交通部中央氣象局委託研究計畫成果報告

船舶海象觀測作業系統

Observation System of Maritime Meteorology

計畫類別：☑ 國內 ☐ 國內

計畫編號：CWB 86-20-02

執行期間：85年7月1日至86年6月30日

計畫主持人：李賢文

國立臺灣海洋大學 海洋科學系

中華民國八十六年六月三十日

前言：

為了促進商船參與海洋氣象資料之收集，我們需要建立一套全自動的船舶海洋氣象觀測系統，在不影響船員工作之下，將可以推廣鼓勵商船使用共同從事台灣海洋氣象觀測工作，增進海洋氣象預報之能力。

若要建立我國海洋氣象即時的資料，最佳方式為擁有專用的海洋氣象觀測船。但考量目前政府財政上的困難和氣象即時資料的重要性，運用民間船舶所觀測的海洋氣象資料作有系統的記錄，並加以自動化及同步傳輸至中央氣象局，應為目前較可行的方式。

但是目前民間船舶所觀測的海洋氣象資料，均為目測儀器值，資料的準確性不足，所以需要將其所觀測的海洋氣象資料電子化，和訂定統一的資料格式及傳輸流程，以利資料傳輸。

中央氣象局根據所收集的海洋氣象資料作預報研判，並適時提供海上航行船隻海洋氣象資訊，以確保航行安全。建立海洋氣象資料庫作為國家建設之參考，同時可與他國交換資料，促進國際合作。

我國目前鄰進的海域，除了若干島嶼設有氣象觀測站之外，海洋上並無足夠的氣象觀測點，收集即時的氣象觀測資料，因此缺乏精確的即時海域氣象資料，運用民間船舶觀測海洋氣象資料，可彌補海洋氣象預報責任區內觀測資料之不足。

依據【行政院第十四次科技顧問會議結論與建議處理辦法中，議題參之主題一，會議結論第四條第五點】，中央氣象局擬定「推動商船參與海洋氣象觀測草案」，以鼓勵商船參與海洋氣象觀測工作。以便商船一方面將所觀測的海洋氣象資料傳輸至中央氣象局作海洋氣象預報參考，另一方面中央氣象局將完整的海洋氣象預報以通訊網方式提供責任區內的商船，以確保航行安全。如此雙方面得以互惠互利的方式共同執行本項計畫。

為了達到上述「推動商船參與海洋氣象觀測」的目標，首先需要建立船舶海象自動觀測與記錄系統，如此商船才在船上有限的人力之下，執行海象觀測作業。本計畫擇定國立臺灣海洋大學所管理的「海研二號」，建立船舶海洋氣象觀測自動系統，以供未來推動商船參與海洋氣象觀測之基礎。

方法：

擇定「海研二號」研究船，建立為海洋氣象觀測系統自動化的船舶研究船上現有的氣象觀測儀器、定位系統與電腦設備及性能如下：

1. 風速風向計：觀測風速風向，將電子信號送到風速風向平均電路板。但是船舶在航行中所觀測得風速風向為相對風速風向，並非實際真風速風向，須參考電羅經的船艏向及全球定位系統的船速，經由電腦程式計算後，才能獲得真風速風向。其計算公式如下：

   $W = V - U$

   其中 U 為船速向量（由電羅經與GPS定出），

   V 為船上風速儀測定的風速向量，

   W 為實際風速向量。

2. 氣溫計：測量海平面空氣溫度並將電子信號送到JUNCTION BOX。
3. 氣壓計：測量海平面大氣壓力並將電子信號送到JUNCTION BOX。
4. 相對濕度計：測量海平面空氣的相對濕度並將電子信號送到JUNCTION BOX。
5. 風速風向平均電路板：由於風速及風向並非固定不變，若單取一時點的觀測值誤差較大，因此取一時段的平均作為觀測值，此電路板的功能除了將一段時間的風速風向平均外，還負責將電子信號送到JUNCTION BOX 。

