_DegLat) * 60)
            l_SecLat = (((Abs(l_NewLat) - l_DegLat) * 60) - Int((Abs(l_NewLat) - l_DegLat) * 60)) * 60
            l_DegLon = Int(Abs(l_NewLon))
            l_MinLon = Int((Abs(l_NewLon) - l_DegLon) * 60)
            l_SecLon = (((Abs(l_NewLon) - l_DegLon) * 60) - Int((Abs(l_NewLon) - l_DegLon) * 60)) * 60
            If Sgn(l_NewLat) = -1 Then
                l_LatNS = "S"
            End If

            If Sgn(l_NewLon) = 1 Then
                l_LonEW = "E"
            End If

            FindLLBearingDist = Format(l_DegLat, "00") & " " & Format(l_MinLat, "00") & " " & Format(l
_SecLat, "00.0000") & " " & Format(l_DegLon, "000") & " " & Format(l_MinLon, "00") & " " & Format(l_Se
cLon, "00.0000")


    End Select
End Function


Public Function TruncateLine(p_OriginOffset As Single, p_OutsideOffset As Single, p_Radius As Single,
p_DstOriginOffset As Single) As Single
    Dim l_x As Single
    Dim l_AngleA As Single
    Dim l_AngleB As Single
    Dim l_AngleC As Single

    l_AngleA = Abs(p_OriginOffset - p_OutsideOffset)
    If l_AngleA >= 180 Then
        l_AngleA = 360 - l_AngleA
    End If

    If l_AngleA = 90 Then
```

```
        'de  bug.print "Found a 90 deg angle!, Correcting now to 89.99"
        l_AngleA = 89
    End If

    l_x = (p_DstOriginOffset * Sin(l_AngleA * PI / 180)) / p_Radius
    If Abs(l_x) = 1 Then

    l_x = 0.9999 * l_x
    End If

    l_AngleB = Atn(l_x / Sqr(-l_x * l_x + 1)) * (180 / PI)

    l_AngleC = 180 - l_AngleA - l_AngleB
    If l_AngleB = 0 Then
        l_AngleB = 0.0001
    End If

    TruncateLine = Abs(p_DstOriginOffset * Sin(l_AngleC * PI / 180) / Sin(l_AngleB * PI / 180))
End Function

Public Sub ShorelineExtract(p_RadarOrigin As String, p_FileGeo As Integer, p_Output As Integer, p_Radi
us As Integer, p_Mag As Single)
    Dim l_LatMapString As String
    Dim l_LongMapString As String
    Dim l_MapLat As String
    Dim l_MapLon As String
    Dim l_MapFile(3) As String
    Dim l_maploop As Integer
    Dim l_FileShoreLine As Integer
    Dim l_ReadCount As Integer
    Dim l_GeoSearch As String
    Dim l_georead1 As String
    Dim l_georead2 As String
    Dim l_georead3 As String
    Dim l_Position2 As String
    Dim l_Position1 As String
    Dim l_LatLonSplit As Integer
    Dim l_CharCount As Integer
    Dim l_LatNegTest As String
    Dim l_ConvertedShoreString As String
    Dim l_LenStringShore As Integer
    Dim l_ConvertedCircularOrigin As String
    Dim l_BearingDistance As String
    Dim l_ShoreLineCount As Single

    'de  bug.print "Shoreline maps"
    l_LatMapString = left(Trim(p_RadarOrigin), 4)
    l_LongMapString = Mid(Trim(p_RadarOrigin), 16, 4)

    'l_MapLat = IIf(left(l_LatMapString, 1) = "N", CInt(Mid(l_LatMapString, 2, 3)), CInt(Mid(l_LatMapS
tring, 2, 3)) * -1)
    If left(l_LatMapString, 1) = "N" Then
        l_MapLat = CInt(Mid(l_LatMapString, 2, 3))
    Else
        l_MapLat = CInt(Mid(l_LatMapString, 2, 3)) * -1
    End If

    'l_MapLon = IIf(left(l_LongMapString, 1) = "E", CInt(Mid(l_LongMapString, 2, 3)), CInt(Mid(l_LongM
apString, 2, 3)) * -1)
    If left(l_LongMapString, 1) = "E" Then
        l_MapLon = CInt(Mid(l_LongMapString, 2, 3))
    Else
        l_MapLon = CInt(Mid(l_LongMapString, 2, 3)) * -1
    End If

    l_ConvertedCircularOrigin = CheckLLFormat(RTrim(p_RadarOrigin))
    'E36toE180_N65toN30  Asia

