

**System Installation Report**

GBON AWS Stations

|  |  |
| --- | --- |
| Project Reference: | Job6685 |
| Project Name: | UNDP GBON SOFF 3 x Research Grade Stations Rwanda |
| Location: | Rwanda |

|  |  |  |
| --- | --- | --- |
| Rwanda Meteorology Agency (METEO RWANDA) |  | Campbell Scientific Africa Pty. (Ltd) |
| 23V5+Q33, KN 96 St,  Rwanda Kigali  Iyakaremye Vedaste (PhD)  Data Observations, Quality Control and Processing Division Manager  [v.iyakaremye@meteorwanda.gov.rw](mailto:v.iyakaremye@meteorwanda.gov.rw)  Cell (+250) 783622984 |  | 1A Meson Street, Technopark, Stellenbosch, Western Cape, 7600, South Africa  [sales@campbellsci.co.za](mailto:sales@campbellsci.co.za)  support@campbellsci.co.za  +27 21 880 9960  [www.campbellsci.co.za](http://www.campbellsci.co.za) |

i. General Notices & Document information

a. Acknowledgements and Trademarks

Campbell Scientific acknowledges the proprietary rights in the trademark and product names of other companies mentioned in this document.

b. Copyright Notice

2023 Campbell Scientific©

The information contained herein is the property of Campbell Scientific and must not be reproduced (wholly or in part) used or disclosed without the prior written permission of the copyright owner and then on condition that this notice is included in any reproduction or disclosure. The copyright and the foregoing restriction on copying, use and disclosure extend to all media in which this information may be embodied including magnetic storage.

c. Security Notice

This document is commercial-in-confidence and is to be modified by Campbell Scientific only. The contents of this document are to be considered confidential and are not to be distributed to persons not directly involved with the end user that the document is prepared for.

d. Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revisions | Brief Change History | Prepared by | Approved by | Date |
| 00 | Original | S.C Boyi | J. van Jaarsveldt | 2025-06-01 |
|  |  |  |  |  |

ii. Warranty

Limited warranty

Products sold by Campbell Scientific Africa are warranted to be free from defects in materials and workmanship under normal use and service for a period of 12 months from the date of shipment as indicated on your invoice. Some products may offer the option for an extended warranty, the details of which will be specified on the invoice. Please note that extended third-party warranties may have additional conditions outlined in the documentation provided with the product.

This warranty does not cover damage resulting from normal wear and tear, surges, or from failure to properly maintain or calibrate the product as recommended. Consumable items and items with limited life expectancy, such as batteries, mechanical relays, fine-wire thermocouples, and desiccant, are only warranted to perform to specification at the time of delivery.

Campbell Scientific Africa's obligation under this warranty is limited to repairing or replacing defective products, at our discretion. This repair or replacement is the sole and exclusive remedy provided under this warranty.

The customer is responsible for all costs associated with removing, reinstalling, and shipping defective products to Campbell Scientific Africa. However, Campbell Scientific Africa will cover the cost of carriage (excluding any local Customs or clearance fees) for all repairs or replacements covered under warranty.

Under ex-works terms, the customer assumes responsibility for any expenses related to work performed or items provided beyond the warranty coverage, subject to fair and reasonable assessment. Additionally, the customer is responsible for shipping costs to our location and our expenses for returning the repaired or replacement goods to them.

|  |  |
| --- | --- |
| Warranty start date: | 01 June 2025 (From date of installation) |
| Warranty end date: | 01 June 2026 |

iii. Support, Repairs, Returns & Calibrations

Products may not be returned without prior authorization. Products shipped to Campbell Scientific require a Returned Materials Authorization (RMA) or Repair Reference number and must be clean and uncontaminated by harmful substances, such as hazardous materials, chemicals, insects, and pests. Please complete the required forms prior to shipping equipment.

To obtain a Returned Materials Authorization or Repair Reference number, send an email to Campbell Scientific Africa. Please write the issued number clearly on the outside of the shipping container and ship as directed. For all returns, the customer must provide a “Statement of Product Cleanliness and Decontamination” or “Declaration of Hazardous Material and Decontamination” form and comply with the requirements specified in it.

Campbell Scientific Africa is unable to process any returns until we receive this statement. If the statement is not received within three days of product receipt or is incomplete, the product will be returned to the customer at the customer’s expense. Campbell Scientific Africa reserves the right to refuse service on products that were exposed to contaminants that may cause health or safety concerns for our employees.

Technical Support

For any assistance, please reach out to our technical support team using one of the following communication channels.

Email (Primary channel): support@campbellsci.co.za

WhatsApp: +27 79 528 8014

Telephonically: +27 21 880 9960

Operating hours Refer to business hours on our website.

Response time 2-3 working days.

Our support services are unavailable beyond regular operating hours, weekends, or public holidays. In anticipation of an upcoming site visit, we kindly request advance notification (2-3 business days). Failure to provide prior notice may impact the assurance of technical support availability.

Contents

[i. General Notices & Document information 2](#_Toc202944151)

[ii. Warranty 3](#_Toc202944152)

[iii. Support, Repairs, Returns & Calibrations 4](#_Toc202944153)

[1. Introduction 6](#_Toc202944154)

[2. Customer & contractor details 6](#_Toc202944155)

[3. Site details 6](#_Toc202944156)

[3.1 Site Area Map 8](#_Toc202944157)

[4. System elements 10](#_Toc202944158)

[4.1 Sensor details 10](#_Toc202944159)

[4.2 Power 11](#_Toc202944160)

[4.3 Communication 12](#_Toc202944161)

[5. Installation photographs 14](#_Toc202944162)

[6. Wis2 Environment 21](#_Toc202944163)

[Appendix A: Wiring diagram 22](#_Toc202944164)

[Appendix B: Calibration certificates 25](#_Toc202944165)

[Appendix C: Datalogger Program 34](#_Toc202944166)

1. Introduction

This System Installation Report (SIR) serves as a comprehensive document post-commissioning, providing crucial details about the site's location, characteristics, and technical specifications, alongside calibration certificates and relevant information essential for data users to grasp the site's intricacies and guarantee the reliability of measurement data. It includes visual documentation with photographs depicting the site and its surrounding terrain.

2. Customer & contractor details

|  |  |
| --- | --- |
| Customer | Contact details |
| Rwanda Meteorology Agency (METEO RWANDA) | Mr Iyakaremye Vedaste (PhD)  Data Observations, Quality Control and Processing Division Manager  [v.iyakaremye@meteorwanda.gov.rw](mailto:v.iyakaremye@meteorwanda.gov.rw)  Cell (+250) 783622984 |

|  |  |
| --- | --- |
| Contractor(s) | Scope of work |
| Campbell Scientific Africa Pty (Ltd)  1A Meson Street, Technopark, Stellenbosch, South Africa  PO Box 2450, Somerset West, 7129  Tel: +27 (0)21 880 9960  Fax: + 27(0)21 880 0240  Email: sales@campbellsci.co.za | Supply & Installation of 3x GBON/SOFF Automatic Weather Stations. |

3. Site details

|  |  |
| --- | --- |
| System Name: | Kazo AWS |
| Installation date: | 27 May 2025 |
| Decommissioning date: | - |
| Location coordinates: | -2.18346°, 30.5147° |
| Elevation (AMSL): | 1664 m |
| Time Zone (UTC): | +2:00 |
| Contact person for site access: | Mr Desiree Ntirengaya  Position: Instruments Maintenance and Calibration Officer  Email: [d.ntirengaya@meteorwanda.gov.rw](mailto:d.ntirengaya@meteorwanda.gov.rw)  Cell (+250) 783 543 995 |

|  |  |
| --- | --- |
| System Name: | Nyagatare AWS |
| Installation date: | 28 May 2025 |
| Decommissioning date: | - |
| Location coordinates: | -1.29466°, 30.3315° |
| Elevation (AMSL): | 1394 m |
| Time Zone (UTC): | +2:00 |
| Contact person for site access: | Mr Desiree Ntirengaya  Position: Instruments Maintenance and Calibration Officer  Email: [d.ntirengaya@meteorwanda.gov.rw](mailto:d.ntirengaya@meteorwanda.gov.rw)  Cell (+250) 783 543 995 |

|  |  |
| --- | --- |
| System Name: | Rubengera AWS |
| Installation date: | 27 May 2025 |
| Decommissioning date: | - |
| Location coordinates: | -2.06726°, 30.4105° |
| Elevation (AMSL): | 1591 m |
| Time Zone (UTC): | +2:00 |
| Contact person for site access: | Mr Desiree Ntirengaya  Position: Instruments Maintenance and Calibration Officer  Email: [d.ntirengaya@meteorwanda.gov.rw](mailto:d.ntirengaya@meteorwanda.gov.rw)  Cell (+250) 783 543 995 |

3.1 Site Area Map

The Site Area Map provides a visual representation of the geographical layout and spatial organization of the installation location. This detailed depiction outlines key features such as buildings, infrastructure, access points, and surrounding landmarks.

|  |
| --- |
|  |
| Figure 1: Ngoma Kazo Office (Google Earth) |
|  |

|  |
| --- |
|  |
| Figure 2: Nyagatare Office (Google Earth) |
|  | |
| Figure 3: Rubengera Karongi (Google Earth) | |

|  |
| --- |
|  |

4. System elements

4.1 Sensor details

**Table 1**: Ngoma

|  |  |  |  |
| --- | --- | --- | --- |
| **Equipment (Measurement Parameter)** | **Manufacturer & Model** | **Serial Number** | **Parameter Variable** |
| Datalogger | Campbell Scientific  CR1000XE | 70620 | - |
| GPS | Garmin GPS16XHVS | 1A4325247 | GPS\_Latitude  GPS\_Longitude  GPS\_Elevation |
| Precipitation | RM Young 52203 – Unheated Rain Gauge | 19546 | Rain |
| Temperature | Campbell Scientific  TempVUE10 | 1422 | AirTemp |
| Relative Humidity | EE181 | 24016000352CA | RH |
| Barometric Pressure | Campbell Scientific  BaroVUE10 | 3798 | BPress |
| Wind Direction | Thies 4.3151.10.020 | 7210492 | WDir |
| Wind Speed | Thies 4.3352.10.000 | 1124668 | WSpd |

**Table 2**: Nyagatare

|  |  |  |  |
| --- | --- | --- | --- |
| **Equipment (Measurement parameter)** | **Manufacturer & Model** | **Serial Number** | **Parameter Variable** |
| Datalogger | Campbell Scientific  CR1000Xe | 70621 | - |
| GPS | Garmin GPS16XHVS | 1A4328219 | GPS\_Latitude  GPS\_Longitude  GPS\_Elevation |
| Precipitation | RM Young 52203 – Unheated Rain Gauge | 19548 | Rain |
| Temperature | Campbell Scientific  TempVue10 | 1424 | AirTemp |
| Relative Humidity | EE181 | 245016000359EA | RH |
| Barometric Pressure | Campbell Scientific  BaroVue10 | 3804 | BPress |
| Wind Direction | Thies 4.3151.10.020 | 72104267 | WDir |
| Wind Speed | Thies 4.3352.10.000 | 11242671 | WSpd |

**Table 3**: Karongi

|  |  |  |  |
| --- | --- | --- | --- |
| **Equipment (Measurement parameter)** | **Manufacturer & model** | **Serial number** | **Parameter variable** |
| Datalogger | Campbell Scientific  CR1000Xe | 70623 | - |
| GPS | Germin GPS16XHVS | 1A4329400 | GPS\_Latitude  GPS\_Longitude  GPS\_Elavation |
| Precipitation | RM Young 52203 – Unheated Rain Gauge | 19540 | Rain |
| Temperature | Campbell Scientific  TempVue10 | 1482 | AirTemp |
| Relative Humidity | EE181 | 2450160003458D | RH |
| Barometric Pressure | Campbell Scientific  BaroVue10 | 3824 | BPress |
| Wind Direction | Thies 4.3151.10.020 | 8210551 | WDir |
| Wind Speed | Thies 4.3352.10.000 | 11242673 | WSpd |

4.2 Power

|  |  |
| --- | --- |
| **Power supply type** | 12V DC |
| **Battery** | 1 x 12 V (33Ah) |
| **PV module** | 50W |
| **Charge controller** | CS MPPT Regulator |

4.3 Communication

|  |  |  |
| --- | --- | --- |
| **Station Name** | **Kazo** | |
| **GSM Modem Model** | Campbell Scientific Cell215  SN: 15120 | |
| **Communication protocol** | GSM - TDMA | |
| **Communication**  **Antenna** | ANT 7DB -7 dB Gain Dual Band Magnetic Base | |
| **Pakbus** | 118 | |
| **Connection URL/IP** | callback.csafrica.co.za | |
| **Port** | 11005 | |
| **Neighbour** | 4005 | |
| **Security codes** | Security Code 1: 6201  Security Code 2: 6202  Security Code 3: 6203 | |
| **TCP password** | Csaf110054005# | |
| **Pakbus encryption key** | Cr1000Xe70620# | |
| **Network details** | Service Provider:  Cell Number | MTN  +250796664768 |

|  |  |  |
| --- | --- | --- |
| **Station Name** | **Nyagatare** | |
| **GSM Modem Model** | Campbell Scientific Cell215  SN: | |
| **Communication protocol** | GSM - TDMA | |
| **Communication**  **Antenna** | ANT 7DB -7 dB Gain Dual Band Magnetic Base | |
| **Pakbus** | 119 | |
| **Connection URL/IP** | Callback.csafrica.co.za | |
| **Port** | 11005 | |
| **Neighbour** | 4005 | |
| **Security codes** | Security Code 1: 6211  Security Code 2: 6212  Security Code 3: 6213 | |
| **TCP password** | Csaf110054005# | |
| **Pakbus encryption key** | Cr1000Xe70620# | |
| **Network details** | Service Provider:  Cell Number | MTN  - |

|  |  |  |
| --- | --- | --- |
| **Station Name** | Karongi | |
| **GSM Modem Model** | Campbell Scientific Cell215  SN: | |
| **Communication protocol** | GSM - TDMA | |
| **Communication**  **Antenna** | ANT 7DB -7 dB Gain Dual Band Magnetic Base | |
| **Pakbus** | 120 | |
| **Connection URL/IP** | Callback.csafrica.co.za | |
| **Port** | 11005 | |
| **Neighbour** | 4005 | |
| **Security codes** | Security Code 1: 6231  Security Code 2: 6232  Security 3: 6233 | |
| **TCP password** | Csaf110054005# | |
| **Pakbus encryption key** | Cr3502978# | |
| **Network details** | Service Provider:  Cell Number | MTN  +250796491487 |

5. Installation photographs

This section provides an overview of the installation, including the main station components of the system. Additionally, detailed records of the instrument manufacturer, model, and serial numbers are provided, along with relevant information regarding installation levels and orientation when applicable. This section aims to ensure a thorough understanding of the equipment and its positioning within the measurement environment.

|  |  |
| --- | --- |
|  |  |
|  |  |
| A white box on a metal pole  AI-generated content may be incorrect. |  |
| A group of metal poles in a grassy area  AI-generated content may be incorrect. |  |

Figure 5: Installed System – Kazo AWS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

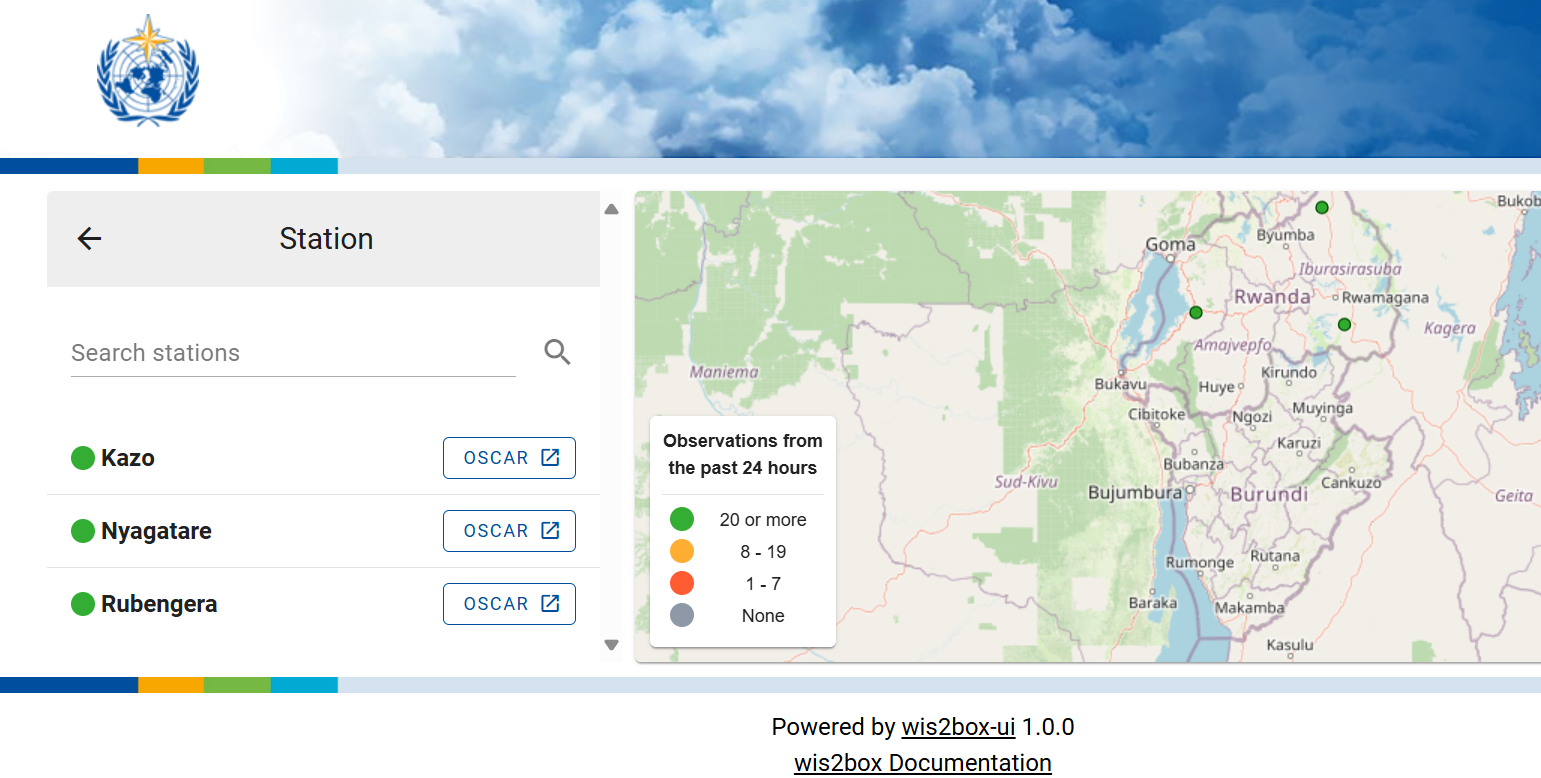
Figure 6: Installed System – Nyangatare AWS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

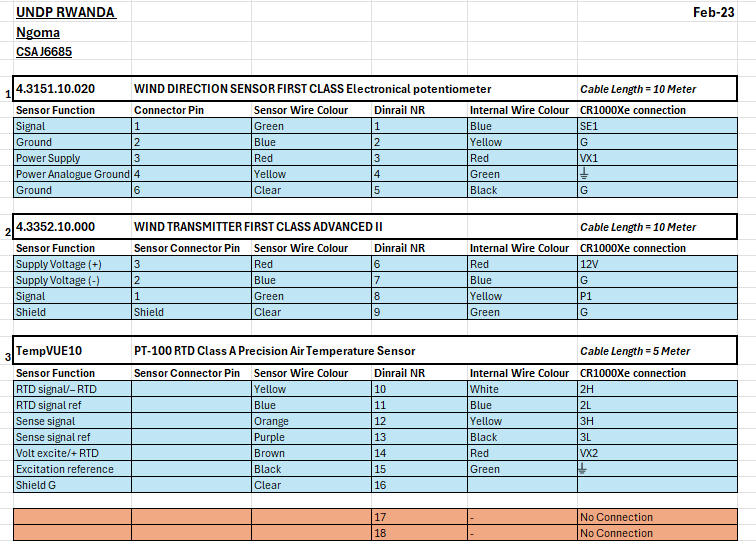
Figure 7: Installed System – Karongi AWS

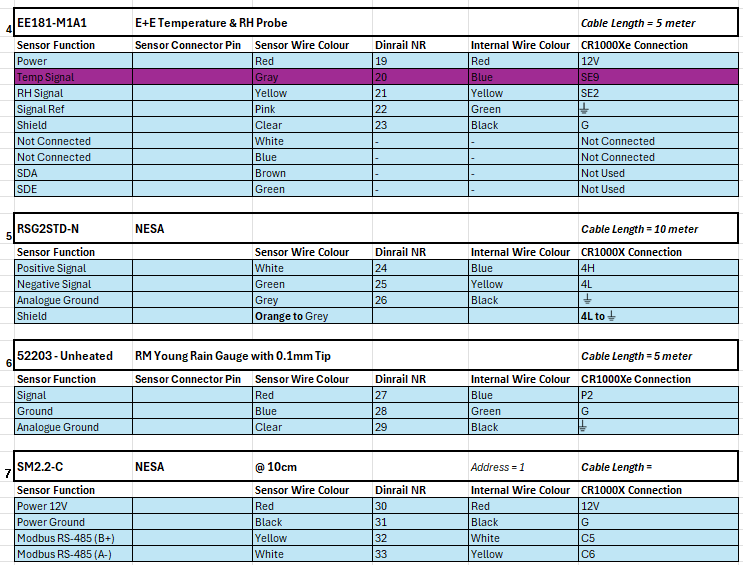
6. Wis2 Environment

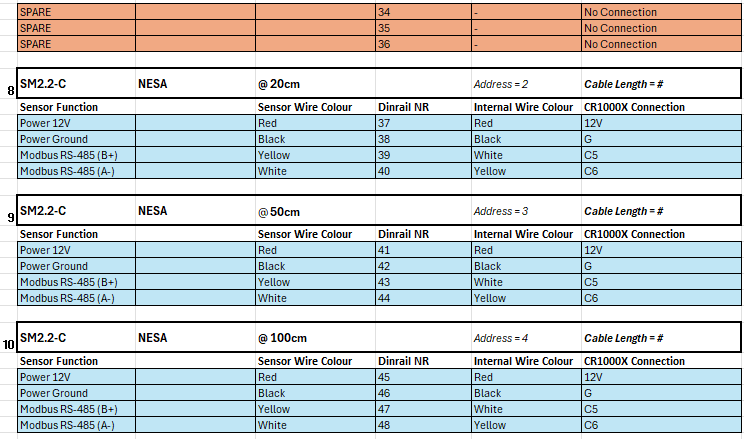
The stations have been successfully installed and configured to transmit data via telemetry to the WIS2 environment, as well as to the on-site servers at HQ. The figure below illustrates the status of the stations within the WMO WIS2 environment.