6. JUNCTION BOX：接收風速風向平均電路板、氣溫計、氣壓計及相對濕度計的電子信號，將電子信號送到AD-CONVERT。

7. AD-CONVERT-I：接收JUNCTION BOX送來的類比信號，轉換成數位信號後送入PC。

8. 電羅經：將船艏向的信號經COM2送入PC。

9. 全球衛星定位系統：將船位及船舶速度信號經COM1送入PC。

10. 個人電腦：負責記錄海象觀測資料及計算真風速風向。

為了建立自動化系統，必須將氣象觀測儀器記錄的資料予以數位化，並撰寫電腦程式，以控制及整合各項觀測資料，並予以存檔。本系統之架構如圖一。

結果：

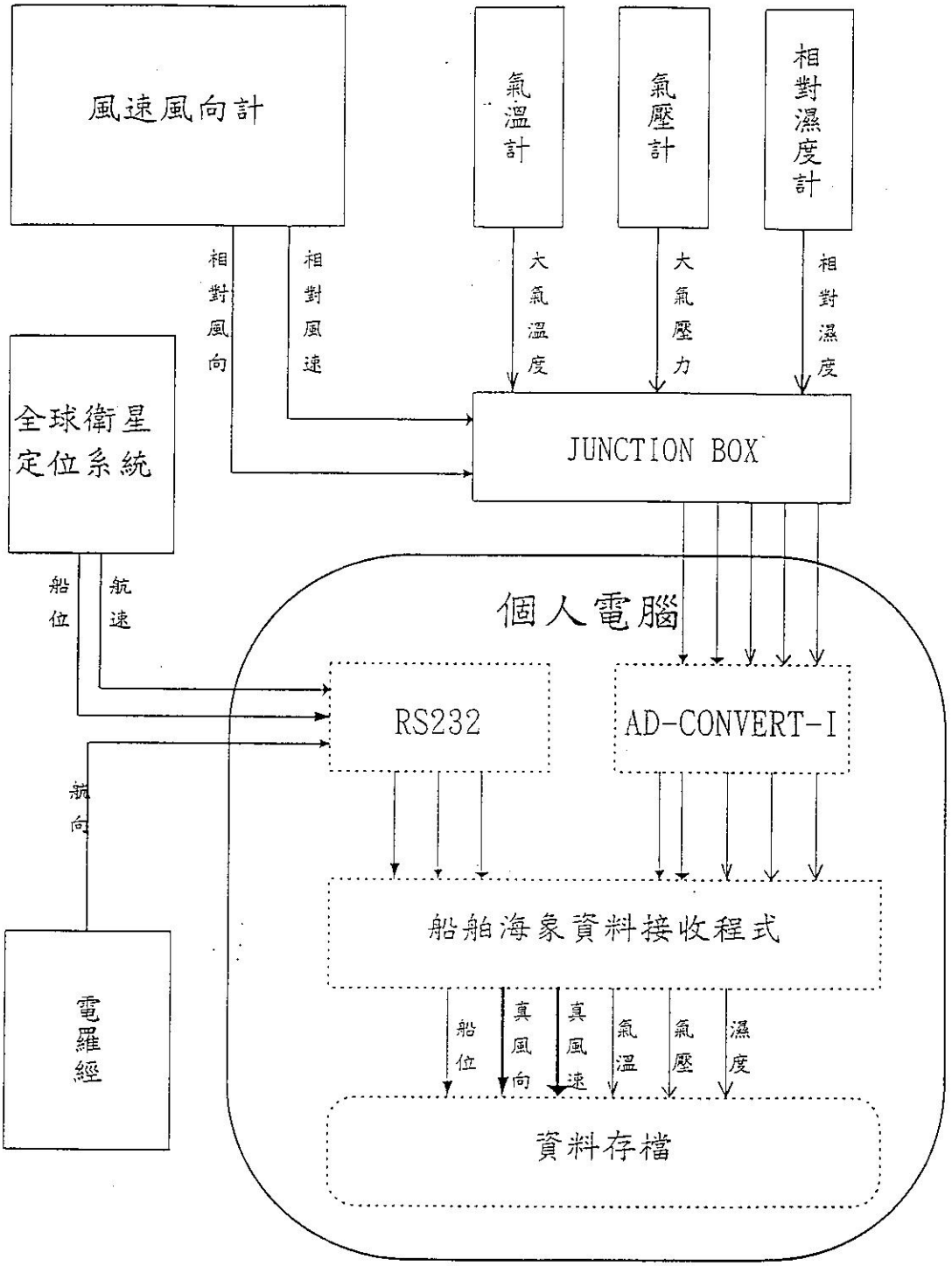
本研究成功地將電羅經訊號接入電腦中，如此可以精確地獲得船首向，可避免停船的GPS之船首向失真之問題，如此可精確地計算出真風向，而不受限於船舶移動的影響。

