RWIS Deployment Update for Campbell-Datalogger-Based-RPU

Jeff Worthington, ITS Engineer

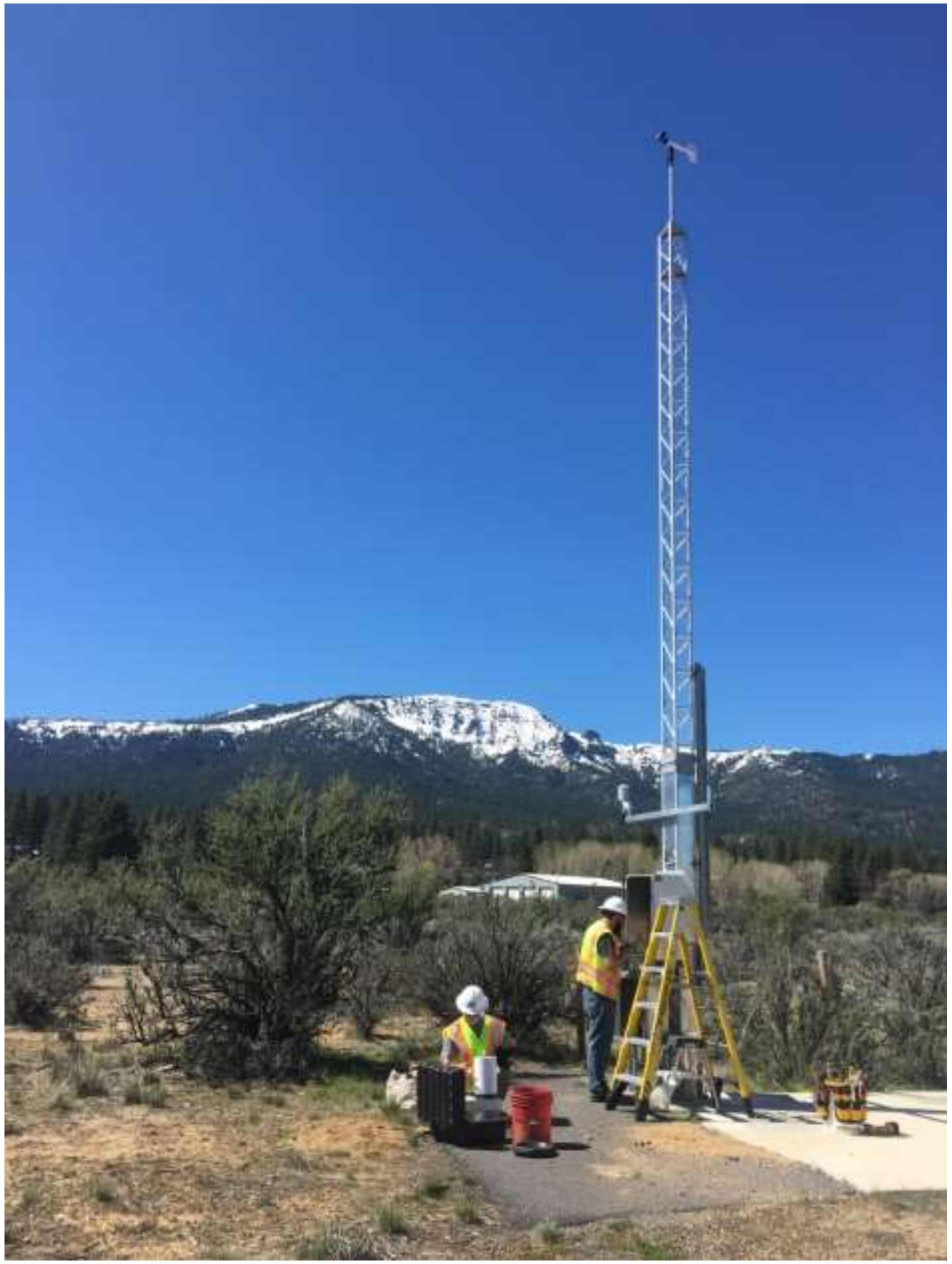
Caltrans, District 2

Western States Forum

June 22, 2016

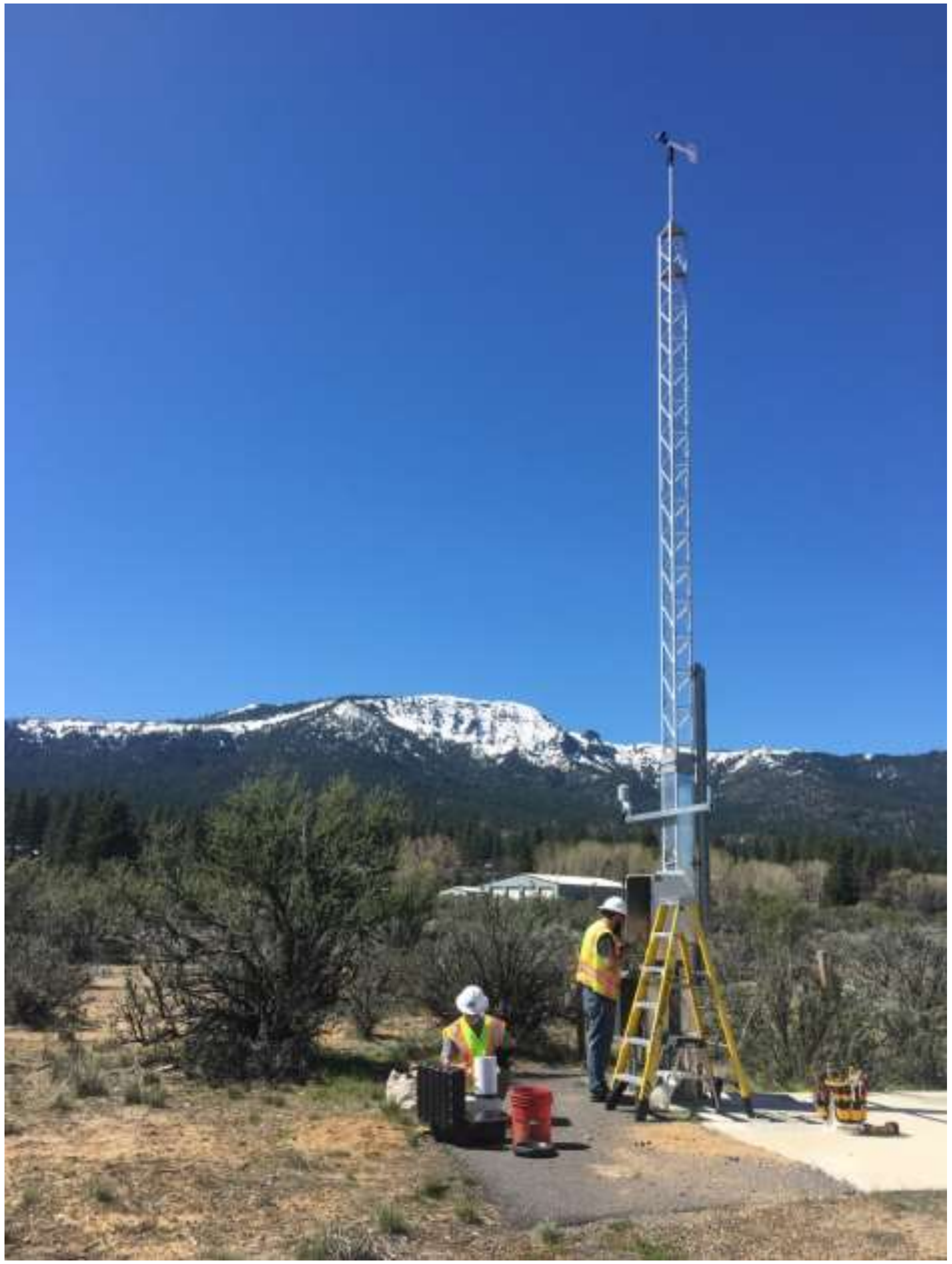
CR1000 RPU Deployment

- **Definitions and Background**
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



CR1000 RPU Deployment

- Definitions and Background
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



Definitions and Background

"A Road Weather Information System (RWIS) is comprised of Environmental Sensor Stations (ESS) in the field, a communication system for data transfer, and central systems to collect field data from numerous ESS. These stations measure atmospheric, pavement and/or water level conditions. Central RWIS hardware and software are used to process observations from ESS to develop nowcasts or forecasts, and display or disseminate road weather information in a format that can be easily interpreted by a manager. RWIS data are used by road operators and maintainers to support decision making."

U.S. Department of Transportation – Federal Highway Administration
http://www.ops.fhwa.dot.gov/weather/faq.htm

Definitions and Background

Acronyms used in this presentation

RWIS = Road Weather Information *System*

ESS = Environmental Sensor Station

RPU = Remote Processor Unit, and supporting hardware

RSS = Road Surface Sensor

ICWS=Icy Curves Warning System

IPS = In Pavement Sensor

NIPS = Non Invasive Pavement Sensor

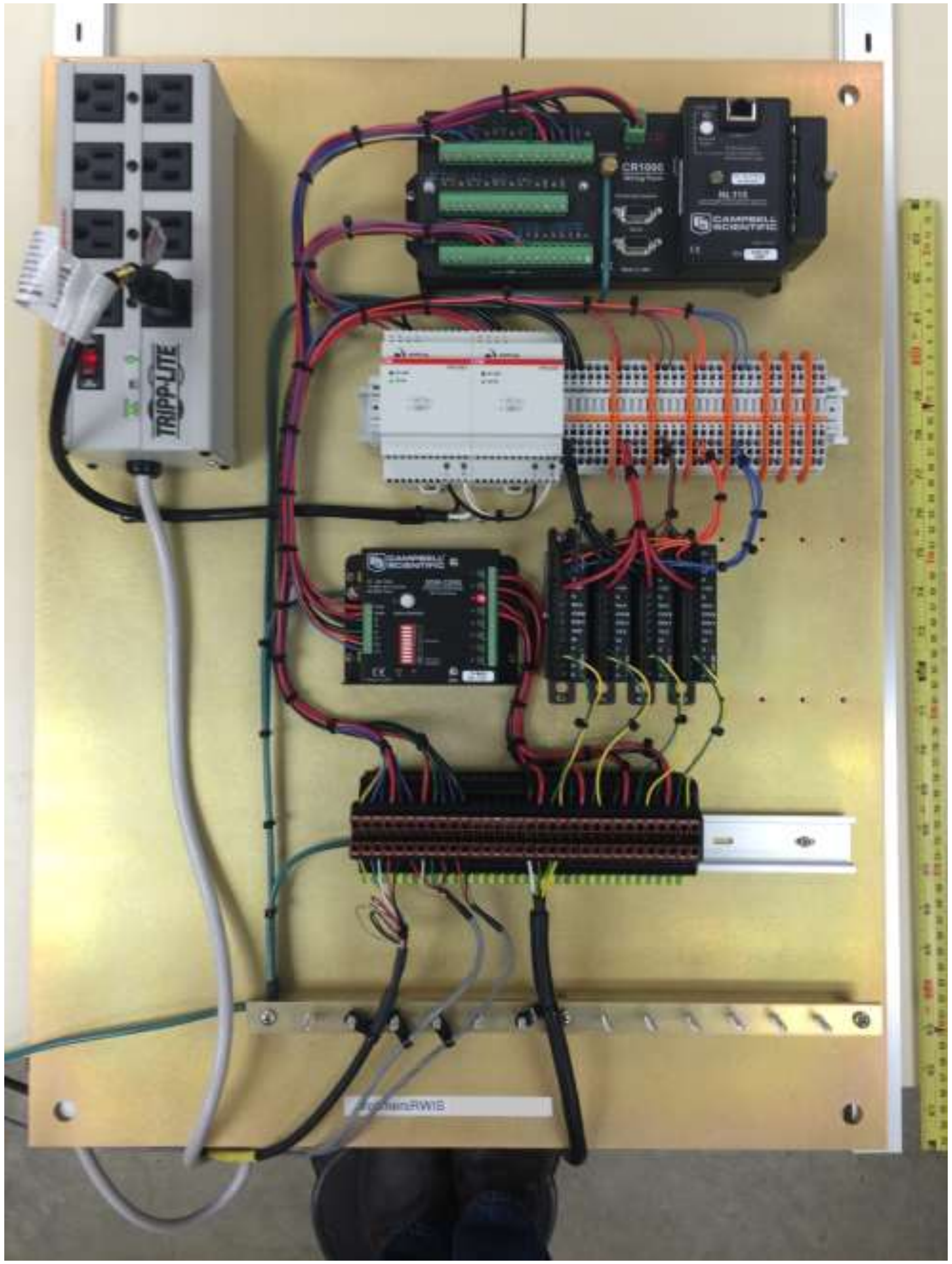
BOM = Bill of Materials

NEMA = National Electrical Manufacturers Association

NTCIP = National Transportation Communications for Intelligent Transportation System Protocol.

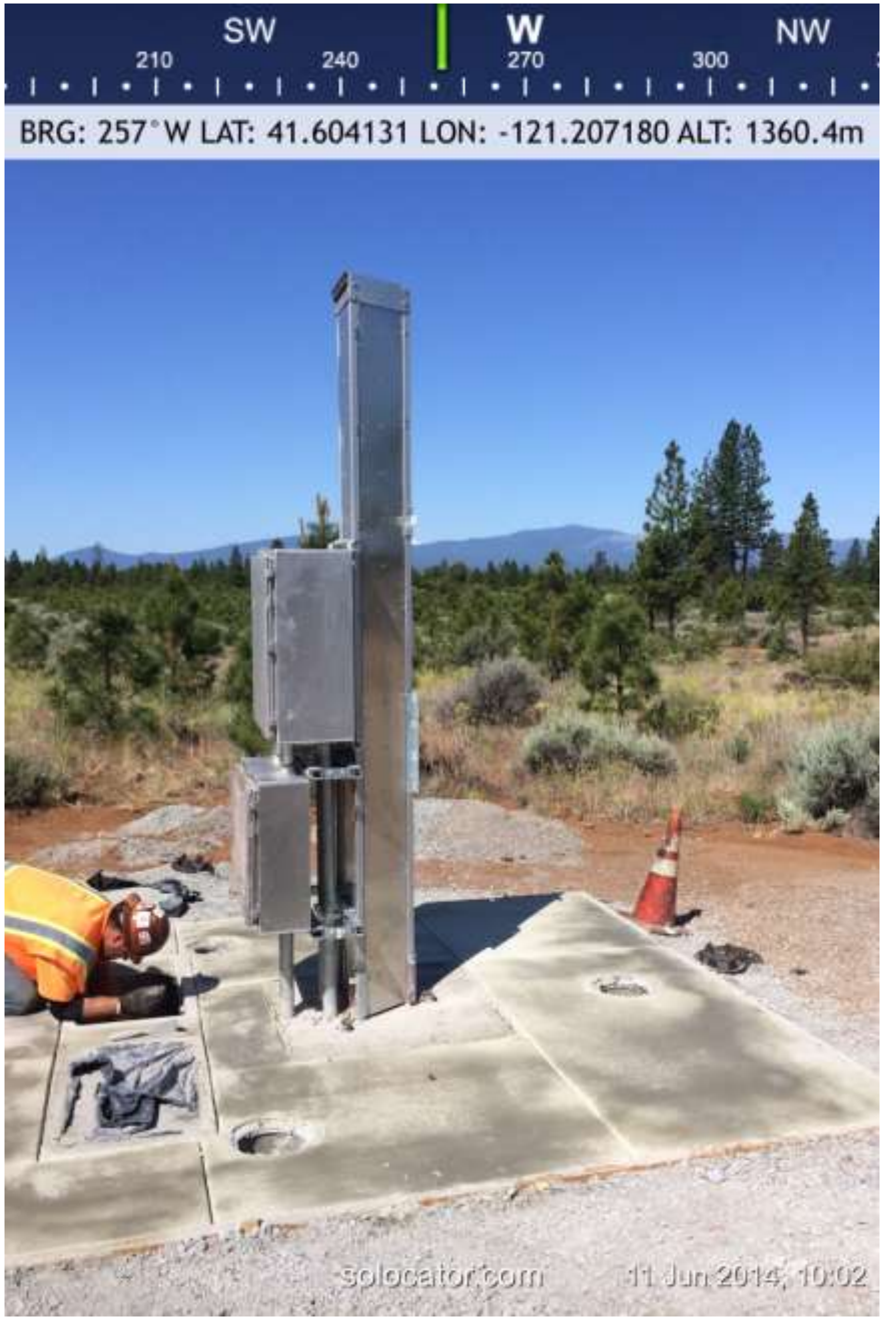
RPU – Caltrans D2 Campbell version 1

- The first RPU installed in field. Vollmers, CA



ESS – D2 standard site under construction

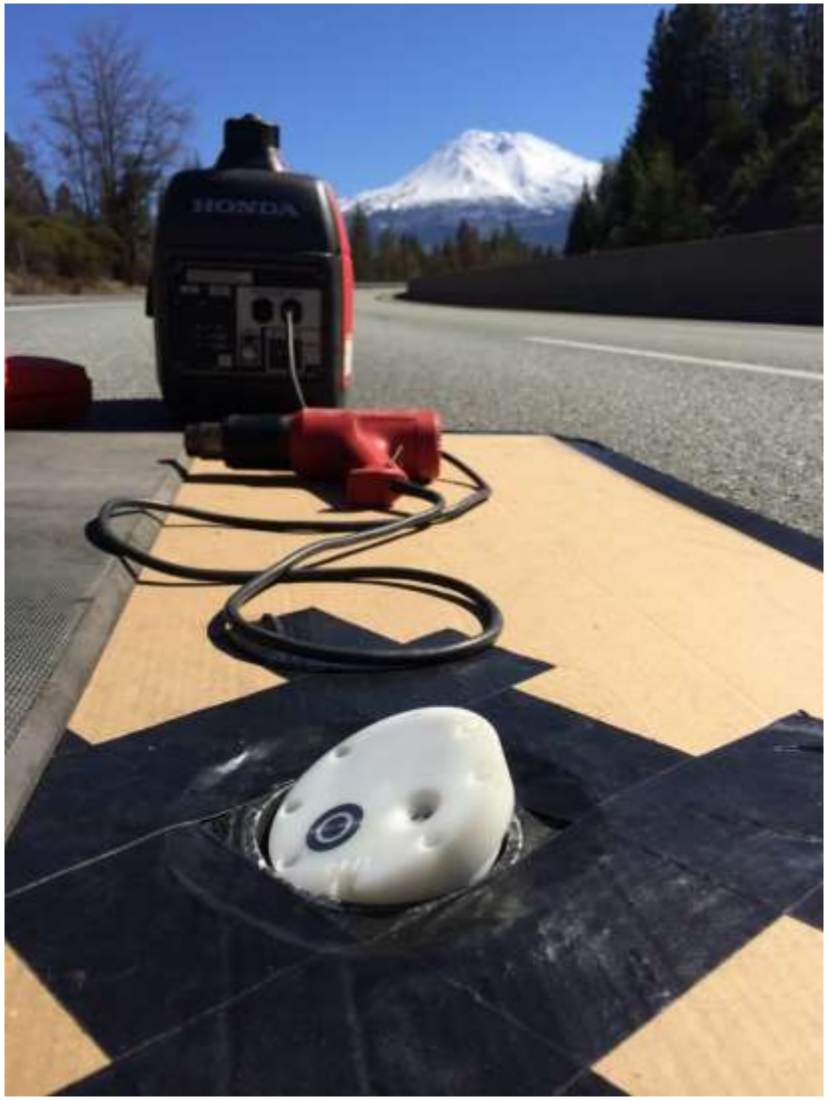
- RPU cabinet on top
- Communication cabinet below
- Completed in 2014
- Located in Perez, CA



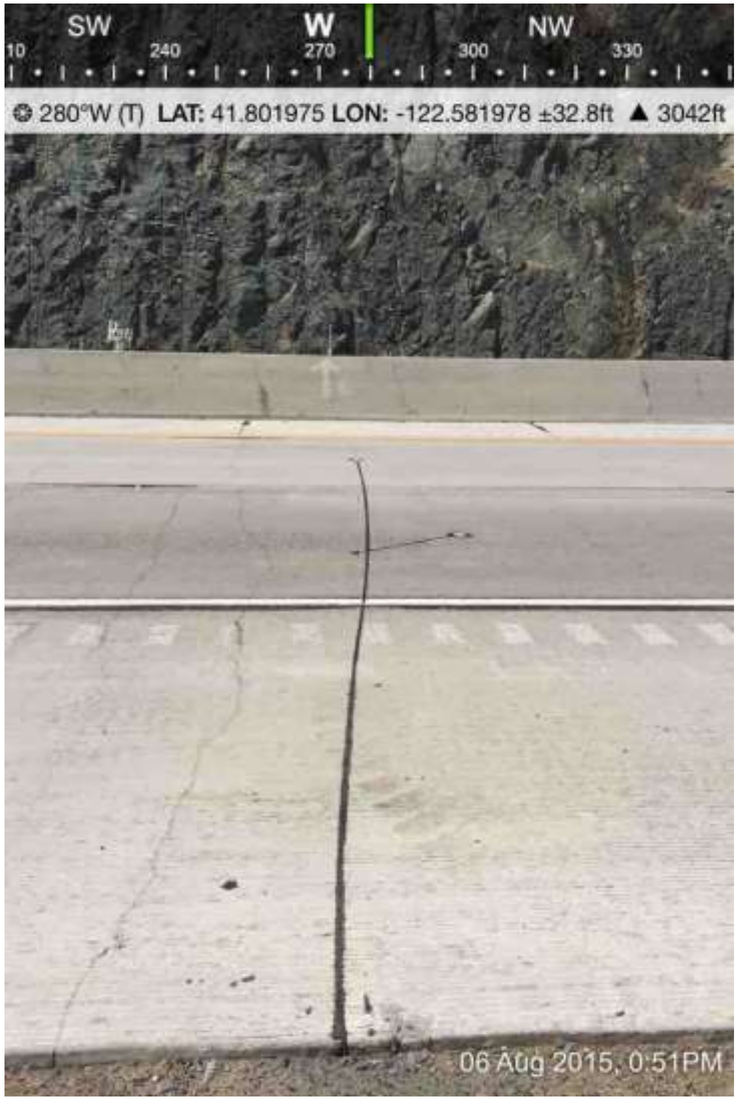
ESS – D2 Mini RPU

- Supports 8 Lufft RSS
- No Tower Instruments
- Came on line in 2015
- Located on Anderson Grade Summit, Yreka, CA

RSS in pavement



Lufft RSS are removable from a housing.



Notice saw cuts for wired power and data.