    If (l_MapLon >= 36) And (l_MapLon <= 180) And (l_MapLat <= 65) And (l_MapLat >= 30) Then
    l_MapFile(0) = "E36toE180_N65toN30"
    End If

    'E2toE60_N72toN53   Russia
    If (l_MapLon >= 2) And (l_MapLon <= 60) And (l_MapLat <= 72) And (l_MapLat >= 53) Then
        l_MapFile(0) = "E2toE60_N72toN53"
    End If
    'E55toE120_N72toN53  Siberia

    If (l_MapLon >= 55) And (l_MapLon <= 120) And (l_MapLat <= 72) And (l_MapLat >= 53) Then
        l_MapFile(0) = "E55toE120_N72toN53"
    End If

    'E67toE120_N32toS30  Indian Subcontinent

    If (l_MapLon >= 67) And (l_MapLon <= 120) And (l_MapLat <= 32) And (l_MapLat >= -30) Then
        l_MapFile(0) = "E67toE120_N32toS30"
    End If

    'E110toE180_N32toS70 Australia, Oceania

    If (l_MapLon >= 110) And (l_MapLon <= 180) And (l_MapLat <= 32) And (l_MapLat >= -70) Then
        l_MapFile(0) = "E110toE180_N32toS70"
    End If

    'W20toE70_N40toS70   Africa, Middle East

    If (l_MapLon >= -20) And (l_MapLon <= 70) And (l_MapLat <= 40) And (l_MapLat >= -70) Then
        l_MapFile(0) = "W20toE70_N40toS70"
    End If

    'W100toW50_N76toN53  North East Canada

    If (l_MapLon >= -100) And (l_MapLon <= -50) And (l_MapLat <= 76) And (l_MapLat >= 53) Then
        l_MapFile(0) = "W100toW50_N76toN53"
    End If

    'W175toW50_N55toN10 North and Central America

    If (l_MapLon >= -180) And (l_MapLon <= -50) And (l_MapLat <= 55) And (l_MapLat >= 10) Then
        l_MapFile(0) = "W175toW50_N55toN10"
        l_MapFile(1) = "E36toE180_N65toN30"
        l_MapFile(2) = "W180toW100_N73toN50"
    End If

    'W175toW55_N75toN55 Northern Canada, Alaska

    If (l_MapLon >= -175) And (l_MapLon <= -55) And (l_MapLat <= 75) And (l_MapLat >= 55) Then
        l_MapFile(0) = "W175toW55_N75toN55"
    End If

    'W180toW0_N12toS70  South America

    If (l_MapLon >= -180) And (l_MapLon <= 0) And (l_MapLat <= 12) And (l_MapLat >= -70) Then
        l_MapFile(0) = "W180toW0_N12toS70"
    End If

    'W180toW100_N73toN50 Alaska

    If (l_MapLon >= -180) And (l_MapLon <= -100) And (l_MapLat <= 73) And (l_MapLat >= 50) Then
        l_MapFile(0) = "W180toW100_N73toN50"
    End If

    'W20toE40_N70toN30 Europe

    If (l_MapLon >= -20) And (l_MapLon <= 40) And (l_MapLat <= 70) And (l_MapLat >= 30) Then
        l_MapFile(0) = "W20toE40_N70toN30"
    End If

    'W66toE10_N85toN53 Iceland,Greenland
    If (l_MapLon >= -66) And (l_MapLon <= 10) And (l_MapLat <= 85) And (l_MapLat >= 53) Then
        l_MapFile(0) = "W66toE10_N85toN53"
    End If
    For l_maploop = 0 To 2
    g_ShorelineMissing = False
    l_FileShoreLine = FreeFile
    If Dir(g_ProgPath & "CommonData\USGS Shoreline\" & l_MapFile(l_maploop) & ".dat") = "" Then
        'MsgBox (g_ProgPath & "CommonData\USGS Shoreline\" & l_MapFile(0) & ".dat is missing")
        g_ShorelineMissing = True
    Else
        Open g_ProgPath & "CommonData\USGS Shoreline\" & l_MapFile(l_maploop) & ".dat" For Input As #l
_FileShoreLine
        l_ReadCount = 1
        If p_Output = 0 Then
            Print #p_FileGeo, ";Shoreline Data " & "p_Radius " & l_ConvertedCircularOrigin & " " & Tri
m(p_RadarOrigin)
        End If
        l_georead1 = ""
        l_georead2 = ""
        l_georead3 = ""
        Input #l_FileShoreLine, l_GeoSearch
        Do While Not EOF(l_FileShoreLine)
            Input #l_FileShoreLine, l_GeoSearch