Appendix A: Wiring diagram







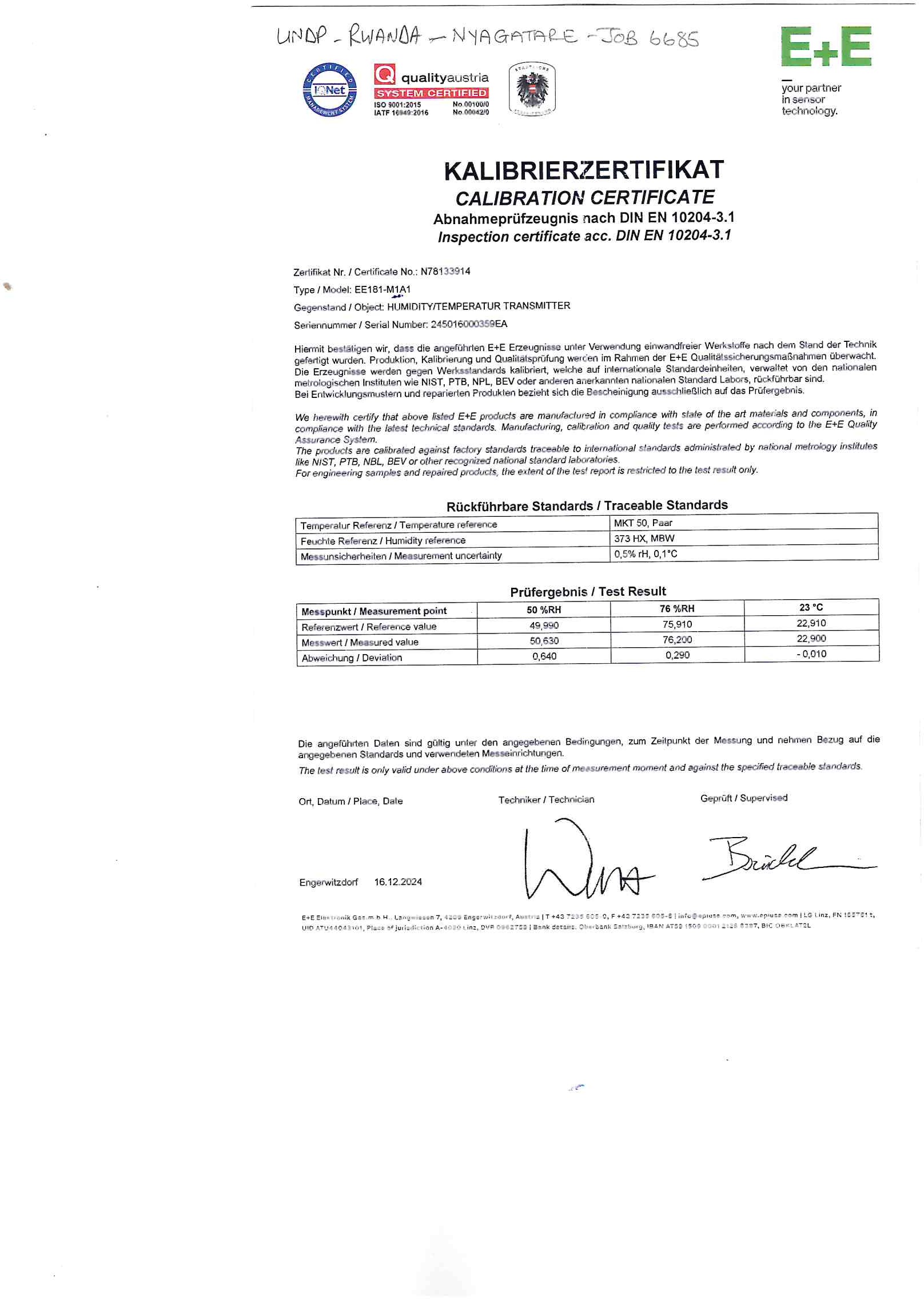
A blue and white chart with black text

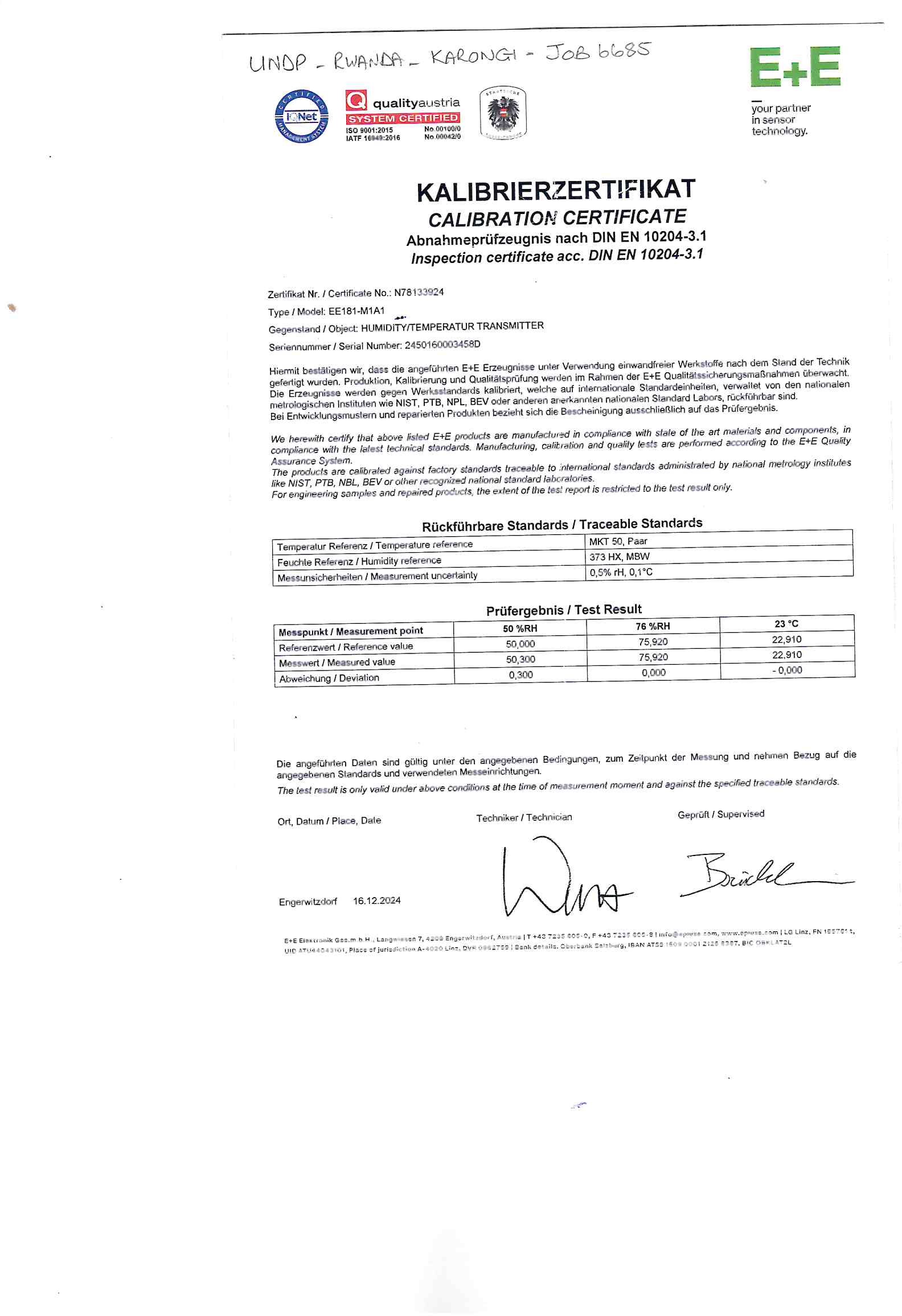
AI-generated content may be incorrect.

A table with blue and white lines

AI-generated content may be incorrect.

Appendix B: Calibration certificates





A close-up of a document

AI-generated content may be incorrect.A close-up of a certificate

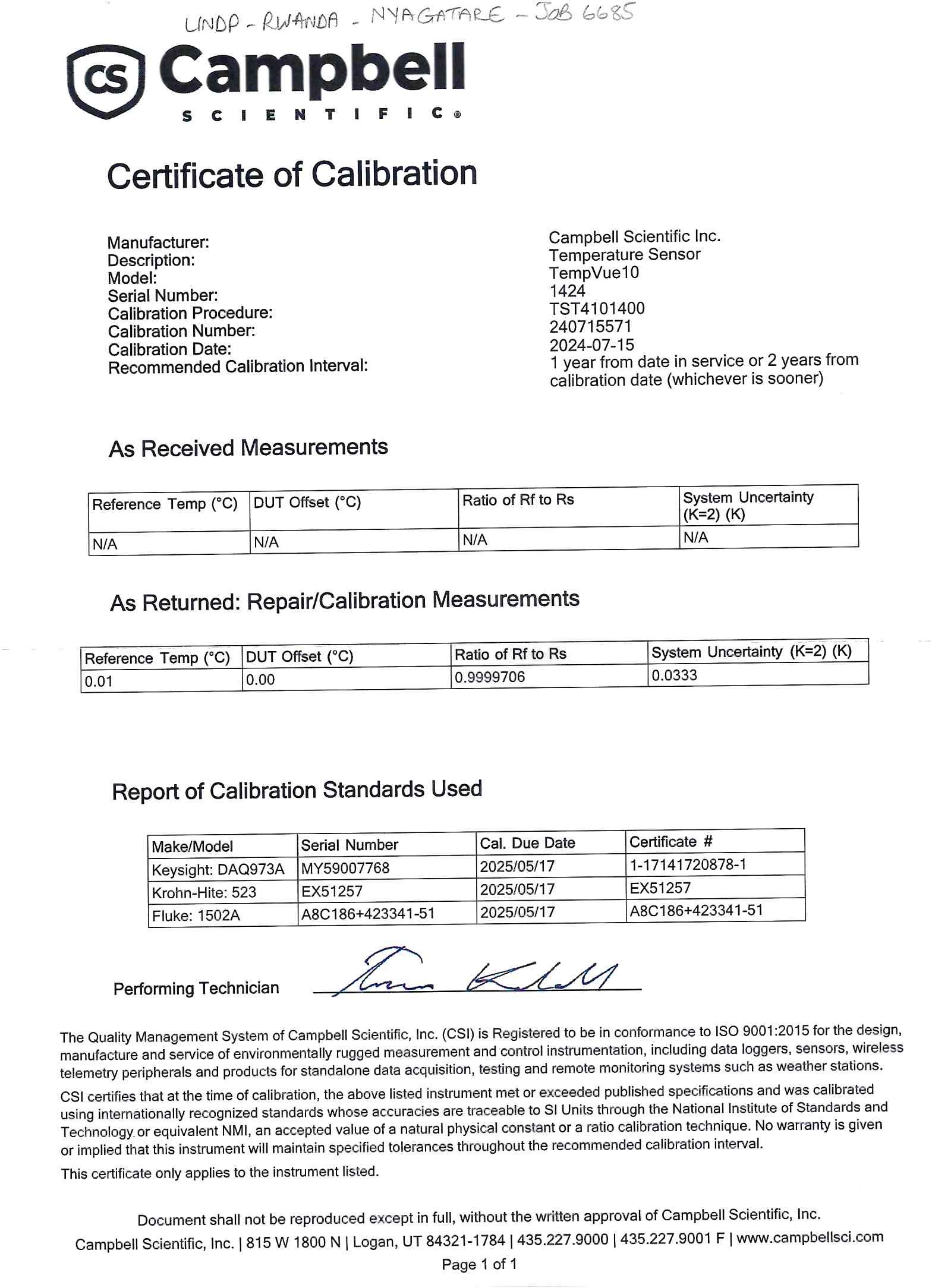
AI-generated content may be incorrect.

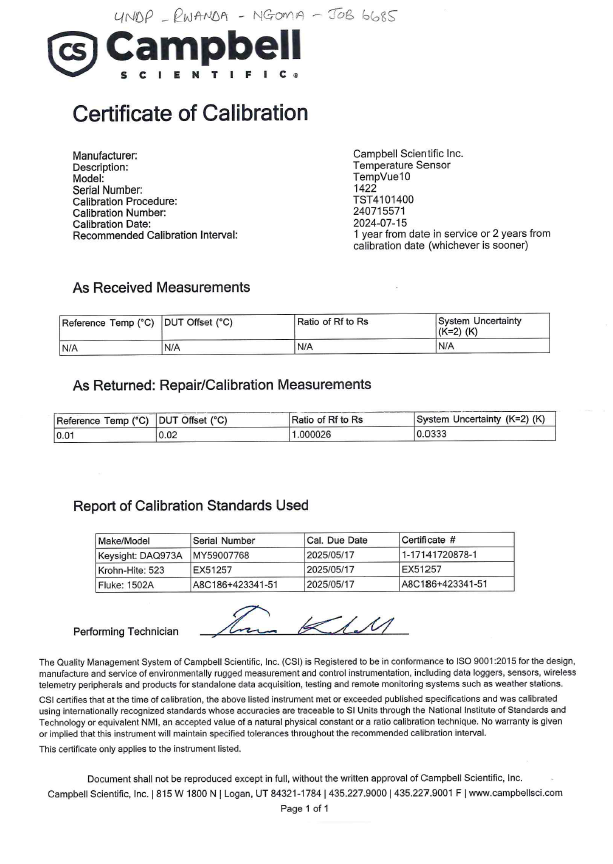
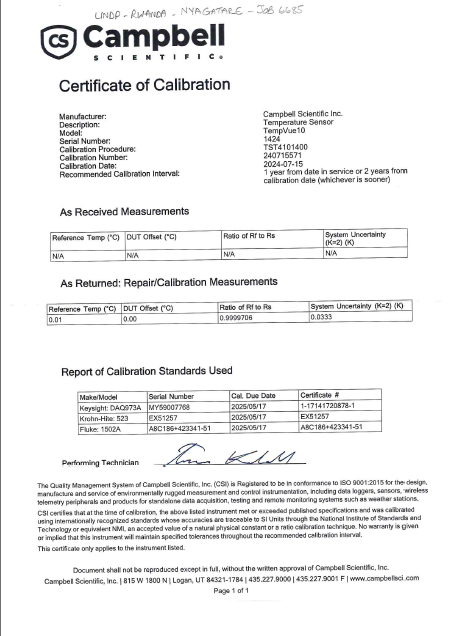
A close-up of a certificate

AI-generated content may be incorrect.

A close-up of a certificate

AI-generated content may be incorrect.





Appendix C: Datalogger Program

*The program is up-to-date at the time of installation; however, it is advisable to download the latest version as updates may have been performed after the installation.*

'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

'\* Generic program \*

'\* 1. User settings must be entered in the public table - readings will be invalid until this has been done. \*

'\* 2. Values must then be stored on the CPU/USR drive by setting the various StoreSettingsFlags to 1 \*

'\* This will ensure that settings will be restored after program restarts or program uploads \*

'\* 3. Therefore protect the CPU / USR drives by backing up the entire logger before upgrading the OS \*

'\* Also do not change the size of the USR drive which will format the USR drive \*

'\* 4. Variables included in the tables are aliased to generic names - This is to keep table field names unchanged \*

'\* when sensors are replaced with a different type. \*

'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

'--------------------------------------------------------------------------------------------------

' Client: UNDP RWANDA

' StationID: Set in public table

' CSA Job: 6685

'--------------------------------------------------------------------------------------------------

' Description: Campbell Scientific CR1000Xe datalogger

' Thies 4.3351.10.000 Anemometer | Thies Sensor: \_WSpd GBON QC: \_WSpd2M \_WSpd2MGust \_WSpd10M \_WSpd10MGust | P1 |

' Thies 4.3151.10.020 Wind Vane | Thies Sensor: \_WDir GBON QC: \_WDir2m \_WDir10m | SE1(ActivePot) VX1(SWVX Instruction) |

' TV10 AirTemp | TV10 Sensor: \_AirTemp GBON QC: \_AirTemp1M \_AirTempMin24H \_AirTempMax24H | 2L2H3L3H VX2 |

' EE181 RH | EE181 Sensor: \_RH GBON QC: \_RH1M | SE2 |

' Calculated DewPointTemp | TV10 Sensor: \_AirTemp EE181 Sensor: \_RH GBON QC: \_DewPointTemp10M | Calculated value |

' R.M.Young 52203 0.1mm Rain gauge | RMY52203 Sensor: \_Rain GBON QC: \_RainIntensity1M \_Rain1M \_Rain1H \_Rain3H, Rain24H | P2 |

' Campbell BaroVUE10 Barometric Pressure | BV10 Sensor: \_BPress \_BPTemp \_QFE \_QNH \_QNH\_ICAO \_QFF GBON QC: \_BPress1M \_QNH1M \_BPressChange3H \_BPressTendency3H | C3 SDI12 Address 0 |

' NESA RSG2STDN Pyranometer | RSG2STDN Sensor: \_SlrW \_SlrMJ GBON QC \_SlrJ1H \_SlrJ24H | 4H4L |

' NESA SM2C Volumentric Water Content | SM2C\_VWC\_1 \_EC\_1 \_SoilTemp\_1 | C5C6 Address 1 |

' NESA SM2C Volumentric Water Content | SM2C\_VWC\_2 \_EC\_2 \_SoilTemp\_2 | C5C6 Address 2 |

' NESA SM2C Volumentric Water Content | SM2C\_VWC\_3 \_EC\_3 \_SoilTemp\_3 | C5C6 Address 3 |

' NESA SM2C Volumentric Water Content | SM2C\_VWC\_4 \_EC\_4 \_SoilTemp\_4 | C5C6 Address 4 |

' Garmin GPS16XHVS GPS | GPS16XHVS \_GPS | C1C2 |

' Calculated Solar Position | Sun \_Azimuth \_ Elevation \_Up | --- |

' Calculated Sunshine Hours | Sun\_Hours GBON QC \_SunHours1H \_SunHours24H | --- |

' Power Supply: ##### AC Charge Controller

' Campbell MPPT4A Solar Charge Controller | MPPT4A\_VSolar \_VBat \_Vout \_ISolar \_Iout \_ChargerState \_ChargerTemp | C7 SDI12 Address 0 |

' 50 Watt Solar Panel

' 12V/33Ah SLA Battery

' Comms: Cell215 modem (CSIO port) setup to do callback

' MQTT of SYNOP table conforming to GBON BUFR Template 307096 - BUFR template for synoptic reports from fixed land stations suitable for SYNOP Data

'

'--------------------------------------------------------------------------------------------------

' Note all changes made during installation/maintenance and save as new file with revision and date

' Version: 1 Revision: 0 Date: 20250329 Author: Nkosi and FlR CSAf Description: Integration and lab test

' Version: # Revision: # Date: YYYYMMDD Author: ### #### Description: #############################

' Version: # Revision: # Date: YYYYMMDD Author: ### #### Description: #############################

' Version: # Revision: # Date: YYYYMMDD Author: ### #### Description: #############################

'--------------------------------------------------------------------------------------------------

'-Constants------------------------------------------------------------------

AngleDegrees

'-Declare JobID Variables and Units------------------------------------------------------------

Public JobID\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*"

Public JobID As String \* 30 = "UNDP\_Rwanda\_AWS\_6685" : ReadOnly JobID 'ID given by Campbell Scientific Africa - Do not change

'-Declare StationID Variables and Units--------------------------------------------------------

Public StationID\_Block As String = "\*\*\*\*\*\*\*\*\*\*"

Public StationID As String \* 30 ' ID given by the client - Set in public table

' Persistence

Public StationID\_WriteFlag As Boolean

Dim StationID\_WriteTime\_Float

Public StationID\_WriteTime As String \* 25

' Sub: Reading station Identification

Sub StationID\_Read

Dim Closefile, Openfile As Long

'StationID

Openfile = FileOpen("CPU:StationID.txt","r",-1)

FileReadLine(Openfile,StationID,25)

Closefile=FileClose(Openfile)

'Saved time

Calfile(StationID\_WriteTime\_Float,1,"CPU:StationID\_WriteTime.cal",1)

StationID\_WriteTime = SecsSince1990(StationID\_WriteTime\_Float,4)

EndSub

' Sub: Writing of station Identification

Sub StationID\_Write

Dim Closefile, Openfile As Long

'SN

Openfile = FileOpen ("CPU:StationID.txt","w",-1)

FileWrite (Openfile,StationID,0)

Closefile=FileClose (Openfile)

'Probe settings time

StationID\_WriteTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(StationID\_WriteTime\_Float,1,"CPU:StationID\_WriteTime.cal",0)

EndSub

'-Declare Station Coordinates Variables and Units-----------------------------------------

Public StationCoordinates As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*"

Public Latitude = NAN : Units Latitude = DecDeg 'Decimal degrees (Positive if north of the equator, negative if south of the equator)

Public Longitude = NAN : Units Longitude = DecDeg 'Decimal degrees (Positive if east of greenwich, negative if west of greenwich)