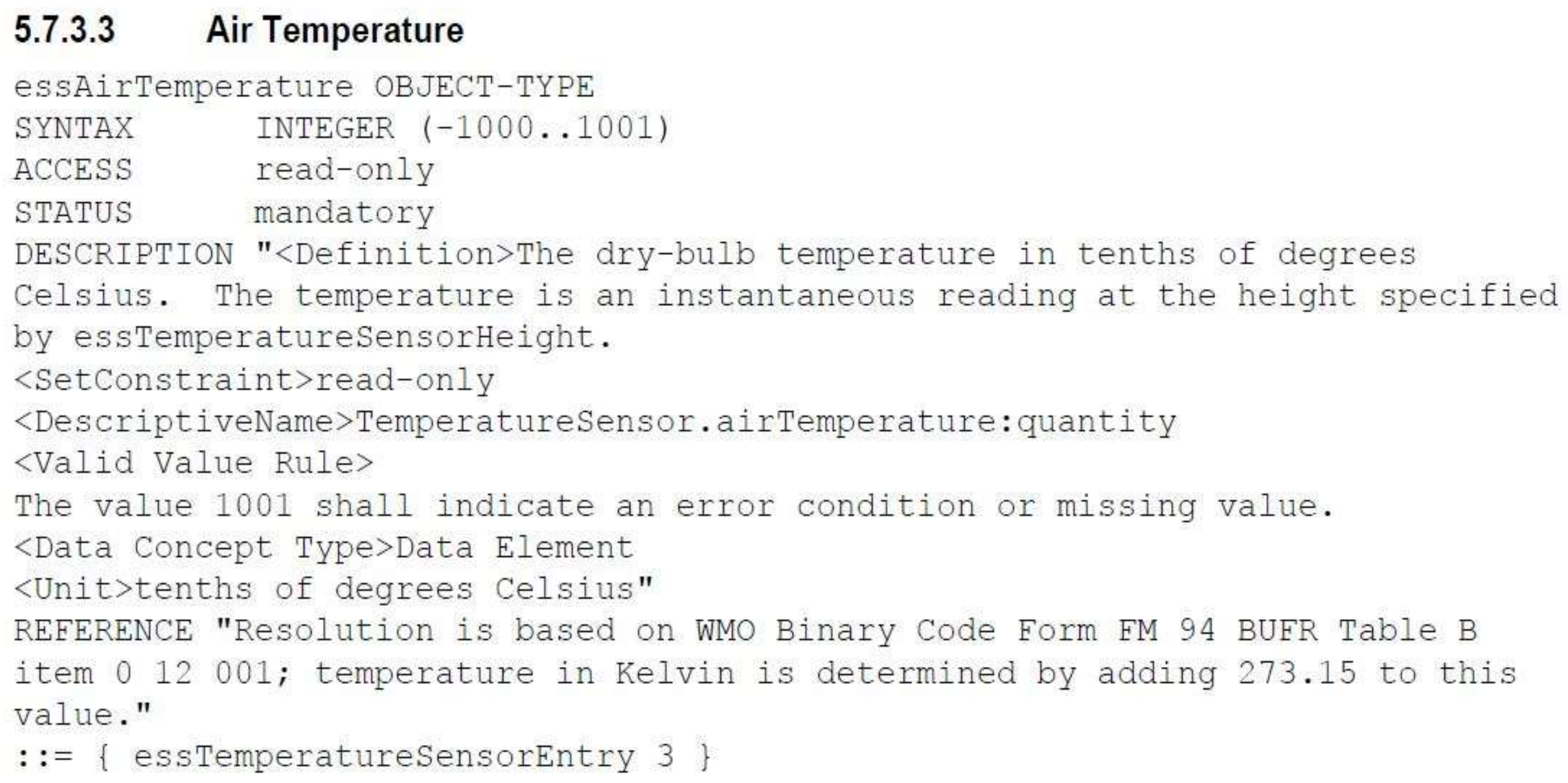
Definitions and Background

NTCIP related information

- The NTCIP 1204 standard defines ESS data (type and units)
- Version 1 of 1204 was approved in 1998
- Version 2 of 1204 was approved in 2006
- Version 3 of 1204 was approved in 2014
- We currently follow the version 2 standard
- For atmospheric, pavement, and site definition, we provide 58 NTCIP ESS values

Definitions and Background

Ess variable sample

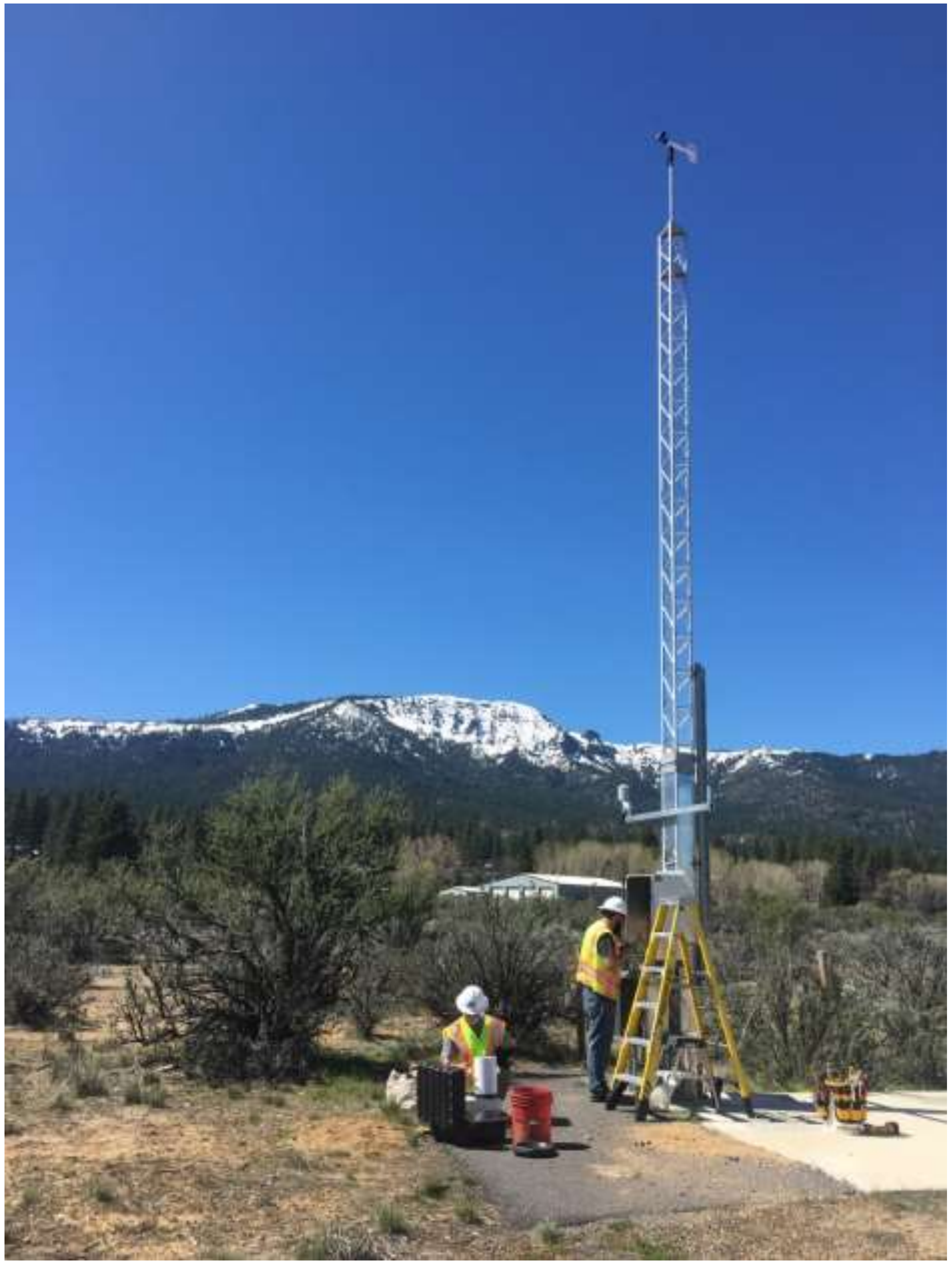


5.7.3.3 Air Temperature

```
essAirTemperature OBJECT-TYPE
SYNTAX      INTEGER (-1000..1001)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION "<Definition>The dry-bulb temperature in tenths of degrees
Celsius.  The temperature is an instantaneous reading at the height specified
by essTemperatureSensorHeight.
<SetConstraint>read-only
<DescriptiveName>TemperatureSensor.airTemperature:quantity
<Valid Value Rule>
The value 1001 shall indicate an error condition or missing value.
<Data Concept Type>Data Element
<Unit>tenths of degrees Celsius"
REFERENCE "Resolution is based on WMO Binary Code Form FM 94 BUFR Table B
item 0 12 001; temperature in Kelvin is determined by adding 273.15 to this
value."
::= { essTemperatureSensorEntry 3 }
```

CR1000 RPU Deployment

- **Definitions and Background**
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



RPU Redesign – Why?

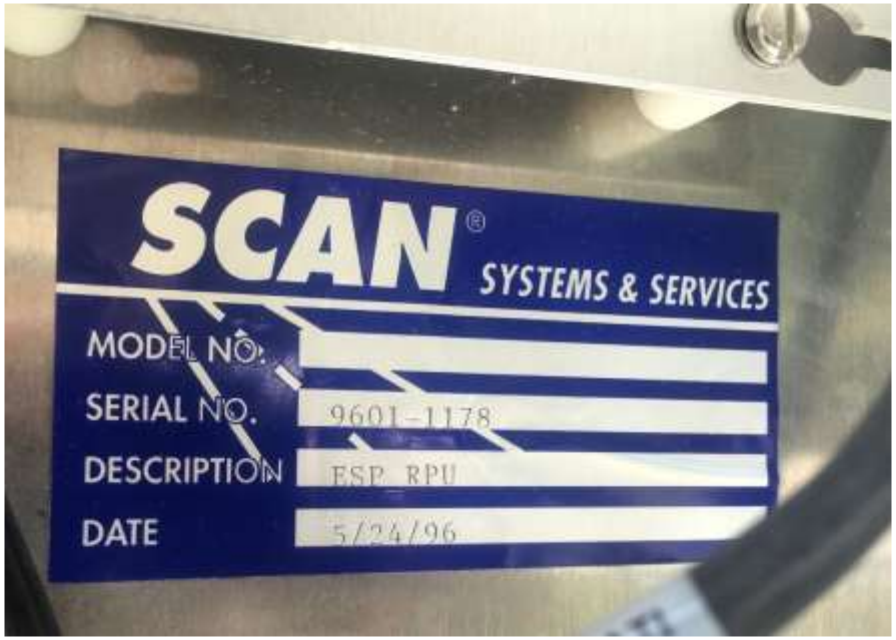
- Vendor configurations not always standardized/or open sourced.
- Vendors will not always be there. Hardware ages. Need for replacement parts.
- Expensive support agreements.
- Flexibility with future design and integration with off-the-shelf components.
- District 2 Mantra – Is it Accurate, Timely, and Reliable?

RPU Redesign – Why?

- District 2 Mantra – Is it Accurate, Timely, and Reliable?
  - ➢Ask yourself if every instrument at your RWIS station can be ground- truthed?
    - ✓Goal is for all instruments to be calibrated in the field.
    - ✓Avoid ‘black-box’ components. Want open source.
  - ➢Want a reliable RPU and reliable communications to that RPU, especially when you need it most!
    - ✓Design for your temperature extremes.
    - ✓Resistance to water, sunlight, pollution in cabinet; splices; and cabling.
    - ✓Separate your communications equipment into a separate cabinet.
    - ✓Battery backup for power loss.
  - ➢Ease of Troubleshooting:
    - ✓Can I get the same information remotely as if I were physically at the site?
    - ✓Prefer not to have to drive to site for a serial connection or a reboot.
    - ✓Will I need to call support and wait for that one person assigned to the entire West Coast?

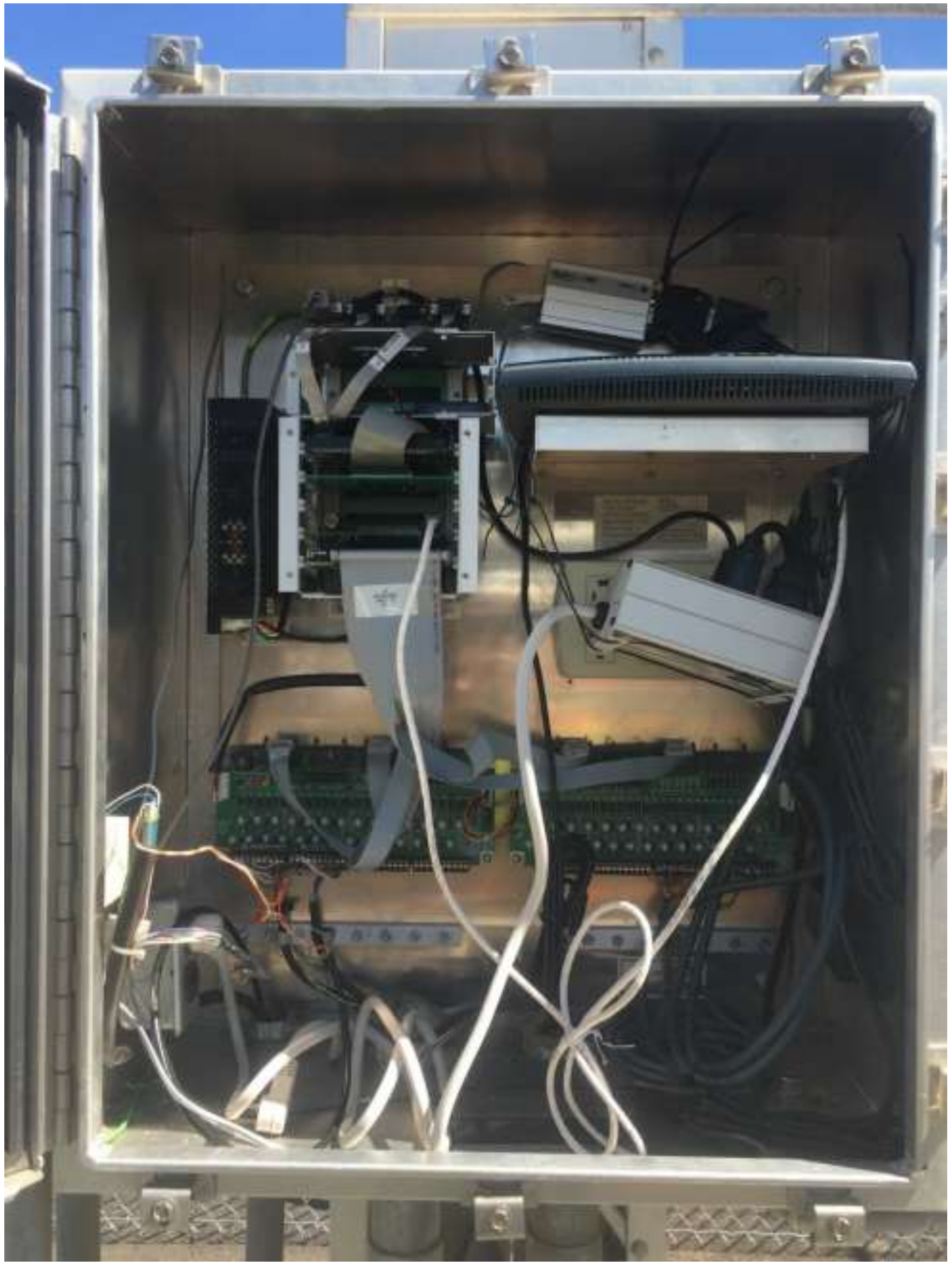
Older Site Example

- Visited this site April, 2016
- 20 years old!
- Kudos to Vendor
- Although original vendor has been sold (twice) since install
- Solution ended up being a reboot

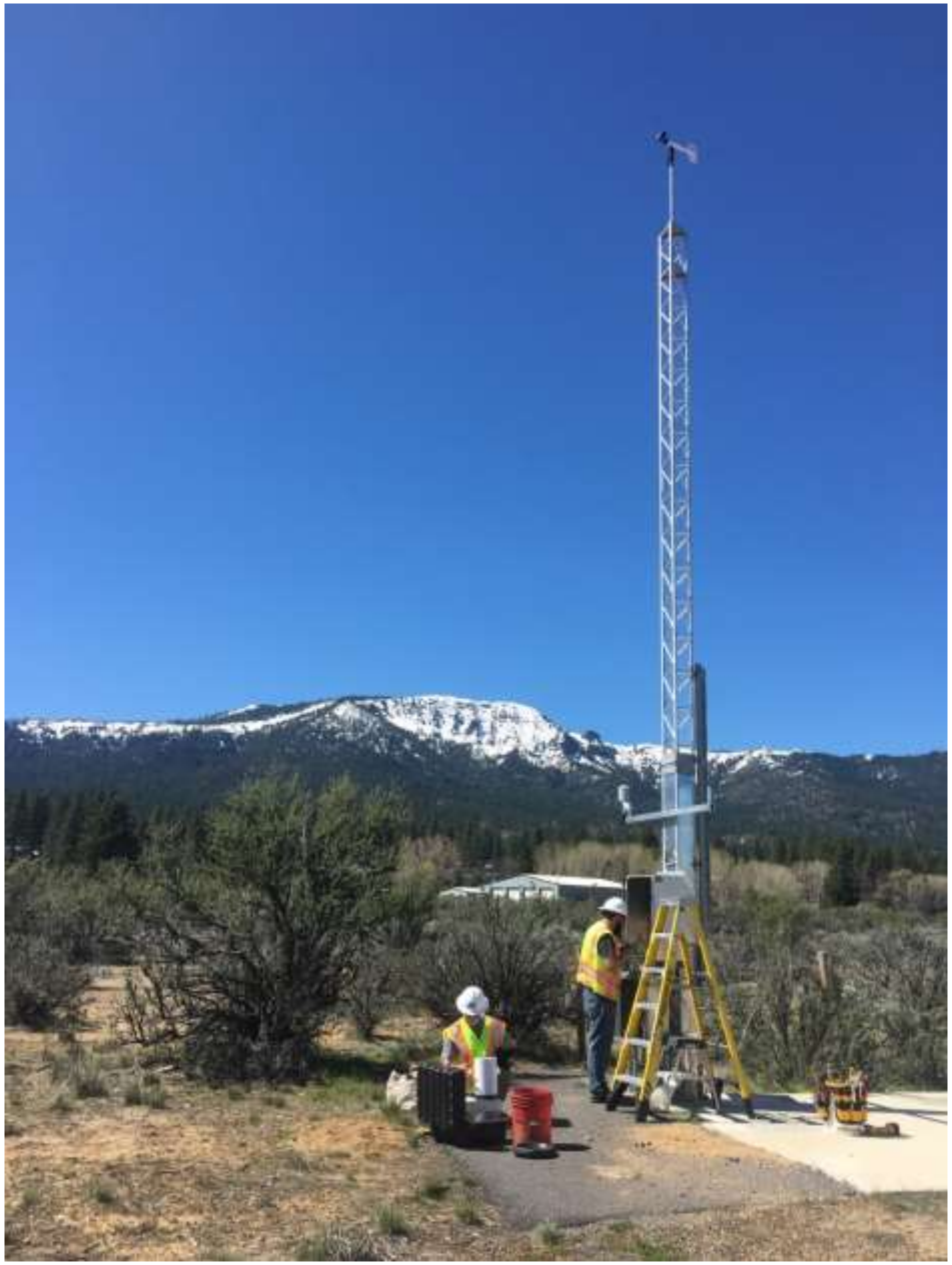


Older Site Example #2

- Router in same cabinet
- Modem in same cabinet
- Extra surge suppression, with cords and zip ties.



CR1000 RPU Deployment



RWIS Stations for District 2

- District 2 Roadside Weather.
  - ➢ We have highways from 200 feet above sea level to 6000 feet in elevation.
  - ➢ Temperature extremes range from 118F in the summer to -30F in the winter.
  - ➢ Cabinet temperatures will approach 145F on the hottest days.
  - ➢ Primary safety weather concerns are monitoring Icy Roads and High Wind Advisory areas.
    - ✓ We have two automated ICWS.
    - ✓ We have two primary corridors for frequent wind advisory.
    - ✓ Most of our RWIS stations have multiple Road Surface Sensors.
  - ➢ We have dry, high desert areas, as well as extremely wet areas during the rainy season.

RWIS Stations for District 2

- Instruments used at our RWIS stations;
  - Atmospheric = Temp/RH/Wind speed and direction/Rainfall
  - Pavement = Two generations of In-Pavement-Sensors (Vaisala FP2000 vs. Lufft Intelligent sensors) as well as the start of Out-Of-Pavement NIPS. (Mike will cover these details shortly.) Usually one per lane of traffic.
  - Subsurface = Any site with surface sensors will also contain one subsurface probe.
  - These provide most of the 58 NTCIP ESS variables. (the remaining are essentially static variables.)

RWIS Stations for District 2

NTCIP Wind Data

(12 Values)

essAvgWindSpeed

essAvgWindDirection

essSpotWindSpeed

essSpotWindDirection

essMaxWindGustSpeed

essMaxWindGustDir

windSensorAvgSpeed(i)

windSensorAvgDirection(i)

windSensorSpotSpeed(i)

windSensorSpotDirection(i)

windSensorGustDirection(i)

windSensorSituation(i)

NTCIP Rainfall Data

(10 Values)

essPrecipitationOneHour

essPrecipitationThreeHours

essPrecipitationSixHours

essPrecipitationTwelveHours

essPrecipitation24Hours

essPrecipRate

essPrecipYesNo

essPrecipitationStartTime

essPrecipitationEndTime

essPrecipSituation

NTCIP Atmospheric Data

(6 Values)

essAirTemperature(i)

essMaxTemp

essMinTemp

essRelativeHumidity

essWetBulbTemp

essDewPointTemp

RWIS Stations for District 2

NTCIP Pavement Data

(11 Values) x (number of sensors)

essSurfaceStatus(i)

essPavementSensorError(i)

essSurfaceTemperature(i)

essSurfaceWaterDepth(i)

essSurfaceIceOrWaterDepth(i)

essSurfaceSalinity(i)

essSurfaceFreezePoint(i)

essSubSurfaceTemperature(i)

essSubSurfaceDepth(i)

essSubSurfaceSensorLocation(i)

essSubSurfaceType(i)

NTCIP Static Data

19 Values

essNtcipCategory

essNtcipSiteDescription

essTypeofStation

essLongitude

essLatitude

essReferenceHeight

essNumTemperatureSensors

essTemperatureSensorHeight

windSensorTableNumSensors

windSensorHeight(1)

windSensorLocation(1)

essPavementType(i)

essPavementExposure(i)

essPavementElevation(i)

essPavementSensorType(i)

essPavementSensorLocation(i)

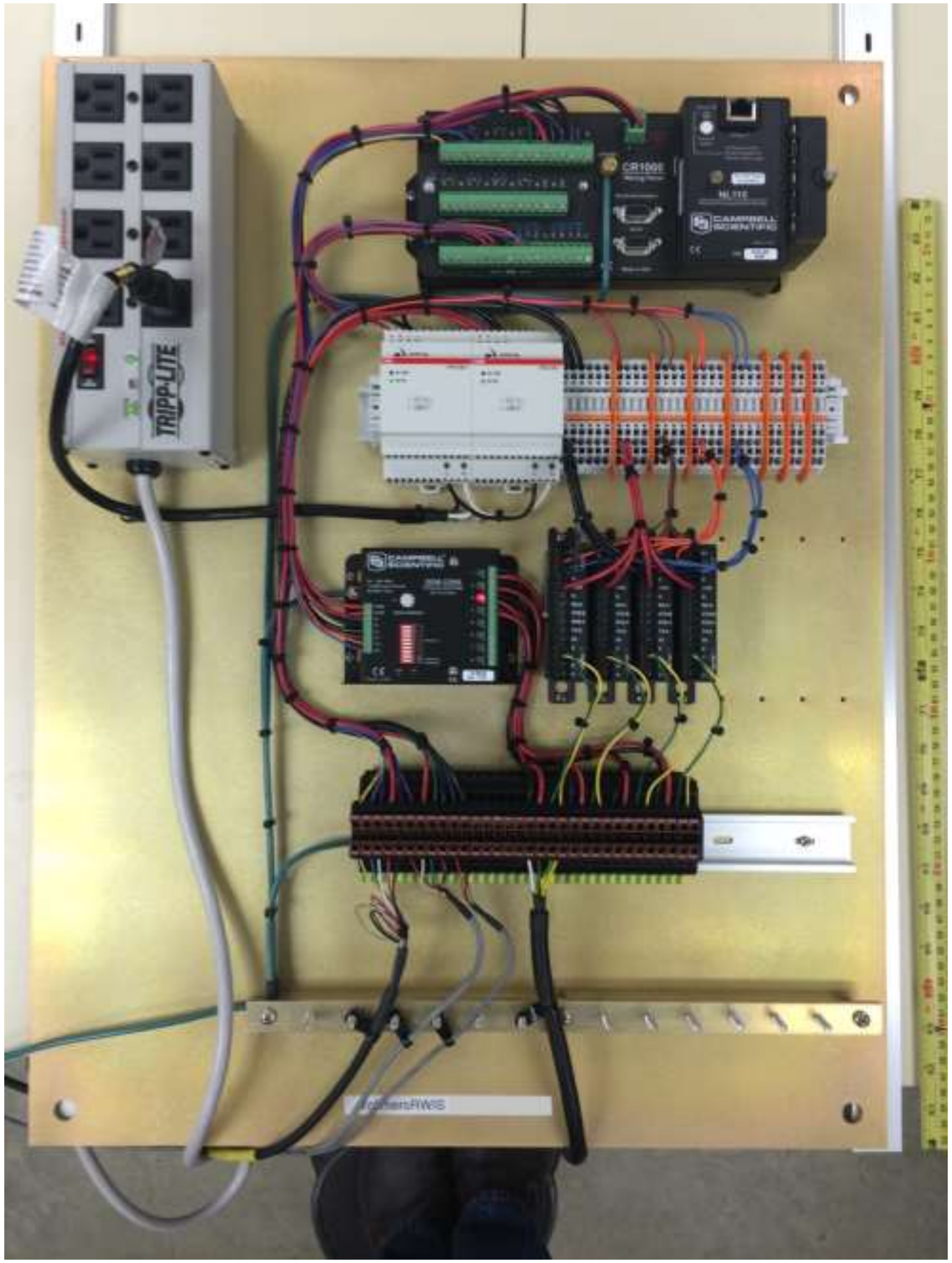
essSurfaceBlackIceSignal(i)

numEssSubsurfaceSensors

numEssPavementSensors

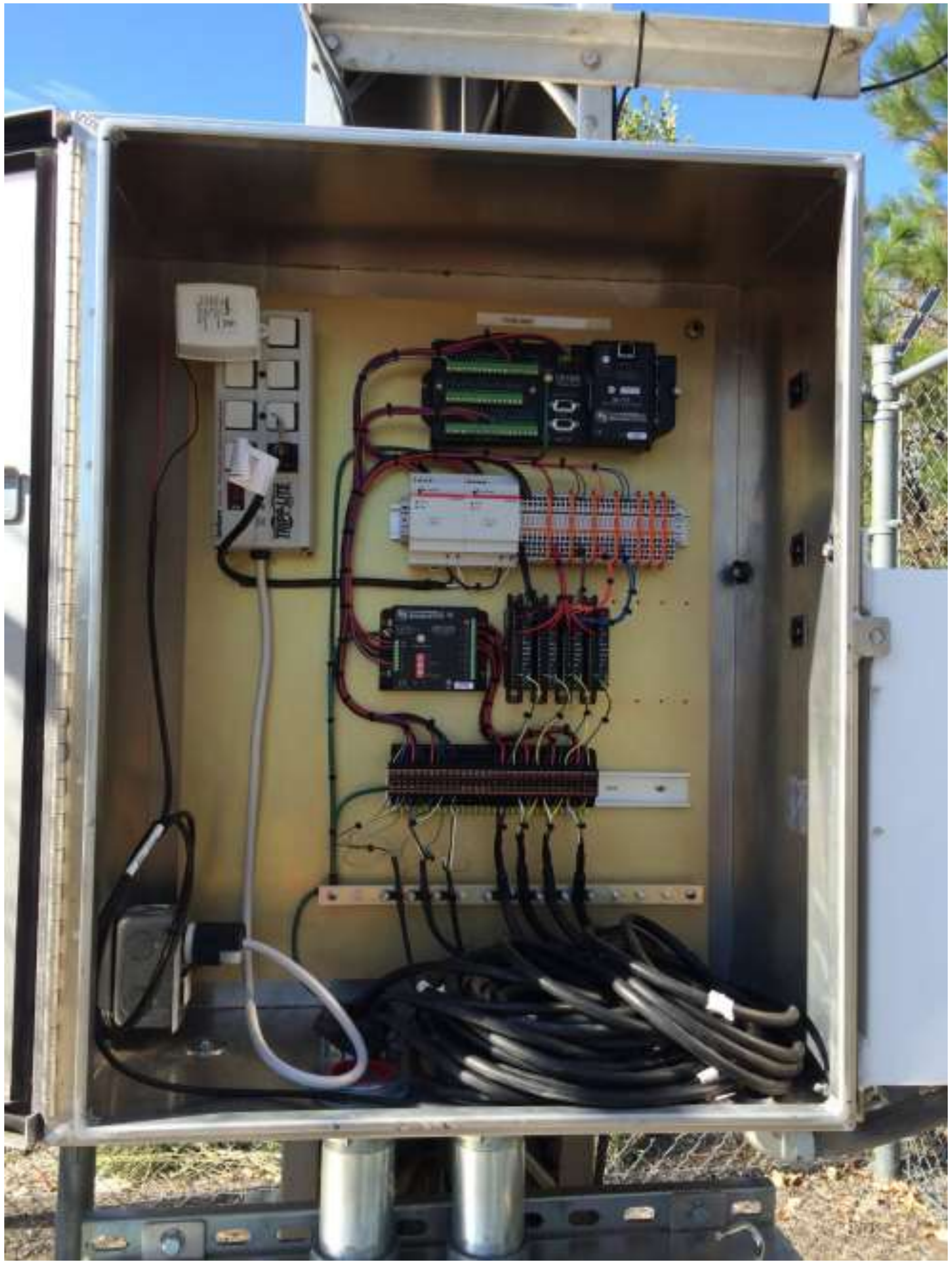
RPU – Caltrans D2 Campbell version 1

- Fits into standard Nema-4 cabinet. (30"x24"x12")
- Swappable with Vaisala RPU
- Supports Temp/RH/Wind/Rain
- Supports up to 8 RSS
- Note that the communication equipment is located elsewhere.
- Approximate Cost $7000



RPU – Caltrans D2 Campbell version 1

- Same RPU as previous slide located inside the NEMA 4 cabinet.