            If Trim(l_GeoSearch) = "# -b" Then
                l_Position2 = ";break"
                l_ReadCount = l_ReadCount + 1
            Else
                l_LenStringShore = Len(Trim(l_GeoSearch))
                l_CharCount = 8

                Do While l_CharCount < 13
                    l_CharCount = l_CharCount + 1
                    If Asc(Mid(Trim(l_GeoSearch), l_CharCount, 1)) = vbKeyTab Then l_LatLonSplit = l_C
harCount
                Loop

                'l_LatNegTest = IIf(Mid(l_GeoSearch, l_LatLonSplit + 1, 1) <> "-", " ", "")
                If Mid(l_GeoSearch, l_LatLonSplit + 1, 1) <> "-" Then
                    l_LatNegTest = " "
                Else
                    l_LatNegTest = ""
                End If

                l_ConvertedShoreString = l_LatNegTest & Format(CSng(right(l_GeoSearch, l_LenStringShor
e - l_LatLonSplit)), "00.000000") & Format(CSng(left(l_GeoSearch, l_LatLonSplit - 1)), "000.000000")
                ' 'de  bug.print "CCO:" & l_ConvertedCircularOrigin
                l_BearingDistance = LL2BearingDistance(l_ConvertedCircularOrigin, l_ConvertedShoreStri
ng, p_Mag)

                If CSng(Mid(l_BearingDistance, 7, 5)) <= CSng(p_Radius) - 1 Then
                    If p_Output = 0 Then 'p_Output DETERMINES THE FORMAT EITHER ll OR xy
                        l_Position2 = FindLLBearingDist(l_ConvertedShoreString, "000.00/000.00", 2, 0)
                        'de  bug.print "stop and step Shoreline Extract p_Output 0"
                    End If

                    If p_Output = 1 Then
                        l_BearingDistance = BearingDistCheck(l_BearingDistance)
                        If l_BearingDistance = "error" Then
                            l_Position2 = ";error"
                        End If

                        If l_BearingDistance <> "error" Then
                            'de  bug.print "stop and step Shoreline Extract output 1"

                            l_Position2 = FindLLBearingDist(l_ConvertedCircularOrigin, l_BearingDistan
ce, 5, 0) 'p_Mag replaced with 0
                        End If
                    End If
                End If

            End If

            If l_Position2 <> l_Position1 And l_Position2 <> ";break" Then
                If p_Output = 0 Then
                    Print #p_FileGeo, l_Position1 & " " & l_Position2 & " shoreline"
                End If
                If p_Output = 1 Then
                    If left(l_Position2, 1) <> ";" Then
                        Print #p_FileGeo, l_Position1 & " " & l_Position2 & " shoreline"

                        l_ShoreLineCount = l_ShoreLineCount + 1
                    End If
                End If
            End If
            l_Position1 = l_Position2
        Loop
        'de  bug.print "Shoreline elements:" & l_ShoreLineCount

        Close #l_FileShoreLine

    End If
   Next l_maploop
End Sub

Public Function HEADINGCHECK(p_Heading As Single) As Single
    'de  bug.print "HEADINGCHECK IN geodeticcalc"
    If p_Heading < 0 Then
        p_Heading = p_Heading + 360
        HEADINGCHECK = p_Heading
    End If

    'If p_Heading > 360 Then
    Do While p_Heading > 360
        p_Heading = p_Heading - 360
        Loop