Public Elevation = NAN : Units Elevation = M 'Metres above sea level

Public UTC\_Offset = NAN : Units UTC\_Offset = Hours 'GMT + # Hours

' Persistence

Public Location\_WriteFlag As Boolean ' write flag

Dim Location\_WriteTime\_Float

Public Location\_WriteTime As String \* 25 ' write time

' Sub:Read location

Sub Location\_Read

Calfile(Latitude,1,"CPU:Latitude.cal",1) 'Latitude

Calfile(Longitude,1,"CPU:Longitude.cal",1) 'Longitude

Calfile(Elevation,1,"CPU:Elevation.cal",1) 'Elevation

Calfile(UTC\_Offset,1,"CPU:UTC\_Offset.cal",1) 'UTC\_Offset

Calfile(Location\_WriteTime\_Float,1,"CPU:Location\_WriteTime.cal",1)

Location\_WriteTime = SecsSince1990(Location\_WriteTime\_Float,4)

EndSub

' Sub:Write location

Sub Location\_Write

Calfile(Latitude,1,"CPU:Latitude.cal",0) 'Latitude

Calfile(Longitude,1,"CPU:Longitude.cal",0) 'Longitude

Calfile(Elevation,1,"CPU:Elevation.cal",0) 'Elevation

Calfile(UTC\_Offset,1,"CPU:UTC\_Offset.cal",0) 'UTC\_Offset

Location\_WriteTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(Location\_WriteTime\_Float,1,"CPU:Location\_WriteTime.cal",0)

EndSub

' Declare Solar Position Variables and Units---------------------------------

Public SolarPosition\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Dim SolPos(5)

Public Sun\_Azimuth : Units Sun\_Azimuth=degrees

Public Sun\_Elevation : Units Sun\_Elevation=degrees

Public Sun\_Up As Boolean

' Declare Garmin GPS16XHVS\_GPS Variables and Units---------------------------

Public GPS16XHVS\_GPS\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias GPS16XHVS\_GPS\_Block = GPS\_Block

Const GPS16XHVS\_GPS\_UTCOffset = 2 'Enter UTC offset in hours where station will be installed

Dim GPS16XHVS\_GPS\_NmeaSentence(2) As String \* 100

Dim GPS16XHVS\_GPS\_Data(15)

Public GPS16XHVS\_GPS\_FixInvalid : Alias GPS16XHVS\_GPS\_FixInvalid = GPS\_FixInvalid

Public GPS16XHVS\_GPS\_FixGPS : Alias GPS16XHVS\_GPS\_FixGPS = GPS\_FixGPS

Public GPS16XHVS\_GPS\_FixDiffGPS : Alias GPS16XHVS\_GPS\_FixDiffGPS = GPS\_FixDiffGPS

Public GPS16XHVS\_GPS\_FixEstimated : Alias GPS16XHVS\_GPS\_FixEstimated = GPS\_FixEstimated

Public GPS16XHVS\_GPS\_Ready : Alias GPS16XHVS\_GPS\_Ready = GPS\_Ready

Dim GPS16XHVS\_GPS\_LatDD, GPS16XHVS\_GPS\_Lat\_MM\_MMMM,GPS16XHVS\_GPS\_LonDDD, GPS16XHVS\_GPS\_Lon\_MM\_MMMM

Public GPS16XHVS\_GPS\_Latitude : Units GPS16XHVS\_GPS\_Latitude = decDeg : Alias GPS16XHVS\_GPS\_Latitude = GPS\_Latitude

Public GPS16XHVS\_GPS\_Longitude : Units GPS16XHVS\_GPS\_Longitude = decDeg : Alias GPS16XHVS\_GPS\_Longitude = GPS\_Longitude

Public GPS16XHVS\_GPS\_Elevation : Units GPS16XHVS\_GPS\_Elevation = m : Alias GPS16XHVS\_GPS\_Elevation = GPS\_Elevation

Public GPS16XHVS\_GPS\_ClockChanges : Alias GPS16XHVS\_GPS\_ClockChanges = GPS\_ClockChanges

' Declare UTC Time variables and units---------------------------------------

Public UTCTime\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*"

Dim UTC\_Offset\_Long As Long

Dim SSLocal As Long

Dim SSUTC As Long

Dim SSLocalDateString As String \* 25

Dim SSUTCDateString As String \* 25

Public UYear As Long : Dim UYearStr As String ' UTC year

Public UMonth As Long : Dim UMonthStr As String ' UTC month

Public UDay As Long : Dim UDayStr As String ' UCT day

Public UHour As Long : Dim UHourStr As String ' UCT hour

Public UMinute As Long : Dim UMinuteStr As String' UCT minute

'-Declare SYNOP Variables and Units------------------------------------------

' User settings

Public SYNOP\_Settings\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*"

Public WMO\_Block As String \* 30 ' WMO block number

Public WMO\_Station\_ID As String \* 50 ' WMO station number

Public WMO\_Station\_Name As String \* 50 ' Station or site name

Public WMO\_Station\_Type As String \* 30 ' Type of station, 0 automatic, 1 manned, 2 hybrid

' Persistence

Public SYNOP\_WriteFlag As Boolean ' write flag

Dim SYNOP\_WriteTime\_Float

Public SYNOP\_WriteTime As String \* 25 ' write time

' Sub:Read SYNOP

Sub SYNOP\_Read

Dim Closefile, Openfile As Long

Openfile = FileOpen("CPU:WMO\_Block.txt","r",-1)

FileReadLine(Openfile,WMO\_Block,30)

Closefile=FileClose(Openfile)

Openfile = FileOpen("CPU:WMO\_Station\_ID.txt","r",-1)

FileReadLine(Openfile,WMO\_Station\_ID,50)

Closefile=FileClose(Openfile)

Openfile = FileOpen("CPU:WMO\_Station\_Name.txt","r",-1)

FileReadLine(Openfile,WMO\_Station\_Name,50)

Closefile=FileClose(Openfile)

Openfile = FileOpen("CPU:WMO\_Station\_Type.txt","r",-1)

FileReadLine(Openfile,WMO\_Station\_Type,30)

Closefile=FileClose(Openfile)

Calfile(SYNOP\_WriteTime\_Float,1,"CPU:SYNOP\_WriteTime.cal",1)

SYNOP\_WriteTime = SecsSince1990(SYNOP\_WriteTime\_Float,4)

EndSub

' Sub:Write SYNOP

Sub SYNOP\_Write

Dim Closefile, Openfile As Long

Openfile = FileOpen ("CPU:WMO\_Block.txt","w",-1)

FileWrite (Openfile,WMO\_Block,0)

Closefile=FileClose (Openfile)

Openfile = FileOpen ("CPU:WMO\_Station\_ID.txt","w",-1)

FileWrite (Openfile,WMO\_Station\_ID,0)

Closefile=FileClose (Openfile)

Openfile = FileOpen ("CPU:WMO\_Station\_Name.txt","w",-1)

FileWrite (Openfile,WMO\_Station\_Name,0)

Closefile=FileClose (Openfile)

Openfile = FileOpen ("CPU:WMO\_Station\_Type.txt","w",-1)

FileWrite (Openfile,WMO\_Station\_Type,0)

Closefile=FileClose (Openfile)

SYNOP\_WriteTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(SYNOP\_WriteTime\_Float,1,"CPU:SYNOP\_WriteTime.cal",0)

EndSub

' Declare Thies\_WSpd Variables and Units-------------------------------------

Public Thies\_WSpd\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias Thies\_WSpd\_Block = WSpd\_Block

Public Thies\_WSpd : Units Thies\_WSpd = m/s : Alias Thies\_WSpd = WSpd ' Value from sensor, QC not yet performed.

Public Thies\_WSpd3s : Units Thies\_WSpd3s = m/s : Alias Thies\_WSpd3s = WSpd3s ' Value from sensor, QC not yet performed.

' QC working variables

' Startup initialize

Dim Thies\_WSpdStartup

Dim Thies\_WSpdBufferCount

' 3 Second buffer

Dim Thies\_WSpd\_Prev

Dim Thies\_WSpd3sec = NAN

Dim Thies\_WSpd3secN

Dim Thies\_WSpd\_Delta

' 120 Second buffer

Dim Thies\_WSpd120s = NAN

Dim Thies\_WSpd120sN

Dim Thies\_WSpd120s3secMax = NAN

Dim Thies\_WSpd120s3secMaxN

' 600 Second buffer

Dim Thies\_WSpd600s = NAN

Dim Thies\_WSpd600sN

Dim Thies\_WSpd600s3secMax = NAN

Dim Thies\_WSpd600s3secMaxN

' 60 minute buffer

Dim Thies\_WSpd60mBuffer(30) ' Based on the 2 minute average values

Dim Thies\_WSpd60mBufferMin(30)

Dim Thies\_WSpd60mBufferMax(30)

Dim Thies\_WSpd60mDelta

Dim Thies\_WSpd2mDelta

' QC performed

Public Thies\_WSpd2m = NAN : Units Thies\_WSpd2m = m/s : Alias Thies\_WSpd2m = WSpd2m ' QC applied to wind direction 2 minute

Public Thies\_WSpd10m = NAN : Units Thies\_WSpd10m = m/s : Alias Thies\_WSpd10m = WSpd10m ' QC applied to wind direction 10 minute

Public Thies\_WSpd2mGust = NAN : Units Thies\_WSpd2mGust = m/s : Alias Thies\_WSpd2mGust = WSpd2mGust ' QC applied to wind direction 2 minute Gust

Public Thies\_WSpd10mGust = NAN : Units Thies\_WSpd10mGust = m/s : Alias Thies\_WSpd10mGust = WSpd10mGust ' QC applied to wind direction 10 minute Gust

' QC flags raised

Public Thies\_WSpd\_StuckSuspected = 1 : Alias Thies\_WSpd\_StuckSuspected = WSpd\_StuckSuspected

Public Thies\_WSpd2m\_JumpSuspected : Alias Thies\_WSpd2m\_JumpSuspected = WSpd2m\_JumpSuspected

Public Thies\_WSpd10m\_JumpSuspected : Alias Thies\_WSpd10m\_JumpSuspected = WSpd10m\_JumpSuspected

Public Thies\_WSpd2m\_JumpError = 1 : Alias Thies\_WSpd2m\_JumpError = WSpd2m\_JumpError

Public Thies\_WSpd10m\_JumpError = 1 : Alias Thies\_WSpd10m\_JumpError = WSpd10m\_JumpError

Public Thies\_WSpd2m\_Incomplete = 1 : Alias Thies\_WSpd2m\_Incomplete = WSpd2m\_Incomplete

Public Thies\_WSpd10m\_Incomplete = 1 : Alias Thies\_WSpd10m\_Incomplete = WSpd10m\_Incomplete

Public Thies\_WSpd\_Settings\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias Thies\_WSpd\_Settings\_Block = WSpd\_Settings\_Block

Public Thies\_WSpd\_Height = NAN : Units Thies\_WSpd\_Height = m : Alias Thies\_WSpd\_Height = WSpd\_Height' Height of Wind sensor, units metre

' constants

Public Thies\_WSpd\_SensorType = 0 : Alias Thies\_WSpd\_SensorType = WSpd\_SensorType : ReadOnly WSpd\_SensorType 'Wind sensor type

Public Thies\_WSpd\_ReportTimeSignificance = 2 : Alias Thies\_WSpd\_ReportTimeSignificance = WSpd\_ReportTimeSignificance : ReadOnly WSpd\_ReportTimeSignificance 'Wind reporting time offset, hours

Public Thies\_WSpd\_ReportPeriod10M = -10 : Alias Thies\_WSpd\_ReportPeriod10M = WSpd\_ReportPeriod10M : ReadOnly Thies\_WSpd\_ReportPeriod10M 'Wind reporting period, units minute

' Persistence

Public Thies\_WSpd\_WriteSettingsFlag As Boolean : Alias Thies\_WSpd\_WriteSettingsFlag = WSpdWriteSettingsFlag

Dim Thies\_WSpd\_SavedTime\_Float

Public Thies\_WSpd\_SavedTime As String \* 25 : Alias Thies\_WSpd\_SavedTime = WSpd\_SavedTime

' Read settings

Sub Thies\_WSpd\_ReadSettings

Calfile(Thies\_WSpd\_Height,1,"CPU:Thies\_WSpd\_Height.cal",1)

Calfile(Thies\_WSpd\_SavedTime\_Float,1,"CPU:Thies\_WSpd\_SavedTime.cal",1)

Thies\_WSpd\_SavedTime = SecsSince1990(Thies\_WSpd\_SavedTime\_Float,4)

EndSub

' Write setting

Sub Thies\_WSpd\_WriteSettings

Calfile(WSpd\_Height,1,"CPU:Thies\_WSpd\_Height.cal",0)

Thies\_WSpd\_SavedTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(Thies\_WSpd\_SavedTime\_Float,1,"CPU:Thies\_WSpd\_SavedTime.cal",0)

EndSub

' Declare Thies\_WDir Variables and Units-------------------------------------

Public Thies\_WDir\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias Thies\_WDir\_Block = WDir\_Block

Public Thies\_WDir : Units Thies\_WDir = deg : Alias Thies\_WDir = WDir ' Value from sensor, QC not yet performed.

' QC working variables

' Startup initialize

Dim Thies\_WDirStartup

Dim Thies\_WDirBufferCount

' 120 Second buffer

Dim Thies\_WDirSin120s = NAN

Dim Thies\_WDirCos120s = NAN

Dim Thies\_WDir120s = NAN

Dim Thies\_WDir120sN

' 600 Second buffer

Dim Thies\_WDirSin600s = NAN

Dim Thies\_WDirCos600s = NAN

Dim Thies\_WDir600s = NAN

Dim Thies\_WDir600sN

' 60 minute buffer

Dim Thies\_WDir60mBuffer(30) ' based on the 2 minute average values

Dim Thies\_WDirSin60mAccum

Dim Thies\_WDirCos60mAccum

Dim Thies\_WDirSin60m = NAN

Dim Thies\_WDirCos60m = NAN

Dim Thies\_WDir60m = NAN

Dim Thies\_WDir60mN

Dim Thies\_WDir60mOffset = NAN

Dim Thies\_WDir60mOffsetBuffer(30)

Dim Thies\_WDir60mBackMin

Dim Thies\_WDir60mVeerMax

Dim Thies\_WDir60mVariance

' QC performed

Public Thies\_WDir2m = NAN : Units Thies\_WDir2m = deg : Alias Thies\_WDir2m = WDir2m ' QC applied to wind direction 2 minute

Public Thies\_WDir10m = NAN : Units Thies\_WDir10m = deg : Alias Thies\_WDir10m = WDir10m ' QC applied to wind direction 10 minute

' QC flags raised

Public Thies\_WDir\_StuckSuspected = 1 : Alias Thies\_WDir\_StuckSuspected = WDir\_StuckSuspected

Public Thies\_WDir2m\_Incomplete = 1 : Alias Thies\_WDir2m\_Incomplete = WDir2m\_Incomplete

Public Thies\_WDir10m\_Incomplete = 1 : Alias Thies\_WDir10m\_Incomplete = WDir10m\_Incomplete

' Declare TV10\_AirTemp Variables and Units-----------------------------------

Public TV10\_AirTemp\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias TV10\_AirTemp\_Block = AirTemp\_Block

Dim TV10\_AirTemp\_Ratio : Alias TV10\_AirTemp\_Ratio = AirTemp\_Ratio

Public TV10\_AirTemp : Units TV10\_AirTemp = DegC : Alias TV10\_AirTemp = AirTemp ' Value from sensor, QC not yet performed.

' QC working variables

' Startup initialize

Dim TV10\_AirTempStartup

Dim TV10\_AirTempBufferCount

' 60 Second buffer

Dim TV10\_AirTemp\_Prev

Dim TV10\_AirTemp60s = NAN

Dim TV10\_AirTemp60sN

Dim TV10\_AirTemp\_Delta

' 1440 minute buffer

Dim TV10\_AirTemp1440mBuffer(1440) : Dim TV10\_AirTempTime1440mBuffer(1440) As Long

Dim TV10\_AirTemp1440mBufferMin(2)

Dim TV10\_AirTemp1440mBufferMax(2)

Dim TV10\_AirTemp60mDelta

Dim TV10\_AirTemp2mDelta

' QC performed

Public TV10\_AirTemp1M = NAN : Units TV10\_AirTemp1M = DegC : Alias TV10\_AirTemp1M = AirTemp1M ' QC applied to air temperature

Public TV10\_AirTempK1M = NAN : Units TV10\_AirTempK1M = K : Alias TV10\_AirTempK1M = AirTempK1M' QC applied to air temperature

' QC flags raised

Public TV10\_AirTemp\_StuckSuspected = 1 : Alias TV10\_AirTemp\_StuckSuspected = AirTemp\_StuckSuspected

Public TV10\_AirTemp\_JumpSuspected : Alias TV10\_AirTemp\_JumpSuspected = AirTemp\_JumpSuspected

Public TV10\_AirTemp\_JumpError = 1 : Alias TV10\_AirTemp\_JumpError = AirTemp\_JumpError

' Derived parameters

Public TV10\_AirTempMin24H : Units TV10\_AirTempMin24H = DegC : Alias TV10\_AirTempMin24H =AirTempMin24H

Public TV10\_AirTempMax24H : Units TV10\_AirTempMax24H = DegC : Alias TV10\_AirTempMax24H =AirTempMax24H

Public TV10\_AirTempKMin24H : Units TV10\_AirTempKMin24H = K : Alias TV10\_AirTempKMin24H =AirTempKMin24H

Public TV10\_AirTempKMax24H : Units TV10\_AirTempKMax24H = K : Alias TV10\_AirTempKMax24H =AirTempKMax24H

Public TV10\_AirTempMinTime24H As String : Alias TV10\_AirTempMinTime24H =AirTempMinTime24H

Public TV10\_AirTempMaxTime24H As String : Alias TV10\_AirTempMaxTime24H =AirTempMaxTime24H

Public TV10\_AirTemp\_Settings\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias TV10\_AirTemp\_Settings\_Block = AirTemp\_Settings\_Block

Public TV10\_AirTemp\_Cal = NAN : Units TV10\_AirTemp\_Cal = RfRs : Alias TV10\_AirTemp\_Cal = AirTemp\_Cal

Public TV10\_AirTemp\_Height = NAN : Units TV10\_AirTemp\_Height = m : Alias TV10\_AirTemp\_Height = AirTemp\_Height' Height of Temp sensor, units metre

' constants

Public TV10\_AirTemp\_ReportPeriod24H = -24 : Alias TV10\_AirTemp\_ReportPeriod24H = AirTemp\_ReportPeriod24H : ReadOnly TV10\_AirTemp\_ReportPeriod24H 'Temperature reporting period, units hours

Public TV10\_AirTemp\_ReportTimeOffset24H = 0 : Alias TV10\_AirTemp\_ReportTimeOffset24H = AirTemp\_ReportTimeOffset24H : ReadOnly AirTemp\_ReportTimeOffset24H 'Temperature reporting time offset, hours

' Persistence

Public TV10\_AirTemp\_WriteSettingsFlag As Boolean : Alias TV10\_AirTemp\_WriteSettingsFlag = AirTempWriteSettingsFlag

Dim TV10\_AirTemp\_SavedTime\_Float

Public TV10\_AirTemp\_SavedTime As String \* 25 : Alias TV10\_AirTemp\_SavedTime = AirTemp\_SavedTime

' Read settings

Sub TV10\_AirTemp\_ReadSettings

Calfile(TV10\_AirTemp\_Cal,1,"CPU:TV10\_AirTemp\_Cal.cal",1)

Calfile(TV10\_AirTemp\_Height,1,"CPU:TV10\_AirTemp\_Height.cal",1)

Calfile(TV10\_AirTemp\_SavedTime\_Float,1,"CPU:TV10\_AirTemp\_SavedTime.cal",1)

TV10\_AirTemp\_SavedTime = SecsSince1990(TV10\_AirTemp\_SavedTime\_Float,4)

EndSub

' Write setting

Sub TV10\_AirTemp\_WriteSettings

Calfile(TV10\_AirTemp\_Cal,1,"CPU:TV10\_AirTemp\_Cal.cal",0)

Calfile(AirTemp\_Height,1,"CPU:TV10\_AirTemp\_Height.cal",0)

TV10\_AirTemp\_SavedTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(TV10\_AirTemp\_SavedTime\_Float,1,"CPU:TV10\_AirTemp\_SavedTime.cal",0)

EndSub

' Declare EE181\_RH Variables and Units---------------------------------------

Public EE181\_RH\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias EE181\_RH\_Block = RH\_Block

Public EE181\_RH : Units EE181\_RH = % : Alias EE181\_RH = RH ' Value from sensor, QC not yet performed.

' QC working variables

' Startup initialize

Dim EE181\_RHStartup

Dim EE181\_RHBufferCount

' 60 Second buffer

Dim EE181\_RH\_Prev

Dim EE181\_RH60s = NAN

Dim EE181\_RH60sN

Dim EE181\_RH\_Delta

' 1440 minute buffer

Dim EE181\_RH1440mBuffer(1440) : Dim EE181\_RHTime1440mBuffer(1440) As Long

Dim EE181\_RH1440mBufferMin(2)

Dim EE181\_RH1440mBufferMax(2)

Dim EE181\_RH60mDelta

Dim EE181\_RH2mDelta

' QC performed

Public EE181\_RH1M = NAN : Units EE181\_RH1M = % : Alias EE181\_RH1M = RH1M ' QC applied to air temperature

' QC flags raised

Public EE181\_RH\_StuckSuspected = 1 : Alias EE181\_RH\_StuckSuspected = RH\_StuckSuspected

Public EE181\_RH\_JumpSuspected : Alias EE181\_RH\_JumpSuspected = RH\_JumpSuspected

Public EE181\_RH\_JumpError = 1 : Alias EE181\_RH\_JumpError = RH\_JumpError

' Derived parameters

Public EE181\_RHMin24H : Units EE181\_RHMin24H = % : Alias EE181\_RHMin24H =RHMin24H

Public EE181\_RHMax24H : Units EE181\_RHMax24H = % : Alias EE181\_RHMax24H =RHMax24H

Public EE181\_RHMinTime24H As String : Alias EE181\_RHMinTime24H =RHMinTime24H

Public EE181\_RHMaxTime24H As String : Alias EE181\_RHMaxTime24H =RHMaxTime24H

' Declare Calculated\_DewPointTemp Variables and Units------------------------

Public Calculated\_DewPointTemp\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias Calculated\_DewPointTemp\_Block = DewPointTemp\_Block

Public Calculated\_DewPointTemp : Units Calculated\_DewPointTemp = DegC : Alias Calculated\_DewPointTemp = DewPointTemp ' Value from sensor, QC not yet performed.