RWIS Station V1 - Design BOM, circa 2012

Sensor Instruments;

- RM Young 5103 Wind Sensor.
- RM Young 52202 Heated Tipping Bucket.
- HC2S3 Rotronic HygroClip2 Temperature/RH Probe.
- Lufft IRS21 Intelligent Road Surface Sensor with subsurface probe.

RWIS Station V1 - Design BOM, circa 2012

RPU Components;

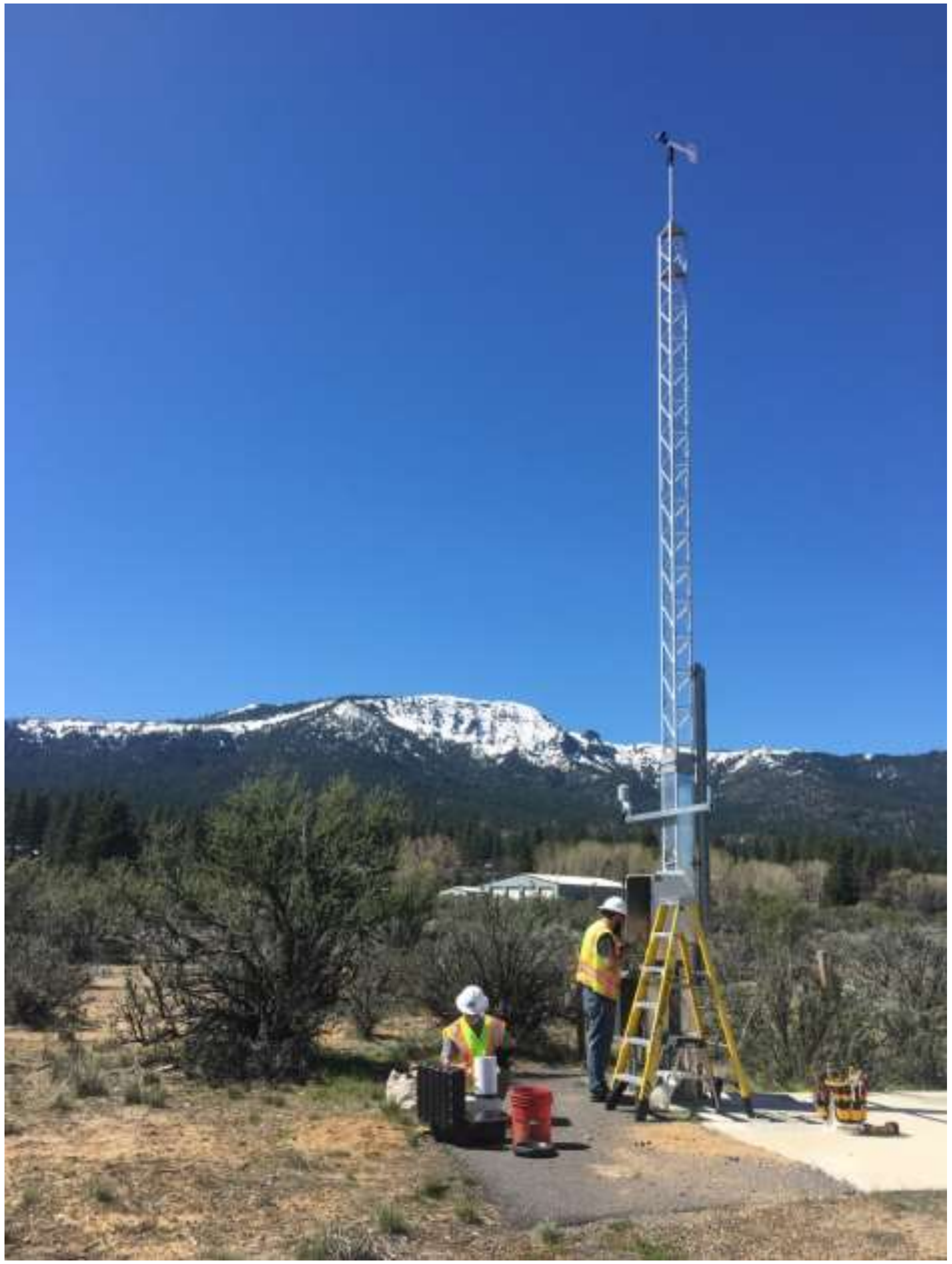
- Campbell Scientific CR1000-Extended Temp (-55 to +85C)
- Campbell Scientific NL116 Network Interface-Extended Temp
- (2) DC 12V Power Supplies
- Aluminum Fabricated Backboard; 3/16” thick; Alodine coated.
- (48) Surge Voltage Protection blocks, one for each external wired connection.
- (4) Campbell Scientific SIO1 SDM module (serial comm)
- Campbell Scientific SDM-CD8S DC controller
- Miscellaneous wiring and terminal blocks

Tower Instrumentation
Approximate Cost $2300

In-Pavement Sensing
Approximate Cost $8000 per lane

CR1000 RPU Deployment

- **Definitions and Background**
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



Campbell RWIS Station - Timeline

- 2009: Design started. (Credit to Ken Beals.)
- Mid-2012: Full scale RWIS station installed on tower next to our EELAB.
- End-2012: V1.0 Design completed. (again Credit to Ken Beals.)
- End-2012: Multiple construction projects had Lufft IRS21s in road.
- End-2012: IRS21 discontinued, IRS31 announced.
- End-2012: Retirement of key personnel at Caltrans and Campbell.
- End-2012: *Only one Campbell site active, at the EELAB.*

Campbell RWIS Station - Timeline

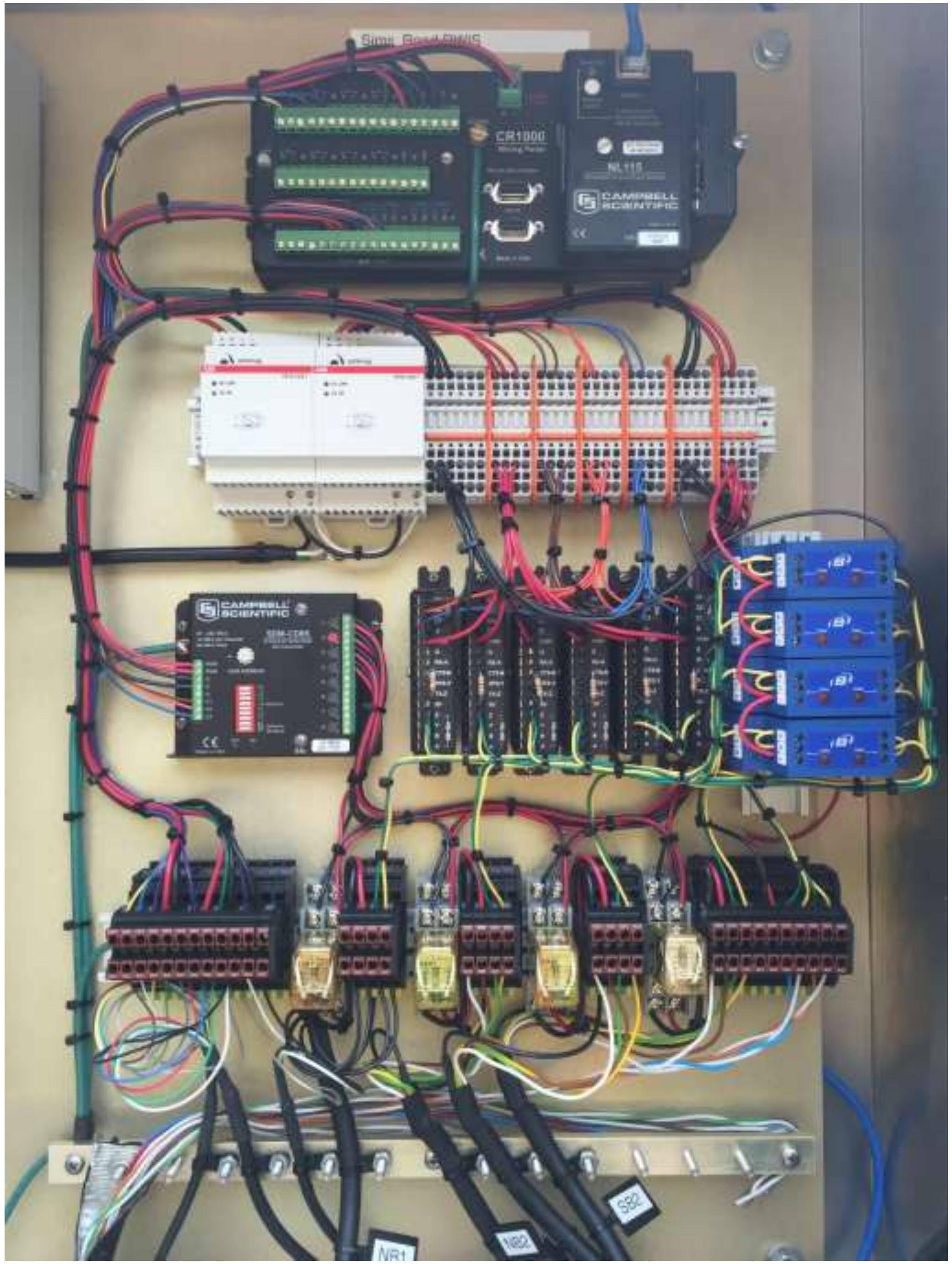
- Winter 2012/2013: No RSS connected to EELAB RPU, only able to monitor RPU with tower instruments.
- Fall-2013: First deployment of Campbell RPU at two field sites. North Weed and Vollmers. *Three total Campbell sites active.*
- Nov-2013: Lufft IRS31pro RSS announced.
  - IRS31 model silently phased out. Traded our inventory for the 'pro' model.
  - Lufft IRS31Pro represented a different hardware and software approach than the IRS21. More details to come.
- Dec 30th, 2013: First Lufft RSS Failure at North Weed.

Campbell RWIS Station - Timeline

- Mid-2014: Lufft RSS failure(s) diagnosed. (Ultimately water intrusion, but blame put on 'galvanic corrosion' – More to follow by Mike.)
- Mid-2014: Analysis of Lufft RSS installation procedures. New Caltrans publication created by Mike Beyer clarifies and improves saw cuts, epoxy, splicing, termination resistors, torque wrench, laying of cable.
- Mid-2014: New RPU circuit design related to Lufft IRS21 RSS galvanic corrosion issues completed. *Existing RPUs had to be rebuilt.* Complete isolation of 12V DC power with DPDT relays.
- Mid-2014: Support for Non-Invasive-Pavement-Sensor added to RPU (hardware and software).
- Fall-2014: Two competing NIPS (Vaisala and IceSight) installed at Sims Rd. Will be compared with Lufft IRS21s at same location.

RPU at Sims Rd.

- Power switching to Lufft IRS21 RSS now uses DPDT relays.
- B&B 232 to 485 converters used to optically isolate the serial communications.
- Four Lufft RSS sensors
- Two Out of Pavement sensors
- Running out of real estate!



Campbell RWIS Station - Timeline

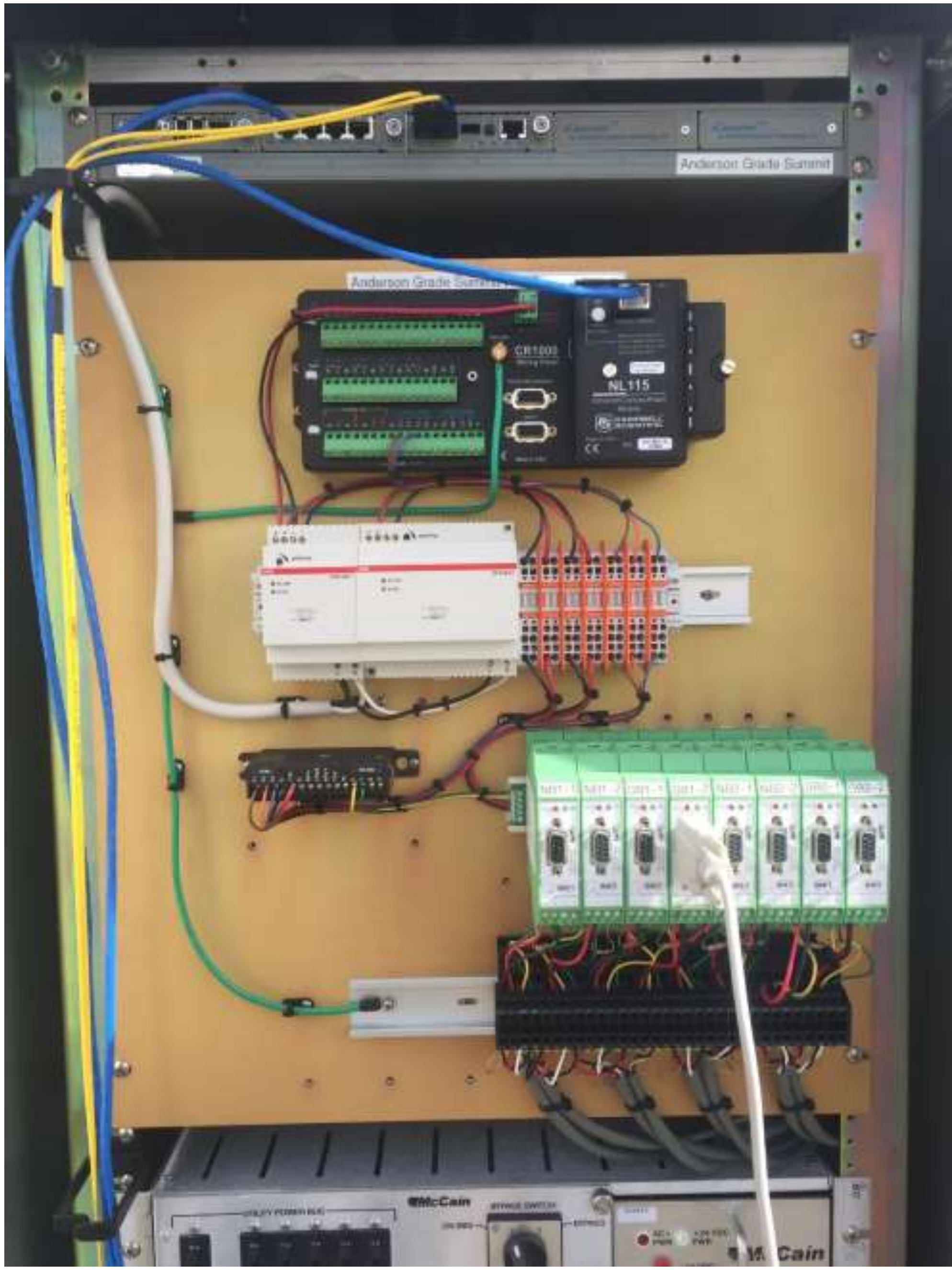
- End-2014: Three more Campbell RPUs implemented in field. Dunsmuir, Perez, Sims Rd. *Six total Campbell sites in field.*
- End-2014: At Dunsmuir and Perez sites are the first Lufft IRS31Pros.
  - Different 12V DC power connection to sensor. (No switching needed.)
  - Different software query method. (New binary protocol instead of ASCII)
- Winter 2014/2015: Experienced the most mild winter we have had in 20 years or more! Best freeze/ice event at Sims Rd. test site occurred in April 2015 with the temperature barely at freezing.

Campbell RWIS Station - Timeline

- Mid-2015: Miniaturized design of RPU completed for McCain ITS Node cabinet at Anderson Grade. Supports both versions of Lufft RSS. Another design update to now use the Lufft ISOCON module for both 12V DC power and RS485 communications.
- Fall-2015: Two more Campbell RPUs implemented at Anderson Grade. *Eight total Campbell sites in field.*
- Fall-2015: Texas Electronics tipping bucket implemented as a result of ease-of-calibration procedures. Good reaction to inch/hour rainfall rate.
- Winter 2015/2016:
  - NIPS comparison analysis at Snowman Summit RWIS station with older Vaisala FP2000 road sensors.
  - NIPS comparison analysis at Dunsmuir RWIS station with newest Lufft IRS31Pro road sensors.

Mini RPU at Anderson Grade

- Eight Lufft RSS sensors
- Lufft ISOCON and IRS21CON unit provides both power and data isolation to RSS.
- Only need a single Campbell Scientific SIO1 module.
- All RSS are physically connected to the same serial port.
- Both generations (IRS21 and IRS31Pro supported by this design.)
- Nice elegant solution.

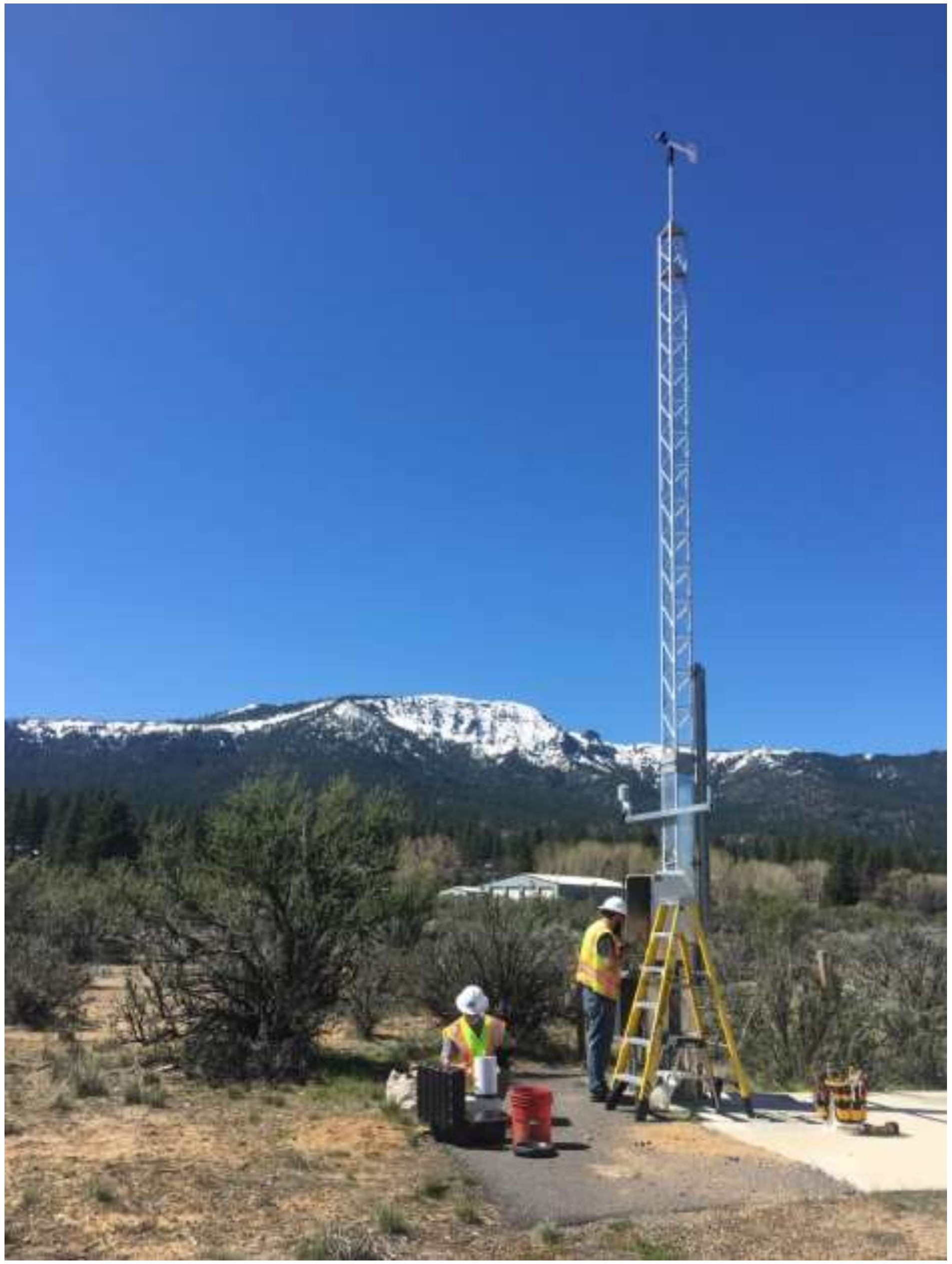


Campbell RWIS Station - Timeline

- Software: A continuous process. Last mentioned, but certainly not least!
- Software: Several major milestones as hardware and troubleshooting continued to evolve. The current CR Basic code is significantly different than original.
- Will cover this in more detail later in a software section.

CR1000 RPU Deployment

- Definitions and Background
- Why the Redesign?
- District 2 RWIS Station Profile
- Implementation Timeline
- Implementation Pros and Cons
- Software Notes
- Conclusion



Campbell RWIS Station V1 – Pros

- Tower instrumentation: Choices solid. Tipping bucket change was a nice enhancement and easy swap. No hardware or software changes were necessary.
- Campbell Scientific CR1000 Datalogger: Solid choice. Very robust. NTCIP version 2.0 support. Designed for harsh environments. Flexible in handling multiple instrument and serial device connections. Great vendor support. They always answer the phone and are practically in our time zone! Periodic firmware updates. Future looks even better with the CR6 Datalogger.
- RPU backboard: Good, but will be updated on next fabrication. Essentially there are an additional 20 screw holes added for flexibility in locating serial communication equipment and future datalogger support (CR6).
- Testing: Definitely a good idea to stage the deployment and not upgrade all sites at once. Waiting for all weather conditions requires patience and time!

Campbell RWIS Station V1 - Pros

- Lufft RSS: The removable housings were useful when swapping sensors for the model change, or the need to remove them prior to paving jobs.
- Lufft IRS31Pro RSS: sensor has some nice features.
  - UMB protocol consistent, predictable, checksum, choice of 20 different measurements with different units either averaged or single readings. The UMB protocol is not a proprietary secret, but actually has a pretty good detailed manual. (see appendix A for more details.)
  - The UMB protocol is used in all Lufft instruments and could ease development if someone were to choose those.
  - Software and hardware tool is available to load firmware and allows customization to change reaction to environment, such as transitions from dry to moist to wet. They can be calibrated.