        HEADINGCHECK = p_Heading
    'End If
End Function

Public Function LL2MAPBD(p_Origin As String, p_Target As String, p_Var As Single) As String
    Dim l_OriginLon As Single
    Dim l_OriginLat As Single
    Dim l_TargetLon As Single
    Dim l_TargetLat As Single
    Dim l_Adjacent As Single
    Dim l_Opposite As Single
    Dim l_Hypotenuse As Single
    Dim l_UnconvertedHeading As Single
    Dim l_TrueBearing As Single
    Dim l_Loxodrome As Single
    Dim l_AdjHyp As Double

    p_Target = Trim(p_Target)
    If left(p_Target, 1) <> "-" Then
        p_Target = " " & p_Target
    End If

    p_Origin = Trim(p_Origin)
    If left(p_Origin, 1) <> "-" Then
        p_Origin = " " & p_Origin
    End If

    l_OriginLat = CSng(left(p_Origin, 10))
    'de  bug.print "------------"
    'de  bug.print p_Origin

    'de  bug.print "OriginLat:" & l_OriginLat

    l_OriginLon = CSng(right(p_Origin, 11))
    If l_OriginLon > 0 Then
        l_OriginLon = l_OriginLon - 360
    End If

    'de  bug.print "OriginLon:" & l_OriginLon
    'de  bug.print p_Target
    'de  bug.print "p_Target:" & Left(p_Target, 10)
    l_TargetLat = CSng(left(p_Target, 10))
    'de  bug.print "TargetLat:" & l_TargetLat

    l_TargetLon = CSng(right(p_Target, 11))
    If l_TargetLon > 0 Then
        l_TargetLon = l_TargetLon - 360
    End If

    'de  bug.print "TargetLon:" & l_TargetLon

    l_Adjacent = Cos((l_OriginLat + l_TargetLat) / 2 * 3.14159265 / 180) * (l_OriginLon - l_TargetLon)
 * -60
    'de  bug.print "Adjacent:" & l_Adjacent

    l_Opposite = (l_TargetLat - l_OriginLat) * 60
    'de  bug.print "Opposite:" & l_Opposite

    l_Hypotenuse = Sqr(l_Adjacent ^ 2 + l_Opposite ^ 2)
    'If l_Hypotenuse <= 500 Then
    'de  bug.print "Hypotenuse:" & l_Hypotenuse
    'End If

    If l_Hypotenuse = 0 Then
        l_AdjHyp = 0
    Else
        l_AdjHyp = CSng(l_Adjacent / l_Hypotenuse)
    End If
    'de  bug.print "AdjHyp1:" & l_AdjHyp

    If l_AdjHyp = 1 Then
        l_AdjHyp = 0.999999999999999
    End If
    'de  bug.print "AdjHyp2:" & l_AdjHyp

    If l_AdjHyp = -1 Then
        l_AdjHyp = -0.999999999999999
    End If
    'de  bug.print "AdjHyp3:" & l_AdjHyp

    l_UnconvertedHeading = (Atn(-l_AdjHyp / Sqr(-l_AdjHyp * l_AdjHyp + 1)) + 2 * Atn(1)) * 180 / 3.141
59265
    'de  bug.print "UnconvertedHeading:" & l_UnconvertedHeading
    If (Sgn(l_Opposite)) = -1 Then
        l_TrueBearing = 90 + l_UnconvertedHeading

        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If

        LL2MAPBD = Format(l_Loxodrome, "000.00") & "/" & Format(l_Hypotenuse, "000.00")
    Else
        'l_TrueBearing = IIf(Sgn(l_Adjacent) = -1, 450 - l_UnconvertedHeading, 90 - l_UnconvertedHeadi
ng)
        If Sgn(l_Adjacent) = -1 Then
            l_TrueBearing = 450 - l_UnconvertedHeading
        Else
            l_TrueBearing = 90 - l_UnconvertedHeading
        End If

        'l_Loxodrome = IIf(l_TrueBearing + p_Var < 0, 360 + (l_TrueBearing + p_Var), l_TrueBearing + p
_Var)
        If l_TrueBearing + p_Var < 0 Then
            l_Loxodrome = 360 + (l_TrueBearing + p_Var)
        Else
            l_Loxodrome = l_TrueBearing + p_Var
        End If

        LL2MAPBD = Format(l_Loxodrome, "000.00") & "/" & Format(l_Hypotenuse, "000.00")
    End If
End Function