' QC working variables

' Startup initialize

Dim Calculated\_DewPointTempStartup

Dim Calculated\_DewPointTempBufferCount

' 60 Second buffer

Dim Calculated\_DewPointTemp\_Prev

Dim Calculated\_DewPointTemp60s = NAN

Dim Calculated\_DewPointTemp60sN

Dim Calculated\_DewPointTemp\_Delta

' 1440 minute buffer

Dim Calculated\_DewPointTemp1440mBuffer(1440) : Dim Calculated\_DewPointTempTime1440mBuffer(1440) As Long

Dim Calculated\_DewPointTemp1440mBufferMin(2)

Dim Calculated\_DewPointTemp1440mBufferMax(2)

Dim Calculated\_DewPointTemp60mDelta

Dim Calculated\_DewPointTemp2mDelta

' QC performed

Public Calculated\_DewPointTemp1M = NAN : Units Calculated\_DewPointTemp1M = DegC : Alias Calculated\_DewPointTemp1M = DewPointTemp1M ' QC applied to DewPoint temperature

' QC flags raised

Public Calculated\_DewPointTemp\_StuckSuspected = 1 : Alias Calculated\_DewPointTemp\_StuckSuspected = DewPointTemp\_StuckSuspected

Public Calculated\_DewPointTemp\_JumpSuspected : Alias Calculated\_DewPointTemp\_JumpSuspected = DewPointTemp\_JumpSuspected

Public Calculated\_DewPointTemp\_JumpError = 1 : Alias Calculated\_DewPointTemp\_JumpError = DewPointTemp\_JumpError

' Derived parameters

Public Calculated\_DewPointTemp10M : Units Calculated\_DewPointTemp10M = DegC : Alias Calculated\_DewPointTemp10M =DewPointTemp10M

Public Calculated\_DewPointTempK10M : Units Calculated\_DewPointTempK10M = K : Alias Calculated\_DewPointTempK10M =DewPointTempK10M

Public Calculated\_DewPointTempMin24H : Units Calculated\_DewPointTempMin24H = DegC : Alias Calculated\_DewPointTempMin24H =DewPointTempMin24H

Public Calculated\_DewPointTempMax24H : Units Calculated\_DewPointTempMax24H = DegC : Alias Calculated\_DewPointTempMax24H =DewPointTempMax24H

Public Calculated\_DewPointTempKMin24H : Units Calculated\_DewPointTempKMin24H = K : Alias Calculated\_DewPointTempKMin24H =DewPointTempKMin24H

Public Calculated\_DewPointTempKMax24H : Units Calculated\_DewPointTempKMax24H = K : Alias Calculated\_DewPointTempKMax24H =DewPointTempKMax24H

Public Calculated\_DewPointTempMinTime24H As String : Alias Calculated\_DewPointTempMinTime24H =DewPointTempMinTime24H

Public Calculated\_DewPointTempMaxTime24H As String : Alias Calculated\_DewPointTempMaxTime24H =DewPointTempMaxTime24H

' Declare RMY52203\_Rain Variables and Units----------------------------------

Public RMY52203\_Rain\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias RMY52203\_Rain\_Block = Rain\_Block

Public RMY52203\_Rain : Units RMY52203\_Rain = mm : Alias RMY52203\_Rain = Rain

' QC working variables

Dim RMY52203\_Rain60s : Units RMY52203\_Rain60s = mm : Alias RMY52203\_Rain60s = Rain60s

' QC performed

Public RMY52203\_RainIntensity1M : Units RMY52203\_RainIntensity1M = mm/h : Alias RMY52203\_RainIntensity1M = RainIntensity1M

Public RMY52203\_Rain1M : Units RMY52203\_Rain1M = mm : Alias RMY52203\_Rain1M = Rain1M

Public RMY52203\_Rain1H : Units RMY52203\_Rain1H = mm : Alias RMY52203\_Rain1H = Rain1H

Public RMY52203\_Rain3H : Units RMY52203\_Rain3H = mm : Alias RMY52203\_Rain3H = Rain3H

Public RMY52203\_Rain24H : Units RMY52203\_Rain24H = mm : Alias RMY52203\_Rain24H = Rain24H

Public RMY52203\_Rain\_Settings\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias RMY52203\_Rain\_Settings\_Block = Rain\_Settings\_Block

Public RMY52203\_Rain\_Height = NAN : Units RMY52203\_Rain\_Height = m : Alias RMY52203\_Rain\_Height = Rain\_Height' Height of Rain sensor, units metre

' constants

Public RMY52203\_Rain\_ReportPeriod1H = -1 : Alias RMY52203\_Rain\_ReportPeriod1H = Rain\_ReportPeriod1H : ReadOnly RMY52203\_Rain\_ReportPeriod1H 'Rain reporting period, units hours

Public RMY52203\_Rain\_ReportPeriod3H = -3 : Alias RMY52203\_Rain\_ReportPeriod3H = Rain\_ReportPeriod3H : ReadOnly RMY52203\_Rain\_ReportPeriod3H 'Rain reporting period, units hours

Public RMY52203\_Rain\_ReportPeriod24H = -24 : Alias RMY52203\_Rain\_ReportPeriod24H = Rain\_ReportPeriod24H : ReadOnly RMY52203\_Rain\_ReportPeriod24H 'Rain reporting period, units hours

' Persistence

Public RMY52203\_Rain\_WriteSettingsFlag As Boolean : Alias RMY52203\_Rain\_WriteSettingsFlag = RainWriteSettingsFlag

Dim RMY52203\_Rain\_SavedTime\_Float

Public RMY52203\_Rain\_SavedTime As String \* 25 : Alias RMY52203\_Rain\_SavedTime = Rain\_SavedTime

' Read settings

Sub RMY52203\_Rain\_ReadSettings

Calfile(RMY52203\_Rain\_Height,1,"CPU:RMY52203\_Rain\_Height.cal",1)

Calfile(RMY52203\_Rain\_SavedTime\_Float,1,"CPU:RMY52203\_Rain\_SavedTime.cal",1)

RMY52203\_Rain\_SavedTime = SecsSince1990(RMY52203\_Rain\_SavedTime\_Float,4)

EndSub

' Write setting

Sub RMY52203\_Rain\_WriteSettings

Calfile(Rain\_Height,1,"CPU:RMY52203\_Rain\_Height.cal",0)

RMY52203\_Rain\_SavedTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(RMY52203\_Rain\_SavedTime\_Float,1,"CPU:RMY52203\_Rain\_SavedTime.cal",0)

EndSub

' Declare BV10\_BPress Variables and Units------------------------------------

Public BV10\_BPress\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias BV10\_BPress\_Block = BPress\_Block

Dim BV10\_SDI12(3)

Public BV10\_BPress : Units BV10\_BPress = hPa : Alias BV10\_BPress = BPress ' Value from sensor, QC not yet performed.

' Optionally include

Public BV10\_BPTemp : Units BV10\_BPTemp = degC : Alias BV10\_BPTemp = BPTemp

' Calculated values

Public BV10\_tQFE = NAN : Units BV10\_tQFE = degC : Alias BV10\_tQFE = tQFE

Public BV10\_QFE : Units BV10\_QFE = hPa : Alias BV10\_QFE = QFE'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Public BV10\_QNH : Units BV10\_QNH = hPa : Alias BV10\_QNH = QNH 'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Dim BV10\_hISA

Public BV10\_QNH\_ICAO : Units BV10\_QNH\_ICAO = hPa : Alias BV10\_QNH\_ICAO = QNH\_ICAO 'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Public BV10\_tQFF = NAN : Units BV10\_tQFF = k : Alias BV10\_tQFF = tQFF

Public BV10\_QFF : Units BV10\_QFF = hPa : Alias BV10\_QFF = QFF 'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

' QC working variables

' Startup initialize

Dim BV10\_BPressStartup

Dim BV10\_BPressBufferCount

' 60 Second buffer

Dim BV10\_BPress\_Prev

Dim BV10\_BPress60s = NAN

Dim BV10\_BPress60sN

Dim BV10\_Bpress\_Delta

' 180 minute buffer

Dim BV10\_BPress180mBuffer(180)

Dim BV10\_BPress180mBufferMin(2)

Dim BV10\_BPress180mBufferMax(2)

Dim BV10\_Bpress60mDelta

Dim BV10\_Bpress2mDelta

' QC performed

Public BV10\_BPress1M = NAN : Units BV10\_BPress1M = hPa : Alias BV10\_BPress1M = BPress1M ' QC applied to barometric pressure

Public BV10\_BPressPa1M = NAN : Units BV10\_BPressPa1M = Pa : Alias BV10\_BPressPa1M = BPressPa1M ' QC applied to barometric pressure

' QC flags raised

Public BV10\_Bpress\_StuckSuspected = 1 : Alias BV10\_Bpress\_StuckSuspected = BPress\_StuckSuspected

Public BV10\_BPress\_JumpSuspected : Alias BV10\_BPress\_JumpSuspected = BPress\_JumpSuspected

Public BV10\_BPress\_JumpError = 1 : Alias BV10\_BPress\_JumpError = BPress\_JumpError

' Derived parameters

Public BV10\_tQFE1M = NAN : Units BV10\_tQFE1M = degC : Alias BV10\_tQFE1M = tQFE1M

Public BV10\_QFE1M : Units BV10\_QFE1M = hPa : Alias BV10\_QFE1M = QFE1M'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Public BV10\_QNH1M : Units BV10\_QNH1M = hPa : Alias BV10\_QNH1M = QNH1M 'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Public BV10\_QNHPa1M : Units BV10\_QNHPa1M = Pa : Alias BV10\_QNHPa1M = QNHPa1M 'Calculated value(done in logger) 'Be sure the sensor setup correlates with this

Public BV10\_BPressChange3H : Units BV10\_BPressChange3H = hPa : Alias BV10\_BPressChange3H = BPressChange3H

Public BV10\_BPressPaChange3H : Units BV10\_BPressPaChange3H = Pa : Alias BV10\_BPressPaChange3H = BPressPaChange3H

Public BV10\_BPressTendency3H : Alias BV10\_BPressTendency3H = BPressTendency3H

Public BV10\_BPress\_Elevation = NAN : Units BV10\_BPress\_Elevation = m : Alias BV10\_BPress\_Elevation = BPress\_Elevation ReadOnly BPress\_Elevation 'Readonly: Calculated value

Public BV10\_BPress\_Settings\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias BV10\_BPress\_Settings\_Block = BPress\_Settings\_Block

' To calculate QFE, QFF, QNH enter these heights and standard temperature

Public BV10\_hQFE = NAN: Units BV10\_hQFE = m : Alias BV10\_hQFE = hQFE' : ReadOnly BV10\_hQFE

Public BV10\_tstdQFE = NAN : Units BV10\_tstdQFE = degC : Alias BV10\_tstdQFE = tstdQFE' : ReadOnly BV10\_tstdQFE 'Temp value if air temp is not available

Public BV10\_hQNH = NAN : Units BV10\_hQNH = m : Alias BV10\_hQNH = hQNH' : ReadOnly BV10\_hQNH

' Persistence

'Public BV10\_WriteSettingsBlock As String = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*" : Alias BV10\_WriteSettingsBlock = BPress\_WriteSettingsBlock

Public BV10\_WriteSettingsFlag As Boolean : Alias BV10\_WriteSettingsFlag = BPress\_WriteSettingsFlag ' settings flag

Dim BV10\_SettingsTimeFloat

Public BV10\_SettingsTime As String \* 25 : Alias BV10\_SettingsTime = BPress\_SettingsTime ' Settings Time

' Sub:Read zero values

Sub BV10\_ReadSettings

Calfile(BV10\_hQFE,1,"CPU:BV10\_hQFE.cal",1) 'BV10\_hQFE

Calfile(BV10\_tstdQFE,1,"CPU:BV10\_tstdQFE.cal",1) 'BV10\_tstdQFE

Calfile(BV10\_hQNH,1,"CPU:BV10\_hQNH.cal",1) 'BV10\_hQNH

Calfile(BV10\_SettingsTimeFloat,1,"CPU:BV10\_SettingsTime.cal",1)

BV10\_SettingsTime = SecsSince1990(BV10\_SettingsTimeFloat,4)

EndSub

' Sub:Write zero values

Sub BV10\_WriteSettings

Calfile(BV10\_hQFE,1,"CPU:BV10\_hQFE.cal",0) 'BV10\_hQFE

Calfile(BV10\_tstdQFE,1,"CPU:BV10\_tstdQFE.cal",0) 'BV10\_tstdQFE

Calfile(BV10\_hQNH,1,"CPU:BV10\_hQNH.cal",0) 'BV10\_hQNH

BV10\_SettingsTimeFloat = SecsSince1990(Public.Timestamp,1)

Calfile(BV10\_SettingsTimeFloat,1,"CPU:BV10\_SettingsTime.cal",0)

EndSub

' Declare RSG2STDN\_SlrW \_SlrMJ variables and units---------------------------

Public RSG2STDN\_SlrW\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias RSG2STDN\_SlrW\_Block = SolRad\_Block

Public RSG2STDN\_SlrW\_SN As String \* 10 = "####" : Alias RSG2STDN\_SlrW\_SN = SlrW\_SN

Public RSG2STDN\_SlrW\_Sensitivity = NAN : Alias RSG2STDN\_SlrW\_Sensitivity = SlrW\_Sensitivity

Dim RSG2STDN\_SlrmV : Units RSG2STDN\_SlrmV = mV : Alias RSG2STDN\_SlrmV = SlrmV

Public RSG2STDN\_SlrW : Units RSG2STDN\_SlrW = W/m^2 : Alias RSG2STDN\_SlrW = SlrW

Public RSG2STDN\_SlrMJ : Units RSG2STDN\_SlrMJ = MJ/m^2 : Alias RSG2STDN\_SlrMJ = SlrMJ ' ET0 work with this so keep it in

' QC working variables

Dim RSG2STDN\_SlrJ : Units RSG2STDN\_SlrJ = J/m^2 : Alias RSG2STDN\_SlrJ = SlrJ

Dim RSG2STDN\_SlrJ60s : Units RSG2STDN\_SlrJ60s = J/m^2 : Alias RSG2STDN\_SlrJ60s = SlrJ60s

' QC performed

Public RSG2STDN\_SlrJ1H : Units RSG2STDN\_SlrJ1H = J/m^2 : Alias RSG2STDN\_SlrJ1H = SlrJ1H

Public RSG2STDN\_SlrJ24H : Units RSG2STDN\_SlrJ24H = J/m^2 : Alias RSG2STDN\_SlrJ24H = SlrJ24H

' constants

Public RSG2STDN\_SlrJ\_ReportPeriod1H = -1 : Alias RSG2STDN\_SlrJ\_ReportPeriod1H = SlrJ\_ReportPeriod1H : ReadOnly SlrJ\_ReportPeriod1H 'SlrJ reporting period, units hours

Public RSG2STDN\_SlrJ\_ReportPeriod24H = -24 : Alias RSG2STDN\_SlrJ\_ReportPeriod24H = SlrJ\_ReportPeriod24H : ReadOnly SlrJ\_ReportPeriod24H 'SlrJ reporting period, units hours

' Persistence

Public RSG2STDN\_SlrW\_WriteSettingsFlag As Boolean : Alias RSG2STDN\_SlrW\_WriteSettingsFlag = SlrW\_WriteSettingsFlag

Dim RSG2STDN\_SlrW\_SavedTime\_Float

Public RSG2STDN\_SlrW\_SavedTime As String \* 25 : Alias RSG2STDN\_SlrW\_SavedTime = SlrW\_SavedTime

' Read settings

Sub RSG2STDN\_SlrW\_ReadSettings

Dim Closefile, Openfile As Long

Openfile = FileOpen("CPU:RSG2STDN\_SlrW\_SN.txt","r",-1)

FileReadLine(Openfile,RSG2STDN\_SlrW\_SN,25)

Closefile=FileClose(Openfile)

Calfile(RSG2STDN\_SlrW\_Sensitivity,1,"CPU:RSG2STDN\_SlrW\_Sensitivity.cal",1)

Calfile(RSG2STDN\_SlrW\_SavedTime\_Float,1,"CPU:RSG2STDN\_SlrW\_SavedTime.cal",1)

RSG2STDN\_SlrW\_SavedTime = SecsSince1990(RSG2STDN\_SlrW\_SavedTime\_Float,4)

EndSub

' Write setting

Sub RSG2STDN\_SlrW\_WriteSettings

Dim Closefile, Openfile As Long

Openfile = FileOpen ("CPU:RSG2STDN\_SlrW\_SN.txt","w",-1)

FileWrite (Openfile,RSG2STDN\_SlrW\_SN,0)

Closefile=FileClose (Openfile)

Calfile(RSG2STDN\_SlrW\_Sensitivity,1,"CPU:RSG2STDN\_SlrW\_Sensitivity.cal",0)

RSG2STDN\_SlrW\_SavedTime\_Float = SecsSince1990(Public.Timestamp,1)

Calfile(RSG2STDN\_SlrW\_SavedTime\_Float,1,"CPU:RSG2STDN\_SlrW\_SavedTime.cal",0)

EndSub

' Declare Sun\_Hours Variables and Units--------------------------------------

Public Sun\_Hours\_Block As String = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Public PotSlrW : Units PotSlrW=W/m^2

Public Sun\_Hours : Units Sun\_Hours=Hours

' QC working variables

Dim SunHours60s : Units SunHours60s = minutes ' Calculation is hours but expressed in minutes

' QC performed

Public SunHours1H : Units SunHours1H = minutes ' Calculation is hours but expressed in minutes

Public SunHours24H : Units SunHours24H = minutes ' Calculation is hours but expressed in minutes

' constants

Public SunHours\_ReportPeriod1H = -1 : ReadOnly SunHours\_ReportPeriod1H 'SlrJ reporting period, units hours

Public SunHours\_ReportPeriod24H = -24 : ReadOnly SunHours\_ReportPeriod24H 'SlrJ reporting period, units hours

' Declare SM2C \_VWC\_1 \_EC\_1 \_SoilTemp\_1 Variables and Units------------------

Public SM2C\_VWC\_1\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias SM2C\_VWC\_1\_Block = VWC\_1\_Block

Dim SM2C\_ModbusData\_1(3) As Long, SM2C\_ModbusResult\_1

Public SM2C\_VWC\_1 : Units SM2C\_VWC\_1 = % : Alias SM2C\_VWC\_1 = VWC\_1

Public SM2C\_EC\_1 : Units SM2C\_EC\_1 = mS/cm : Alias SM2C\_EC\_1 = EC\_1

Public SM2C\_SoilTemp\_1 : Units SM2C\_SoilTemp\_1=DegC : Alias SM2C\_SoilTemp\_1 = SoilTemp\_1

' Declare SM2C \_VWC\_2 \_EC\_2 \_SoilTemp\_2 Variables and Units------------------

Public SM2C\_VWC\_2\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias SM2C\_VWC\_2\_Block = VWC\_2\_Block

Dim SM2C\_ModbusData\_2(3) As Long, SM2C\_ModbusResult\_2

Public SM2C\_VWC\_2 : Units SM2C\_VWC\_2 = % : Alias SM2C\_VWC\_2 = VWC\_2

Public SM2C\_EC\_2 : Units SM2C\_EC\_2 = mS/cm : Alias SM2C\_EC\_2 = EC\_2

Public SM2C\_SoilTemp\_2 : Units SM2C\_SoilTemp\_2=DegC : Alias SM2C\_SoilTemp\_2 = SoilTemp\_2

' Declare SM2C \_VWC\_3 \_EC\_3 \_SoilTemp\_3 Variables and Units------------------

Public SM2C\_VWC\_3\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias SM2C\_VWC\_3\_Block = VWC\_3\_Block

Dim SM2C\_ModbusData\_3(3) As Long, SM2C\_ModbusResult\_3

Public SM2C\_VWC\_3 : Units SM2C\_VWC\_3 = % : Alias SM2C\_VWC\_3 = VWC\_3

Public SM2C\_EC\_3 : Units SM2C\_EC\_3 = mS/cm : Alias SM2C\_EC\_3 = EC\_3

Public SM2C\_SoilTemp\_3 : Units SM2C\_SoilTemp\_3=DegC : Alias SM2C\_SoilTemp\_3 = SoilTemp\_3

' Declare SM2C \_VWC\_4 \_EC\_4 \_SoilTemp\_4 Variables and Units------------------

Public SM2C\_VWC\_4\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias SM2C\_VWC\_4\_Block = VWC\_4\_Block

Dim SM2C\_ModbusData\_4(3) As Long, SM2C\_ModbusResult\_4

Public SM2C\_VWC\_4 : Units SM2C\_VWC\_4 = % : Alias SM2C\_VWC\_4 = VWC\_4

Public SM2C\_EC\_4 : Units SM2C\_EC\_4 = mS/cm : Alias SM2C\_EC\_4 = EC\_4

Public SM2C\_SoilTemp\_4 : Units SM2C\_SoilTemp\_4=DegC : Alias SM2C\_SoilTemp\_4 = SoilTemp\_4

' Declare MPPT4A4A Variables, Units and Settings-----------------------------

Public MPPT4A\_SolarCharger\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*" : Alias MPPT4A\_SolarCharger\_Block = SolarCharger\_Block

' SDI12

Dim MPPT4A\_SDI12\_1(10) : Dim MPPT4A\_SDI12\_2(10) : Dim MPPT4A\_SDI12\_X(1)

' Values from charger

Public MPPT4A\_PanelVoltage : Units MPPT4A\_PanelVoltage = V : Alias MPPT4A\_PanelVoltage = SolarCharger\_PanelVoltage