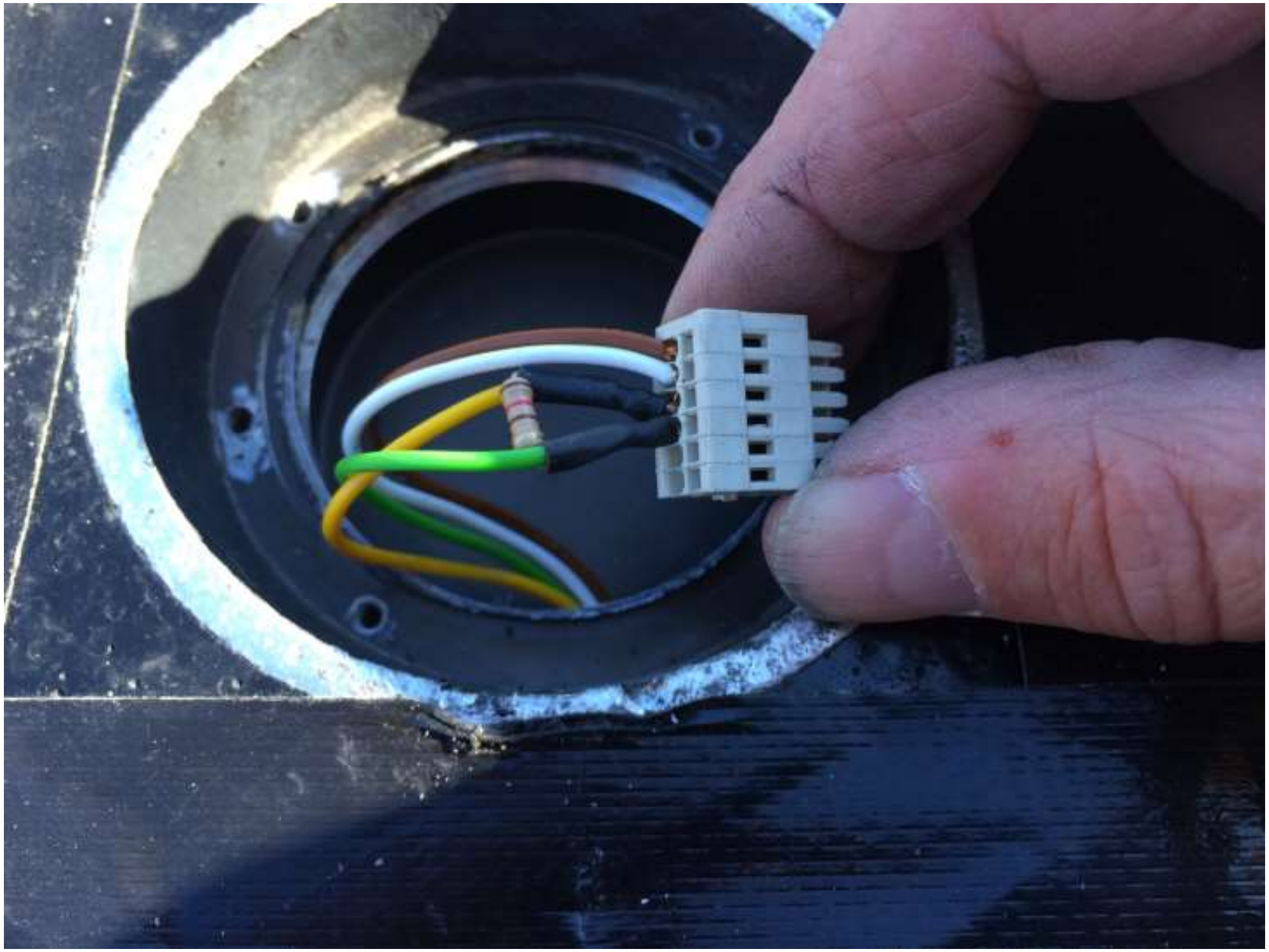
Campbell RWIS Station V1 - Cons

- Lufft RSS model change from IRS21 to IRS31Pro was significant. This affected both hardware design and software. We already had the older model installed in places, and needed to accommodate for both. (However the IRS21CON module makes this much easier.)
- The *installation* of a sensor in the pavement is of *utmost importance*; requires clear instructions and skilled personnel to monitor this critical process. Much rework was done by the ITS staff to install termination resistors and redo splices. To date, none of the existing sites have had their RSS installed by the new standard document.
- Troubleshooting a sensor in the pavement is difficult and resource intensive. Requires lane closures, multiple personnel, fair weather, perhaps multiple site visits. Hard to duplicate conditions in a controlled test environment; waiting for snow/ice is tedious.
- LUFFT RSS so far have a failure rate that is too high. Motivates us to continue the Out-of-Pavement Sensor evaluation.

Setting up for work on In-Pavement Sensor. Lane closure on I-5.

Mike Re-Splicing cable per new Caltrans D2 spec.

Installing a termination resistor in the road per new Caltrans D2 Spec.

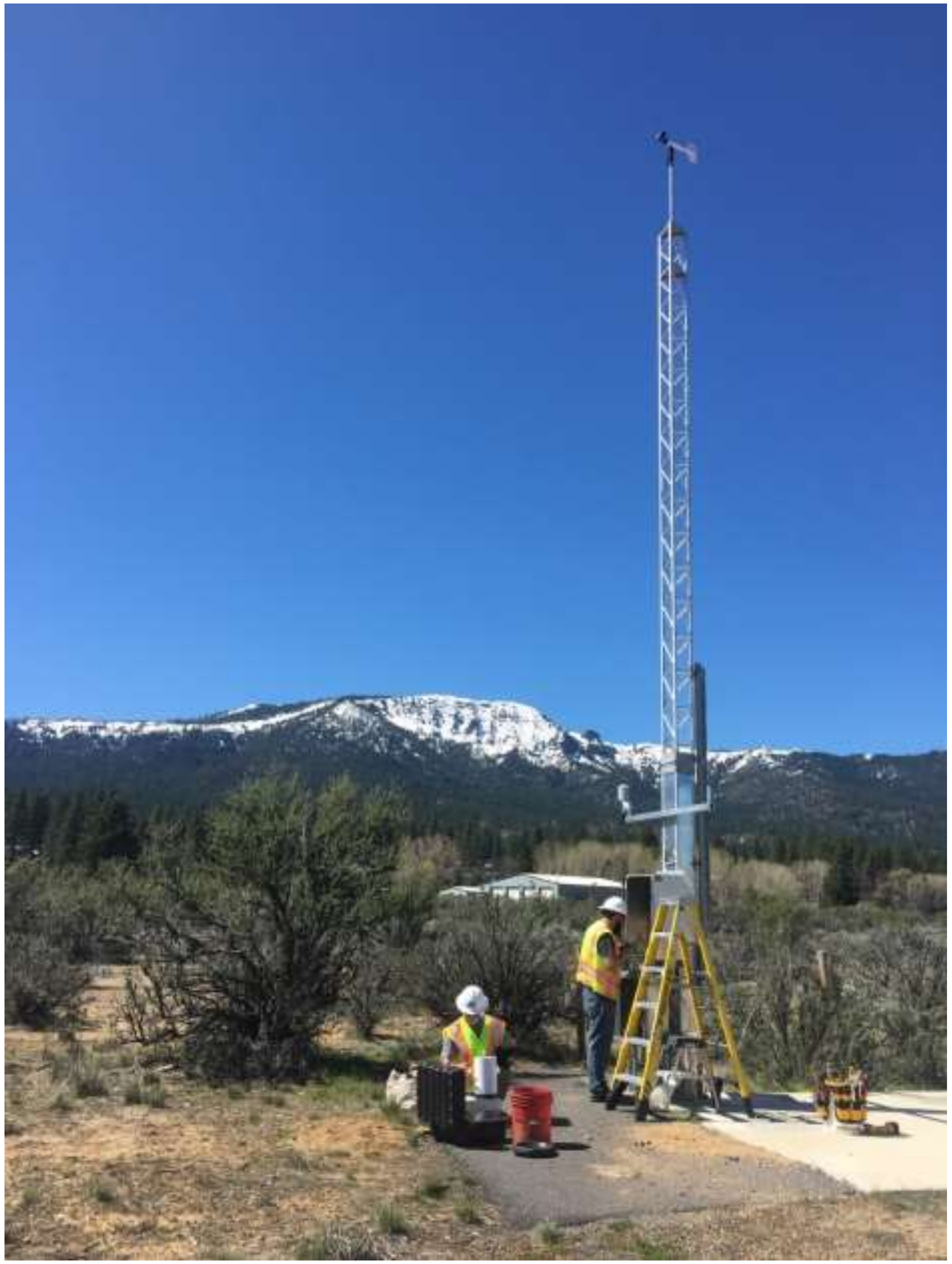


Campbell RWIS Station V1 - Gotchas

- The CR1000, CR Basic, tower instruments and pavement sensors all have a learning curve. The Campbell Scientific CR1000 manual is over 600 pages in length, as is the manual for the LoggerNet software support tool. Every instrument has a manual that needs to be understood. Each serial device will have a different way to request and parse the data which has to be developed and tested in the lab.
- The retirement of key personnel was a small setback to the learning, despite excellent documentation and notes provided by them. Resulted in much time reacting to situations rather than proactive until proficient.
- CR Basic code evolutionary changes were significant in time spent and scope. It has undergone multiple revisions and enhancements and could easily stand on its own presentation or even a week-long seminar!

CR1000 RPU Deployment

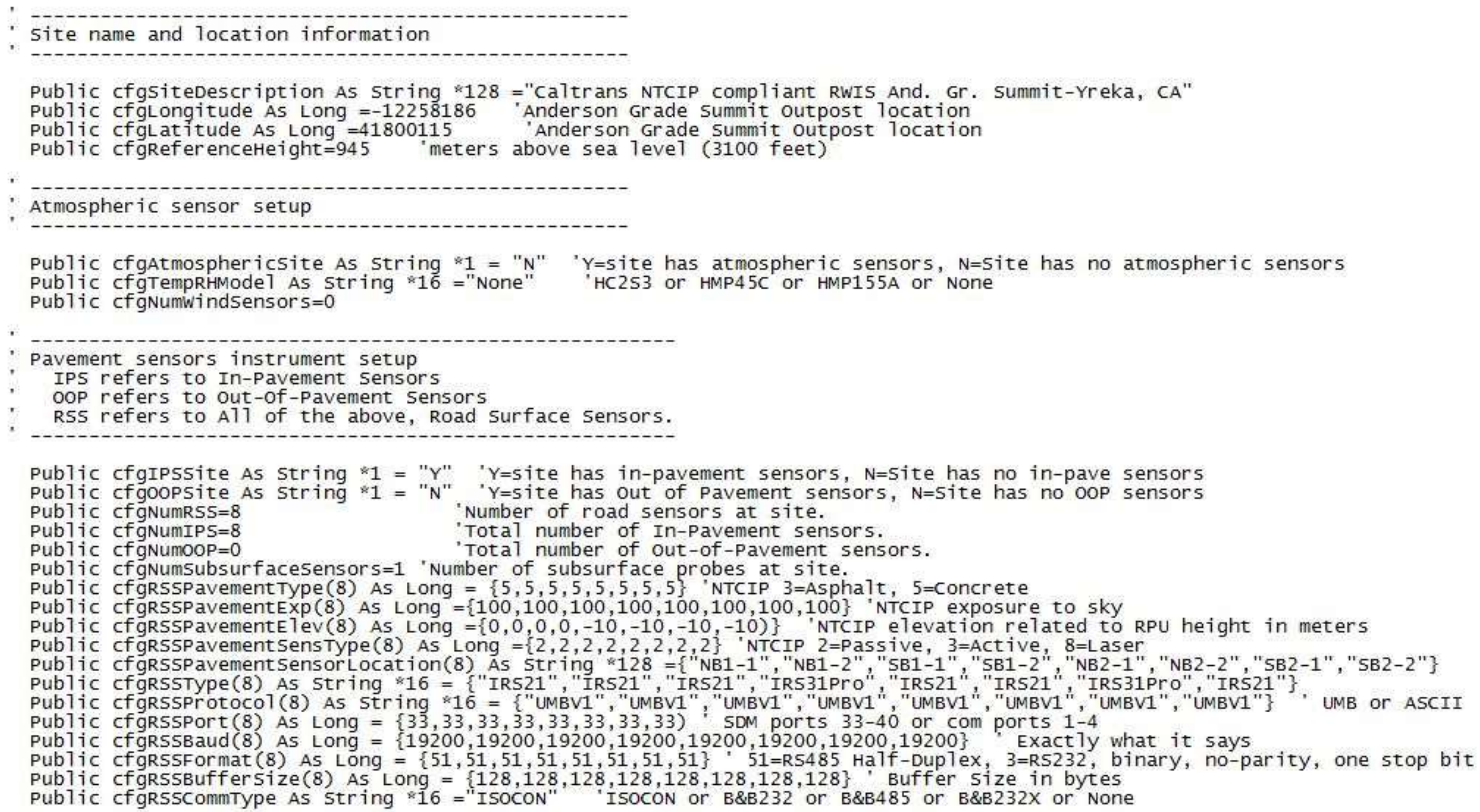
- **Definitions and Background**
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



Campbell RWIS Station - Software

- For reference, original code module could print on three pages of paper. Current code encompasses multiple modules and will print on over 30 pages of paper. (But is actually easier to implement and understand.)
- Even the first two Campbell sites had differences, and therefore each needed a different program under the original single module design. Managing multiple programs for multiple sites would not be desirable.
- Site configuration file developed to accommodate site differences.

CR Basic code for SiteConfig file. Handles all differences for each site.

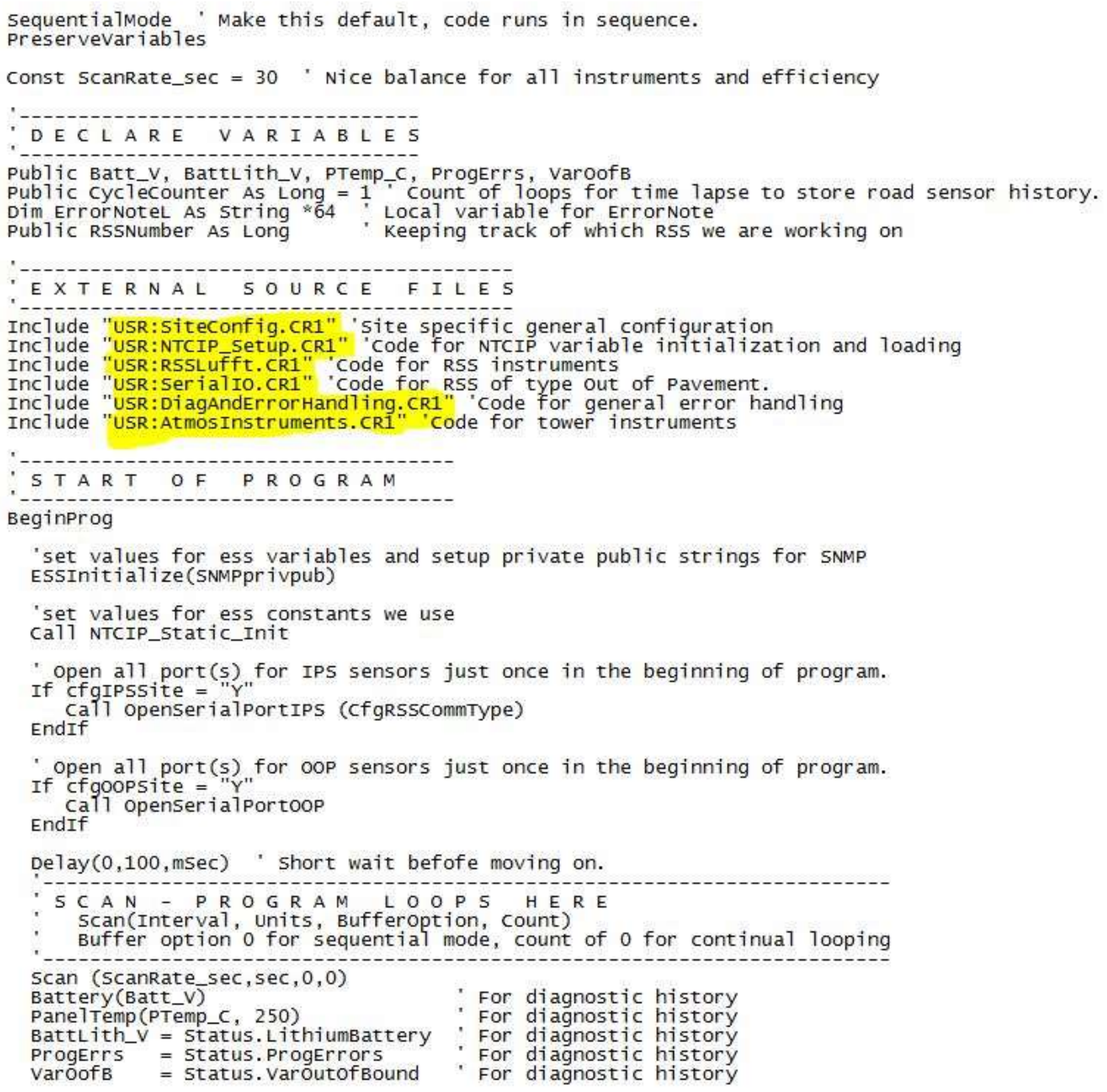


```
' ---------------------------------------------------
' Site name and location information
' ---------------------------------------------------

  Public cfgSiteDescription As String *128 ="Caltrans NTCIP compliant RWIS And. Gr. Summit-Yreka, CA"
  Public cfgLongitude As Long =-12258186   'Anderson Grade Summit Outpost location
  Public cfgLatitude As Long =41800115      'Anderson Grade Summit Outpost location
  Public cfgReferenceHeight=945    'meters above sea level (3100 feet)

' ---------------------------------------------------
' Atmospheric sensor setup
' ---------------------------------------------------

  Public cfgAtmosphericSite As String *1 = "N"  'Y=site has atmospheric sensors, N=Site has no atmospheric sensors
  Public cfgTempRHModel As String *16 ="None"    'HC2S3 or HMP45C or HMP155A or None
  Public cfgNumWindSensors=0

' -------------------------------------------------------
' Pavement sensors instrument setup
'   IPS refers to In-Pavement Sensors
'   OOP refers to Out-Of-Pavement Sensors
'   RSS refers to All of the above, Road Surface Sensors.
' -------------------------------------------------------

  Public cfgIPSSite As String *1 = "Y"  'Y=site has in-pavement sensors, N=Site has no in-pave sensors
  Public cfgOOPSite As String *1 = "N"  'Y=site has Out of Pavement sensors, N=Site has no OOP sensors
  Public cfgNumRSS=8                    'Number of road sensors at site.
  Public cfgNumIPS=8                    'Total number of In-Pavement sensors.
  Public cfgNumOOP=0                    'Total number of Out-of-Pavement sensors.
  Public cfgNumSubsurfaceSensors=1 'Number of subsurface probes at site.
  Public cfgRSSPavementType(8) As Long = {5,5,5,5,5,5,5,5} 'NTCIP 3=Asphalt, 5=Concrete
  Public cfgRSSPavementExp(8) As Long ={100,100,100,100,100,100,100,100} 'NTCIP exposure to sky
  Public cfgRSSPavementElev(8) As Long ={0,0,0,0,-10,-10,-10,-10)}  'NTCIP elevation related to RPU height in meters
  Public cfgRSSPavementSensType(8) As Long ={2,2,2,2,2,2,2,2} 'NTCIP 2=Passive, 3=Active, 8=Laser
  Public cfgRSSPavementSensorLocation(8) As String *128 ={"NB1-1","NB1-2","SB1-1","SB1-2","NB2-1","NB2-2","SB2-1","SB2-2"}
  Public cfgRSSType(8) As String *16 = {"IRS21","IRS21","IRS21","IRS31Pro","IRS21","IRS21","IRS31Pro","IRS21"}
  Public cfgRSSProtocol(8) As String *16 = {"UMBV1","UMBV1","UMBV1","UMBV1","UMBV1","UMBV1","UMBV1","UMBV1"}  ' UMB or ASCII
  Public cfgRSSPort(8) As Long = {33,33,33,33,33,33,33,33)  ' SDM ports 33-40 or com ports 1-4
  Public cfgRSSBaud(8) As Long = {19200,19200,19200,19200,19200,19200,19200,19200}  ' Exactly what it says
  Public cfgRSSFormat(8) As Long = {51,51,51,51,51,51,51,51} ' 51=RS485 Half-Duplex, 3=RS232, binary, no-parity, one stop bit
  Public cfgRSSBufferSize(8) As Long = {128,128,128,128,128,128,128,128} ' Buffer Size in bytes
  Public cfgRSSCommType As String *16 ="ISOCON"   'ISOCON or B&B232 or B&B485 or B&B232X or None
```

Campbell RWIS Station - Software

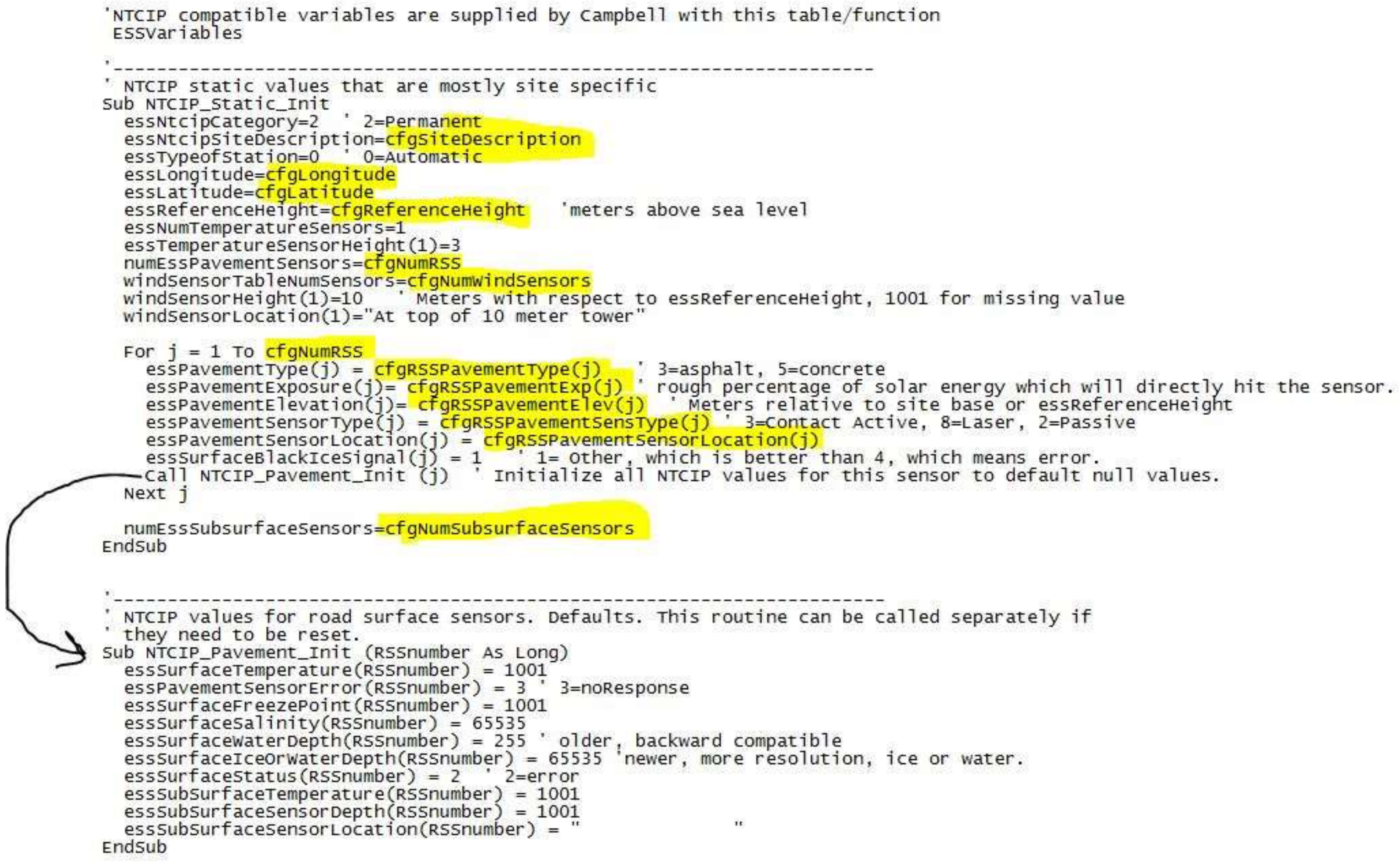
- Modularized design continued to evolve and proved beneficial in making code adjustments to one area, while not affecting another area. Easier to focus on small portions of code.
- There is currently only one main module, which is the only one that compiles, and inserts the sub modules with the CR Basic “Include” statement.
- Standard sub modules are;
  - Site Configuration
  - NTCIP setup/initialization
  - Serial IO (Setup and write/read to serial ports)
  - Tower Instruments
  - In-Pavement Instruments
  - Non-Invasive Pavement Instruments