Public Function FindTabs(datastring As String, tabs2count As Integer)
Dim placecount As Integer
Dim tabcount As Integer

placecount = 1
        tabcount = 1
          Do While tabcount < tabs2count
          If Asc(Mid(datastring, placecount, 1)) = vbKeyTab Then
          pubTabLocate(tabcount) = placecount
          tabcount = tabcount + 1
          End If
          placecount = placecount + 1
          Loop
End Function
Public Function tabdata(datastring As String, tabstart As Integer) As String
tabdata = Mid(datastring, pubTabLocate(tabstart) + 1, pubTabLocate(tabstart + 1) - pubTabLocate(tabsta
rt) - 1)
End Function
Public Function IsOnScope(strOrigin As String, strTarget As String, MVar As Single) As String
Dim l_XYStr As String
Dim l_currX As Single
Dim l_currY As Single
Dim l_split As Variant
Dim l_FoundX As Boolean
Dim l_FoundY As Boolean

IsOnScope = False 'until proven true
l_XYStr = LL2XY(strOrigin, strTarget, MVar)
l_split = Split(l_XYStr, "/")
''de bug.printg_XOffset & " " & g_YOffset
        l_currX = g_AntennaX + (g_ScaleFactor * (l_split(0))) '- g_XOffset  'already
        l_currY = g_AntennaY - (g_ScaleFactor * (l_split(1))) '- g_YOffset
        l_FoundX = True
        l_FoundY = True 'innocent until proven guilty

        If l_currX >= frmGUI.pctMap.ScaleWidth - 12 Then
                l_FoundX = False
            End If

            If l_currX <= 12 Then
                l_FoundX = False
            End If

            If l_currY >= frmGUI.pctMap.ScaleHeight - 12 Then
                l_FoundY = False
            End If

            If l_currY < 12 Then
                l_FoundY = False
            End If

            If l_FoundY = True And l_FoundX = True Then

            IsOnScope = True
            'de bug.print"IsOnScope TRUE " & l_currX & " " & l_currY
            End If

End Function
Public Function LL2XY(strOrigin As String, strTarget As String, MVar As Single) As String
    Dim Xstr As String
    Dim Ystr As String
    Dim Distance As Single
    Dim Xaxis As Single
    Dim Yaxis As Single
    Dim OriginLat As Single
    Dim originLon As Single
    Dim TgtLat As Single
    Dim TgtLon As Single
    Dim Adjacent As Single
    Dim Opposite As Single
    Dim Hypotenuse As Single
    Dim adjhyp As Double
    Dim UnconvertedHdg As Single
    Dim ConvertedBearing  As Single
    Dim TrueBearing As Single
    Dim brg As Single
    Dim l_StrOrigin  As String
    Dim l_StrTarget  As String

    l_StrOrigin = strOrigin
    l_StrTarget = strTarget

    l_StrTarget = Trim(l_StrTarget)
    If left(l_StrTarget, 1) <> "-" Then l_StrTarget = " " & l_StrTarget
    l_StrOrigin = Trim(l_StrOrigin)
    If left(l_StrOrigin, 1) <> "-" Then l_StrOrigin = " " & l_StrOrigin

    OriginLat = CSng(left(l_StrOrigin, 10))
    originLon = CSng(right(l_StrOrigin, 11))
    TgtLat = CSng(left(l_StrTarget, 10))
    TgtLon = CSng(right(l_StrTarget, 11))
    Adjacent = Cos((OriginLat + TgtLat) / 2 * 3.14159265 / 180) * (originLon - TgtLon) * -60
    Opposite = (TgtLat - OriginLat) * 60
    Hypotenuse = Sqr(Adjacent ^ 2 + Opposite ^ 2)
    If Hypotenuse = 0 Then adjhyp = 0 Else adjhyp = CSng(Adjacent / Hypotenuse)

    If adjhyp = 1 Then adjhyp = 0.99999999 '9999999
    If adjhyp = -1 Then adjhyp = -0.9999999 '99999999
    UnconvertedHdg = (Atn(-adjhyp / Sqr(-adjhyp * adjhyp + 1)) + 2 * Atn(1)) * 180 / 3.14159265
    If (Sgn(Opposite)) = -1 Then
        TrueBearing = 90 + UnconvertedHdg
    Else
        'AUTOFIXIIf function found
        TrueBearing = IIf(Sgn(Adjacent) = -1, 450 - UnconvertedHdg, 90 - UnconvertedHdg)
    End If