另外我們是以BASIC語言撰寫電腦程式，淺顯易懂、修改容易，便於日後維護，適合推廣使用。

目前本系統所觀測記錄存檔的資料樣本如附錄。

誌謝：

本計畫得以完成，主要歸功於「海研二號」研究船探測技士曾安源先生蘇兆輝先生與黃余達先生，他們三位在本計畫中兼任研究助理，負責各項儀器的操作維護及程式設計。另外也感謝「海研二號」全體船員之協助。



圖一 船舶海象觀測系統架構圖

海 象 觀 測 系 統 原 始 程 式

```
' Program:      Marine Star system for Receiving & Recording data
' Programmer:   David Sue

    ON ERROR GOTO ErrorHandler

'---------------------------- MAIN Loop ---------------------------------

    GOSUB IniInitialization

    DO
        GOSUB CheckInkey
        GOSUB GetGPSData
        GOSUB getgyrodata
        GOSUB CheckGPSData
        GOSUB GetWeatherData
        GOSUB WindDirectionAndSpeed
        GOSUB AirHumidity
        GOSUB AirTemperature
        GOSUB SolarRadiation
        GOSUB BarometricPressure
        TimeTmp$ = LEFT$(TIME$, 2) + MID$(TIME$, 4, 2) + RIGHT$(TIME$, 2)
        GOSUB DisplayData
        GOSUB RecordData
    LOOP

    GOSUB ExitSystem

IniInitialization:

    CLS
    SCREEN 12

    LINE (380, 310)-(450, 310), 3
    LINE (380, 360)-(450, 360), 4
    LINE (380, 405)-(450, 405), 2
    LOCATE 20, 42: PRINT " Ship "
    LOCATE 23, 42: PRINT "R-Wind"
    LOCATE 26, 42: PRINT "T-Wind"

    ViewXMin = 20
    ViewYMin = 250
    ViewLength = 200
    VIEW (ViewXMin, ViewYMin)-(ViewXMin + ViewLength, ViewYMin + ViewLength), ,
6
    WINDOW (-50, -50)-(50, 50)
```

```
    LOCATE 2, 1: PRINT "Press Esc to stop."
REM     LOCATE 6, 1: PRINT " Time     TDir TSpd   Temp   Solar   Baro     Latitude
Longitude "
REM     LOCATE 7, 1: PRINT "======   ==== ====   ====   =====   ====
========== ==========="
    LOCATE 4, 1: PRINT " Time     TDir   TSpd   Temp     Baro      Humidity   Solar
Latitude      Longitude "
    LOCATE 5, 1: PRINT "======   ====   ====   ====   ======    ========
=====   ========== ==========="
    LOCATE 8, 9: PRINT " SDir     SSpd"
    LOCATE 9, 9: PRINT " ====     ====="
    LOCATE 12, 9: PRINT " RDir    RSpd"
    LOCATE 13, 9: PRINT " ====    ====="
REM     LOCATE 9, 25: PRINT "Humidity"
REM     LOCATE 10, 25: PRINT "========"

    AvgInterval = 1
    DisplayTimeChoice = 1       ' Choice of Seconds/Display
    RecordTimeChoice = 2        ' Choice of Seconds/Record
                                  ' SELECT CASE RecordTimeChoice/DisplayTimeChoice
                                  ' CASE 0:          Unlimited
                                  ' CASE 1:          10 Seconds / Record
                                  ' CASE 2:          1 Minutes / Record
                                  ' CASE 3:          10 Minutes / Record
                                  ' CASE 4:          1 Hours    / Record
                                  ' END SELECT

    floppy$ = "T"
    pi = 3.1415926#
    AvgMemory = AvgInterval * 10
    RawDataCount = 6
    OldDate$ = " "
    DIM WindSpeed(AvgMemory), WindDirect(AvgMemory)