Snippet of part of main module. Sub modules are highlighted.



```
SequentialMode  ' Make this default, code runs in sequence.
PreserveVariables

Const ScanRate_sec = 30  ' Nice balance for all instruments and efficiency

'--------------------------------
' D E C L A R E   V A R I A B L E S
'--------------------------------
Public Batt_V, BattLith_V, PTemp_C, ProgErrs, VarOofB
Public CycleCounter As Long = 1 ' Count of loops for time lapse to store road sensor history.
Dim ErrorNoteL As String *64   ' Local variable for ErrorNote
Public RSSNumber As Long       ' Keeping track of which RSS we are working on

'----------------------------------------
' E X T E R N A L   S O U R C E   F I L E S
'----------------------------------------
Include "USR:SiteConfig.CR1" 'Site specific general configuration
Include "USR:NTCIP_Setup.CR1" 'Code for NTCIP variable initialization and loading
Include "USR:RSSLufft.CR1" 'Code for RSS instruments
Include "USR:SerialIO.CR1" 'Code for RSS of type Out of Pavement.
Include "USR:DiagAndErrorHandling.CR1" 'Code for general error handling
Include "USR:AtmosInstruments.CR1" 'Code for tower instruments

'-----------------------------------
' S T A R T   O F   P R O G R A M
'-----------------------------------
BeginProg

  'set values for ess variables and setup private public strings for SNMP
  ESSInitialize(SNMPprivpub)

  'set values for ess constants we use
  Call NTCIP_Static_Init

  ' Open all port(s) for IPS sensors just once in the beginning of program.
  If cfgIPSSite = "Y"
     Call OpenSerialPortIPS (CfgRSSCommType)
  EndIf

  ' Open all port(s) for OOP sensors just once in the beginning of program.
  If cfgOOPSite = "Y"
     Call OpenSerialPortOOP
  EndIf

  Delay(0,100,mSec)  ' Short wait befofe moving on.
  '-----------------------------------------------------------------------------
  ' S C A N  -  P R O G R A M   L O O P S   H E R E
  '   Scan(Interval, Units, BufferOption, Count)
  '   Buffer option 0 for sequential mode, count of 0 for continual looping
  '-----------------------------------------------------------------------------
  Scan (ScanRate_sec,sec,0,0)
  Battery(Batt_V)                     ' For diagnostic history
  PanelTemp(PTemp_C, 250)             ' For diagnostic history
  BattLith_V = Status.LithiumBattery  ' For diagnostic history
  ProgErrs   = Status.ProgErrors      ' For diagnostic history
  VarOofB    = Status.VarOutOfBound   ' For diagnostic history
```

Setting up NTCIP variables on boot. Configuration values are highlighted

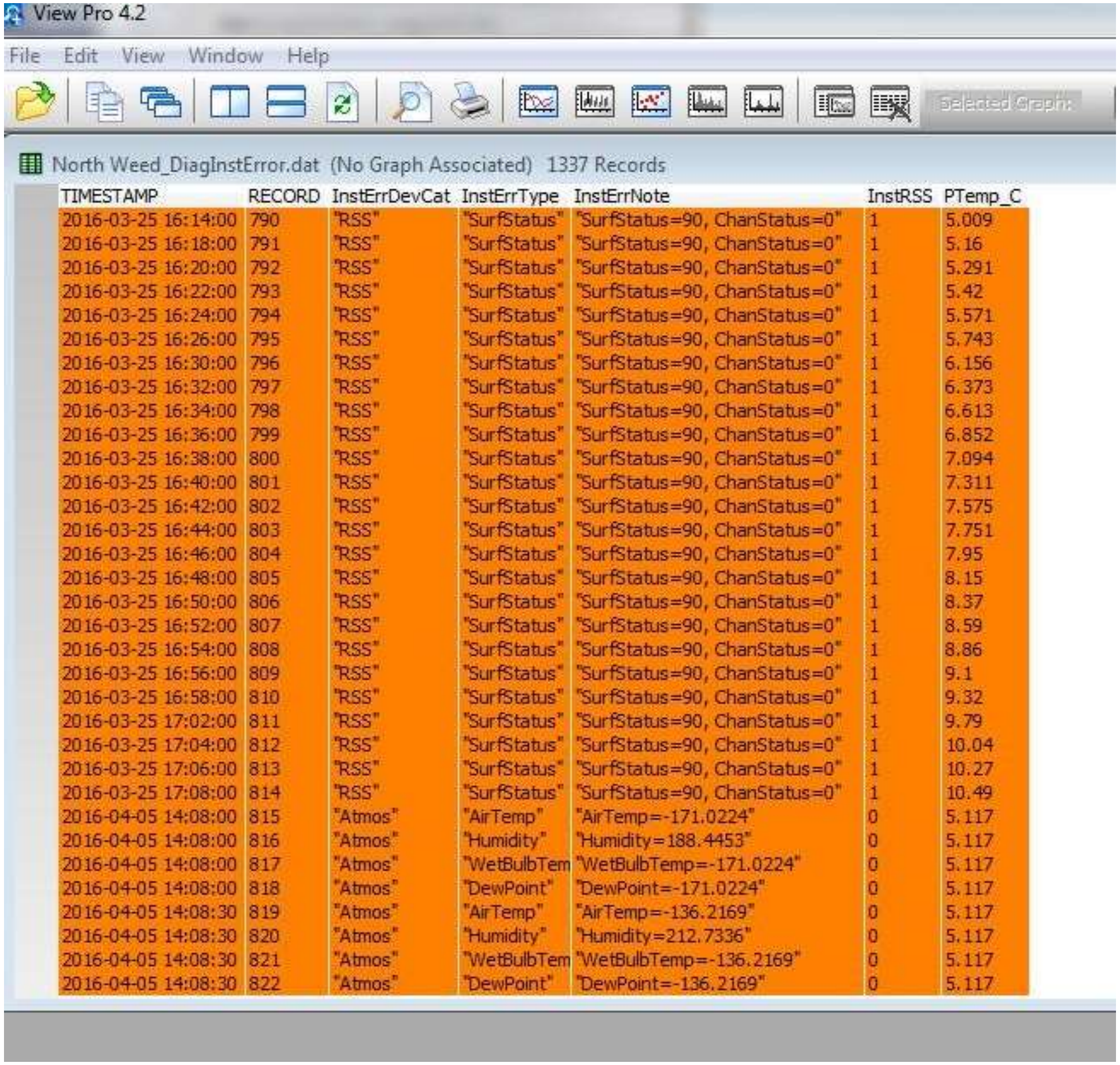


```
'NTCIP compatible variables are supplied by Campbell with this table/function
 ESSVariables

'------------------------------------------------------------------------
' NTCIP static values that are mostly site specific
Sub NTCIP_Static_Init
  essNtcipCategory=2   ' 2=Permanent
  essNtcipSiteDescription=cfgSiteDescription
  essTypeofStation=0   ' 0=Automatic
  essLongitude=cfgLongitude
  essLatitude=cfgLatitude
  essReferenceHeight=cfgReferenceHeight   'meters above sea level
  essNumTemperatureSensors=1
  essTemperatureSensorHeight(1)=3
  numEssPavementSensors=cfgNumRSS
  windSensorTableNumSensors=cfgNumWindSensors
  windSensorHeight(1)=10    ' Meters with respect to essReferenceHeight, 1001 for missing value
  windSensorLocation(1)="At top of 10 meter tower"

  For j = 1 To cfgNumRSS
    essPavementType(j) = cfgRSSPavementType(j)    ' 3=asphalt, 5=concrete
    essPavementExposure(j)= cfgRSSPavementExp(j) ' rough percentage of solar energy which will directly hit the sensor.
    essPavementElevation(j)= cfgRSSPavementElev(j)   ' Meters relative to site base or essReferenceHeight
    essPavementSensorType(j) = cfgRSSPavementSensType(j) ' 3=Contact Active, 8=Laser, 2=Passive
    essPavementSensorLocation(j) = cfgRSSPavementSensorLocation(j)
    essSurfaceBlackIceSignal(j) = 1    ' 1= other, which is better than 4, which means error.
    Call NTCIP_Pavement_Init (j)   ' Initialize all NTCIP values for this sensor to default null values.
  Next j

  numEssSubsurfaceSensors=cfgNumSubsurfaceSensors
EndSub


'------------------------------------------------------------------------
' NTCIP values for road surface sensors. Defaults. This routine can be called separately if
' they need to be reset.
Sub NTCIP_Pavement_Init (RSSnumber As Long)
  essSurfaceTemperature(RSSnumber) = 1001
  essPavementSensorError(RSSnumber) = 3 ' 3=noResponse
  essSurfaceFreezePoint(RSSnumber) = 1001
  essSurfaceSalinity(RSSnumber) = 65535
  essSurfaceWaterDepth(RSSnumber) = 255 ' older, backward compatible
  essSurfaceIceOrWaterDepth(RSSnumber) = 65535 'newer, more resolution, ice or water.
  essSurfaceStatus(RSSnumber) = 2   ' 2=error
  essSubSurfaceTemperature(RSSnumber) = 1001
  essSubSurfaceSensorDepth(RSSnumber) = 1001
  essSubSurfaceSensorLocation(RSSnumber) = "              "
EndSub
```

Campbell RWIS Station - Software

- Refined error checking for NTCIP values, as well as tower instruments and pavement instruments.
  - Combine detailed error reporting into a single file and manage by type of instrument, and type of error. Only need to go to one place to look for errors. Date and timestamps are there with a descriptive text string explaining the detected error.
  - 24 hour summary diagnostic file records totals for any errors, warnings, panel temperature ranges. Provides a history of error counts on each day. Are numbers increasing? Decreasing? Helps zero in on errors more quickly, rather than having to peruse through tons of information. Reset these counts each day.
  - Diagnostics for low 3.6v lithium battery and/or DC input voltage to datalogger.

Errors at North Weed RPU



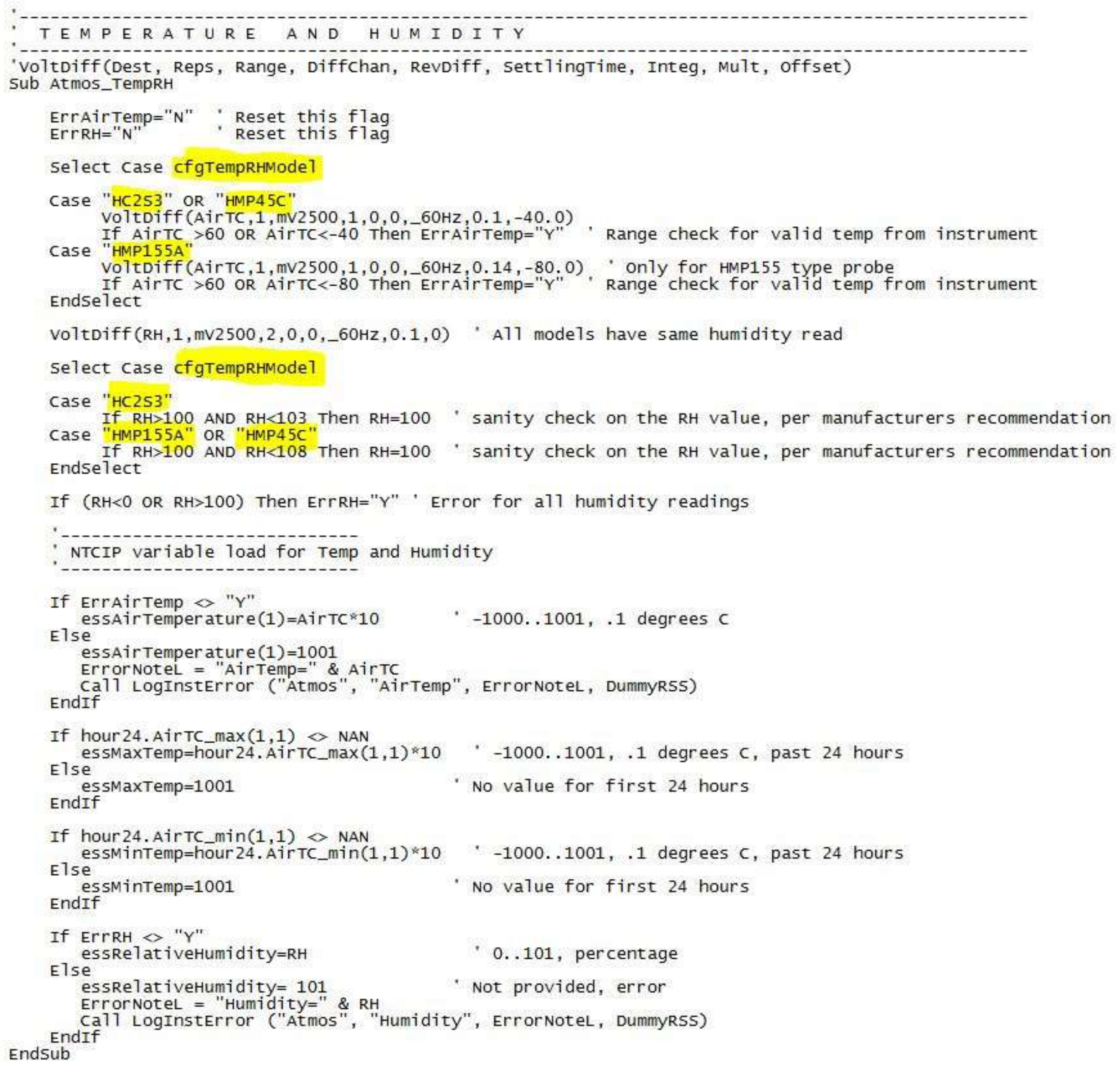
North Weed_DiagInstError.dat (No Graph Associated) 1337 Records

TIMESTAMP	RECORD	InstErrDevCat	InstErrType	InstErrNote	InstRSS	PTemp_C
2016-03-25 16:14:00	790	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.009
2016-03-25 16:18:00	791	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.16
2016-03-25 16:20:00	792	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.291
2016-03-25 16:22:00	793	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.42
2016-03-25 16:24:00	794	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.571
2016-03-25 16:26:00	795	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	5.743
2016-03-25 16:30:00	796	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	6.156
2016-03-25 16:32:00	797	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	6.373
2016-03-25 16:34:00	798	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	6.613
2016-03-25 16:36:00	799	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	6.852
2016-03-25 16:38:00	800	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	7.094
2016-03-25 16:40:00	801	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	7.311
2016-03-25 16:42:00	802	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	7.575
2016-03-25 16:44:00	803	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	7.751
2016-03-25 16:46:00	804	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	7.95
2016-03-25 16:48:00	805	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	8.15
2016-03-25 16:50:00	806	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	8.37
2016-03-25 16:52:00	807	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	8.59
2016-03-25 16:54:00	808	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	8.86
2016-03-25 16:56:00	809	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	9.1
2016-03-25 16:58:00	810	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	9.32
2016-03-25 17:02:00	811	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	9.79
2016-03-25 17:04:00	812	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	10.04
2016-03-25 17:06:00	813	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	10.27
2016-03-25 17:08:00	814	"RSS"	"SurfStatus"	"SurfStatus=90, ChanStatus=0"	1	10.49
2016-04-05 14:08:00	815	"Atmos"	"AirTemp"	"AirTemp=-171.0224"	0	5.117
2016-04-05 14:08:00	816	"Atmos"	"Humidity"	"Humidity=188.4453"	0	5.117
2016-04-05 14:08:00	817	"Atmos"	"WetBulbTem	"WetBulbTemp=-171.0224"	0	5.117
2016-04-05 14:08:00	818	"Atmos"	"DewPoint"	"DewPoint=-171.0224"	0	5.117
2016-04-05 14:08:30	819	"Atmos"	"AirTemp"	"AirTemp=-136.2169"	0	5.117
2016-04-05 14:08:30	820	"Atmos"	"Humidity"	"Humidity=212.7336"	0	5.117
2016-04-05 14:08:30	821	"Atmos"	"WetBulbTem	"WetBulbTemp=-136.2169"	0	5.117
2016-04-05 14:08:30	822	"Atmos"	"DewPoint"	"DewPoint=-136.2169"	0	5.117

Temperature and Humidity code section.

Site configuration variables highlighted.

You can also see the error checking and NTCIP variable load from the instrument.

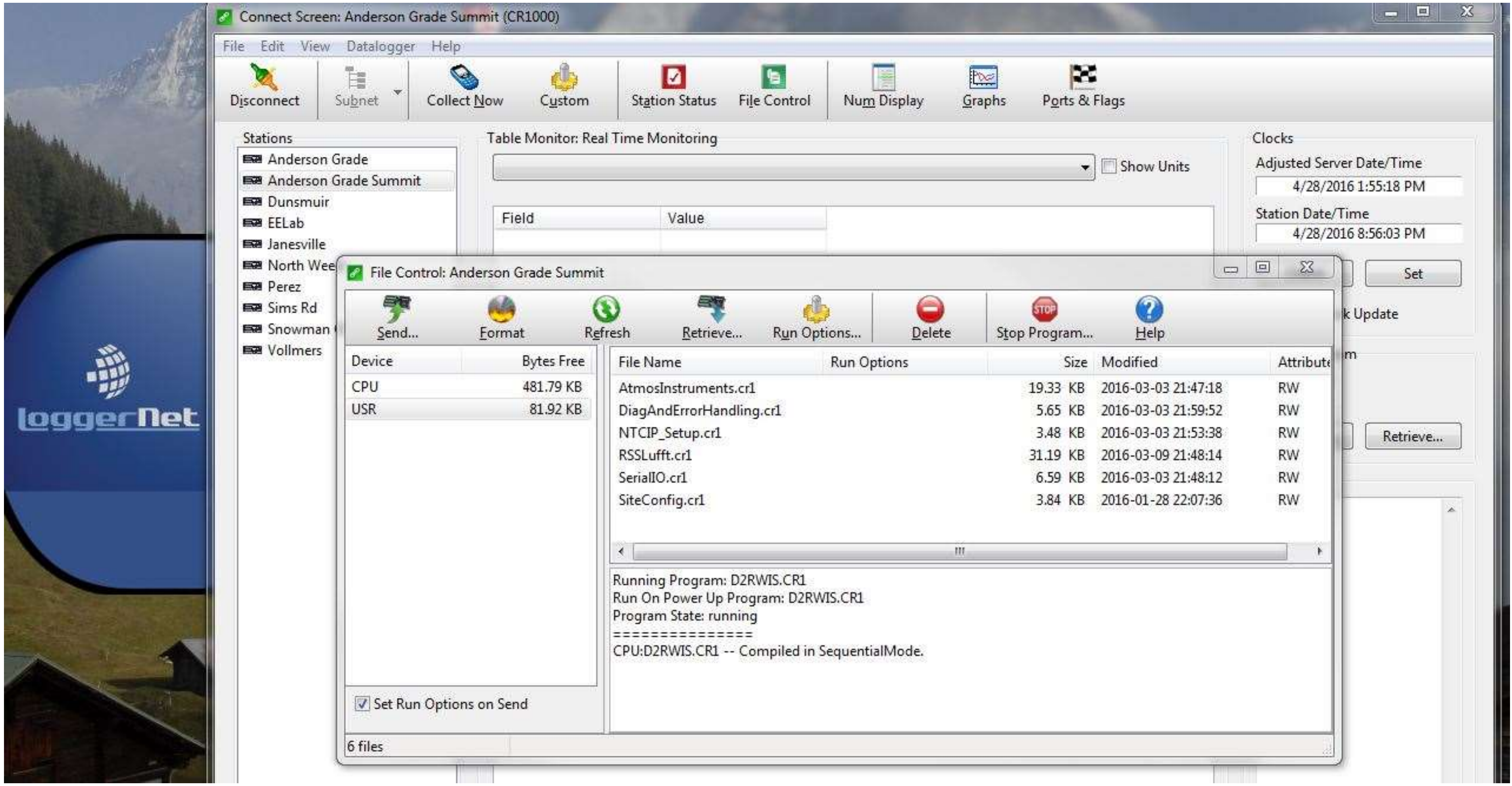


```
'
'-----------------------------------------------------------------------------------------------
'  T E M P E R A T U R E   A N D   H U M I D I T Y
'-----------------------------------------------------------------------------------------------
'VoltDiff(Dest, Reps, Range, DiffChan, RevDiff, SettlingTime, Integ, Mult, Offset)
Sub Atmos_TempRH

    ErrAirTemp="N"   ' Reset this flag
    ErrRH="N"        ' Reset this flag

    Select Case cfgTempRHModel

    Case "HC2S3" OR "HMP45C"
         VoltDiff(AirTC,1,mV2500,1,0,0,_60Hz,0.1,-40.0)
         If AirTC >60 OR AirTC<-40 Then ErrAirTemp="Y"   ' Range check for valid temp from instrument
    Case "HMP155A"
         VoltDiff(AirTC,1,mV2500,1,0,0,_60Hz,0.14,-80.0)  ' Only for HMP155 type probe
         If AirTC >60 OR AirTC<-80 Then ErrAirTemp="Y"   ' Range check for valid temp from instrument
    EndSelect

    VoltDiff(RH,1,mV2500,2,0,0,_60Hz,0.1,0)  ' All models have same humidity read

    Select Case cfgTempRHModel

    Case "HC2S3"
         If RH>100 AND RH<103 Then RH=100   ' sanity check on the RH value, per manufacturers recommendation
    Case "HMP155A" OR "HMP45C"
         If RH>100 AND RH<108 Then RH=100   ' sanity check on the RH value, per manufacturers recommendation
    EndSelect

    If (RH<0 OR RH>100) Then ErrRH="Y" ' Error for all humidity readings

    '-----------------------------
    ' NTCIP variable load for Temp and Humidity
    '-----------------------------

    If ErrAirTemp <> "Y"
       essAirTemperature(1)=AirTC*10        ' -1000..1001, .1 degrees C
    Else
       essAirTemperature(1)=1001
       ErrorNoteL = "AirTemp=" & AirTC
       Call LogInstError ("Atmos", "AirTemp", ErrorNoteL, DummyRSS)
    EndIf

    If hour24.AirTC_max(1,1) <> NAN
       essMaxTemp=hour24.AirTC_max(1,1)*10   ' -1000..1001, .1 degrees C, past 24 hours
    Else
       essMaxTemp=1001                      ' No value for first 24 hours
    EndIf

    If hour24.AirTC_min(1,1) <> NAN
       essMinTemp=hour24.AirTC_min(1,1)*10   ' -1000..1001, .1 degrees C, past 24 hours
    Else
       essMinTemp=1001                      ' No value for first 24 hours
    EndIf

    If ErrRH <> "Y"
       essRelativeHumidity=RH                ' 0..101, percentage
    Else
       essRelativeHumidity= 101             ' Not provided, error
       ErrorNoteL = "Humidity=" & RH
       Call LogInstError ("Atmos", "Humidity", ErrorNoteL, DummyRSS)
    EndIf
EndSub
```

Campbell RWIS Station - Software

- Campbell Scientific LoggerNet software tool compiles the source directly on the datalogger. Any datalogger can have new source code loaded and compiled remotely. (e.g. Site visits are not required for program changes.) We can see the values of all program variables remotely using this same tool.
- LoggerNet tool can also provide a console interface where you can monitor the traffic on your serial ports. (Very useful for serial connected devices.)

Campbell LoggerNet session



Adjusted Server Date/Time: 4/28/2016 1:55:18 PM

Station Date/Time: 4/28/2016 8:56:03 PM

Device	Bytes Free
CPU	481.79 KB
USR	81.92 KB

File Name	Run Options	Size	Modified	Attribute
AtmosInstruments.cr1		19.33 KB	2016-03-03 21:47:18	RW
DiagAndErrorHandling.cr1		5.65 KB	2016-03-03 21:59:52	RW
NTCIP_Setup.cr1		3.48 KB	2016-03-03 21:53:38	RW
RSSLufft.cr1		31.19 KB	2016-03-09 21:48:14	RW
SerialIO.cr1		6.59 KB	2016-03-03 21:48:12	RW
SiteConfig.cr1		3.84 KB	2016-01-28 22:07:36	RW

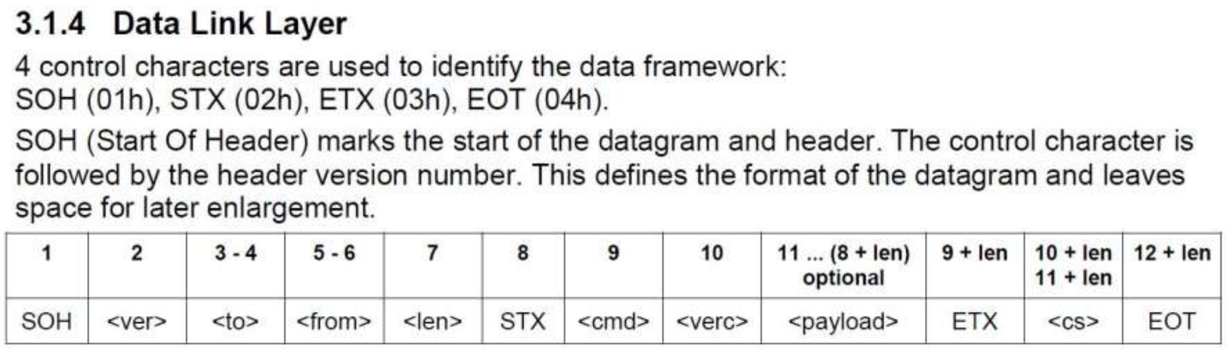
```
Running Program: D2RWIS.CR1
Run On Power Up Program: D2RWIS.CR1
Program State: running
===============
CPU:D2RWIS.CR1 -- Compiled in SequentialMode.
```

☑ Set Run Options on Send

6 files

Campbell RWIS Station - Software

- Created a CR Basic module with a library of functions to handle all the UMB protocol specifics we use from Lufft.
- Here is a snippet of their UMB packet structure, but see *Appendix A* for a detailed description on the development process for the UMB protocol.