    'AUTOFIXIIf function found
    brg = IIf(TrueBearing < 0, 360 + (TrueBearing), TrueBearing)

    Distance = Hypotenuse

    If brg > 360 Then brg = brg - 360
    If brg < 0 Then brg = brg + 360


    'AUTOFIXIIf function found
    ConvertedBearing = IIf(brg - MVar < 270, 90 - (brg - MVar), 450 - (brg - MVar))
    Yaxis = Sin(ConvertedBearing * 3.14159 / 180) * Distance
    Xaxis = Cos(ConvertedBearing * 3.14159 / 180) * Distance


    Xstr = Format(Xaxis, "000.000")
    If Len(Xstr) = 7 Then Xstr = " " & Xstr
    Ystr = Format(Yaxis, "000.000")
    If Len(Ystr) = 7 Then Ystr = " " & Ystr


    LL2XY = Xstr & "/" & Ystr

End Function
Public Function RwyLL2XY(strOrigin As String, strTarget As String, MVar As Single) As String
    Dim Xstr As String
    Dim Ystr As String
    Dim Distance As Single
    Dim Xaxis As Single
    Dim Yaxis As Single
    Dim OriginLat As Single
    Dim originLon As Single
    Dim TgtLat As Single
    Dim TgtLon As Single
    Dim Adjacent As Single
    Dim Opposite As Single
    Dim Hypotenuse As Single
    Dim adjhyp As Double
    Dim UnconvertedHdg As Single
    Dim ConvertedBearing  As Single
    Dim TrueBearing As Single
    Dim brg As Single
    Dim l_StrOrigin  As String
    Dim l_StrTarget  As String

    l_StrOrigin = strOrigin
    l_StrTarget = strTarget

    l_StrTarget = Trim(l_StrTarget)
    If left(l_StrTarget, 1) <> "-" Then l_StrTarget = " " & l_StrTarget
    l_StrOrigin = Trim(l_StrOrigin)
    If left(l_StrOrigin, 1) <> "-" Then l_StrOrigin = " " & l_StrOrigin

    OriginLat = CSng(left(l_StrOrigin, 10))
    originLon = CSng(right(l_StrOrigin, 11))
    TgtLat = CSng(left(l_StrTarget, 10))
    TgtLon = CSng(right(l_StrTarget, 11))
    Adjacent = Cos((OriginLat + TgtLat) / 2 * 3.14159265 / 180) * (originLon - TgtLon) * -60
    Opposite = (TgtLat - OriginLat) * 60
    Hypotenuse = Sqr(Adjacent ^ 2 + Opposite ^ 2)
    If Hypotenuse = 0 Then adjhyp = 0 Else adjhyp = CSng(Adjacent / Hypotenuse)

    If adjhyp = 1 Then adjhyp = 0.99999999 '9999999
    If adjhyp = -1 Then adjhyp = -0.9999999 '99999999
    UnconvertedHdg = (Atn(-adjhyp / Sqr(-adjhyp * adjhyp + 1)) + 2 * Atn(1)) * 180 / 3.14159265
    If (Sgn(Opposite)) = -1 Then
        TrueBearing = 90 + UnconvertedHdg
    Else
        'AUTOFIXIIf function found
        TrueBearing = IIf(Sgn(Adjacent) = -1, 450 - UnconvertedHdg, 90 - UnconvertedHdg)
    End If

    'AUTOFIXIIf function found
    brg = IIf(TrueBearing < 0, 360 + (TrueBearing), TrueBearing)

    Distance = Hypotenuse

    If brg > 360 Then brg = brg - 360
    If brg < 0 Then brg = brg + 360


    'AUTOFIXIIf function found
    ConvertedBearing = IIf(brg - MVar < 270, 90 - (brg - MVar), 450 - (brg - MVar))
    Yaxis = Sin(ConvertedBearing * 3.14159 / 180) * Distance
    Xaxis = Cos(ConvertedBearing * 3.14159 / 180) * Distance


    Xstr = Format(Xaxis, "000.000")
    If Len(Xstr) = 7 Then Xstr = " " & Xstr
    Ystr = Format(Yaxis, "000.000")
    If Len(Ystr) = 7 Then Ystr = " " & Ystr


    RwyLL2XY = Xstr & "/" & Ystr

End Function
```