    DIM rawdata(RawDataCount - 1), filename$(5)

    filename$(1) = "com1:4800,n,8,1,cs0.ds0"
    OPEN filename$(1) FOR INPUT AS #1
    filename$(2) = "com2:4800,n,8,1,cs0.ds0"
    OPEN filename$(2) FOR INPUT AS #2

    RETURN

ExitSystem:

    CLOSE
    STOP
    END
```

```
GetGPSData:

    GPSData$ = ""
    gpsidflag = 0
    DO WHILE gpsidflag < 3
        LINE INPUT #1, GPSData$
        GDHead$ = MID$(GPSData$, 2, 6)
        IF GDHead$ = "$GPGLL" THEN
            gpsidflag = 1
            GPSLat$ = MID$(GPSData$, 9, 2) + CHR$(248) + MID$(GPSData$, 11, 6) + "'"
+ MID$(GPSData$, 18, 1)
            GPSLon$ = MID$(GPSData$, 20, 3) + CHR$(248) + MID$(GPSData$, 23, 6) + "'"
+ MID$(GPSData$, 30, 1)
        END IF
        IF GDHead$ = "$GPVTG" AND gpsidflag = 1 THEN
            gpsidflag = 2
            GPSTrack$ = MID$(GPSData$, 9, 3)
            GPSSpeed$ = MID$(GPSData$, 21, 5)
        END IF
        IF GDHead$ = "$GPZDA" AND gpsidflag = 2 THEN
            gpsidflag = 3
            GPSDate$ = MID$(GPSData$, 16, 10)
            GPSTime$ = MID$(GPSData$, 9, 6)
        END IF
        LOOP

    RETURN
getgyrodata:
      gyrodata$ = ""
      gyroflag = 0
      DO WHILE gyroflag < 1
      LINE INPUT #2, gyrodata$
   IF MID$(gyrodata$, 2, 6) = "$AGHDT" THEN gyroflag = 1
      head$ = MID$(gyrodata$, 9, 5)

      LOOP
      RETURN



CheckGPSData:

    IF OldGPSLat$ <> GPSLat$ THEN OldGPSLat$ = GPSLat$ ' GPSLatitude
    IF OldGPSLon$ <> GPSLon$ THEN OldGPSLon$ = GPSLon$ ' GPSLongitude
    IF OldGPSTrack$ <> GPSTrack$ THEN OldGPSTrack$ = GPSTrack$ ' GPSGroundTrack
    IF OldGPSSpeed$ <> GPSSpeed$ THEN OldGPSSpeed$ = GPSSpeed$ '
GPSGroundSpeed

    RETURN
```

```
GetWeatherData:

    FOR idx = 0 TO RawDataCount - 1
        OUT 770, idx
        OUT 769, 0
        rawdata(idx) = INP(768) / 16 + INP(769) * 16 - 2048
        IF rawdata(idx) < 0 THEN rawdata(idx) = 0
    NEXT idx

    bpraw = rawdata(0)
    wsraw = rawdata(1)
    wdraw = rawdata(2)
    srraw = rawdata(3)
    atraw = rawdata(4)
    ahRaw = rawdata(5)

    RETURN

WindDirectionAndSpeed:

    ShipSpeed = VAL(GPSSpeed$)
REM     ShipDirect = VAL(GPSTrack$)
    shipdirect = VAL(head$)
    WindShipSpeed = INT(wsraw / 20.48)
    WindShipDirect = INT(wdraw / 5.69)
    AntiWindShipDirect = WindShipDirect + 180

    GOSUB CalculateWindVector               ' Calculate Wind Vector

    IF TrueWindVectorSum < AvgMemory THEN TrueWindVectorSum =
TrueWindVectorSum + 1
    TrueWindVectorCounter = (TrueWindVectorCounter + 1) MOD AvgMemory
    WindSpeed(TrueWindVectorCounter) = TrueWindSpeed
    WindDirect(TrueWindVectorCounter) = TrueWindDirect
    WindVectorX = 0: WindVectorY = 0
    FOR n = 0 TO TrueWindVectorSum
        TVWRad = WindDirect(n) * (pi / 180)