3.1.4 Data Link Layer

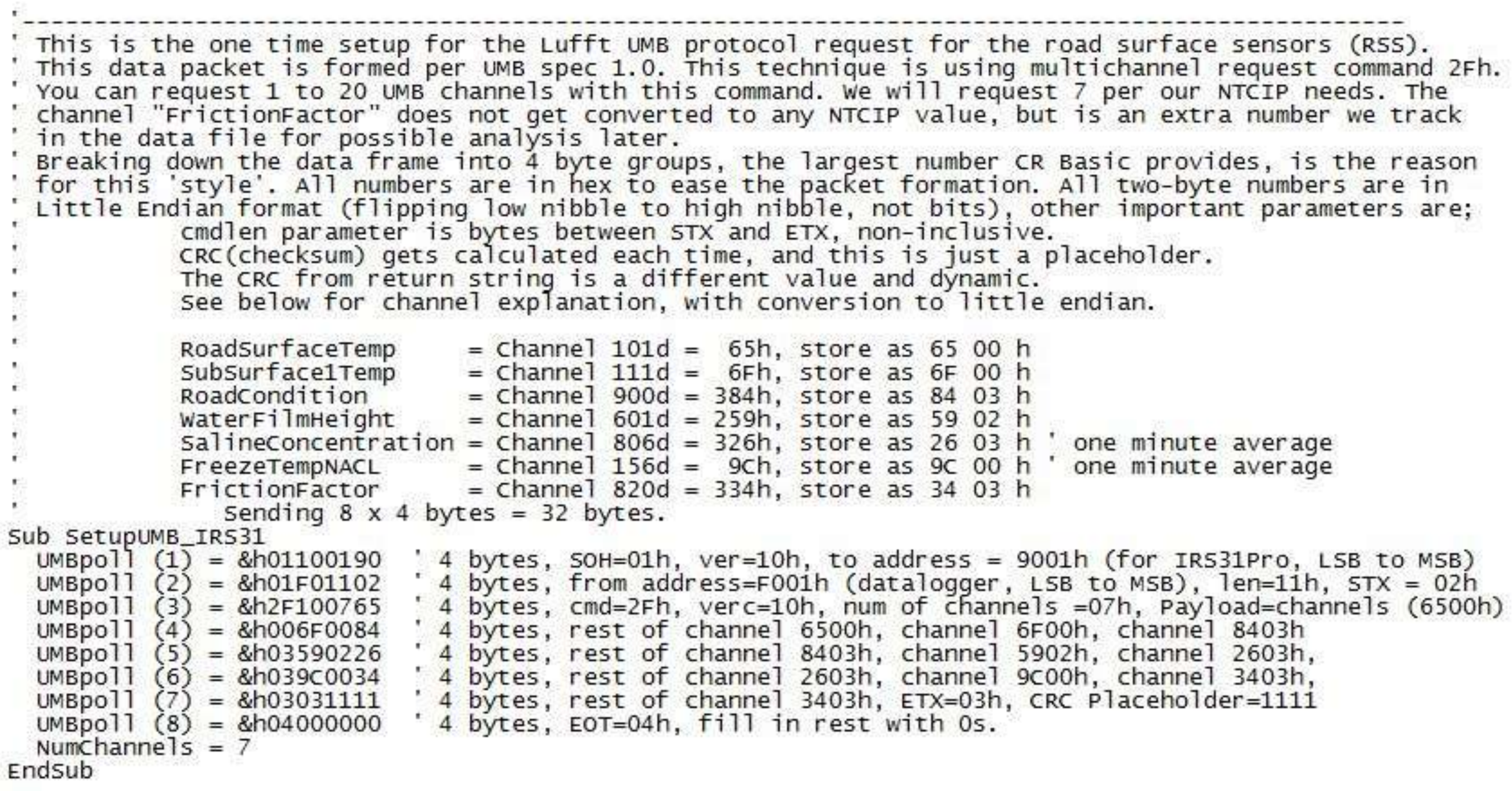
4 control characters are used to identify the data framework:
SOH (01h), STX (02h), ETX (03h), EOT (04h).

SOH (Start Of Header) marks the start of the datagram and header. The control character is followed by the header version number. This defines the format of the datagram and leaves space for later enlargement.

1	2	3 - 4	5 - 6	7	8	9	10	11 ... (8 + len) optional	9 + len	10 + len 11 + len	12 + len
SOH	<ver>	<to>	<from>	<len>	STX	<cmd>	<verc>	<payload>	ETX	<cs>	EOT

Campbell RWIS Station - Software

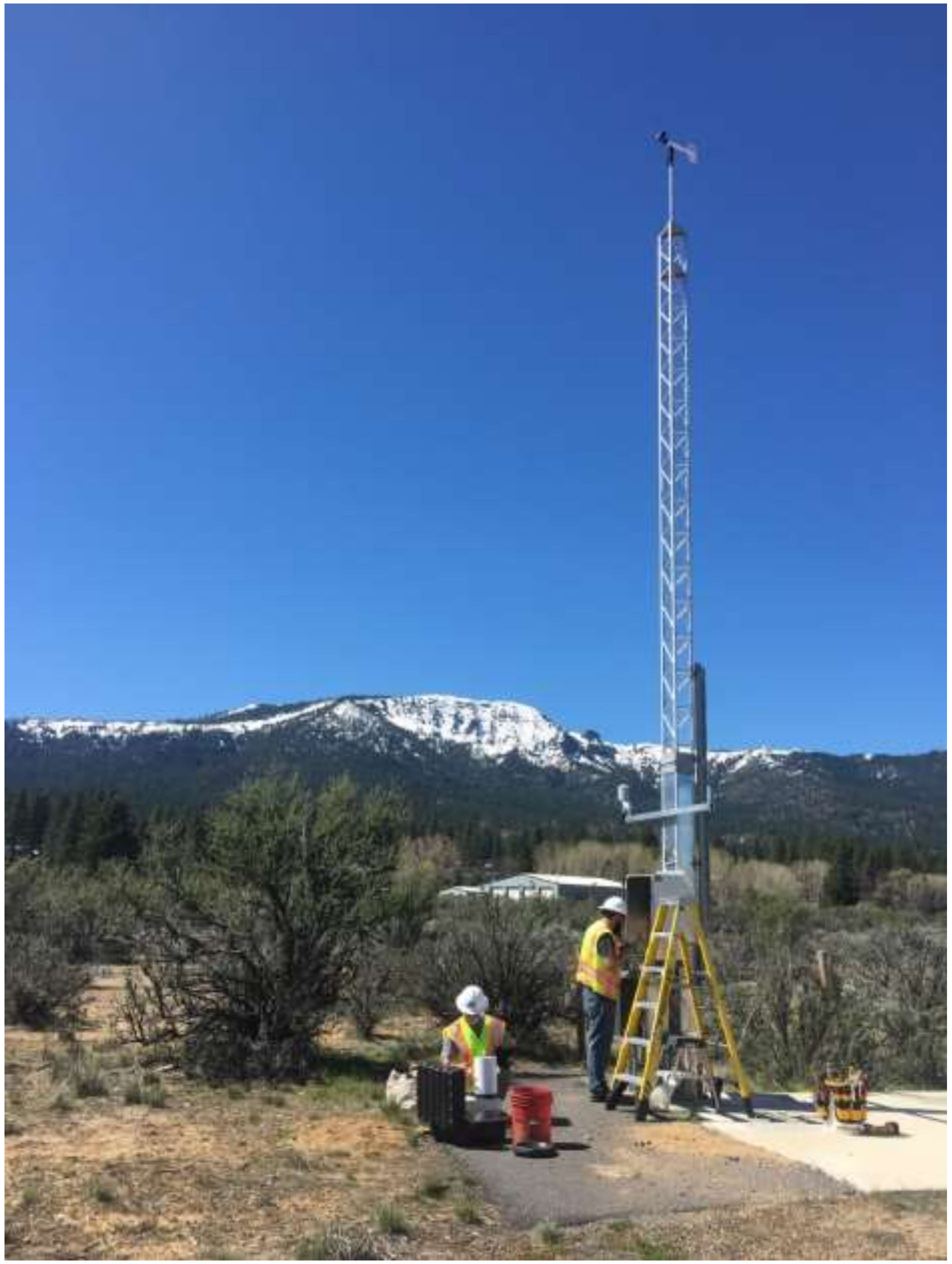
This is how the send packet formation is setup in CR Basic. Notice that we actually request seven channels of information. The seventh is a value that tries to represent road friction which is interpreted and not supported by NTCIP. We use this internally only.



```
'-------------------------------------------------------------------------------------------------
' This is the one time setup for the Lufft UMB protocol request for the road surface sensors (RSS).
' This data packet is formed per UMB spec 1.0. This technique is using multichannel request command 2Fh.
' You can request 1 to 20 UMB channels with this command. We will request 7 per our NTCIP needs. The
' channel "FrictionFactor" does not get converted to any NTCIP value, but is an extra number we track
' in the data file for possible analysis later.
' Breaking down the data frame into 4 byte groups, the largest number CR Basic provides, is the reason
' for this 'style'. All numbers are in hex to ease the packet formation. All two-byte numbers are in
' Little Endian format (flipping low nibble to high nibble, not bits), other important parameters are;
'           cmdlen parameter is bytes between STX and ETX, non-inclusive.
'           CRC(checksum) gets calculated each time, and this is just a placeholder.
'           The CRC from return string is a different value and dynamic.
'           See below for channel explanation, with conversion to little endian.
'
'           RoadSurfaceTemp      = Channel 101d =  65h, store as 65 00 h
'           SubSurface1Temp      = Channel 111d =  6Fh, store as 6F 00 h
'           RoadCondition        = Channel 900d = 384h, store as 84 03 h
'           WaterFilmHeight      = Channel 601d = 259h, store as 59 02 h
'           SalineConcentration = Channel 806d = 326h, store as 26 03 h ' one minute average
'           FreezeTempNACL       = Channel 156d =  9Ch, store as 9C 00 h ' one minute average
'           FrictionFactor       = Channel 820d = 334h, store as 34 03 h
'              Sending 8 x 4 bytes = 32 bytes.
Sub SetupUMB_IRS31
  UMBpoll (1) = &h01100190  ' 4 bytes, SOH=01h, ver=10h, to address = 9001h (for IRS31Pro, LSB to MSB)
  UMBpoll (2) = &h01F01102  ' 4 bytes, from address=F001h (datalogger, LSB to MSB), len=11h, STX = 02h
  UMBpoll (3) = &h2F100765  ' 4 bytes, cmd=2Fh, verc=10h, num of channels =07h, Payload=channels (6500h)
  UMBpoll (4) = &h006F0084  ' 4 bytes, rest of channel 6500h, channel 6F00h, channel 8403h
  UMBpoll (5) = &h03590226  ' 4 bytes, rest of channel 8403h, channel 5902h, channel 2603h,
  UMBpoll (6) = &h039C0034  ' 4 bytes, rest of channel 2603h, channel 9C00h, channel 3403h,
  UMBpoll (7) = &h03031111  ' 4 bytes, rest of channel 3403h, ETX=03h, CRC Placeholder=1111
  UMBpoll (8) = &h04000000  ' 4 bytes, EOT=04h, fill in rest with 0s.
  NumChannels = 7
EndSub
```

CR1000 RPU Deployment

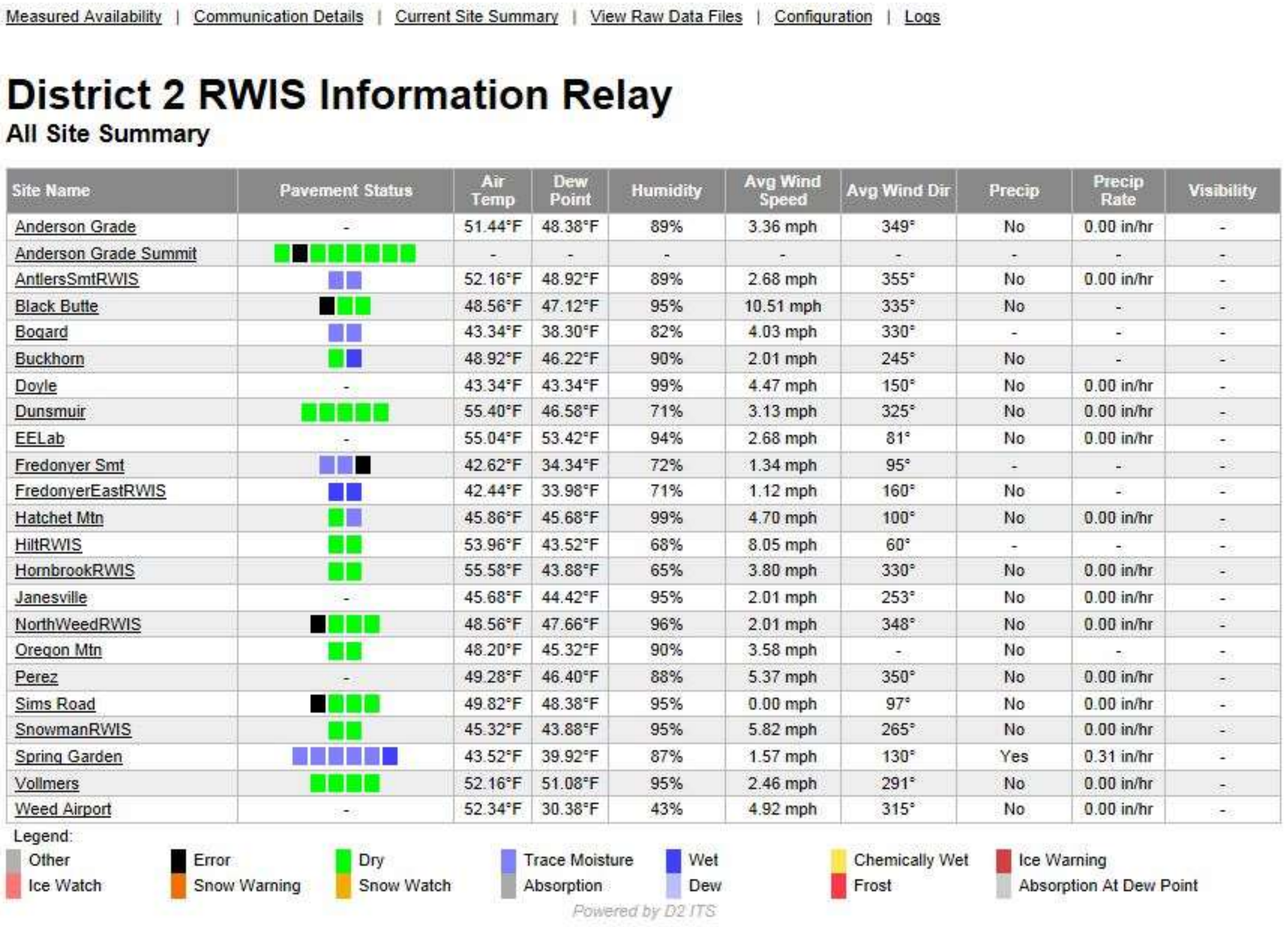
- **Definitions and Background**
- **Why the Redesign?**
- **District 2 RWIS Station Profile**
- **Implementation Timeline**
- **Implementation Pros and Cons**
- **Software Notes**
- **Conclusion**



Campbell RWIS Station - Conclusion

- Current RPU design is stable and major learning curve is over. (See *Appendix B* for updated BOM for V2)
- District 2 ESS count: 23 total RWIS stations; 10 Campbell; 5 Vaisala LX RPU; 8 Vaisala ESS.
- District 2 RSS count: 55 total RSS; 9 Lufft IRS31Pro; 14 Lufft IRS21; 30 Vaisala FP2000; 1 IceSight NIPS; 1 Vaisala NIPS. (See RWIS relay screen shot.)

Campbell RWIS Station - Conclusion



Measured Availability | Communication Details | Current Site Summary | View Raw Data Files | Configuration | Logs

District 2 RWIS Information Relay

All Site Summary

Site Name	Pavement Status	Air Temp	Dew Point	Humidity	Avg Wind Speed	Avg Wind Dir	Precip	Precip Rate	Visibility
Anderson Grade	-	51.44°F	48.38°F	89%	3.36 mph	349°	No	0.00 in/hr	-
Anderson Grade Summit	🟩⬛🟩🟩🟩🟩🟩🟩	-	-	-	-	-	-	-	-
AntlersSmtRWIS	🟪🟪	52.16°F	48.92°F	89%	2.68 mph	355°	No	0.00 in/hr	-
Black Butte	⬛🟩🟩	48.56°F	47.12°F	95%	10.51 mph	335°	No	-	-
Bogard	🟪🟪	43.34°F	38.30°F	82%	4.03 mph	330°	-	-	-
Buckhorn	🟩🟦	48.92°F	46.22°F	90%	2.01 mph	245°	No	-	-
Doyle	-	43.34°F	43.34°F	99%	4.47 mph	150°	No	0.00 in/hr	-
Dunsmuir	🟩🟩🟩🟩🟩	55.40°F	46.58°F	71%	3.13 mph	325°	No	0.00 in/hr	-
EELab	-	55.04°F	53.42°F	94%	2.68 mph	81°	No	0.00 in/hr	-
Fredonyer Smt	🟪🟪⬛	42.62°F	34.34°F	72%	1.34 mph	95°	-	-	-
FredonyerEastRWIS	🟦🟦	42.44°F	33.98°F	71%	1.12 mph	160°	No	-	-
Hatchet Mtn	🟩🟪	45.86°F	45.68°F	99%	4.70 mph	100°	No	0.00 in/hr	-
HiltRWIS	🟩🟩	53.96°F	43.52°F	68%	8.05 mph	60°	-	-	-
HornbrookRWIS	🟩🟩	55.58°F	43.88°F	65%	3.80 mph	330°	No	0.00 in/hr	-
Janesville	-	45.68°F	44.42°F	95%	2.01 mph	253°	No	0.00 in/hr	-
NorthWeedRWIS	⬛🟩🟩🟩	48.56°F	47.66°F	96%	2.01 mph	348°	No	0.00 in/hr	-
Oregon Mtn	🟩🟩	48.20°F	45.32°F	90%	3.58 mph	-	No	-	-
Perez	-	49.28°F	46.40°F	88%	5.37 mph	350°	No	0.00 in/hr	-
Sims Road	⬛🟩🟩🟩	49.82°F	48.38°F	95%	0.00 mph	97°	No	0.00 in/hr	-
SnowmanRWIS	🟩🟩	45.32°F	43.88°F	95%	5.82 mph	265°	No	0.00 in/hr	-
Spring Garden	🟪🟪🟪🟪🟪🟦	43.52°F	39.92°F	87%	1.57 mph	130°	Yes	0.31 in/hr	-
Vollmers	🟩🟩🟩🟩	52.16°F	51.08°F	95%	2.46 mph	291°	No	0.00 in/hr	-
Weed Airport	-	52.34°F	30.38°F	43%	4.92 mph	315°	No	0.00 in/hr	-

Legend:

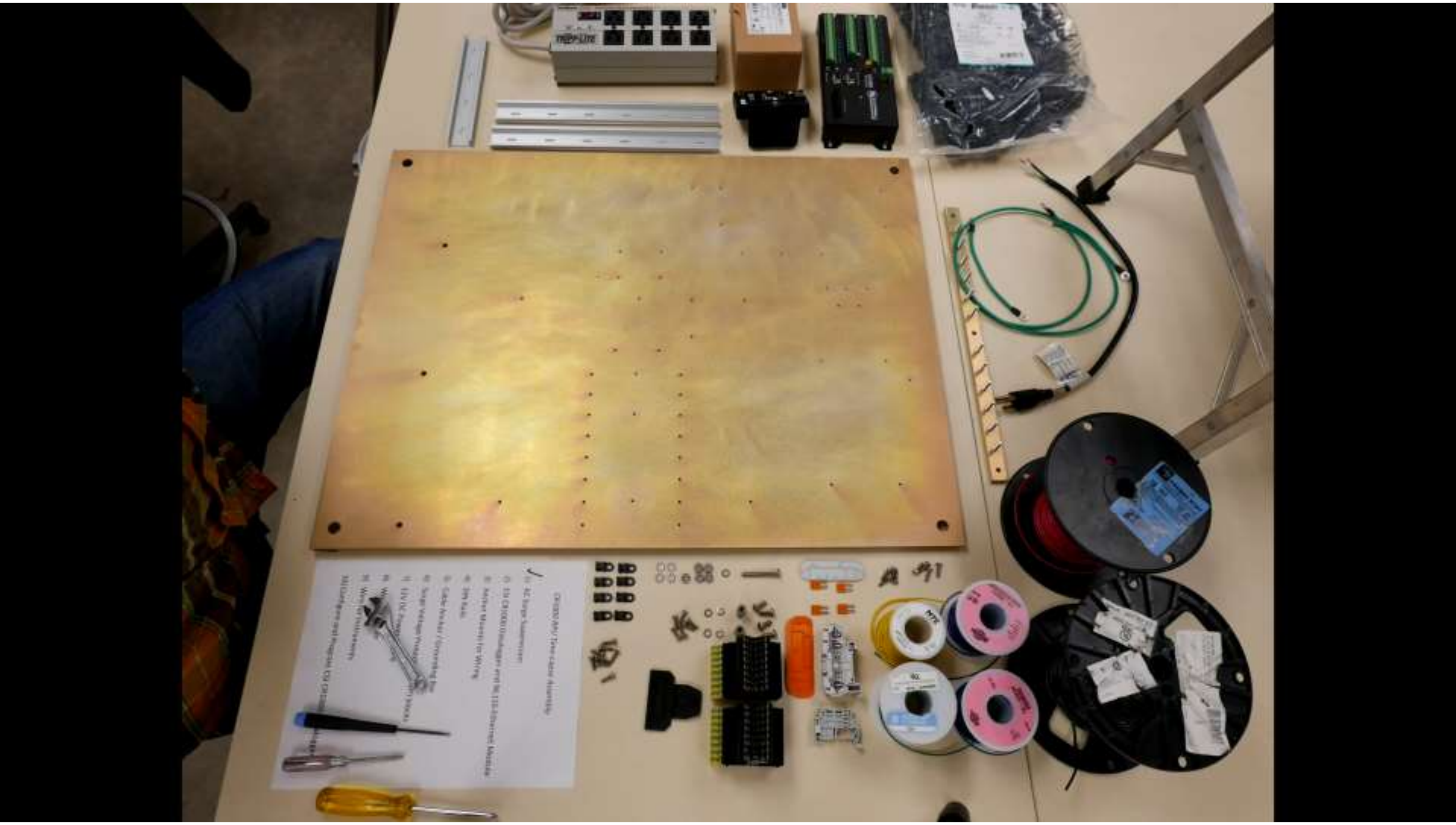
⬜ Other	⬛ Error	🟩 Dry	🟪 Trace Moisture	🟦 Wet	🟨 Chemically Wet	🟥 Ice Warning
🟥 Ice Watch	🟧 Snow Warning	🟧 Snow Watch	⬜ Absorption	🟪 Dew	🟥 Frost	⬜ Absorption At Dew Point

Powered by D2 ITS

Campbell RWIS Station – Future Plans

- We continue to move forward with implementation of the Campbell design. Several capital projects have RWIS stations that will be added, and we will continue to replace the older existing sites with the Campbell design.
- Pavement sensing continues to be a bit of a challenge in terms of reliable sensors and time spent with maintenance. The next presentation by Mike Beyer will provide more details.
- Software will continue to evolve with plans such as; querying devices for their ID/Firmware; advanced error checking/diagnostics; integration of the NIPS into standardized code; installation at a site containing an ICWS.
- Adapt RPU design with a CR6 datalogger.
- Most existing Vaisala RPUs can be swapped out in a day.
- Campbell RPU can be assembled and configured in a few hours. (see time lapse video.)

Time-lapse assembly of Campbell RPU



Thanks!

The entire D2 ITS staff of course:

Mike Beyer

Jeff Cullins

Lonnie Hobbs

Keith Koeppen

Jeremiah Pearce

Ian Turnbull (the boss)

Ken Beals (retired)

Lee Anna Dructor (retired)

D3 ITS:

Michael Mullen

Campbell Scientific:

John Markham

Questions?