Public MPPT4A\_PanelCurrent : Units MPPT4A\_PanelCurrent = A : Alias MPPT4A\_PanelCurrent = SolarCharger\_PanelCurrent

Public MPPT4A\_PanelPower : Units MPPT4A\_PanelPower = W : Alias MPPT4A\_PanelPower = SolarCharger\_PanelPower

Public MPPT4A\_LoadVoltage : Units MPPT4A\_LoadVoltage = V : Alias MPPT4A\_LoadVoltage = SolarCharger\_LoadVoltage

Public MPPT4A\_LoadCurrent : Units MPPT4A\_LoadCurrent = A : Alias MPPT4A\_LoadCurrent = SolarCharger\_LoadCurrent

Public MPPT4A\_BatteryVoltage : Units MPPT4A\_BatteryVoltage = V : Alias MPPT4A\_BatteryVoltage = SolarCharger\_BatteryVoltage

Public MPPT4A\_BoardTemp : Units MPPT4A\_BoardTemp = degC : Alias MPPT4A\_BoardTemp = SolarCharger\_BoardTemp

Public MPPT4A\_State : Alias MPPT4A\_State = SolarCharger\_State

' Settings to and from charger

Dim MPPT4A\_SetMode = 0 : Dim MPPT4A\_Mode

Dim MPPT4A\_SetBulkFloatVoltage = 13 : Units MPPT4A\_SetBulkFloatVoltage = V : Dim MPPT4A\_BulkFloatVoltage : Units MPPT4A\_BulkFloatVoltage = V

Dim MPPT4A\_SetFloatVoltage = 13.4 : Units MPPT4A\_SetFloatVoltage = V : Dim MPPT4A\_FloatVoltage : Units MPPT4A\_FloatVoltage = V

Dim MPPT4A\_SetCurrentLimit = 3 : Units MPPT4A\_SetCurrentLimit = A : Dim MPPT4A\_CurrentLimit : Units MPPT4A\_CurrentLimit = A

Dim MPPT4A\_SetAbsorbTimeLimit = 4 : Units MPPT4A\_SetAbsorbTimeLimit = Hours : Dim MPPT4A\_AbsorbTimeLimit : Units MPPT4A\_AbsorbTimeLimit = Hours

Dim MPPT4A\_SetAbsorbFullCurrent = 0.3 : Units MPPT4A\_SetAbsorbFullCurrent = A : Dim MPPT4A\_AbsorbFullCurrent : Units MPPT4A\_AbsorbFullCurrent = A

Dim MPPT4A\_SetVCalSlope = 1020 : Dim MPPT4A\_VCalSlope

Dim MPPT4A\_SetICalSlope = 1000 : Dim MPPT4A\_ICalSlope

'-Declare LoggerInternal Variables and Units---------------------------------

Public LoggerInternal\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*"

Public ProgramName As String \* 50

Public ProgramSignature As String \* 6

Public LoggerSerialNumber As String \* 6

Public LoggerBattery : Units LoggerBattery = V

Public LoggerTemp : Units LoggerTemp = degC

Public LoggerLithiumBatt : Units LoggerLithiumBatt = V

Public ScanTime1 : Units ScanTime1 = mSec

Public ScanTime2 : Units ScanTime2 = mSec

Public ScanTime3 : Units ScanTime3 = mSec

Public ScanTime4 : Units ScanTime4 = mSec

Dim rTime(9)

Public CardStatus As String \* 50

Public WatchdogErrors : Units WatchdogErrors = Count

'-Declare Modem Variables and Units------------------------------------------

Public Modem\_Block As String \* 12 = "\*\*\*\*\*\*\*\*\*\*\*\*"

Public CellState As String \* 50

Public CellSignalStrength

Public CellSignalQuality

Dim CellInfoRaw As String \* 500

Public CellInfo(15) As String \* 50

Dim ModemReset : Units ModemReset = Flag

Dim ModemLowPower : Units ModemLowPower = Flag

Dim PingRetry

Public PingTime As Long : Units PingTime = msec

Dim IPFailCount As Long

'-Declare data tables--------------------------------------------------------

DataTable(Table2min, True, -1) 'Data table for 2-minute records, 2-minute wind speed and direction logged for aerodromes

DataInterval(0, 2, min, -1)

'-Memory card

CardOut (0,-1 )

'-JobID & StationID

Sample(1,JobID,String)

Sample(1,StationID,String)

' Thies\_WSpd(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

' For reference filtering in a regular table can be performed if required for e.g. exclude Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN

Minimum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Maximum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Average (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

StdDev (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

Totalize(1,Thies\_WSpd\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpError,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' Thies\_WDir(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

WindVector(1,1,Thies\_WDir,ieee4,Thies\_WDir=NAN,0,0,4)

FieldNames("Thies\_WDir\_Avg,Thies\_WDir\_Std")

Totalize(1,Thies\_WDir\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WDir2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

'-Logger internal

'Sample(1,LoggerSerialNumber,String)

'Sample(1,ProgramName,String)

'Sample(1,ProgramSignature,String)

'Average (1,LoggerBattery,FP2,False)

'Average (1,LoggerTemp,FP2,False)

'Average(1,LoggerLithiumBatt,fp2,false)

'Average(1,PingTime,ieee4,PingTime=0)

Totalize(1,1,ieee4,false)

FieldNames("ScanCount")

EndTable

'-Declare data tables--------------------------------------------------------

DataTable(Table10m, True, -1) 'Data table for 10-minute records

DataInterval(0, 10, min, -1)

'-Memory card

'CardOut (0,-1 )

'-JobID & StationID

Sample(1,JobID,String)

Sample(1,StationID,String)

' Thies\_WSpd(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Maximum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Average (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

StdDev (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

Totalize(1,Thies\_WSpd\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpError,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' Thies\_WDir(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

WindVector(1,1,Thies\_WDir,ieee4,Thies\_WDir=NAN,0,0,4)

FieldNames("Thies\_WDir\_Avg,Thies\_WDir\_Std")

Totalize(1,Thies\_WDir\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WDir2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' TV10 \_AirTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,TV10\_AirTemp,ieee4,TV10\_AirTemp = NAN)

Totalize(1,TV10\_AirTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' TV10 \_AirTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,EE181\_RH,ieee4,EE181\_RH = NAN)

Totalize(1,EE181\_RH\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' Calculated \_DewPointTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,Calculated\_DewPointTemp,ieee4,Calculated\_DewPointTemp = NAN)

Totalize(1,Calculated\_DewPointTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RMY52203\_Rain(GBON QC not performed on rain)

'..Total

Totalize(1,RMY52203\_Rain,fp2,false)

'..Intensity

Minimum(1,RMY52203\_RainIntensity1M,fp2,false,false)

Maximum(1,RMY52203\_RainIntensity1M,fp2,false,false)

Average(1,RMY52203\_RainIntensity1M,fp2,false)

'..Running totals

Sample(1,RMY52203\_Rain1M,fp2)

Sample(1,RMY52203\_Rain1H,fp2)

Sample(1,RMY52203\_Rain3H,fp2)

Sample(1,RMY52203\_Rain24H,fp2)

' BV10 \_BPress(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,BV10\_BPress,ieee4,BV10\_BPress = NAN)

Average(1,BV10\_BPTemp,ieee4,BV10\_BPTemp = NAN)

Average(1,BV10\_QFE,IEEE4,BV10\_QFE=NAN)

Average(1,BV10\_QNH,IEEE4,BV10\_QNH=NAN)

Average(1,BV10\_QNH\_ICAO,IEEE4,BV10\_QNH\_ICAO=NAN)

Average(1,BV10\_QFF,IEEE4,BV10\_QFF=NAN)

Totalize(1,BV10\_Bpress\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RSG2STDN\_SlrW \_SlrMJ(GBON QC not performed on Solar)

Average(1,RSG2STDN\_SlrW,ieee4,RSG2STDN\_SlrW=NAN)'Excluded NANs

Totalize(1,RSG2STDN\_SlrMJ,ieee4,RSG2STDN\_SlrMJ=NAN)'Excluded NANs

'-Solpos

Sample (1,Sun\_Azimuth,FP2)

Sample (1,Sun\_Elevation,FP2)

'-Sun\_Hours(GBON QC not performed on Solar)

Totalize(1,Sun\_Hours,ieee4,false)

' SM2C \_VWC\_1 \_EC\_1 \_SoilTemp\_1(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_1,ieee4,SM2C\_VWC\_1=NAN)

Average(1,SM2C\_EC\_1,ieee4,SM2C\_EC\_1=NAN)

Average(1,SM2C\_SoilTemp\_1,fp2,SM2C\_SoilTemp\_1=NAN)

' SM2C \_VWC\_2 \_EC\_2 \_SoilTemp\_2(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_2,ieee4,SM2C\_VWC\_2=NAN)

Average(1,SM2C\_EC\_2,ieee4,SM2C\_EC\_2=NAN)

Average(1,SM2C\_SoilTemp\_2,fp2,SM2C\_SoilTemp\_2=NAN)

' SM2C \_VWC\_3 \_EC\_3 \_SoilTemp\_3(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_3,ieee4,SM2C\_VWC\_3=NAN)

Average(1,SM2C\_EC\_3,ieee4,SM2C\_EC\_3=NAN)

Average(1,SM2C\_SoilTemp\_3,fp2,SM2C\_SoilTemp\_3=NAN)

' SM2C \_VWC\_4 \_EC\_4 \_SoilTemp\_4(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_4,ieee4,SM2C\_VWC\_4=NAN)

Average(1,SM2C\_EC\_4,ieee4,SM2C\_EC\_4=NAN)

Average(1,SM2C\_SoilTemp\_4,fp2,SM2C\_SoilTemp\_4=NAN)

' GPS16XHVS\_GPS

Totalize(1,GPS16XHVS\_GPS\_Ready,Long,false)

Sample(1,GPS16XHVS\_GPS\_ClockChanges,Long,false)

Sample(1,GPS16XHVS\_GPS\_Latitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Longitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Elevation,ieee4)

' MPPT4A\_SolarCharger

' Values from charger

Average(1,MPPT4A\_PanelVoltage,fp2, MPPT4A\_PanelVoltage=NAN)

Average(1,MPPT4A\_PanelCurrent,fp2,MPPT4A\_PanelCurrent =NAN)

Average(1,MPPT4A\_PanelPower,fp2, MPPT4A\_PanelPower=NAN)

Average(1,MPPT4A\_LoadVoltage,fp2, MPPT4A\_LoadVoltage=NAN)

Average(1,MPPT4A\_LoadCurrent,fp2, MPPT4A\_LoadCurrent=NAN)

Average(1,MPPT4A\_BatteryVoltage,fp2,MPPT4A\_BatteryVoltage=NAN)

Average(1,MPPT4A\_BoardTemp ,fp2, MPPT4A\_BoardTemp=NAN)

Sample(1,MPPT4A\_State,fp2)

'-Logger internal

Sample(1,LoggerSerialNumber,String)

Sample(1,ProgramName,String)

Sample(1,ProgramSignature,String)

Average (1,LoggerBattery,FP2,False)

Average (1,LoggerTemp,FP2,False)

Average(1,LoggerLithiumBatt,fp2,false)

Average(1,PingTime,ieee4,PingTime=0)

Totalize(1,1,ieee4,false)

FieldNames("ScanCount")

EndTable

DataTable(TableHour, True, -1) 'Data table for hourly records

DataInterval(0, 60, min, -1)

'-Memory card

'CardOut (0,-1 )

'-JobID & StationID

Sample(1,JobID,String)

Sample(1,StationID,String)

' Thies\_WSpd(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Maximum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Average (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

StdDev (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

Totalize(1,Thies\_WSpd\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpError,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' Thies\_WDir(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

WindVector(1,1,Thies\_WDir,ieee4,False,0,0,4)

FieldNames("Thies\_WDir\_Avg,Thies\_WDir\_Std")

Totalize(1,Thies\_WDir\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WDir2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' TV10 \_AirTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,TV10\_AirTemp,ieee4,TV10\_AirTemp = NAN)

Totalize(1,TV10\_AirTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' EE181 \_RH(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,EE181\_RH,ieee4,EE181\_RH = NAN)

Totalize(1,EE181\_RH\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' Calculated \_DewPointTemp(GBON filtering and QC counts in a regular table)

Average(1,Calculated\_DewPointTemp,ieee4,Calculated\_DewPointTemp = NAN)

Totalize(1,Calculated\_DewPointTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RMY52203\_Rain(GBON QC not performed on rain)

'..Total

Totalize(1,RMY52203\_Rain,fp2,false)

'..Intensity

Minimum(1,RMY52203\_RainIntensity1M,fp2,false,false)

Maximum(1,RMY52203\_RainIntensity1M,fp2,false,false)

Average(1,RMY52203\_RainIntensity1M,fp2,false)

'..Running totals

Sample(1,RMY52203\_Rain1M,fp2)

Sample(1,RMY52203\_Rain1H,fp2)

Sample(1,RMY52203\_Rain3H,fp2)

Sample(1,RMY52203\_Rain24H,fp2)

' BV10 \_BPress(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Average(1,BV10\_BPress,ieee4,BV10\_BPress = NAN)

Average(1,BV10\_BPTemp,ieee4,BV10\_BPTemp = NAN)

Average(1,BV10\_QFE,IEEE4,BV10\_QFE=NAN)

Average(1,BV10\_QNH,IEEE4,BV10\_QNH=NAN)

Average(1,BV10\_QNH\_ICAO,IEEE4,BV10\_QNH\_ICAO=NAN)

Average(1,BV10\_QFF,IEEE4,BV10\_QFF=NAN)

Totalize(1,BV10\_Bpress\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RSG2STDN\_SlrW \_SlrMJ

Average(1,RSG2STDN\_SlrW,ieee4,RSG2STDN\_SlrW=NAN)'Excluded NANs

Totalize(1,RSG2STDN\_SlrMJ,ieee4,RSG2STDN\_SlrMJ=NAN)'Excluded NANs

'-Solpos

Sample (1,Sun\_Azimuth,FP2)

Sample (1,Sun\_Elevation,FP2)

'-Sun\_Hours

Totalize(1,Sun\_Hours,ieee4,false)

' SM2C \_VWC\_1 \_EC\_1 \_SoilTemp\_1(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_1,ieee4,SM2C\_VWC\_1=NAN)

Average(1,SM2C\_EC\_1,ieee4,SM2C\_EC\_1=NAN)

Average(1,SM2C\_SoilTemp\_1,fp2,SM2C\_SoilTemp\_1=NAN)

' SM2C \_VWC\_2 \_EC\_2 \_SoilTemp\_2(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_2,ieee4,SM2C\_VWC\_2=NAN)

Average(1,SM2C\_EC\_2,ieee4,SM2C\_EC\_2=NAN)

Average(1,SM2C\_SoilTemp\_2,fp2,SM2C\_SoilTemp\_2=NAN)

' SM2C \_VWC\_3 \_EC\_3 \_SoilTemp\_3(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_3,ieee4,SM2C\_VWC\_3=NAN)

Average(1,SM2C\_EC\_3,ieee4,SM2C\_EC\_3=NAN)

Average(1,SM2C\_SoilTemp\_3,fp2,SM2C\_SoilTemp\_3=NAN)

' SM2C \_VWC\_4 \_EC\_4 \_SoilTemp\_4(GBON QC not performed on Soiltemp)

Average(1,SM2C\_VWC\_4,ieee4,SM2C\_VWC\_4=NAN)

Average(1,SM2C\_EC\_4,ieee4,SM2C\_EC\_4=NAN)

Average(1,SM2C\_SoilTemp\_4,fp2,SM2C\_SoilTemp\_4=NAN)

' GPS16XHVS\_GPS

Totalize(1,GPS16XHVS\_GPS\_Ready,Long,false)

Sample(1,GPS16XHVS\_GPS\_ClockChanges,Long,false)

Sample(1,GPS16XHVS\_GPS\_Latitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Longitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Elevation,ieee4)

' MPPT4A\_SolarCharger

' Values from charger

Average(1,MPPT4A\_PanelVoltage,fp2, MPPT4A\_PanelVoltage=NAN)

Average(1,MPPT4A\_PanelCurrent,fp2,MPPT4A\_PanelCurrent =NAN)

Average(1,MPPT4A\_PanelPower,fp2, MPPT4A\_PanelPower=NAN)

Average(1,MPPT4A\_LoadVoltage,fp2, MPPT4A\_LoadVoltage=NAN)

Average(1,MPPT4A\_LoadCurrent,fp2, MPPT4A\_LoadCurrent=NAN)

Average(1,MPPT4A\_BatteryVoltage,fp2,MPPT4A\_BatteryVoltage=NAN)

Average(1,MPPT4A\_BoardTemp ,fp2, MPPT4A\_BoardTemp=NAN)

Sample(1,MPPT4A\_State,fp2)

'-Logger internal

Sample(1,LoggerSerialNumber,String)

Sample(1,ProgramName,String)

Sample(1,ProgramSignature,String)

Average (1,LoggerBattery,FP2,False)

Average (1,LoggerTemp,FP2,False)

Average(1,LoggerLithiumBatt,fp2,false)

Average(1,PingTime,ieee4,PingTime=0)

Totalize(1,1,ieee4,false)

FieldNames("ScanCount")

EndTable

DataTable(TableDay, True, -1) 'Data table for daily records

DataInterval(0, 1440, min, -1)

'-Memory card

'CardOut (0,-1 )

'-JobID & StationID

Sample(1,JobID,String)

Sample(1,StationID,String)

' Thies\_WSpd(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Maximum (1,Thies\_WSpd3s,IEEE4,Thies\_WSpd3s = NAN,False)

Average (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

StdDev (1,Thies\_WSpd,IEEE4,Thies\_WSpd = NAN)

Totalize(1,Thies\_WSpd\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_JumpError,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WSpd2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' Thies\_WDir(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

WindVector(1,1,Thies\_WDir,ieee4,False,0,0,4)

FieldNames("Thies\_WDir\_Avg,Thies\_WDir\_Std")

Totalize(1,Thies\_WDir\_StuckSuspected,fp2,TimeIsBetween(0,119,120,sec))

Totalize(1,Thies\_WDir2m\_Incomplete,fp2,TimeIsBetween(0,119,120,sec))

' TV10 \_AirTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum(1,TV10\_AirTemp,ieee4,TV10\_AirTemp = NAN,true)

Maximum(1,TV10\_AirTemp,ieee4,TV10\_AirTemp = NAN,true)

Average(1,TV10\_AirTemp,ieee4,TV10\_AirTemp = NAN)

Totalize(1,TV10\_AirTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,TV10\_AirTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' EE181 \_RH(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum(1,EE181\_RH,ieee4,EE181\_RH = NAN,true)

Maximum(1,EE181\_RH,ieee4,EE181\_RH = NAN,true)

Average(1,EE181\_RH,ieee4,EE181\_RH = NAN)

Totalize(1,EE181\_RH\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,EE181\_RH\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' Calculated \_DewPointTemp(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum(1,Calculated\_DewPointTemp,ieee4,Calculated\_DewPointTemp = NAN,true)

Maximum(1,Calculated\_DewPointTemp,ieee4,Calculated\_DewPointTemp = NAN,true)

Average(1,Calculated\_DewPointTemp,ieee4,Calculated\_DewPointTemp = NAN)

Totalize(1,Calculated\_DewPointTemp\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,Calculated\_DewPointTemp\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RMY52203\_Rain(GBON QC not performed on rain)

'..Total

Totalize(1,RMY52203\_Rain,fp2,false)

'..Intensity

Minimum(1,RMY52203\_RainIntensity1M,fp2,false,true)

Maximum(1,RMY52203\_RainIntensity1M,fp2,false,true)

Average(1,RMY52203\_RainIntensity1M,fp2,false)

'..Running totals

Maximum(1,RMY52203\_Rain1M,fp2,false,true)

Maximum(1,RMY52203\_Rain1H,fp2,false,true)

Maximum(1,,RMY52203\_Rain3H,fp2,false,true)

Maximum(1,RMY52203\_Rain24H,fp2,false,true)

' BV10 \_BPress(GBON QC counts in a regular table, filtering is not performed in the regular tables and only flags logged)

Minimum(1,BV10\_BPress,ieee4,BV10\_BPress = NAN,true)

Maximum(1,BV10\_BPress,ieee4,BV10\_BPress = NAN,true)

Average(1,BV10\_BPress,ieee4,BV10\_BPress = NAN)

Average(1,BV10\_BPTemp,ieee4,BV10\_BPTemp=NAN)

Average(1,BV10\_QFE,IEEE4,BV10\_QFE=NAN)

Minimum(1,BV10\_QNH,IEEE4,BV10\_QNH=NAN,true)

Maximum(1,BV10\_QNH,IEEE4,BV10\_QNH=NAN,true)

Average(1,BV10\_QNH,IEEE4,BV10\_QNH=NAN)

Average(1,BV10\_QNH\_ICAO,IEEE4,BV10\_QNH\_ICAO=NAN)

Average(1,BV10\_QFF,IEEE4,BV10\_QFF=NAN)

Totalize(1,BV10\_Bpress\_StuckSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpSuspected,fp2,TimeIsBetween(0,59,60,sec))

Totalize(1,BV10\_BPress\_JumpError,fp2,TimeIsBetween(0,59,60,sec))

' RSG2STDN\_SlrW \_SlrMJ

Maximum(1,RSG2STDN\_SlrW,ieee4,RSG2STDN\_SlrW=NAN,true) 'Exclude NANs, Include time of peak

Average(1,RSG2STDN\_SlrW,ieee4,RSG2STDN\_SlrW=NAN)'Excluded NANs

Totalize(1,RSG2STDN\_SlrMJ,ieee4,RSG2STDN\_SlrMJ=NAN)'Excluded NANs

'-Solpos

Sample (1,Sun\_Azimuth,FP2)

Sample (1,Sun\_Elevation,FP2)

'-Sun\_Hours

Totalize(1,Sun\_Hours,ieee4,false)