        WindVectorX = WindVectorX + WindSpeed(n) * COS(TVWRad)
        WindVectorY = WindVectorY + WindSpeed(n) * SIN(TVWRad)
    NEXT n
    WindVectorX = WindVectorX / TrueWindVectorSum: IF WindVectorX = 0 THEN
WindVectorX = .000001
    WindVectorY = WindVectorY / TrueWindVectorSum
    AvgTrueWindSpeed = SQR(WindVectorY * WindVectorY + WindVectorX *
WindVectorX)
    AvgTrueWindDirect = INT((ATN(WindVectorY / WindVectorX)) * (180 / pi) + .5)
    IF WindVectorX < 0 THEN
        AvgTrueWindDirect = AvgTrueWindDirect + 180
```

```
    ELSEIF WindVectorY < 0 THEN
        AvgTrueWindDirect = AvgTrueWindDirect + 360
    END IF
    IF AvgTrueWindSpeed <> oldAvgTrueWindSpeed THEN oldAvgTrueWindSpeed =
AvgTrueWindSpeed
    IF AvgTrueWindDirect <> oldAvgTrueWindDirect THEN oldAvgTrueWindDirect =
AvgTrueWindDirect

    RETURN

AirHumidity:

    AirHumidity = INT(ahRaw / 32) + 32

    RETURN

AirTemperature:

    IF atraw < 164 THEN atraw = 164              'Range is -30 to +50 deg C
    IF atraw > 819 THEN atraw = 819
    AirTemp = INT((atraw - 164) / 8.188) - 30    ' CalculateAirTemperature
    IF AirTemp <> oldAirTemp THEN oldAirTemp = AirTemp

    RETURN

SolarRadiation:

' ----- Calculate Solar Radiation ------

    Solar = INT(srraw / .8673)      ' JOULES / meter / minute
    'Solar = INT(srraw / 1.44)      'if in WATTS      5V = 1422.5 W/M
    IF Solar <> OldSolar THEN OldSolar = Solar

    RETURN

BarometricPressure:

' ----- Calculate Barometric Pressure -----

    Baro = (bpraw / 8.6) + 930      '0V = 930 mb and 2V = 1030 mb
    IF Baro <> oldbaro THEN oldbaro = Baro

    RETURN

CalculateWindVector:

    AntiWindShipDirectRad = AntiWindShipDirect * (pi / 180)
    TrueWindSpeedY = ShipSpeed * SIN(AntiWindShipDirectRad)
    TrueWindSpeedX = WindShipSpeed - ShipSpeed * COS(AntiWindShipDirectRad)
```

```
    TrueWindSpeed = SQR(TrueWindSpeedX * TrueWindSpeedX + TrueWindSpeedY *
TrueWindSpeedY)
    IF TrueWindSpeedX = 0 THEN TrueWindSpeedX = .001
    TrueWindCrossWindShipArc = ATN(TrueWindSpeedY / TrueWindSpeedX) * (180 / pi)
    IF TrueWindSpeedX < 0 THEN
        IF TrueWindSpeedY >= 0 THEN
            TrueWindCrossWindShipArc = TrueWindCrossWindShipArc + 180
        ELSE
            TrueWindCrossWindShipArc = TrueWindCrossWindShipArc - 180
        END IF
    END IF
    TrueWindDirect = (shipdirect + (WindShipDirect - TrueWindCrossWindShipArc)) MOD
360

    RETURN

CheckInkey:

    K$ = INKEY$
    IF K$ = "" THEN
        RETURN
    ELSEIF ASC(K$) = 27 THEN
        GOSUB ExitSystem
    END IF

    RETURN

RecordData:

    GOSUB OutputRawData

    IF RecordTimeChoice = 0 THEN
    ELSEIF RecordTimeTmp$ <> MID$(TimeTmpS, 6 - RecordTimeChoice, 1) THEN
        RecordTimeTmp$ = MID$(TimeTmp$, 6 - RecordTimeChoice, 1)
    ELSE
        RETURN
    END IF