References

U.S. Department of Transportation – Federal Highway Administration

http://www.ops.fhwa.dot.gov/weather/faq.htm

National Transportation Comunications for ITS Protocol

https://www.ntcip.org

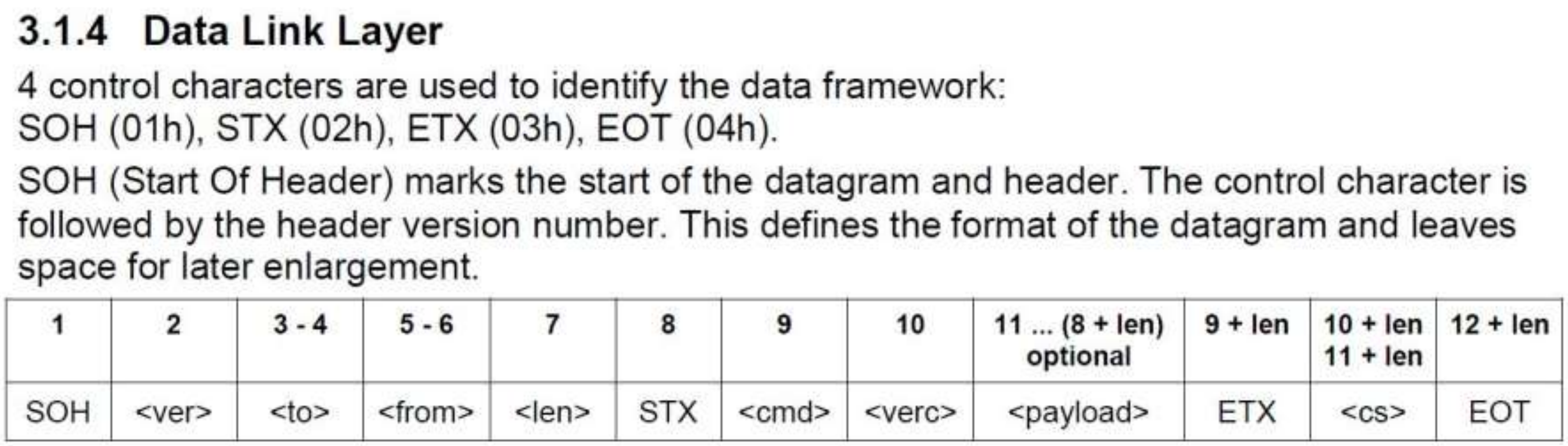
Campbell Scientific

https://www.campbellsci.com/

Lufft

http://www.lufft.com/en/company/

Appendix A – UMB protocol primer for Lufft sensor programming.



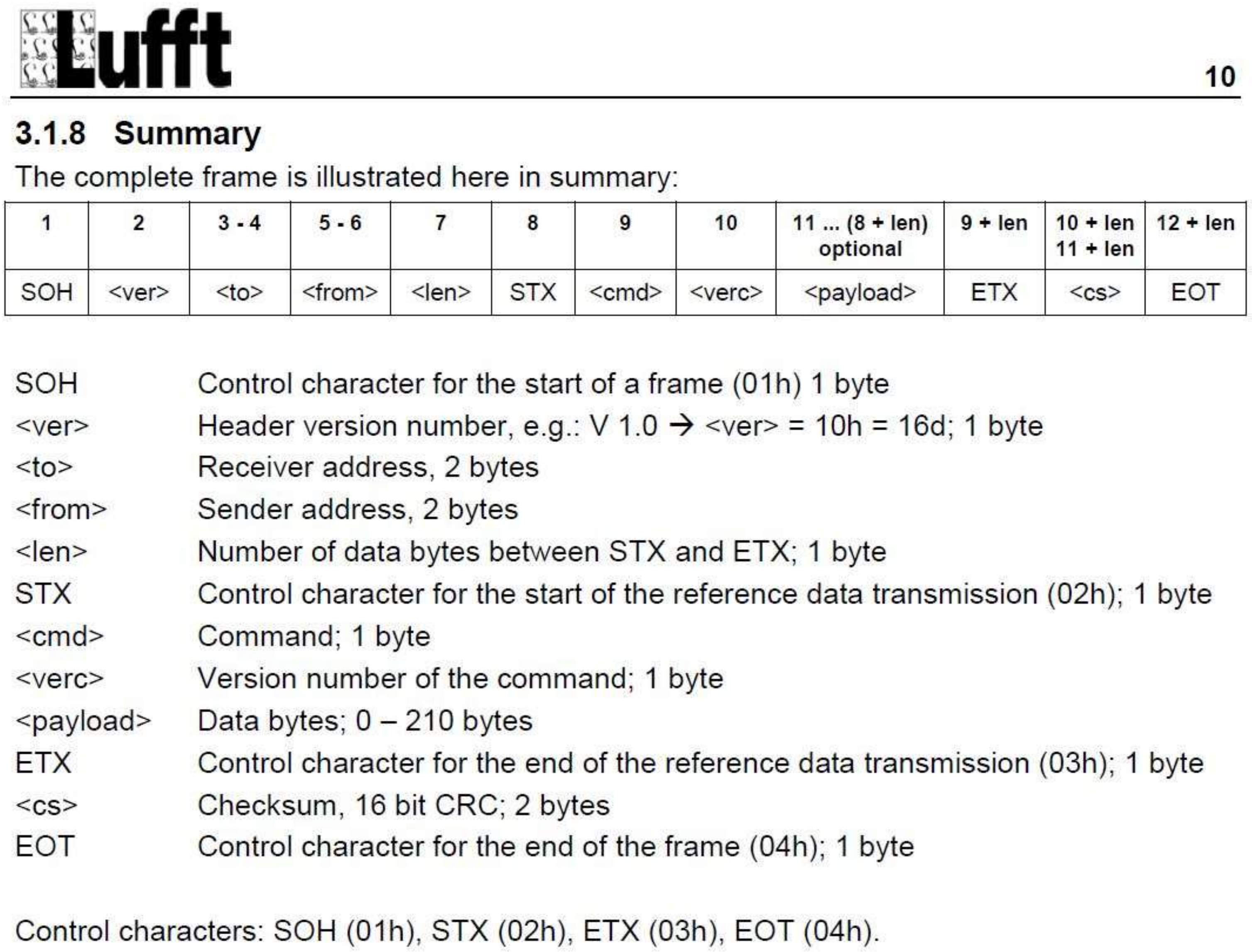
3.1.4 Data Link Layer

4 control characters are used to identify the data framework:
SOH (01h), STX (02h), ETX (03h), EOT (04h).

SOH (Start Of Header) marks the start of the datagram and header. The control character is followed by the header version number. This defines the format of the datagram and leaves space for later enlargement.

1	2	3 - 4	5 - 6	7	8	9	10	11 ... (8 + len) optional	9 + len	10 + len 11 + len	12 + len
SOH	<ver>	<to>	<from>	<len>	STX	<cmd>	<verc>	<payload>	ETX	<cs>	EOT

Appendix A



3.1.8 Summary

The complete frame is illustrated here in summary:

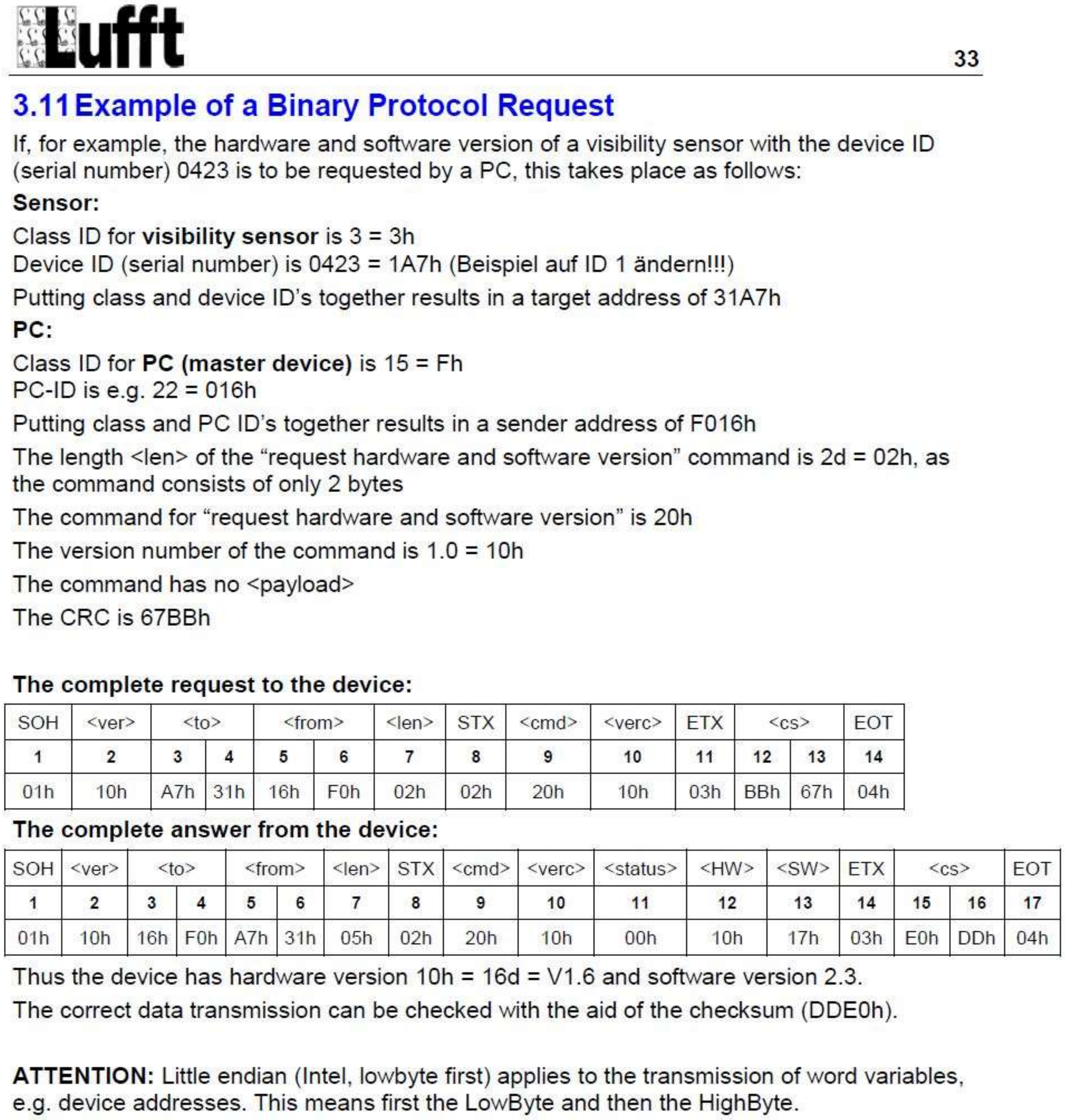
1	2	3 - 4	5 - 6	7	8	9	10	11 ... (8 + len) optional	9 + len	10 + len 11 + len	12 + len
SOH	<ver>	<to>	<from>	<len>	STX	<cmd>	<verc>	<payload>	ETX	<cs>	EOT

SOH	Control character for the start of a frame (01h) 1 byte
<ver>	Header version number, e.g.: V 1.0 → <ver> = 10h = 16d; 1 byte
<to>	Receiver address, 2 bytes
<from>	Sender address, 2 bytes
<len>	Number of data bytes between STX and ETX; 1 byte
STX	Control character for the start of the reference data transmission (02h); 1 byte
<cmd>	Command; 1 byte
<verc>	Version number of the command; 1 byte
<payload>	Data bytes; 0 – 210 bytes
ETX	Control character for the end of the reference data transmission (03h); 1 byte
<cs>	Checksum, 16 bit CRC; 2 bytes
EOT	Control character for the end of the frame (04h); 1 byte

Control characters: SOH (01h), STX (02h), ETX (03h), EOT (04h).

Appendix A

Nice example in Lufft manual of a packet send and receive.



Lufft

3.11 Example of a Binary Protocol Request

If, for example, the hardware and software version of a visibility sensor with the device ID (serial number) 0423 is to be requested by a PC, this takes place as follows:

Sensor:

Class ID for **visibility sensor** is 3 = 3h
Device ID (serial number) is 0423 = 1A7h (Beispiel auf ID 1 ändern!!!)

Putting class and device ID's together results in a target address of 31A7h

PC:

Class ID for **PC (master device)** is 15 = Fh
PC-ID is e.g. 22 = 016h

Putting class and PC ID's together results in a sender address of F016h

The length <len> of the "request hardware and software version" command is 2d = 02h, as the command consists of only 2 bytes

The command for "request hardware and software version" is 20h

The version number of the command is 1.0 = 10h

The command has no <payload>

The CRC is 67BBh

The complete request to the device:

SOH	<ver>	<to>		<from>		<len>	STX	<cmd>	<verc>	ETX	<cs>		EOT
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
01h	10h	A7h	31h	16h	F0h	02h	02h	20h	10h	03h	BBh	67h	04h

The complete answer from the device:

SOH	<ver>	<to>		<from>		<len>	STX	<cmd>	<verc>	<status>	<HW>	<SW>	ETX	<cs>		EOT
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
01h	10h	16h	F0h	A7h	31h	05h	02h	20h	10h	00h	10h	17h	03h	E0h	DDh	04h

Thus the device has hardware version 10h = 16d = V1.6 and software version 2.3.

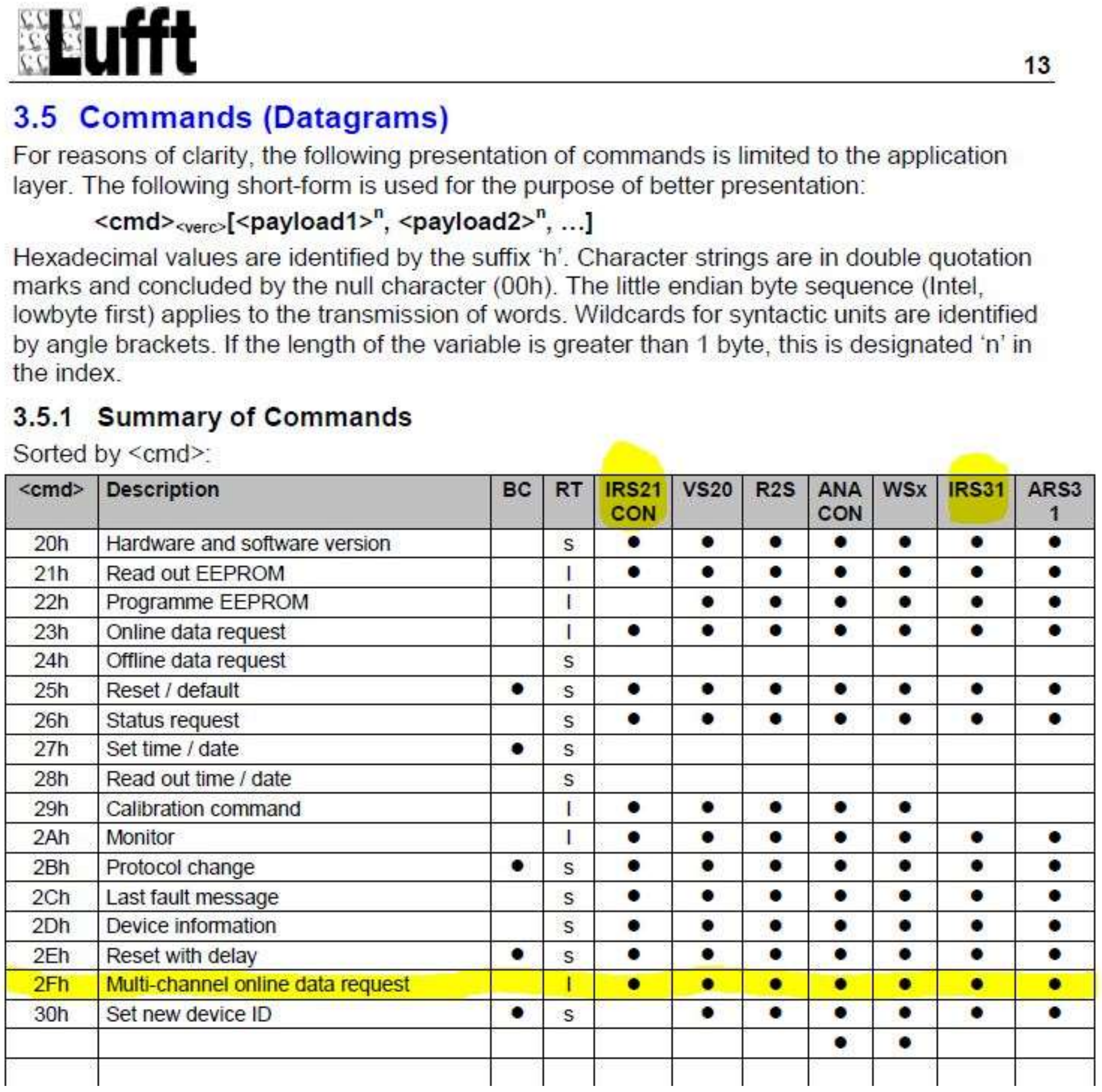
The correct data transmission can be checked with the aid of the checksum (DDE0h).

ATTENTION: Little endian (Intel, lowbyte first) applies to the transmission of word variables, e.g. device addresses. This means first the LowByte and then the HighByte.

Appendix A

Multiple commands are available. We use the 2F command to request multiple "channels", which is essentially different types of data.

Notice that the UMB protocol became supported by the older IRS21 sensors with an interface device called the "IRS21CON"



Lufft

3.5 Commands (Datagrams)

For reasons of clarity, the following presentation of commands is limited to the application layer. The following short-form is used for the purpose of better presentation:

<cmd>$_{\text{<verc>}}$[<payload1>^{n}, <payload2>^{n}, ...]

Hexadecimal values are identified by the suffix 'h'. Character strings are in double quotation marks and concluded by the null character (00h). The little endian byte sequence (Intel, lowbyte first) applies to the transmission of words. Wildcards for syntactic units are identified by angle brackets. If the length of the variable is greater than 1 byte, this is designated 'n' in the index.

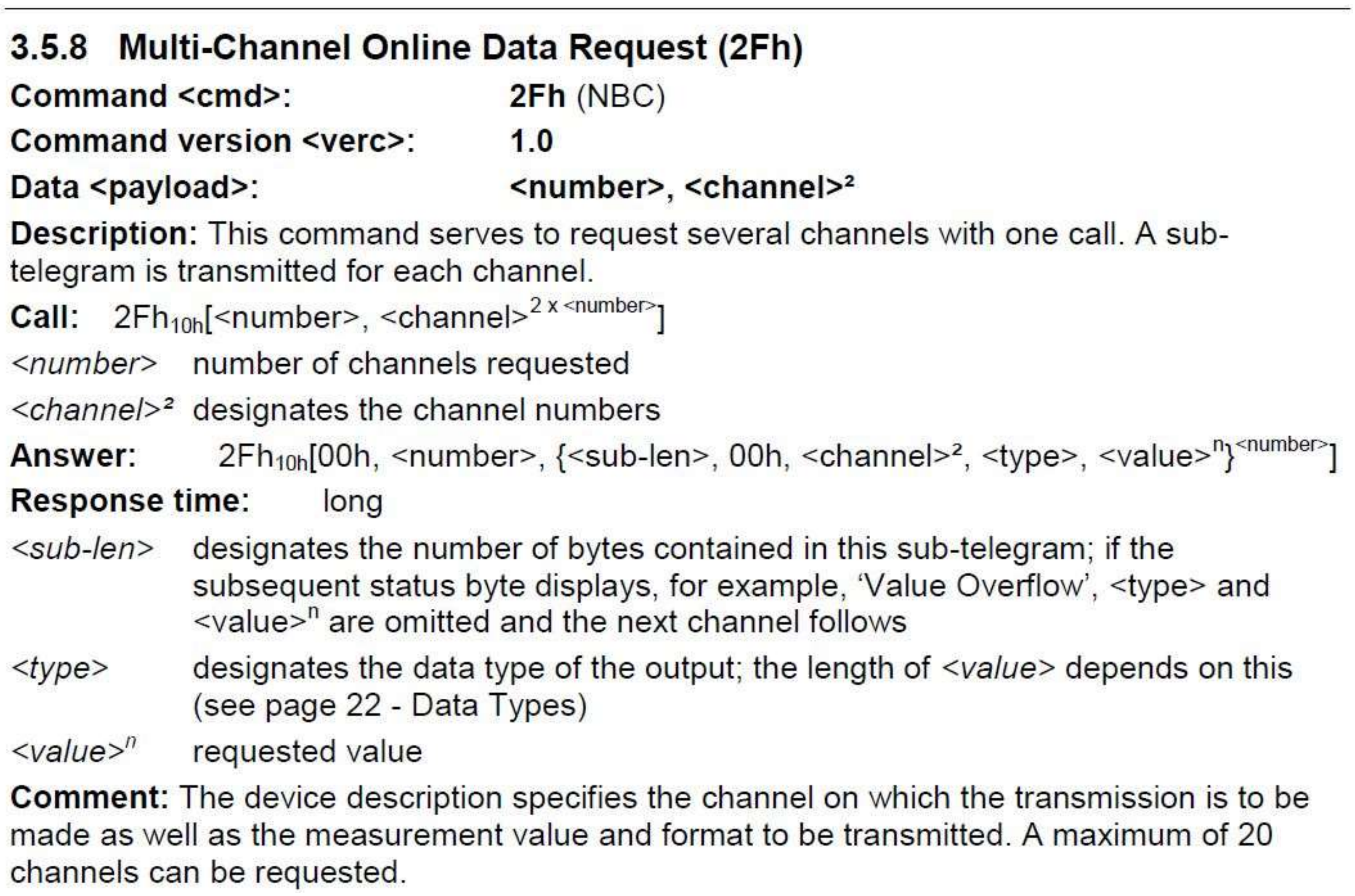
3.5.1 Summary of Commands

Sorted by <cmd>:

<cmd>	Description	BC	RT	IRS21 CON	VS20	R2S	ANA CON	WSx	IRS31	ARS3 1
20h	Hardware and software version		s	●	●	●	●	●	●	●
21h	Read out EEPROM		l	●	●	●	●	●	●	●
22h	Programme EEPROM		l		●	●	●	●	●	●
23h	Online data request		l	●	●	●	●	●	●	●
24h	Offline data request		s							
25h	Reset / default	●	s	●	●	●	●	●	●	●
26h	Status request		s	●	●	●	●	●	●	●
27h	Set time / date	●	s							
28h	Read out time / date		s							
29h	Calibration command		l	●	●	●	●	●		
2Ah	Monitor		l	●	●	●	●	●	●	●
2Bh	Protocol change	●	s	●	●	●	●	●	●	●
2Ch	Last fault message		s	●	●	●	●	●	●	●
2Dh	Device information		s	●	●	●	●	●	●	●
2Eh	Reset with delay	●	s	●	●	●	●	●	●	●
2Fh	Multi-channel online data request		l	●	●	●	●	●	●	●
30h	Set new device ID	●	s		●	●	●	●	●	●
							●	●		

Appendix A

Details of packet formation of the 2F command that we use.



3.5.8 Multi-Channel Online Data Request (2Fh)

Command <cmd>: **2Fh** (NBC)

Command version <verc>: **1.0**

Data <payload>: **<number>, <channel>^{2}**

Description: This command serves to request several channels with one call. A sub-telegram is transmitted for each channel.