' SM2C \_VWC\_1 \_EC\_1 \_SoilTemp\_1

Minimum(1,SM2C\_VWC\_1,ieee4,SM2C\_VWC\_1=NAN,true)

Maximum(1,SM2C\_VWC\_1,ieee4,SM2C\_VWC\_1=NAN,true)

Average(1,SM2C\_VWC\_1,ieee4,SM2C\_VWC\_1=NAN)

Minimum(1,SM2C\_EC\_1,ieee4,SM2C\_EC\_1=NAN,true)

Maximum(1,SM2C\_EC\_1,ieee4,SM2C\_EC\_1=NAN,true)

Average(1,SM2C\_EC\_1,ieee4,SM2C\_EC\_1=NAN)

Minimum(1,SM2C\_SoilTemp\_1,fp2,SM2C\_SoilTemp\_1=NAN,true)

Maximum(1,SM2C\_SoilTemp\_1,fp2,SM2C\_SoilTemp\_1=NAN,true)

Average(1,SM2C\_SoilTemp\_1,fp2,SM2C\_SoilTemp\_1=NAN)

' SM2C \_VWC\_2 \_EC\_2 \_SoilTemp\_2

Minimum(1,SM2C\_VWC\_2,ieee4,SM2C\_VWC\_2=NAN,true)

Maximum(1,SM2C\_VWC\_2,ieee4,SM2C\_VWC\_2=NAN,true)

Average(1,SM2C\_VWC\_2,ieee4,SM2C\_VWC\_2=NAN)

Minimum(1,SM2C\_EC\_2,ieee4,SM2C\_EC\_2=NAN,true)

Maximum(1,SM2C\_EC\_2,ieee4,SM2C\_EC\_2=NAN,true)

Average(1,SM2C\_EC\_2,ieee4,SM2C\_EC\_2=NAN)

Minimum(1,SM2C\_SoilTemp\_2,fp2,SM2C\_SoilTemp\_2=NAN,true)

Maximum(1,SM2C\_SoilTemp\_2,fp2,SM2C\_SoilTemp\_2=NAN,true)

Average(1,SM2C\_SoilTemp\_2,fp2,SM2C\_SoilTemp\_2=NAN)

' SM2C \_VWC\_3 \_EC\_3 \_SoilTemp\_3

Minimum(1,SM2C\_VWC\_3,ieee4,SM2C\_VWC\_3=NAN,true)

Maximum(1,SM2C\_VWC\_3,ieee4,SM2C\_VWC\_3=NAN,true)

Average(1,SM2C\_VWC\_3,ieee4,SM2C\_VWC\_3=NAN)

Minimum(1,SM2C\_EC\_3,ieee4,SM2C\_EC\_3=NAN,true)

Maximum(1,SM2C\_EC\_3,ieee4,SM2C\_EC\_3=NAN,true)

Average(1,SM2C\_EC\_3,ieee4,SM2C\_EC\_3=NAN)

Minimum(1,SM2C\_SoilTemp\_3,fp2,SM2C\_SoilTemp\_3=NAN,true)

Maximum(1,SM2C\_SoilTemp\_3,fp2,SM2C\_SoilTemp\_3=NAN,true)

Average(1,SM2C\_SoilTemp\_3,fp2,SM2C\_SoilTemp\_3=NAN)

' SM2C \_VWC\_4 \_EC\_4 \_SoilTemp\_4

Minimum(1,SM2C\_VWC\_4,ieee4,SM2C\_VWC\_4=NAN,true)

Maximum(1,SM2C\_VWC\_4,ieee4,SM2C\_VWC\_4=NAN,true)

Average(1,SM2C\_VWC\_4,ieee4,SM2C\_VWC\_4=NAN)

Minimum(1,SM2C\_EC\_4,ieee4,SM2C\_EC\_4=NAN,true)

Maximum(1,SM2C\_EC\_4,ieee4,SM2C\_EC\_4=NAN,true)

Average(1,SM2C\_EC\_4,ieee4,SM2C\_EC\_4=NAN)

Minimum(1,SM2C\_SoilTemp\_4,fp2,SM2C\_SoilTemp\_4=NAN,true)

Maximum(1,SM2C\_SoilTemp\_4,fp2,SM2C\_SoilTemp\_4=NAN,true)

Average(1,SM2C\_SoilTemp\_4,fp2,SM2C\_SoilTemp\_4=NAN)

' GPS16XHVS\_GPS

Totalize(1,GPS16XHVS\_GPS\_Ready,Long,false)

Sample(1,GPS16XHVS\_GPS\_ClockChanges,Long,false)

Sample(1,GPS16XHVS\_GPS\_Latitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Longitude,ieee4)

Sample(1,GPS16XHVS\_GPS\_Elevation,ieee4)

' MPPT4A\_SolarCharger

' Values from charger

Maximum(1,MPPT4A\_PanelVoltage,fp2, MPPT4A\_PanelVoltage=NAN,false)

Minimum(1,MPPT4A\_LoadCurrent,fp2, MPPT4A\_LoadCurrent=NAN,false)

Maximum(1,MPPT4A\_LoadCurrent,fp2, MPPT4A\_LoadCurrent=NAN,false)

Average(1,MPPT4A\_LoadCurrent,fp2, MPPT4A\_LoadCurrent=NAN)

'-Logger internal

Sample(1,LoggerSerialNumber,String)

Sample(1,ProgramName,String)

Sample(1,ProgramSignature,String)

Minimum(1,LoggerBattery,ieee4,false,true) 'Exclude NANs, Include time of peak

Maximum(1,LoggerBattery,ieee4,false,true) 'Exclude NANs, Include time of peak

Average (1,LoggerBattery,FP2,False)

Minimum(1,LoggerTemp,ieee4,false,true) 'Exclude NANs, Include time of peak

Maximum(1,LoggerTemp,ieee4,false,true) 'Exclude NANs, Include time of peak

Average (1,LoggerTemp,FP2,False)

Average(1,LoggerLithiumBatt,fp2,false)

Average(1,PingTime,ieee4,PingTime=0)

Totalize(1,1,ieee4,false)

FieldNames("ScanCount")

EndTable

DataTable(SettingsBackup,true,365)

'-JobID & StationID

Sample(1,JobID,String)

Sample(1,StationID,String)

' Location

Sample(1,Latitude,ieee4)

Sample(1,Longitude,ieee4)

Sample(1,Elevation,ieee4)

Sample(1,UTC\_Offset,ieee4)

' PTB330\_BPress

Sample(1,BV10\_hQFE,ieee4)

Sample(1,BV10\_tstdQFE,ieee4)

Sample(1,BV10\_hQNH,ieee4)

' TV10\_AirTemp

Sample(1,TV10\_AirTemp\_Height,fp2)

Sample(1,TV10\_AirTemp\_Cal,ieee4)

'Thies\_WSpd

Sample(1,Thies\_WSpd\_Height,fp2)

' RMY52203\_Rain

Sample(1,RMY52203\_Rain\_Height,fp2)

' RSG2STDN\_SlrW \_SlrMJ

Sample(1,RSG2STDN\_SlrW\_SN, String)

Sample(1,RSG2STDN\_SlrW\_Sensitivity,ieee4)

' Synop

Sample(1,WMO\_Block,String)

Sample(1,WMO\_Station\_ID,String)

Sample(1,WMO\_Station\_Name,String)

Sample(1,WMO\_Station\_Type,String)

EndTable

DataTable(SYNOP,True,-1)

DataInterval (0,60,Min,10)

'DataInterval (0,1,Min,-1) ' Test

'Station Info

Sample(1,WMO\_Block,FP2) ' WMO block number

Sample(1,WMO\_Station\_ID,String) : FieldNames("Station\_ID") ' WMO station number

Sample(1,WMO\_Station\_Name,String) : FieldNames("Station\_Name") ' Station or site name

Sample(1,WMO\_Station\_Type,String) ' Type of station, 0 automatic, 1 manned, 2 hybrid

Sample(1,UYear,Long) : FieldNames("M\_Year") ' UTC year

Sample(1,UMonth,Long) : FieldNames("M\_Month") ' UTC month

Sample(1,UDay,Long) : FieldNames("M\_DayOfMonth") ' UCT day

Sample(1,UHour,Long) : FieldNames("M\_HourOfDay") ' UCT hour

Sample(1,UMinute,Long) : FieldNames("M\_Minutes") ' UCT minute

Sample(1,Latitude,FP2) ' Latitude

Sample(1,Longitude,FP2) ' Longitude

Sample(1,Elevation,FP2) ' Elevation

' Pressure

Sample(1,BPress\_Elevation,FP2) : FieldNames("BP\_Elevation") ' Height of barometer above sea level and ground level, units metre

Sample(1,BPressPa1M,IEEE4) : FieldNames("BP") ' Current barometric pressure, units Pa

Sample(1,QNHPa1M,IEEE4) : FieldNames("QNH") ' Current sealevel adjusted barometric pressure, units Pa

Sample(1,BPressPaChange3H,IEEE4) : FieldNames("BP\_Change") ' Previous 3 hourly pressure change, units Pa

Sample(1,BPressTendency3H,FP2) : FieldNames("BP\_Tendency") ' Previous 3 hourly pressure tendency

' Air Temperature

Sample(1,AirTemp\_Height,FP2) : FieldNames("Temp\_Height") ' Height of Temp sensor, units metre

Sample(1,AirTempK1M,FP2) : FieldNames("AirTempK") ' Current temp, units K

' Dewpoint temperature

Sample(1,DewPointTempK10M,FP2) : FieldNames("DewPointTempK") ' Current DewPointTemp, units K

' Relative humidity

Sample(1,RH1M,FP2) : FieldNames("RH") ' Current RH, units %

' Rain Intensity

Sample(1,RainIntensity1M,ieee4): FieldNames("RainIntensity") ' Current rain intensity, units mm/h

' Wind

Sample(1,WSpd\_Height,FP2) : FieldNames("Wind\_Height") ' Height of Wind sensor, units metre

Sample(1,WSpd\_SensorType,String) : FieldNames("Wind\_Type") ' Sensor type

Sample(1,WSpd\_ReportTimeSignificance,FP2) : FieldNames("Wind\_Sig") ' Time significance 2 time averaged

Sample(1,WSpd\_ReportPeriod10M,FP2) : FieldNames("Wind\_T") ' Wind reporting period in minutes

Sample(1,WDir10m,FP2) : FieldNames("WindDir") ' Current wind direction, units degrees

Sample(1,WSpd10m,FP2) : FieldNames("WSpeed") ' Current wind speed, units m/s

Sample(1,WSpd10mGust,FP2) : FieldNames("WindGust") ' Current wind gust, units m/s

' Air temperature extremes

Sample(1,AirTemp\_ReportPeriod24H,FP2) : FieldNames("Temp\_hr24") ' Temperature reporting period, units hours

Sample(1,AirTemp\_ReportTimeOffset24H,FP2) : FieldNames("Temp24T") ' Temperature reporting time offset, hours

Sample(1,AirTempKMax24H,FP2) : FieldNames("AirTempMaxK") ' Minimum temp in previous 24 hours, units K

Sample(1,AirTempKMin24H,FP2) : FieldNames("AirTempMinK") ' Maximum temp in previous 24 hours, units K

' Rain total

Sample(1,Rain\_Height,FP2) ' Height of Rain sensor, units metre

Sample(1,Rain\_ReportPeriod1H,FP2) : FieldNames("Rain\_hr1") ' Reporting Period, units hours

Sample(1,Rain1H,fp2,False) ' Total rain for the previous 3 hours, units mm

Sample(1,Rain\_ReportPeriod3H,FP2) : FieldNames("Rain\_hr3") ' Reporting Period, units hours

Sample(1,Rain3H,fp2,False) ' Total rain for the hour, units mm

Sample(1,Rain\_ReportPeriod24H,FP2) : FieldNames("Rain\_hr24") ' Reporting Period, units hours

Sample(1,Rain24H,fp2,False) ' Total rain for the previous 24 hours, units mm

' Sun\_Hours

Sample(1,SunHours\_ReportPeriod1H,FP2) : FieldNames("SunHours\_hr1") ' Reporting Period, units hours

Sample(1,SunHours1H,IEEE4,False) ' Total sun hours for the hour,units minutes

Sample(1,SunHours\_ReportPeriod1H,FP2) : FieldNames("SunHours\_hr24") ' Reporting Period, units hours

Sample(1,SunHours24H,IEEE4,False) ' Total sun hours for the previous 24 hours,units minutes

'Solar radiation

Sample(1,SlrJ\_ReportPeriod1H,FP2) : FieldNames("SlrJ\_hr1") ' Reporting Period, units hours

Sample(1,SlrJ1H,IEEE4,False) ' Total solar radiation for the hour,units J/m^2

Sample(1,SlrJ\_ReportPeriod1H,FP2) : FieldNames("SlrJ\_hr24") ' Reporting Period, units hours

Sample(1,SlrJ24H,IEEE4,False) ' Total solar ratiation for the previous 24 hours, units J/m^2

' BPress(GBON QC flags)

'Sample(1,BPress\_StuckSuspected,fp2)

'Sample(1,BPress\_JumpSuspected,fp2)

'Sample(1,BPress\_JumpError,fp2)

' AirTemp(GBON QC flags)

'Sample(1,AirTemp\_StuckSuspected,fp2)

'Sample(1,AirTemp\_JumpSuspected,fp2)

'Sample(1,AirTemp\_JumpError,fp2)

' RH(GBON QC flags)

'Sample(1,RH\_StuckSuspected,fp2)

'Sample(1,RH\_JumpSuspected,fp2)

'Sample(1,RH\_JumpError,fp2)

'DewPointTemp(GBON QC flags)

'Sample(1,DewPointTemp\_StuckSuspected,fp2)

'Sample(1,DewPointTemp\_JumpSuspected,fp2)

'Sample(1,DewPointTemp\_JumpError,fp2)

' WSpd(GBON QC flags)

'Sample(1,WSpd\_StuckSuspected,fp2)

'Sample(1,WSpd10m\_JumpSuspected,fp2)

'Sample(1,WSpd10m\_JumpError,fp2)

'Sample(1,WSpd10m\_Incomplete,fp2)

' WDir(GBON QC flags)

'Sample(1,WDir\_StuckSuspected,fp2)

'Sample(1,WDir10m\_Incomplete,fp2)

' MQTT publish to WIS2

MQTTPublishTable (0,60,60,Min,2,Longitude,Latitude,Elevation)

EndTable

'-Main Program---------------------------------------------------------------

BeginProg

' Default Datalogger Battery Voltage at startup

Battery(LoggerBattery)

'-Open serial port to SM2C \_VWC \_EC

SerialOpen (ComC5, 9600,3, 0, 100, 3)

' Power modem on power up of logger

IPNetPower (5,1)

' Power SW12\_1 on power up of logger

SW12(SW12\_1,1)

' Power SW12\_2 on power up of logger

SW12(SW12\_2,1)

' Power up Thies\_WDir on logger startup

SWVX(VX1,1,1)

' Read station Identification from logger

Call StationID\_Read

' Read Location Settings from logger

Call Location\_Read

' Read TV10\_AirTemp settings and calibration

Call TV10\_AirTemp\_ReadSettings

' Read RMY52203\_Rain settings

Call RMY52203\_Rain\_ReadSettings

' Read Thies\_WSpd settings

Call Thies\_WSpd\_ReadSettings

' Read BV10\_BPress Settings

Call BV10\_ReadSettings

' Read RSG2STDN\_SlrW calibration

Call RSG2STDN\_SlrW\_ReadSettings

' Read SYNOP Settings from logger

Call SYNOP\_Read

' Wait for sensors to power up

Delay(1,10,Sec)

'-Main Scan----------------------------------------------------------------

Scan(1, Sec, 1, 0) ' Do not change

'-Starting time of processing

Timer (1,mSec,2)

'-Logger internal--------------------------------------------------------

LoggerSerialNumber = Status.SerialNumber

ProgramName = Status.Progname

ProgramSignature = Status.ProgSignature

Battery(LoggerBattery)

LoggerLithiumBatt = Status.LithiumBattery

PanelTemp(LoggerTemp,50)

RealTime( rTime)

CardStatus = status.cardstatus

WatchdogErrors = status.watchdogerrors

' Modem Status

CellState = Status.CellState

CellSignalStrength=Status.CellRSRP

CellSignalQuality=Status.CellRSRQ

CellInfoRaw=Status.CellInfo

SplitStr(CellInfo(),CellInfoRaw,CHR(13)&CHR(10),15,5)

RealTime( rTime)

'-Solar Position---------------------------------------------------------

If AirTemp <> NAN Then

SolarPosition(SolPos(),rTime(),UTC\_Offset\*3600,Latitude,Longitude,Elevation,-1,AirTemp)

Else

SolarPosition(SolPos(),rTime(),UTC\_Offset\*3600,Latitude,Longitude,Elevation,-1,LoggerTemp)

EndIf

Sun\_Azimuth = SolPos(1)

Sun\_Elevation = SolPos(2)

If Sun\_Elevation > 0.27 Then Sun\_Up = True Else Sun\_Up = False

' GPS16XHVS--------------------------------------------------------------

GPS (GPS16XHVS\_GPS\_Data(1),ComC1,GPS16XHVS\_GPS\_UTCOffset\*3600,100,GPS16XHVS\_GPS\_NmeaSentence(1))

GPS16XHVS\_GPS\_LatDD = GPS16XHVS\_GPS\_Data(1)

GPS16XHVS\_GPS\_Lat\_MM\_MMMM = GPS16XHVS\_GPS\_Data(2)

GPS16XHVS\_GPS\_Latitude = GPS16XHVS\_GPS\_LatDD +GPS16XHVS\_GPS\_Lat\_MM\_MMMM/60

GPS16XHVS\_GPS\_LonDDD = GPS16XHVS\_GPS\_Data(3)

GPS16XHVS\_GPS\_Lon\_MM\_MMMM = GPS16XHVS\_GPS\_Data(4)

GPS16XHVS\_GPS\_Longitude = GPS16XHVS\_GPS\_LonDDD + GPS16XHVS\_GPS\_Lon\_MM\_MMMM/60

GPS16XHVS\_GPS\_Elevation = GPS16XHVS\_GPS\_Data(10)

If GPS16XHVS\_GPS\_Data(8) = 0 Then GPS16XHVS\_GPS\_FixInvalid = 1 Else GPS16XHVS\_GPS\_FixInvalid = 0

If GPS16XHVS\_GPS\_Data(8) = 1 Then GPS16XHVS\_GPS\_FixGPS = 1 Else GPS16XHVS\_GPS\_FixGPS = 0

If GPS16XHVS\_GPS\_Data(8) = 2 Then GPS16XHVS\_GPS\_FixDiffGPS = 1 Else GPS16XHVS\_GPS\_FixDiffGPS = 0

If GPS16XHVS\_GPS\_Data(8) = 3 Then GPS16XHVS\_GPS\_FixEstimated = 1 Else GPS16XHVS\_GPS\_FixEstimated = 0

GPS16XHVS\_GPS\_Ready = GPS16XHVS\_GPS\_Data(13)

GPS16XHVS\_GPS\_ClockChanges = GPS16XHVS\_GPS\_Data(15)

' SYNOP------------------------------------------------------------------

' UTC Time

UTC\_Offset\_Long = UTC\_Offset

SSLocal=SecsSince1990(Public.Timestamp,1)

SSUTC= SSLocal - (UTC\_Offset \* 3600)

SSUTCDateString = SecsSince1990(SSUTC,1)

UYear = Mid (SSUTCDateString,7,4)

UYearStr = UYear

UMonth = Left(SSUTCDateString,2)

If UMonth < 10 Then UMonthStr = "0" & UMonth Else UMonthStr = UMonth

UDay= Mid (SSUTCDateString,4,2)

If UDay < 10 Then UDayStr = "0" & UDay Else UDayStr = UDay

UHour = Mid (SSUTCDateString,12,2)

If UHour < 10 Then UHourStr = "0" & UHour Else UHourStr = UHour

UMinute = Mid (SSUTCDateString,15,2)

If UMinute < 10 Then UMinuteStr = "0" & UMinute Else UMinuteStr = UMinute

' Thies\_WSpd-------------------------------------------------------------

PulseCount( Thies\_WSpd,1,P1,3,1,0.0462,0.21)

If Thies\_WSpd <= 0.21 Then Thies\_WSpd = 0

AvgRun(Thies\_WSpd3s,1,Thies\_WSpd,3)

'Thies\_WSpd\_Raw = 10\*RND

' QC

' Sample Delta

Thies\_WSpd\_Delta = ABS(Thies\_WSpd - Thies\_WSpd\_Prev)

Thies\_WSpd\_Prev = Thies\_WSpd

' 3s

AvgRun(Thies\_WSpd3sec,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,NAN, Thies\_WSpd),3)

TotalRun(Thies\_WSpd3secN,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,0,1),3)

' 120s

AvgRun(Thies\_WSpd120s,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,NAN, Thies\_WSpd),120)

TotalRun(Thies\_WSpd120sN,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,0,1),120)

Thies\_WSpd2m = Thies\_WSpd120s ' Assign 2 minute running avarage to 2 minute variable

If Thies\_WSpd120sN / 120 > 0.75 Then Thies\_WSpd2m\_Incomplete = 0 Else Thies\_WSpd2m\_Incomplete = 1' Check for 75% minimum amount of values

' 120s 3s gust

MaxRun(Thies\_WSpd120s3secMax,1,IIF(Thies\_WSpd3secN < 3 OR Thies\_WSpd3sec = NAN,NAN,Thies\_WSpd3sec),120)

TotalRun(Thies\_WSpd120s3secMaxN,1,IIF(Thies\_WSpd3secN < 3 OR Thies\_WSpd3sec = NAN,0,1),120)

Thies\_WSpd2mGust = Thies\_WSpd120s3secMax ' Assign 2 minute maximum 3s running avarage to 2 minute gust variable

' 600s

AvgRun(Thies\_WSpd600s,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,NAN, Thies\_WSpd),600)

TotalRun(Thies\_WSpd600sN,1,IIF(Thies\_WSpd\_Delta > 20 OR Thies\_WSpd = NAN,0,1),600)