    GOSUB OutputData

    RETURN

DisplayData:

    IF DisplayTimeChoice = 0 THEN
    ELSEIF DisplayTimeTmp$ <> MID$(TimeTmpS, 6 - DisplayTimeChoice, 1) THEN
        DisplayTimeTmp$ = MID$(TimeTmp$, 6 - DisplayTimeChoice, 1)
    ELSE
        RETURN
```

```
    END IF
    DisplayTime$ = TimeTmp$

    GOSUB ShowData
    GOSUB ShowWindVectorDiagram

    RETURN

OutputData:

    IF DATE$ <> OldDate$ THEN
       IF OldDate$ <> " " THEN
          FOR OFNo = 3 TO 4
REM          GOSUB OutputDataTrailer
             CLOSE #OFNo
          NEXT
       END IF
       OldDate$ = DATE$
       RecordMonth$ = MID$(OldDate$, 9, 2) + MID$(OldDate$, 1, 2)
       recorddate$ = RecordMonth$ + MID$(OldDate$, 4, 2)
REM       filename$(2) = "C:\Marine\DATA\" + RecordMonth$ + "\" + RecordDate$ +
".TXT"
REM       filename$(3) = "B:\" + RecordDate$ + ".TXT"
       filename$(3) = "C:\Marine\DATA\" + RecordMonth$ + "\" + recorddate$ + ".DAT"
       filename$(4) = "B:\" + recorddate$ + ".DAT"
         FOR OFNo = 3 TO 4
          OPEN filename$(OFNo) FOR APPEND AS #OFNo
REM       GOSUB OutputDataHeader
       NEXT
    END IF

    PRINT #3, recorddate$; USING "& ##.# ##.# ## ####.# ###.## ####.#
& & ###.# ##.# ###.# ##.#"; DisplayTime$; AvgTrueWindDirect; AvgTrueWindSpeed;
AirTemp; Baro; AirHumidity; Solar; GPSLat$; GPSLon$; shipdirect; ShipSpeed;
WindShipDirect; WindShipSpeed
    IF floppy$ = "T" THEN
       PRINT #4, recorddate$; USING "& ###.# ##.# ## ####.# ###.##
####.# & & ###.# ##.# ###.# ##.#"; DisplayTime$; AvgTrueWindDirect;
AvgTrueWindSpeed; AirTemp; Baro; AirHumidity; Solar; GPSLat$; GPSLon$; shipdirect;
ShipSpeed; WindShipDirect; WindShipSpeed
    END IF

    RETURN

OutputDataHeader:

    PRINT #OFNo, " Time    WDir WSpd   Temp    Baro    Humidity   Solar    Latitude
Longitude     SDir SSpd   RDir RSpd"
    PRINT #OFNo, "======  ==== ====   ====   ======  ========   =====
```

```
========== ==========  ==== ====  ==== ===="

    RETURN

OutputDataTrailer:

    PRINT #OFNo, "=======  ==== ====  ====  =====  ====  ==========
==========  ==== ====  ==== ===="
    PRINT #OFNo, ""
    PRINT #OFNo, "   Temp   : Temperature (" + CHR$(248) + "C)"
    PRINT #OFNo, "   Solar : Solar Radiation (Joules/meter/minute)"
    PRINT #OFNo, "   Baro   : Barometric Pressure (mb)"
    PRINT #OFNo, "   TDir   : True Wind Direction (" + CHR$(248) + ")"
    PRINT #OFNo, "   TSpd   : True Wind Speed (KNots)"
    PRINT #OFNo, "   SDir   : Ship Direction (" + CHR$(248) + ")"
    PRINT #OFNo, "   SSpd   : Ship Speed (KNots)"
    PRINT #OFNo, "   RDir   : Wind - Ship Resultant Direction (" + CHR$(248) + ")"
    PRINT #OFNo, "   RSpd   : Wind - Ship Resultant Speed (KNots)"

    RETURN

OutputRawData:

REM filename$(5) = "C:\Marine\DATA\Today.RAW"
REM     OPEN filename$(5) FOR OUTPUT AS #5