Thies\_WSpd10m = Thies\_WSpd600s ' Assign 10 minute running avarage to 10 minute variable

If Thies\_WSpd600sN / 600 > 0.75 Then Thies\_WSpd10m\_Incomplete = 0 Else Thies\_WSpd10m\_Incomplete = 1' Check for 75% minimum amount of values

' 600s 3s gust

MaxRun(Thies\_WSpd600s3secMax,1,IIF(Thies\_WSpd3secN < 3 OR Thies\_WSpd3sec = NAN,NAN,Thies\_WSpd3sec),600)

TotalRun(Thies\_WSpd600s3secMaxN,1,IIF(Thies\_WSpd3secN < 3 OR Thies\_WSpd3sec = NAN,0,1),600)

Thies\_WSpd10mGust = Thies\_WSpd600s3secMax ' Assign 10 minute maximum 3s running avarage to 10 minute gust variable

' 60m(based on the 2 minute averages)

' Initialize 60 minute buffer

If Thies\_WSpdStartup = 0 Then

Move(Thies\_WSpd60mBuffer(),30,NAN,1)

Thies\_WSpdStartup = 1

EndIf

If TimeIntoInterval(0,120,sec) Then

' Advance 60 minute buffer

For Thies\_WSpdBufferCount = 30 To 2 Step -1

Thies\_WSpd60mBuffer(Thies\_WSpdBufferCount) = Thies\_WSpd60mBuffer(Thies\_WSpdBufferCount-1)

Next Thies\_WSpdBufferCount

Thies\_WSpd60mBuffer(1) = Thies\_WSpd120s

' Stuck sensor check over previous 60 minutes

MinSpa (Thies\_WSpd60mBufferMin(),30,Thies\_WSpd60mBuffer())

MaxSpa (Thies\_WSpd60mBufferMax(),30,Thies\_WSpd60mBuffer())

Thies\_WSpd60mDelta = ABS(Thies\_WSpd60mBufferMin(1) - Thies\_WSpd60mBufferMax(1))

If Thies\_WSpd60mDelta < 0.5 OR Thies\_WSpd60mDelta = NAN Then Thies\_WSpd\_StuckSuspected = 1 Else Thies\_WSpd\_StuckSuspected = 0

' Jump sensor check over consecutive 2-minute values for the 2-minute period

Thies\_WSpd2m\_JumpSuspected = 0

Thies\_WSpd2m\_JumpError = 0

Thies\_WSpd2mDelta=ABS(Thies\_WSpd60mBuffer(1) - Thies\_WSpd60mBuffer(2))

Select Case Thies\_WSpd2mDelta

Case Is > 10 AND <=20

Thies\_WSpd2m\_JumpSuspected = 1

Case Is > 20

Thies\_WSpd2m\_JumpError = 1

Case Is = NAN

Thies\_WSpd2m\_JumpError = 1

EndSelect

' Jump sensor check over consecutive 2-minute values for the 10-minute period

Thies\_WSpd10m\_JumpSuspected = 0

Thies\_WSpd10m\_JumpError = 0

For Thies\_WSpdBufferCount = 1 To 4

Thies\_WSpd2mDelta=ABS(Thies\_WSpd60mBuffer(Thies\_WSpdBufferCount) - Thies\_WSpd60mBuffer(Thies\_WSpdBufferCount+1))

Select Case Thies\_WSpd2mDelta

Case Is > 10 AND <=20

Thies\_WSpd10m\_JumpSuspected = 1

Case Is > 20

Thies\_WSpd10m\_JumpError = 1

Case Is = NAN

Thies\_WSpd10m\_JumpError = 1

EndSelect

Next Thies\_WSpdBufferCount

EndIf

' Thies\_WDir-------------------------------------------------------------

VoltSe(Thies\_WDir,1,mV5000,1,true,0,50,0.072,0) ' Continuously power with VX channel (SWVX instruction at startup) and read with SE channel (0..input voltage = 0..360 deg)

If Thies\_WDir>=360 OR Thies\_WDir<0 Then Thies\_WDir=0

Thies\_WDir = (Thies\_WDir + 0) MOD 360 ' In case a correction has to be added

'Thies\_WDir = 180 + 100\*RND

' QC

' 120s

AvgRun(Thies\_WDirSin120s,1,IIF(Thies\_WDir = NAN,NAN,SIN(Thies\_WDir)),120)

AvgRun(Thies\_WDirCos120s,1,IIF(Thies\_WDir = NAN,NAN,COS(Thies\_WDir)),120)

TotalRun(Thies\_WDir120sN,1,IIF(Thies\_WDir = NAN,0,1),120)

Thies\_WDir120s = ATN(Thies\_WDirSin120s/Thies\_WDirCos120s)

If Thies\_WDirCos120s < 0 Then

Thies\_WDir120s = (180 + Thies\_WDir120s) MOD 360

ElseIf Thies\_WDirSin120s < 0 Then

Thies\_WDir120s = (360 + Thies\_WDir120s) MOD 360

EndIf

Thies\_WDir2m = Thies\_WDir120s ' Assign 2 minute running avarage to 2 minute variable

If Thies\_WDir120sN / 120 > 0.75 Then Thies\_WDir2m\_Incomplete = 0 Else Thies\_WDir2m\_Incomplete = 1' Check for 75% minimum amount of values

' 600s

AvgRun(Thies\_WDirSin600s,1,IIF(Thies\_WDir = NAN,NAN, SIN(Thies\_WDir)),600)

AvgRun(Thies\_WDirCos600s,1,IIF(Thies\_WDir = NAN,NAN, COS(Thies\_WDir)),600)

TotalRun(Thies\_WDir600sN,1,IIF(Thies\_WDir = NAN,0,1),600)

Thies\_WDir600s = ATN(Thies\_WDirSin600s/Thies\_WDirCos600s)

If Thies\_WDirCos600s < 0 Then

Thies\_WDir600s = (180 + Thies\_WDir600s) MOD 360

ElseIf Thies\_WDirSin600s < 0 Then

Thies\_WDir600s = (360 + Thies\_WDir600s) MOD 360

EndIf

Thies\_WDir10m = Thies\_WDir600s ' Assign 10 minute running avarage to 10 minute variable

If Thies\_WDir600sN / 600 > 0.75 Then Thies\_WDir10m\_Incomplete = 0 Else Thies\_WDir10m\_Incomplete = 1' Check for 75% minimum amount of values

' 60m(based on the 2 minute averages)

' Initialize 60 minute buffers

If Thies\_WDirStartup = 0 Then

Move(Thies\_WDir60mBuffer(),30,NAN,1)

Move(Thies\_WDir60mOffsetBuffer(),30,NAN,1)

Thies\_WDirStartup = 1

EndIf

If TimeIntoInterval(0,120,sec) Then

' Advance 60 minute buffer

For Thies\_WDirBufferCount = 30 To 2 Step -1

Thies\_WDir60mBuffer(Thies\_WDirBufferCount) = Thies\_WDir60mBuffer(Thies\_WDirBufferCount-1)

Next Thies\_WDirBufferCount

Thies\_WDir60mBuffer(1) = Thies\_WDir120s

' Stuck sensor check over previous 60 minutes

' First find mean

Thies\_WDirSin60mAccum = 0

Thies\_WDirCos60mAccum = 0

Thies\_WDir60mN = 0

For Thies\_WDirBufferCount = 1 To 30

If Thies\_WDir60mBuffer(Thies\_WDirBufferCount) <> NAN Then

Thies\_WDirSin60mAccum += SIN(Thies\_WDir60mBuffer(Thies\_WDirBufferCount))

Thies\_WDirCos60mAccum += COS(Thies\_WDir60mBuffer(Thies\_WDirBufferCount))

Thies\_WDir60mN +=1

EndIf

Next Thies\_WDirBufferCount

If Thies\_WDir60mN > 0 Then

Thies\_WDirSin60m = Thies\_WDirSin60mAccum / Thies\_WDir60mN

Thies\_WDirCos60m = Thies\_WDirCos60mAccum / Thies\_WDir60mN

Thies\_WDir60m = ATN(Thies\_WDirSin60m/Thies\_WDirCos60m)

If Thies\_WDirCos60m < 0 Then

Thies\_WDir60m = (180 + Thies\_WDir60m) MOD 360

ElseIf Thies\_WDirSin60m < 0 Then

Thies\_WDir60m = (360 + Thies\_WDir60m) MOD 360

EndIf

Else

Thies\_WDir60m = NAN

EndIf

' Calculate offset from 0 to normalize all values relative to 0

Thies\_WDir60mOffset = (0 + Thies\_WDir60m) MOD 360

' Find minimum back and maximum veer angles

Thies\_WDir60mBackMin = 360

Thies\_WDir60mVeerMax = 0

For Thies\_WDirBufferCount = 1 To 30

If Thies\_WDir60mBuffer(Thies\_WDirBufferCount) <> NAN AND Thies\_WDir60mOffset <> NAN Then

Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) = (Thies\_WDir60mBuffer(Thies\_WDirBufferCount)-Thies\_WDir60mOffset) MOD 360

If Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) < 0 Then Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) = (Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) + 360) MOD 360

' back

If Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) >= 180 AND Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) < 360 Then

If Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) < Thies\_WDir60mBackMin Then Thies\_WDir60mBackMin =Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount)

EndIf

' veer

If Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) > 0 AND Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) <= 180 Then

If Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount) > Thies\_WDir60mVeerMax Then Thies\_WDir60mVeerMax = Thies\_WDir60mOffsetBuffer(Thies\_WDirBufferCount)

EndIf

EndIf

Next Thies\_WDirBufferCount

' find variance

Thies\_WDir60mVariance = 360 - Thies\_WDir60mBackMin + Thies\_WDir60mVeerMax

' rotate back to original offset from 0

Thies\_WDir60mBackMin = (Thies\_WDir60mBackMin + Thies\_WDir60mOffset) MOD 360

Thies\_WDir60mVeerMax = (Thies\_WDir60mVeerMax + Thies\_WDir60mOffset) MOD 360

If Thies\_WDir60mVariance < 10 OR Thies\_WDir60mVariance = NAN Then Thies\_WDir\_StuckSuspected = 1 Else Thies\_WDir\_StuckSuspected = 0

EndIf

' TV10\_AirTemp-----------------------------------------------------------

BrHalf4W(TV10\_AirTemp\_Ratio,1,mV200C,mV5000,2,Vx2,1,2100,True,True,50000,50,TV10\_AirTemp\_Cal,0)

PRTCalc(TV10\_AirTemp,1,TV10\_AirTemp\_Ratio,1,1,0)

' QC

' 60s

TV10\_AirTemp\_Delta = ABS(TV10\_AirTemp - TV10\_AirTemp\_Prev)

AvgRun(TV10\_AirTemp60s,1,IIF(TV10\_AirTemp\_Delta > 2 OR TV10\_AirTemp = NAN,nan, TV10\_AirTemp),60)

TotalRun(TV10\_AirTemp60sN,1,IIF(TV10\_AirTemp\_Delta > 2 OR TV10\_AirTemp = NAN,0,1),60)

TV10\_AirTemp\_Prev = TV10\_AirTemp

' Initialize 1440 minute buffers

If TV10\_AirTempStartup = 0 Then

Move(TV10\_AirTemp1440mBuffer(),1440,NAN,1)

Move(TV10\_AirTempTime1440mBuffer(),1440,0,1)

TV10\_AirTempStartup = 1

EndIf

' 1440m

If TimeIntoInterval(0,60,sec) Then

' Advance 1440 minute buffer

For TV10\_AirTempBufferCount = 1440 To 2 Step -1

TV10\_AirTemp1440mBuffer(TV10\_AirTempBufferCount) = TV10\_AirTemp1440mBuffer(TV10\_AirTempBufferCount-1)

TV10\_AirTempTime1440mBuffer(TV10\_AirTempBufferCount) = TV10\_AirTempTime1440mBuffer(TV10\_AirTempBufferCount-1)

Next TV10\_AirTempBufferCount

TV10\_AirTemp1440mBuffer(1) = TV10\_AirTemp60s

TV10\_AirTempTime1440mBuffer(1) = SecsSince1990(Public.Timestamp,1)

TV10\_AirTemp1M = TV10\_AirTemp1440mBuffer(1)

TV10\_AirTempK1M = TV10\_AirTemp1M + 273.15

' Stuck sensor check over previous 60 minutes

MinSpa (TV10\_AirTemp1440mBufferMin(),60,TV10\_AirTemp1440mBuffer())

MaxSpa (TV10\_AirTemp1440mBufferMax(),60,TV10\_AirTemp1440mBuffer())

TV10\_AirTemp60mDelta = ABS(TV10\_AirTemp1440mBufferMin(1) - TV10\_AirTemp1440mBufferMax(1))

If TV10\_AirTemp60mDelta < 0.1 OR TV10\_AirTemp60mDelta = NAN Then TV10\_AirTemp\_StuckSuspected = 1 Else TV10\_AirTemp\_StuckSuspected = 0

' Jump sensor check over last 2 minutes

TV10\_AirTemp\_JumpSuspected = 0

TV10\_AirTemp\_JumpError = 0

TV10\_AirTemp2mDelta=ABS(TV10\_AirTemp1440mBuffer(1) - TV10\_AirTemp1440mBuffer(2))

Select Case TV10\_AirTemp2mDelta

Case Is > 3 AND <=10

TV10\_AirTemp\_JumpSuspected = 1

Case Is > 10

TV10\_AirTemp\_JumpError = 1

Case Is = NAN

TV10\_AirTemp\_JumpError = 1

EndSelect

EndIf

' Min and Max temperature/timestamp over 24hour period

MinSpa (TV10\_AirTemp1440mBufferMin(),1440,TV10\_AirTemp1440mBuffer())

TV10\_AirTempMin24H = TV10\_AirTemp1440mBufferMin(1)

TV10\_AirTempKMin24H = TV10\_AirTempMin24H + 273.15

TV10\_AirTempMinTime24H = SecsSince1990(TV10\_AirTempTime1440mBuffer(TV10\_AirTemp1440mBufferMin(2)),4)

MaxSpa (TV10\_AirTemp1440mBufferMax(),1440,TV10\_AirTemp1440mBuffer())

TV10\_AirTempMax24H = TV10\_AirTemp1440mBufferMax(1)

TV10\_AirTempKMax24H = TV10\_AirTempMax24H + 273.15

TV10\_AirTempMaxTime24H = SecsSince1990(TV10\_AirTempTime1440mBuffer(TV10\_AirTemp1440mBufferMax(2)),4)

' EE181\_RH---------------------------------------------------------------

VoltSe(EE181\_RH,1,mV1000,2,True,0,50,0.1,0)

' QC

' 60s

EE181\_RH\_Delta = ABS(EE181\_RH - EE181\_RH\_Prev)

AvgRun(EE181\_RH60s,1,IIF(EE181\_RH\_Delta > 5 OR EE181\_RH = NAN,nan, EE181\_RH),60)

TotalRun(EE181\_RH60sN,1,IIF(EE181\_RH\_Delta > 5 OR EE181\_RH = NAN,0,1),60)

EE181\_RH\_Prev = EE181\_RH

' Initialize 1440 minute buffers

If EE181\_RHStartup = 0 Then

Move(EE181\_RH1440mBuffer(),1440,NAN,1)

Move(EE181\_RHTime1440mBuffer(),1440,0,1)

EE181\_RHStartup = 1

EndIf

' 1440m

If TimeIntoInterval(0,60,sec) Then

' Advance 1440 minute buffer

For EE181\_RHBufferCount = 1440 To 2 Step -1

EE181\_RH1440mBuffer(EE181\_RHBufferCount) = EE181\_RH1440mBuffer(EE181\_RHBufferCount-1)

EE181\_RHTime1440mBuffer(EE181\_RHBufferCount) = EE181\_RHTime1440mBuffer(EE181\_RHBufferCount-1)

Next EE181\_RHBufferCount

EE181\_RH1440mBuffer(1) = EE181\_RH60s

EE181\_RHTime1440mBuffer(1) = SecsSince1990(Public.Timestamp,1)

EE181\_RH1M = EE181\_RH1440mBuffer(1)

' Stuck sensor check over previous 60 minutes

MinSpa (EE181\_RH1440mBufferMin(),60,EE181\_RH1440mBuffer())

MaxSpa (EE181\_RH1440mBufferMax(),60,EE181\_RH1440mBuffer())

EE181\_RH60mDelta = ABS(EE181\_RH1440mBufferMin(1) - EE181\_RH1440mBufferMax(1))

If EE181\_RH60mDelta < 1 OR EE181\_RH60mDelta = NAN OR EE181\_RH1440mBufferMin(1) >= 95 Then EE181\_RH\_StuckSuspected = 1 Else EE181\_RH\_StuckSuspected = 0

' Jump sensor check over last 2 minutes

EE181\_RH\_JumpSuspected = 0

EE181\_RH\_JumpError = 0

EE181\_RH2mDelta=ABS(EE181\_RH1440mBuffer(1) - EE181\_RH1440mBuffer(2))

Select Case EE181\_RH2mDelta

Case Is > 10 AND <=15

EE181\_RH\_JumpSuspected = 1

Case Is > 15

EE181\_RH\_JumpError = 1

Case Is = NAN

EE181\_RH\_JumpError = 1

EndSelect

EndIf

' Min and Max temperature/timestamp over 24hour period

MinSpa (EE181\_RH1440mBufferMin(),1440,EE181\_RH1440mBuffer())

EE181\_RHMin24H = EE181\_RH1440mBufferMin(1)

EE181\_RHMinTime24H = SecsSince1990(EE181\_RHTime1440mBuffer(EE181\_RH1440mBufferMin(2)),4)

MaxSpa (EE181\_RH1440mBufferMax(),1440,EE181\_RH1440mBuffer())

EE181\_RHMax24H = EE181\_RH1440mBufferMax(1)

EE181\_RHMaxTime24H = SecsSince1990(EE181\_RHTime1440mBuffer(EE181\_RH1440mBufferMax(2)),4)

' Calculated\_DewPointTemp------------------------------------------------

If TV10\_AirTemp <> NAN AND EE181\_RH <> NAN Then

DewPoint(Calculated\_DewPointTemp,TV10\_AirTemp,EE181\_RH)

Else

Calculated\_DewPointTemp = NAN

EndIf

' QC

' 60s

AvgRun(Calculated\_DewPointTemp60s,1,IIF(TV10\_AirTemp\_Delta > 2 OR EE181\_RH\_Delta > 5 OR TV10\_AirTemp = NAN OR EE181\_RH = NAN OR Calculated\_DewPointTemp = NAN,nan, Calculated\_DewPointTemp),60)

TotalRun(Calculated\_DewPointTemp60sN,1,IIF(TV10\_AirTemp\_Delta > 2 OR EE181\_RH\_Delta > 5 OR TV10\_AirTemp = NAN OR EE181\_RH = NAN OR Calculated\_DewPointTemp = NAN,0,1),60)

Calculated\_DewPointTemp\_Prev = Calculated\_DewPointTemp

' Initialize 1440 minute buffers

If Calculated\_DewPointTempStartup = 0 Then

Move(Calculated\_DewPointTemp1440mBuffer(),1440,NAN,1)

Move(Calculated\_DewPointTempTime1440mBuffer(),1440,0,1)

Calculated\_DewPointTempStartup = 1

EndIf

' 1440m

If TimeIntoInterval(0,60,sec) Then

' Advance 1440 minute buffer

For Calculated\_DewPointTempBufferCount = 1440 To 2 Step -1

Calculated\_DewPointTemp1440mBuffer(Calculated\_DewPointTempBufferCount) = Calculated\_DewPointTemp1440mBuffer(Calculated\_DewPointTempBufferCount-1)

Calculated\_DewPointTempTime1440mBuffer(Calculated\_DewPointTempBufferCount) = Calculated\_DewPointTempTime1440mBuffer(Calculated\_DewPointTempBufferCount-1)

Next Calculated\_DewPointTempBufferCount

Calculated\_DewPointTemp1440mBuffer(1) = Calculated\_DewPointTemp60s

Calculated\_DewPointTempTime1440mBuffer(1) = SecsSince1990(Public.Timestamp,1)

Calculated\_DewPointTemp1M = Calculated\_DewPointTemp1440mBuffer(1)

' Stuck sensor check over previous 60 minutes

Calculated\_DewPointTemp\_StuckSuspected = TV10\_AirTemp\_StuckSuspected OR EE181\_RH\_StuckSuspected

' Jump sensor check over last 2 minutes

Calculated\_DewPointTemp\_JumpSuspected = TV10\_AirTemp\_JumpSuspected OR EE181\_RH\_JumpSuspected

Calculated\_DewPointTemp\_JumpError = TV10\_AirTemp\_JumpError OR EE181\_RH\_JumpError

EndIf

' Average over 10 Minute period

AvgSpa (Calculated\_DewPointTemp10M,10,Calculated\_DewPointTemp1440mBuffer())

Calculated\_DewPointTempK10M = Calculated\_DewPointTemp10M + 273.15

' Min and Max temperature/timestamp over 24hour period

MinSpa (Calculated\_DewPointTemp1440mBufferMin(),1440,Calculated\_DewPointTemp1440mBuffer())

Calculated\_DewPointTempMin24H = Calculated\_DewPointTemp1440mBufferMin(1)

Calculated\_DewPointTempKMin24H = Calculated\_DewPointTempMin24H + 273.15

Calculated\_DewPointTempMinTime24H = SecsSince1990(Calculated\_DewPointTempTime1440mBuffer(Calculated\_DewPointTemp1440mBufferMin(2)),4)

MaxSpa (Calculated\_DewPointTemp1440mBufferMax(),1440,Calculated\_DewPointTemp1440mBuffer())

Calculated\_DewPointTempMax24H = Calculated\_DewPointTemp1440mBufferMax(1)

Calculated\_DewPointTempKMax24H = Calculated\_DewPointTempMax24H + 273.15

Calculated\_DewPointTempMaxTime24H = SecsSince1990(Calculated\_DewPointTempTime1440mBuffer(Calculated\_DewPointTemp1440mBufferMax(2)),4)

' RMY52203\_Rain----------------------------------------------------------

PulseCount(RMY52203\_Rain,1,P2,1,0,0.1,0)

' QC

' Calculate 1 minute rain total which will be used to calculate running totals,

' by using 1 minute instead of 1second, overhead is greatly reduced.

TotalRun(RMY52203\_Rain60s,1,RMY52203\_Rain,60)

' Calculate running totals

If TimeIntoInterval(0,60,sec) Then

' 1m

RMY52203\_Rain1M = RMY52203\_Rain60s

' Rain intensity for the last 60 seconds in mm/h units

RMY52203\_RainIntensity1M = RMY52203\_Rain1M \* 60

' 1h

TotalRun(RMY52203\_Rain1H,1,RMY52203\_Rain1M,60)

' 3h

TotalRun(RMY52203\_Rain3H,1,RMY52203\_Rain1M,180)

' 24h

TotalRun(RMY52203\_Rain24H,1,RMY52203\_Rain1M,1440)

EndIf

' RSG2STDN\_SlrW \_SlrMJ---------------------------------------------------

VoltDiff (RSG2STDN\_SlrmV,1,mV200,4,True,0,50,1,0)

RSG2STDN\_SlrW = (RSG2STDN\_SlrmV\*1000) / RSG2STDN\_SlrW\_Sensitivity 'uV/W/m^2

If RSG2STDN\_SlrW<0 Then RSG2STDN\_SlrW=0

RSG2STDN\_SlrMJ = RSG2STDN\_SlrW \* 10^-6 \* 1 ' This multiplier must be set the same as the scan -from which- the table is called!

' QC

' Calculate 1 minute SlrJ total which will be used to calculate running totals,

' by using 1 minute instead of 1second, overhead is greatly reduced.

RSG2STDN\_SlrJ = RSG2STDN\_SlrW \* 1 ' This multiplier must be set the same as the scan -from which- the table is called!

TotalRun(RSG2STDN\_SlrJ60s,1,RSG2STDN\_SlrJ,60)

' Calculate running totals

If TimeIntoInterval(0,60,sec) Then

' 1h

TotalRun(RSG2STDN\_SlrJ1H,1,RSG2STDN\_SlrJ60s,60)

' 24h

TotalRun(RSG2STDN\_SlrJ24H,1,RSG2STDN\_SlrJ60s,1440)

EndIf

'-Sun\_Hours--------------------------------------------------------------

PotSlrW=SIN(Sun\_Elevation)\*1373 ' Calculate potential radiation for time & position (multiply sine of solar elevation angle by solar constant 1373)

'If the measured value (W/m^2) is greater than 0.4 \* the potential solar radiation (W/m^2)

'and the sun elevation angle (degrees) is greater than 6 degrees) then it has been sunny for the current scan.

If SlrW>0.4\*PotSlrW AND Sun\_Elevation>6 Then

'Calculate sun hours for scan time in seconds

Sun\_Hours=1/3600\* 1 ' This multiplier must be set the same as the scan -from which the table- is called!

Else

'Set sun hours for scan time in seconds to 0

Sun\_Hours=0

EndIf

' QC

' Calculate 1 minute Sun total which will be used to calculate running totals,

' by using 1 minute instead of 1second, overhead is greatly reduced.

TotalRun(SunHours60s,1,Sun\_Hours\*60,60) ' Expressed in minutes for GBON

' Calculate running totals

If TimeIntoInterval(0,60,sec) Then

' 1h

TotalRun(SunHours1H,1,SunHours60s,60)

' 24h

TotalRun(SunHours24H,1,SunHours60s,1440)

EndIf

'-Call tables------------------------------------------------------------

CallTable Table2min

CallTable Table10m

CallTable TableHour

CallTable TableDay

CallTable Synop

' Restart logger on Mondays at 00:00:10 (10 seconds after midnight) (This can be enabled in remote installations where a logger restart is the only way to restore comms)

'If rTime(8) = 2 AND TimeIntoInterval(10,86400,sec) Then FileManage(Status.ProgName,6)

'-Calculate comms time - all comms must take place in less than the scan rate

ScanTime1 = Timer(1,mSec,4)

NextScan

'-Scan2: BV10-------------------------------------------------------------

SlowSequence

Scan(1, Sec, 3, 0)

'-Starting time of processing

Timer (2,mSec,2)

' BV10\_BPress------------------------------------------------------------

SDI12Recorder(BV10\_SDI12(),C3,"0","M!",1,0,-1)

BV10\_BPress = BV10\_SDI12(1)

' Optionally include

BV10\_BPTemp = BV10\_SDI12(2)

'Calculate QFE AND QNH

'Override tQFE using temp sensor

If TV10\_AirTemp <> NAN Then

BV10\_tQFE = TV10\_AirTemp

Else

BV10\_tQFE = BV10\_tstdQFE

EndIf

' Calculate QFE

BV10\_QFE = BV10\_BPress\* (1 + ((BV10\_hQFE \* 9.81 )/(287\*(BV10\_tQFE+273.15))))

' Calculate QNH

BV10\_QNH = BV10\_QFE \* EXP((BV10\_hQNH\*9.81)/(287\*(288.15+((-0.0065\*BV10\_hQNH)/2))))

' Calculate QNH (ICAO)

BV10\_hISA = 44330.77 - 11880.32 \* BV10\_QFE^0.190263

BV10\_QNH\_ICAO= 1013.25 \* (1.0 - ((0.0065\*(BV10\_hISA-BV10\_hQNH))/288.15))^5.25588

' Calculate QFF (Swedish Meteorological and Hydrological Institute)

If BV10\_tQFE < -7 Then BV10\_tQFF = 0.5 \* BV10\_tQFE + 275

If BV10\_tQFE >= -7 AND BV10\_tQFE < 2 Then BV10\_tQFF = 0.535 \* BV10\_tQFE + 275.6

If BV10\_tQFE >= 2 Then BV10\_tQFF = 1.07 \* BV10\_tQFE + 274.5

BV10\_QFF = BV10\_QFE \* EXP ((BV10\_hQNH \*0.034163\*(1 - 0.0026373 \* COS(2 \* GPS\_Latitude)))/BV10\_tQFF)

' QC

' 60s

BV10\_Bpress\_Delta = ABS(BV10\_BPress - BV10\_BPress\_Prev)

AvgRun(BV10\_BPress60s,1,IIF(BV10\_Bpress\_Delta > 0.3 OR BV10\_BPress = NAN,NAN, BV10\_BPress),60)

TotalRun(BV10\_BPress60sN,1,IIF(BV10\_Bpress\_Delta > 0.3 OR BV10\_BPress = NAN,0,1),60)

BV10\_BPress\_Prev = BV10\_BPress

' Initialize 180 minute buffers

If BV10\_BPressStartup = 0 Then

Move(BV10\_BPress180mBuffer(),180,NAN,1)

BV10\_BPressStartup = 1

EndIf

' 180m

If TimeIntoInterval(0,60,sec) Then

' Advance 180 minute buffer

For BV10\_BPressBufferCount = 180 To 2 Step -1

BV10\_BPress180mBuffer(BV10\_BPressBufferCount) = BV10\_BPress180mBuffer(BV10\_BPressBufferCount-1)

Next BV10\_BPressBufferCount

BV10\_BPress180mBuffer(1) = BV10\_BPress60s

BV10\_BPress1M = BV10\_BPress180mBuffer(1)

BV10\_BPressPa1M = BV10\_BPress1M \* 100

' Stuck sensor check over previous 60 minutes

MinSpa (BV10\_BPress180mBufferMin(),60,BV10\_BPress180mBuffer())

MaxSpa (BV10\_BPress180mBufferMax(),60,BV10\_BPress180mBuffer())

BV10\_Bpress60mDelta = ABS(BV10\_BPress180mBufferMin(1) - BV10\_BPress180mBufferMax(1))

If BV10\_Bpress60mDelta < 0.1 OR BV10\_Bpress60mDelta = NAN Then BV10\_Bpress\_StuckSuspected = 1 Else BV10\_Bpress\_StuckSuspected = 0

' Jump sensor check over last 2 minutes

BV10\_BPress\_JumpSuspected = 0

BV10\_BPress\_JumpError = 0

BV10\_Bpress2mDelta=ABS(BV10\_BPress180mBuffer(1) - BV10\_BPress180mBuffer(2))

Select Case BV10\_Bpress2mDelta

Case Is >= 0.5 AND < 2

BV10\_BPress\_JumpSuspected = 1

Case Is >= 2

BV10\_BPress\_JumpError = 1

Case Is = NAN

BV10\_BPress\_JumpError = 1

EndSelect

EndIf

' Elevation

BV10\_BPress\_Elevation = BV10\_hQNH + BV10\_hQFE

' Pressure change

BV10\_BPressChange3H = BV10\_BPress180mBuffer(1) - BV10\_BPress180mBuffer(180)

BV10\_BPressPaChange3H = BV10\_BPressChange3H \* 100

' Pressure tendency

Select Case BV10\_BPressChange3H

Case Is = NAN

BV10\_BPressTendency3H = NAN

Case Is > 0

BV10\_BPressTendency3H = 2

Case Is < 0

BV10\_BPressTendency3H = 7

Case Else

BV10\_BPressTendency3H = 4

EndSelect

'Calculate QFE AND QNH

'Override tQFE using temp sensor

If TV10\_AirTemp1M <> NAN Then

BV10\_tQFE1M = TV10\_AirTemp1M

Else

BV10\_tQFE1M = BV10\_tstdQFE

EndIf

' Calculate QFE

BV10\_QFE1M = BV10\_BPress1M\* (1 + ((BV10\_hQFE \* 9.81 )/(287\*(BV10\_tQFE1M+273.15))))

' Calculate QNH

BV10\_QNH1M = BV10\_QFE1M \* EXP((BV10\_hQNH\*9.81)/(287\*(288.15+((-0.0065\*BV10\_hQNH)/2))))

BV10\_QNHPa1M = BV10\_QNH1M \* 100

'-Calculate comms time - all comms must take place in less than the scan rate

ScanTime2 = Timer(2,mSec,4)

NextScan

EndSequence

'-Scan3: Settings Store, SM2C and MPPT4A-----------------------------------

SlowSequence

Scan(10, Sec, 3, 0)

'-Starting time of processing

Timer (3,mSec,2)

' Write StationID to logger

If StationID\_WriteFlag = true Then

Call StationID\_Write

' Read to test

Call StationID\_Read

CallTable SettingsBackup ' Save to table as well for settings history and backup

StationID\_WriteFlag = false

EndIf

' Write Location settings to logger

If Location\_WriteFlag = true Then

Call Location\_Write

Call Location\_Read ' Read to test

CallTable SettingsBackup ' Save to table as well for settings history and backup

Location\_WriteFlag = false

EndIf

' Write TV10\_AirTemp settings and calibration to user drive

If TV10\_AirTemp\_WriteSettingsFlag = true Then

Call TV10\_AirTemp\_WriteSettings

' Read to test

Call TV10\_AirTemp\_ReadSettings

CallTable SettingsBackup ' Save to table as well for settings history and backup

TV10\_AirTemp\_WriteSettingsFlag = false

EndIf

' Write RMY52203\_Rain settings to user drive

If RMY52203\_Rain\_WriteSettingsFlag = true Then

Call RMY52203\_Rain\_WriteSettings

' Read to test

Call RMY52203\_Rain\_ReadSettings

CallTable SettingsBackup ' Save to table as well for settings history and backup

RMY52203\_Rain\_WriteSettingsFlag = false

EndIf

' Write Thies\_WSpd settings to user drive

If Thies\_WSpd\_WriteSettingsFlag = true Then

Call Thies\_WSpd\_WriteSettings

' Read to test

Call Thies\_WSpd\_ReadSettings

CallTable SettingsBackup ' Save to table as well for settings history and backup

Thies\_WSpd\_WriteSettingsFlag = false

EndIf

' Write BV10\_BPress settings

If BV10\_WriteSettingsFlag = true Then

Call BV10\_WriteSettings

Call BV10\_ReadSettings ' Read to test

CallTable SettingsBackup ' Save to table as well for settings history and backup

BV10\_WriteSettingsFlag = false

EndIf

' Write RSG2STDN\_SlrW Calibration to user drive

If RSG2STDN\_SlrW\_WriteSettingsFlag = true Then

Call RSG2STDN\_SlrW\_WriteSettings

' Read to test

Call RSG2STDN\_SlrW\_ReadSettings

CallTable SettingsBackup ' Save to table as well for settings history and backup

RSG2STDN\_SlrW\_WriteSettingsFlag = false

EndIf

' Write SYNOP settings to logger

If SYNOP\_WriteFlag = true Then

Call SYNOP\_Write

Call SYNOP\_Read ' Read to test

CallTable SettingsBackup ' Save to table as well for settings history and backup

SYNOP\_WriteFlag = false

EndIf

' SM2C \_VWC\_1 \_EC\_1 \_SoilTemp\_1------------------------------------------

' Retrieve values

SerialFlush(Comc5)

ModbusMaster (SM2C\_ModbusResult\_1,Comc5,9600,1,3,SM2C\_ModbusData\_1(),1,3,2,100,1)

If SM2C\_ModbusResult\_1 = 0 Then

SM2C\_SoilTemp\_1= SM2C\_ModbusData\_1(1) / 100

SM2C\_VWC\_1 = SM2C\_ModbusData\_1(2) / 100

SM2C\_EC\_1 =SM2C\_ModbusData\_1(3) / 1000

Else

SM2C\_SoilTemp\_1= NAN

SM2C\_VWC\_1 = NAN

SM2C\_EC\_1 =NAN

EndIf

' SM2C \_VWC\_2 \_EC\_2 \_SoilTemp\_2------------------------------------------

' Retrieve values

SerialFlush(Comc5)

ModbusMaster (SM2C\_ModbusResult\_2,Comc5,9600,2,3,SM2C\_ModbusData\_2(),1,3,2,100,1)

If SM2C\_ModbusResult\_2 = 0 Then

SM2C\_SoilTemp\_2= SM2C\_ModbusData\_2(1) / 100

SM2C\_VWC\_2 = SM2C\_ModbusData\_2(2) / 100

SM2C\_EC\_2 =SM2C\_ModbusData\_2(3) / 1000

Else

SM2C\_SoilTemp\_2= NAN

SM2C\_VWC\_2 = NAN

SM2C\_EC\_2 =NAN

EndIf

' SM2C \_VWC\_3 \_EC\_3 \_SoilTemp\_3------------------------------------------

' Retrieve values

SerialFlush(Comc5)

ModbusMaster (SM2C\_ModbusResult\_3,Comc5,9600,3,3,SM2C\_ModbusData\_3(),1,3,2,100,1)

If SM2C\_ModbusResult\_3 = 0 Then

SM2C\_SoilTemp\_3= SM2C\_ModbusData\_3(1) / 100

SM2C\_VWC\_3 = SM2C\_ModbusData\_3(2) / 100

SM2C\_EC\_3 =SM2C\_ModbusData\_3(3) / 1000

Else

SM2C\_SoilTemp\_3= NAN

SM2C\_VWC\_3 = NAN

SM2C\_EC\_3 =NAN

EndIf

' SM2C \_VWC\_4 \_EC\_4 \_SoilTemp\_4------------------------------------------

' Retrieve values

SerialFlush(Comc5)

ModbusMaster (SM2C\_ModbusResult\_4,Comc5,9600,4,3,SM2C\_ModbusData\_4(),1,3,2,100,1)

If SM2C\_ModbusResult\_4 = 0 Then

SM2C\_SoilTemp\_4= SM2C\_ModbusData\_4(1) / 100

SM2C\_VWC\_4 = SM2C\_ModbusData\_4(2) / 100

SM2C\_EC\_4 =SM2C\_ModbusData\_4(3) / 1000

Else

SM2C\_SoilTemp\_4= NAN

SM2C\_VWC\_4 = NAN

SM2C\_EC\_4 =NAN

EndIf

' MPPT4A\_SolarCharger ---------------------------------------------------

' Get values

SDI12Recorder (MPPT4A\_SDI12\_1(),C7,0,"M!",1.0,0,-1,0)

MPPT4A\_PanelVoltage = MPPT4A\_SDI12\_1(1) / 1000

MPPT4A\_PanelCurrent = MPPT4A\_SDI12\_1(4) / 1000

MPPT4A\_PanelPower = MPPT4A\_PanelVoltage \* MPPT4A\_PanelCurrent

MPPT4A\_LoadVoltage = MPPT4A\_SDI12\_1(3) / 1000

MPPT4A\_LoadCurrent = MPPT4A\_SDI12\_1(5) / 1000

MPPT4A\_BatteryVoltage = MPPT4A\_SDI12\_1(2) / 1000

'MPPT4A\_BoardTemp = MPPT4A\_SDI12\_2(8) 'This is in next SDI12 Call

MPPT4A\_State = MPPT4A\_SDI12\_1(6)

' Get and set settings (TO reset the controller one can send the aXR! command)

SDI12Recorder (MPPT4A\_SDI12\_2(),C7,0,"D1!",1.0,0,-1,0)

If MPPT4A\_SDI12\_2(1) <> NAN Then

MPPT4A\_Mode = MPPT4A\_SDI12\_2(3) : If MPPT4A\_Mode <> MPPT4A\_SetMode Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XP" + MPPT4A\_SetMode + "!",1.0,0,0,0)

MPPT4A\_FloatVoltage = MPPT4A\_SDI12\_2(1) / 1000 : If MPPT4A\_FloatVoltage \* 1000 <> MPPT4A\_SetFloatVoltage \* 1000 Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XV" + MPPT4A\_SetFloatVoltage \* 1000 + "!",1.0,0,0,0)

MPPT4A\_CurrentLimit = MPPT4A\_SDI12\_2(2) / 1000 : If MPPT4A\_CurrentLimit \* 1000 <> MPPT4A\_SetCurrentLimit \* 1000 Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XI" + MPPT4A\_SetCurrentLimit \* 1000 + "!",1.0,0,0,0)

MPPT4A\_AbsorbTimeLimit = MPPT4A\_SDI12\_2(4) : If MPPT4A\_AbsorbTimeLimit <> MPPT4A\_SetAbsorbTimeLimit Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XT" + MPPT4A\_SetAbsorbTimeLimit + "!",1.0,0,0,0)

MPPT4A\_AbsorbFullCurrent = MPPT4A\_SDI12\_2(5) / 1000 : If MPPT4A\_AbsorbFullCurrent \* 1000 <> MPPT4A\_SetAbsorbFullCurrent \*1000 Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XJ" + MPPT4A\_SetAbsorbFullCurrent\*1000 + "!",1.0,0,0,0)

MPPT4A\_VCalSlope = MPPT4A\_SDI12\_2(6) : If MPPT4A\_VCalSlope <> MPPT4A\_SetVCalSlope Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XC" + MPPT4A\_SetVCalSlope + "!",1.0,0,0,0)

MPPT4A\_ICalSlope = MPPT4A\_SDI12\_2(7) : If MPPT4A\_ICalSlope <> MPPT4A\_SetICalSlope Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XB" + MPPT4A\_SetICalSlope + "!",1.0,0,0,0)

MPPT4A\_BoardTemp = MPPT4A\_SDI12\_2(8)

MPPT4A\_BulkFloatVoltage = MPPT4A\_SDI12\_2(9) / 1000 : If MPPT4A\_BulkFloatVoltage \* 1000 <> MPPT4A\_SetBulkFloatVoltage \* 1000 Then SDI12Recorder (MPPT4A\_SDI12\_X(),C7,0,"XY" + MPPT4A\_SetBulkFloatVoltage\*1000 + "!",1.0,0,0,0)

EndIf

'-Calculate comms time - all comms must take place in less than the scan rate

ScanTime3 = Timer(3,mSec,4)

NextScan

EndSequence

' Scan4: Modem power-------------------------------------------------------

SlowSequence

Scan(1,min,0,0)

'-Starting time of processing

Timer (4,mSec,2)

'-First check if modem is enabled

If status.pppinterface <> 0 Then

'-Measure battery voltage

Battery(LoggerBattery)

'-Power down between 23:57 and 23:59

If TimeIsBetween(1437,1439,1440,min) Then

If (ModemReset = false) AND (ModemLowPower = false) Then

ModemReset= true

PPPClose

Delay(1,5,sec)

IPNetPower (5,0)

EndIf

Else

If (ModemReset = true) AND (ModemLowPower = false) Then

ModemReset = false

IPNetPower (5,1)

Delay(1,20,sec)

PPPOpen

EndIf

EndIf

'-Power down below 12V

If (LoggerBattery < 12) AND (ModemLowPower = false) AND (ModemReset = false) Then

ModemLowPower = true

PPPClose

Delay(1,5,sec)

IPNetPower (5,0)

EndIf

'-Power up above 12.2V

If (LoggerBattery > 12.2) AND (ModemLowPower = true) AND (ModemReset = false) Then

ModemLowPower = false

IPNetPower (5,1)

Delay(1,20,sec)

PPPOpen

EndIf

'-IP Keep alive(30 minute)

PingTime = 0

If (ModemReset = false) AND (ModemLowPower = false) Then

SetSetting("DNS(2)","8.8.8.8") 'Add google as seconday DNS

IPRoute("8.8.8.8",1) 'Ping test through modem

IPRoute("callback.csafrica.co.za",1) 'Callback through modem

For PingRetry = 1 To 3

PingTime = PingIP("8.8.8.8",3000)

If PingTime > 0 Then ExitFor

Next PingRetry

If PingTime = 0 Then IPFailCount += 1 Else IPFailCount = 0

If (IPFailCount <> 0) AND (IPFailCount MOD 30 = 0) Then

PPPClose

Delay(1,5,sec)

IPNetPower (5,0)

Delay(1,30,Sec)

IPNetPower (5,1)

Delay(1,15,sec)

PPPOpen

EndIf

EndIf

EndIf

'-Calculate comms time - all comms must take place in less than the scan rate

ScanTime4 = Timer(4,mSec,4)

NextScan

EndSequence

EndProg