REM     RecordRawData$ = ""
     IF RecordTimeTmp$ <> MID$(TIMES, 5 - ((RecordTimeChoice + 3) MOD 4), 1) THEN
        IF RecordTimeTmp$ = MID$(TIMES, 2, 1) THEN
            RecordRawData$ = LEFT$(TIMES, 2)
        END IF
        RecordRawData$ = RecordRawData$ + MID$(TIME$, 4, 2)
        RecordRawData$ = RecordRawData$ + " " + GPSLat$ + " " + GPSLon$
        RecordRawData$ = RecordRawData$ + " " + GPSSpeed$ + " " + GPSTrack$
    END IF
REM     PRINT #5, RecordRawData$
    FOR rawidx = 1 TO 2
        RecordRawData$ = RecordRawData$ + LTRIM$(STR$(rawdata(rawidx))) + " "
        FOR n = 0 TO TrueWindVectorSum
            IF rawidx = 1 THEN
                RecordRawData$ = RecordRawData$ + LTRIM$(STR$(WindSpeed(n))) + " "
            ELSEIF rawidx = 2 THEN
                RecordRawData$ = RecordRawData$ + LTRIM$(STR$(WindDirect(n))) + " "
            END IF
        NEXT n
REM         PRINT #5, RecordRawData$
    NEXT
REM     CLOSE #5

    RETURN
```

```
ShowData:

    LOCATE 6, 1: PRINT USING "&  ###.#  ##.#   ##    ####.#    ###.##     ####
& &"; DisplayTime$; AvgTrueWindDirect; AvgTrueWindSpeed; AirTemp; Baro; AirHumidity;
Solar; GPSLat$; GPSLon$
REM    LOCATE 8, 1: PRINT USING "&  ###  ##.#   ##     ####  ####  & &";
DisplayTime$; AvgTrueWindDirect; AvgTrueWindSpeed; AirTemp; Solar; Baro; GPSLat$;
GPSLon$
    LOCATE 10, 9: PRINT USING "###.#   ##.#"; shipdirect; ShipSpeed;
    LOCATE 14, 9: PRINT USING "###.#   ##.#"; WindShipDirect; WindShipSpeed
REM    LOCATE 11, 25: PRINT USING "   ###.## ####.##"; AirHumidity; ahRaw
REM    LOCATE 11, 25: PRINT USING "   ###.##"; AirHumidity

    RETURN

ShowWindVectorDiagram:

    MaxLength = -1
    ScaleDraw = 1
    AngleShip = 90 - shipdirect
    AngleWind = 90 - (shipdirect + WindShipDirect)
    xsnew = ShipSpeed * COS(AngleShip * pi / 180)
    ysnew = ShipSpeed * SIN(AngleShip * pi / 180)
    xwnew = WindShipSpeed * COS(AngleWind * pi / 180)
    ywnew = WindShipSpeed * SIN(AngleWind * pi / 180)
    IF ShipSpeed > MaxLength THEN MaxLength = ShipSpeed
    IF WindShipSpeed > MaxLength THEN MaxLength = WindShipSpeed
    IF ABS(MaxLength > .1) THEN ScaleDraw = 50 / MaxLength * .8 ELSE ScaleDraw = 0
    LINE (0, 0)-(xsold * ScaleDrawold, ysold * ScaleDrawold), 0
    LINE (0, 0)-(xwold * ScaleDrawold, ywold * ScaleDrawold), 0
    LINE (xwold * ScaleDrawold, ywold * ScaleDrawold)-(xsold * ScaleDrawold, ysold *
ScaleDrawold), 0
    LINE (0, 0)-(xsnew * ScaleDraw, ysnew * ScaleDraw), 3
    LINE (0, 0)-(xwnew * ScaleDraw, ywnew * ScaleDraw), 4
    LINE (xwnew * ScaleDraw, ywnew * ScaleDraw)-(xsnew * ScaleDraw, ysnew *
ScaleDraw), 2
    xwold = xwnew
    ywold = ywnew
    xsold = xsnew
    ysold = ysnew
    ScaleDrawold = ScaleDraw

    RETURN

ErrorHandler:

    RESUME NEXT
```