Call: $2Fh_{10h}$[<number>, <channel>$^{2 \times \text{<number>}}$]

<number>	number of channels requested
<channel>^{2}	designates the channel numbers

Answer: $2Fh_{10h}$[00h, <number>, {<sub-len>, 00h, <channel>^{2}, <type>, <value>^{n}}$^{\text{<number>}}$]

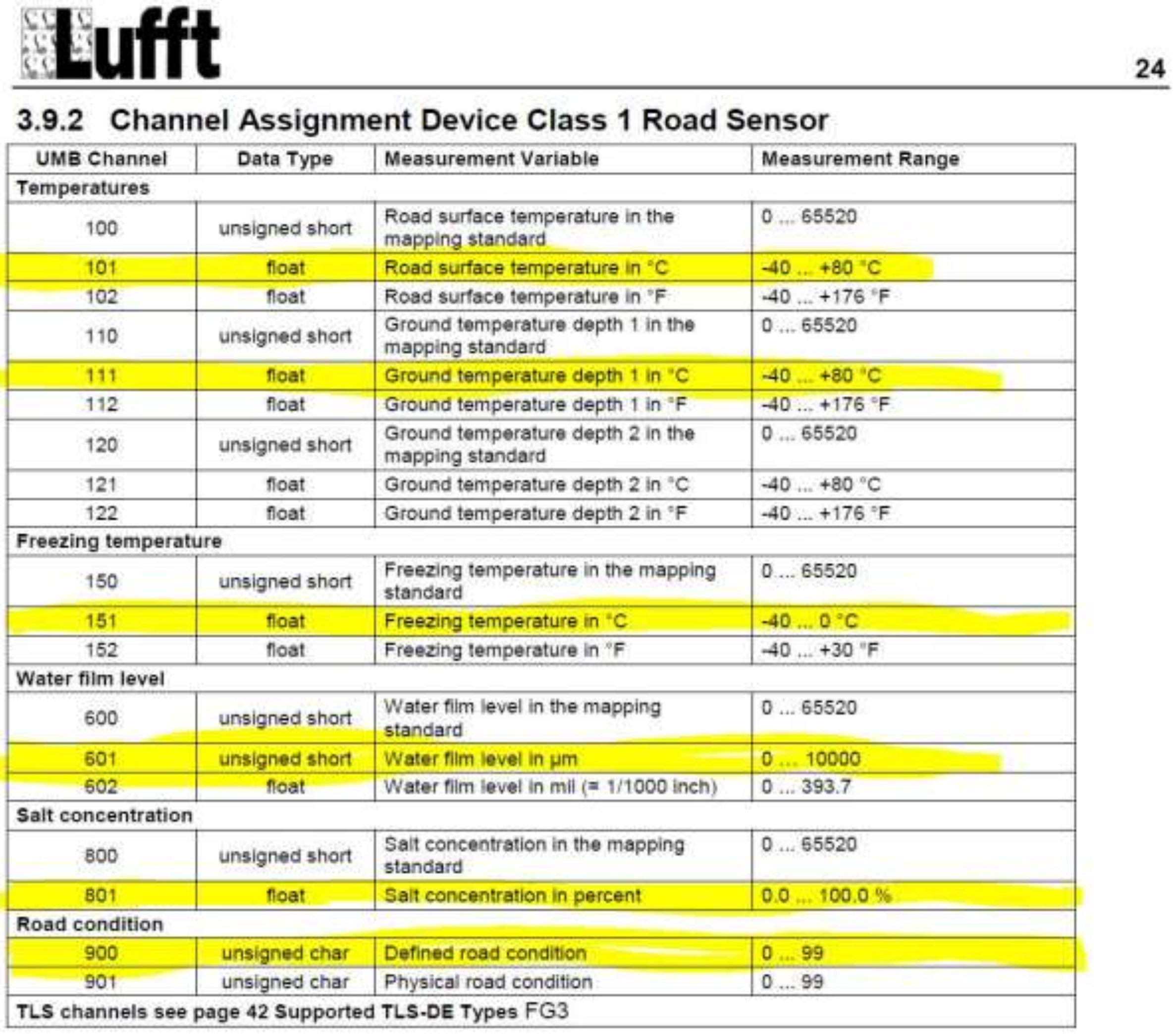
Response time: long

<sub-len>	designates the number of bytes contained in this sub-telegram; if the subsequent status byte displays, for example, 'Value Overflow', <type> and <value>^{n} are omitted and the next channel follows
<type>	designates the data type of the output; the length of *<value>* depends on this (see page 22 - Data Types)
<value>^{n}	requested value

Comment: The device description specifies the channel on which the transmission is to be made as well as the measurement value and format to be transmitted. A maximum of 20 channels can be requested.

Appendix A

Multiple “channels” or types of information are available from the road sensors.
Most of these channels are available in both instantaneous reads, and average reads.
We choose six of these channels to achieve our desired NTCIP reporting.



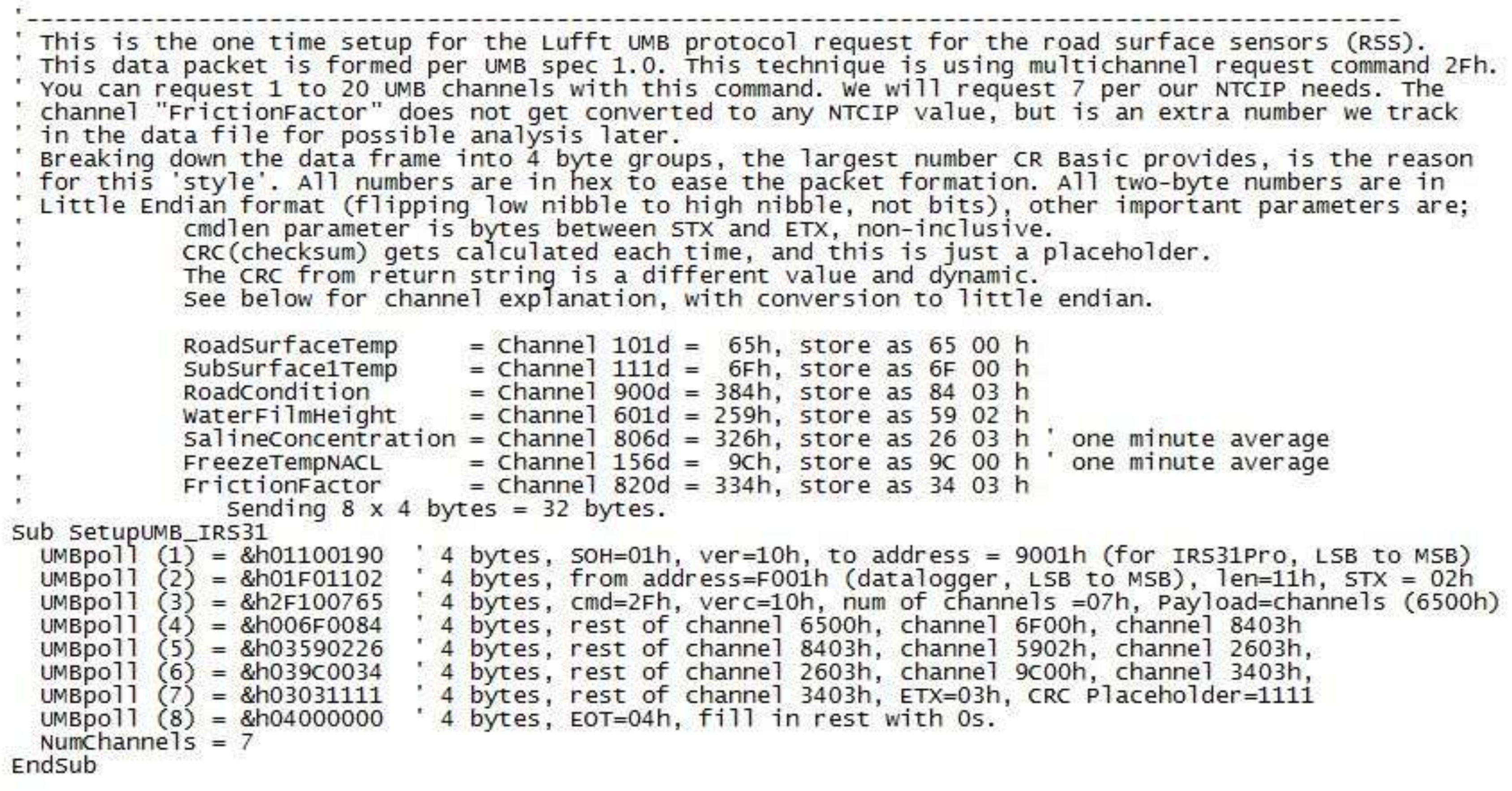
Lufft 24

3.9.2 Channel Assignment Device Class 1 Road Sensor

UMB Channel	Data Type	Measurement Variable	Measurement Range
Temperatures			
100	unsigned short	Road surface temperature in the mapping standard	0 ... 65520
101	float	Road surface temperature in °C	-40 ... +80 °C
102	float	Road surface temperature in °F	-40 ... +176 °F
110	unsigned short	Ground temperature depth 1 in the mapping standard	0 ... 65520
111	float	Ground temperature depth 1 in °C	-40 ... +80 °C
112	float	Ground temperature depth 1 in °F	-40 ... +176 °F
120	unsigned short	Ground temperature depth 2 in the mapping standard	0 ... 65520
121	float	Ground temperature depth 2 in °C	-40 ... +80 °C
122	float	Ground temperature depth 2 in °F	-40 ... +176 °F
Freezing temperature			
150	unsigned short	Freezing temperature in the mapping standard	0 ... 65520
151	float	Freezing temperature in °C	-40 ... 0 °C
152	float	Freezing temperature in °F	-40 ... +30 °F
Water film level			
600	unsigned short	Water film level in the mapping standard	0 ... 65520
601	unsigned short	Water film level in µm	0 ... 10000
602	float	Water film level in mil (= 1/1000 inch)	0 ... 393.7
Salt concentration			
800	unsigned short	Salt concentration in the mapping standard	0 ... 65520
801	float	Salt concentration in percent	0.0 ... 100.0 %
Road condition			
900	unsigned char	Defined road condition	0 ... 99
901	unsigned char	Physical road condition	0 ... 99
TLS channels see page 42 Supported TLS-DE Types FG3			

Appendix A

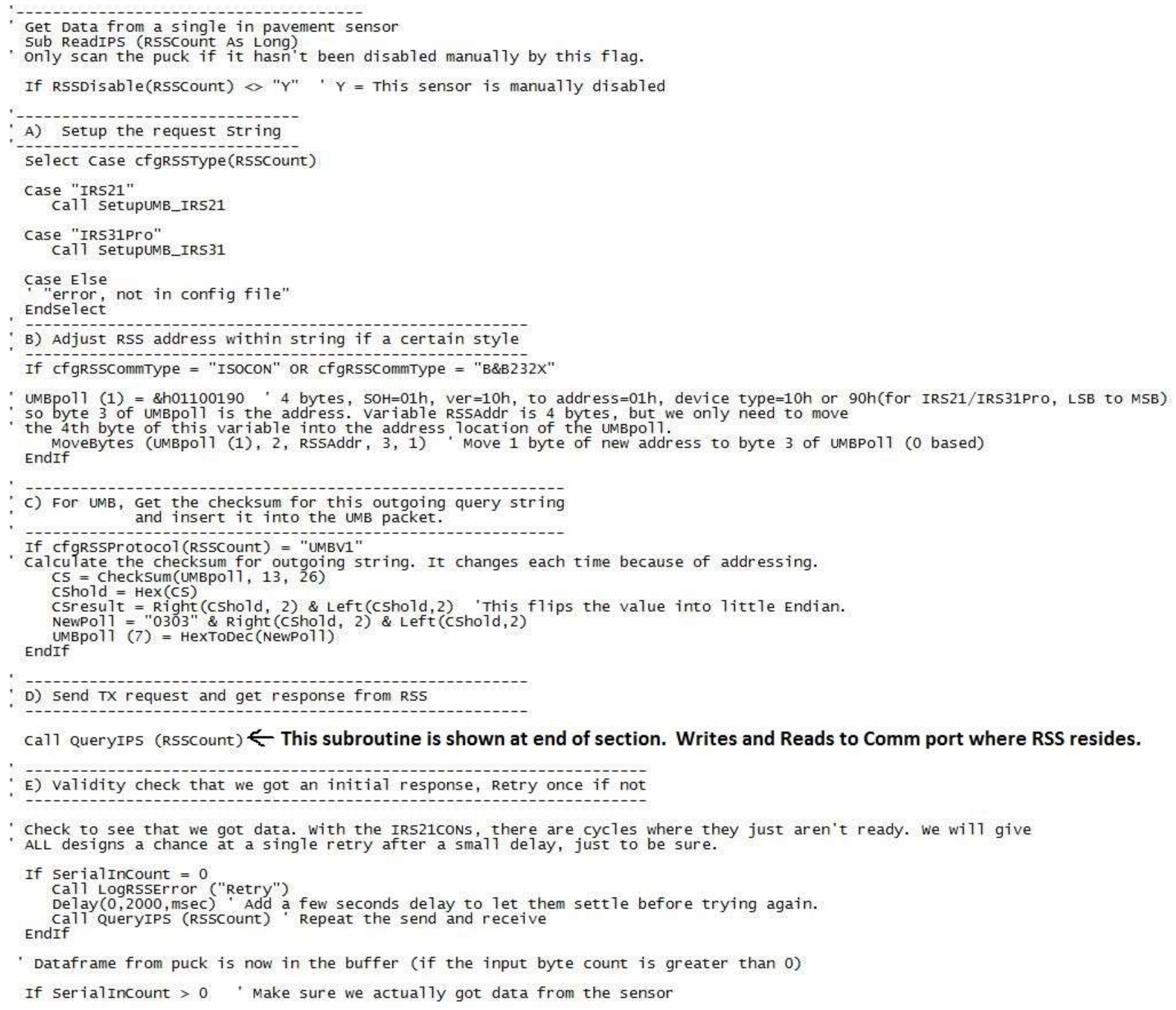
This is how the send packet formation is setup in CR Basic. Notice that we actually request seven channels of information. The seventh is a value that tries to represent road friction which is interpreted and not supported by NTCIP. We use this internally only.



```
'--------------------------------------------------------------------------------------------------
' This is the one time setup for the Lufft UMB protocol request for the road surface sensors (RSS).
' This data packet is formed per UMB spec 1.0. This technique is using multichannel request command 2Fh.
' You can request 1 to 20 UMB channels with this command. We will request 7 per our NTCIP needs. The
' channel "FrictionFactor" does not get converted to any NTCIP value, but is an extra number we track
' in the data file for possible analysis later.
' Breaking down the data frame into 4 byte groups, the largest number CR Basic provides, is the reason
' for this 'style'. All numbers are in hex to ease the packet formation. All two-byte numbers are in
' Little Endian format (flipping low nibble to high nibble, not bits), other important parameters are;
'          cmdlen parameter is bytes between STX and ETX, non-inclusive.
'          CRC(checksum) gets calculated each time, and this is just a placeholder.
'          The CRC from return string is a different value and dynamic.
'          See below for channel explanation, with conversion to little endian.
'
'          RoadSurfaceTemp     = Channel 101d =  65h, store as 65 00 h
'          SubSurface1Temp     = Channel 111d =  6Fh, store as 6F 00 h
'          RoadCondition       = Channel 900d = 384h, store as 84 03 h
'          WaterFilmHeight     = Channel 601d = 259h, store as 59 02 h
'          SalineConcentration = Channel 806d = 326h, store as 26 03 h ' one minute average
'          FreezeTempNACL      = Channel 156d =  9Ch, store as 9C 00 h ' one minute average
'          FrictionFactor      = Channel 820d = 334h, store as 34 03 h
'             Sending 8 x 4 bytes = 32 bytes.
Sub SetupUMB_IRS31
  UMBpoll (1) = &h01100190  ' 4 bytes, SOH=01h, ver=10h, to address = 9001h (for IRS31Pro, LSB to MSB)
  UMBpoll (2) = &h01F01102  ' 4 bytes, from address=F001h (datalogger, LSB to MSB), len=11h, STX = 02h
  UMBpoll (3) = &h2F100765  ' 4 bytes, cmd=2Fh, verc=10h, num of channels =07h, Payload=channels (6500h)
  UMBpoll (4) = &h006F0084  ' 4 bytes, rest of channel 6500h, channel 6F00h, channel 8403h
  UMBpoll (5) = &h03590226  ' 4 bytes, rest of channel 8403h, channel 5902h, channel 2603h,
  UMBpoll (6) = &h039C0034  ' 4 bytes, rest of channel 2603h, channel 9C00h, channel 3403h,
  UMBpoll (7) = &h03031111  ' 4 bytes, rest of channel 3403h, ETX=03h, CRC Placeholder=1111
  UMBpoll (8) = &h04000000  ' 4 bytes, EOT=04h, fill in rest with 0s.
  NumChannels = 7
EndSub
```

Appendix A

The next three slides show the sequence for reading (and validating) data from a UMB device.

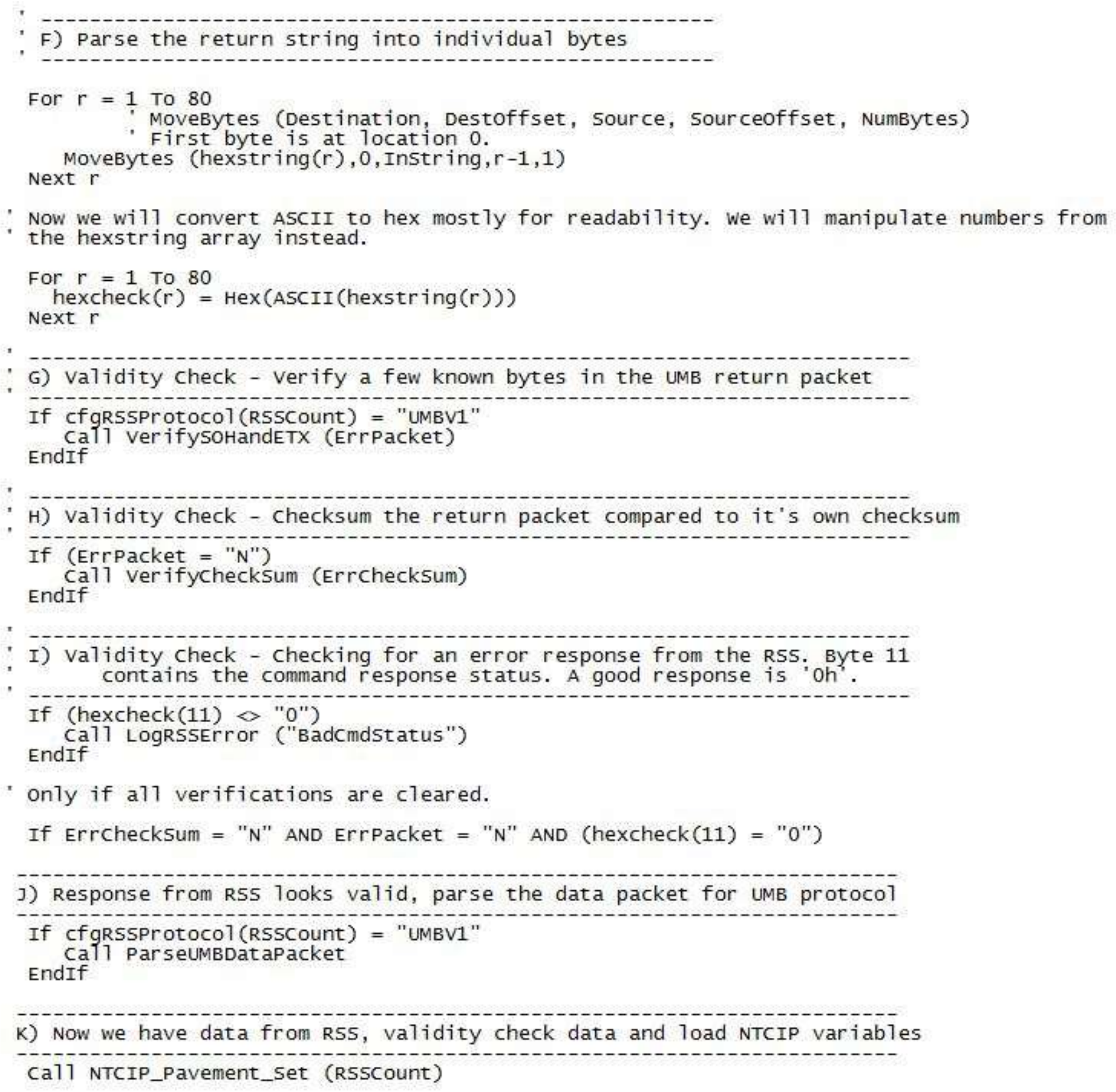


```
'----------------------------------------
' Get Data from a single in pavement sensor
  Sub ReadIPS (RSSCount As Long)
' Only scan the puck if it hasn't been disabled manually by this flag.

  If RSSDisable(RSSCount) <> "Y"   ' Y = This sensor is manually disabled

'-------------------------------
' A)  Setup the request String
'-------------------------------
  Select Case cfgRSSType(RSSCount)

  Case "IRS21"
     Call SetupUMB_IRS21

  Case "IRS31Pro"
     Call SetupUMB_IRS31

  Case Else
  ' "error, not in config file"
  EndSelect
' -----------------------------------------------------
' B) Adjust RSS address within string if a certain style
' -----------------------------------------------------
  If cfgRSSCommType = "ISOCON" OR cfgRSSCommType = "B&B232X"

' UMBpoll (1) = &h01100190   ' 4 bytes, SOH=01h, ver=10h, to address=01h, device type=10h or 90h(for IRS21/IRS31Pro, LSB to MSB)
' so byte 3 of UMBpoll is the address. Variable RSSAddr is 4 bytes, but we only need to move
' the 4th byte of this variable into the address location of the UMBpoll.
     MoveBytes (UMBpoll (1), 2, RSSAddr, 3, 1)   ' Move 1 byte of new address to byte 3 of UMBPoll (0 based)
  EndIf

' ---------------------------------------------------------
' C) For UMB, Get the checksum for this outgoing query string
'             and insert it into the UMB packet.
' ---------------------------------------------------------
  If cfgRSSProtocol(RSSCount) = "UMBV1"
' Calculate the checksum for outgoing string. It changes each time because of addressing.
     CS = CheckSum(UMBpoll, 13, 26)
     CShold = Hex(CS)
     CSresult = Right(CShold, 2) & Left(CShold,2)   'This flips the value into little Endian.
     NewPoll = "0303" & Right(CShold, 2) & Left(CShold,2)
     UMBpoll (7) = HexToDec(NewPoll)
  EndIf

' -----------------------------------------------------
' D) Send TX request and get response from RSS
' -----------------------------------------------------

  Call QueryIPS (RSSCount)
```

This subroutine is shown at end of section. Writes and Reads to Comm port where RSS resides.

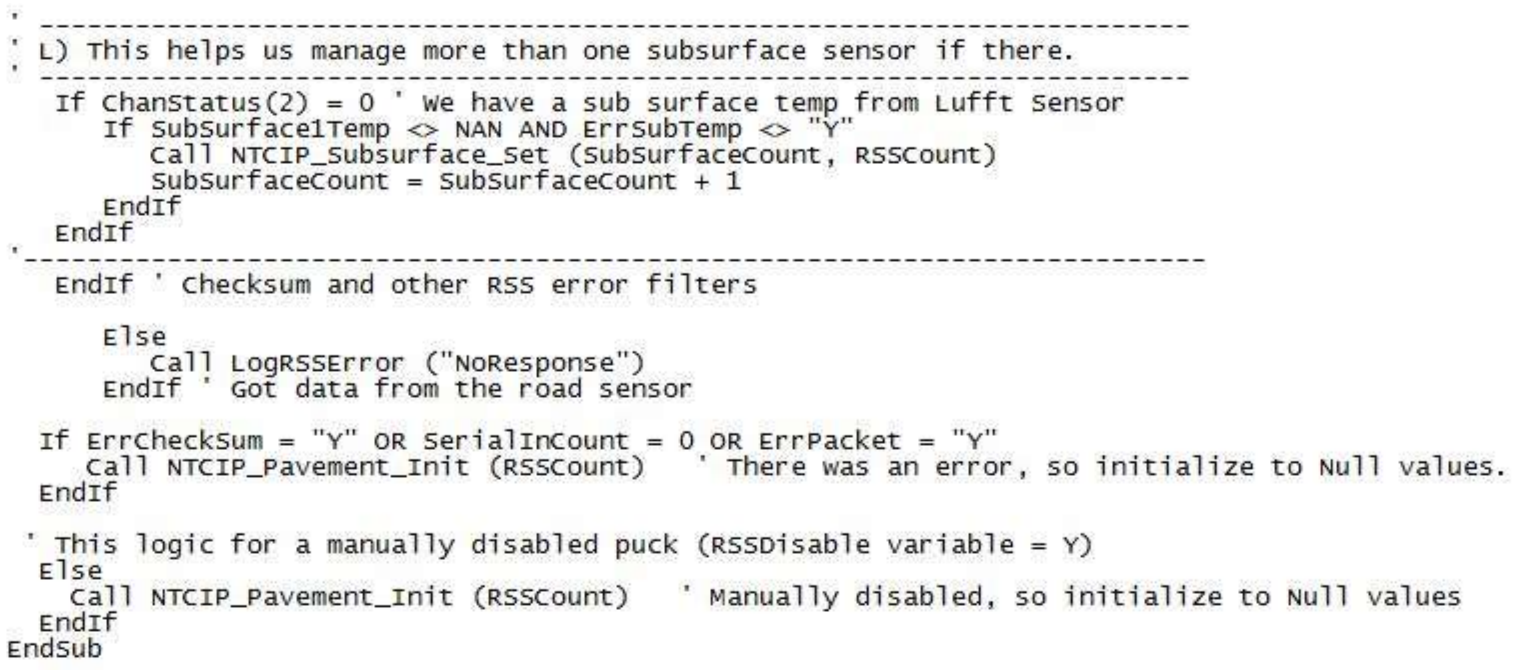
```
' ------------------------------------------------------------------
' E) Validity check that we got an initial response, Retry once if not
' ------------------------------------------------------------------

' Check to see that we got data. With the IRS21CONs, there are cycles where they just aren't ready. We will give
' ALL designs a chance at a single retry after a small delay, just to be sure.

  If SerialInCount = 0
     Call LogRSSError ("Retry")
     Delay(0,2000,msec)  ' Add a few seconds delay to let them settle before trying again.
     Call QueryIPS (RSSCount)  ' Repeat the send and receive
  EndIf

 ' Dataframe from puck is now in the buffer (if the input byte count is greater than 0)

  If SerialInCount > 0   ' Make sure we actually got data from the sensor
```

Appendix A



```
' -------------------------------------------------------
' F) Parse the return string into individual bytes
' -------------------------------------------------------

 For r = 1 To 80
         ' MoveBytes (Destination, DestOffset, Source, SourceOffset, NumBytes)
         ' First byte is at location 0.
    MoveBytes (hexstring(r),0,InString,r-1,1)
 Next r

' Now we will convert ASCII to hex mostly for readability. We will manipulate numbers from
' the hexstring array instead.

 For r = 1 To 80
   hexcheck(r) = Hex(ASCII(hexstring(r)))
 Next r

' ---------------------------------------------------------------------------
' G) Validity Check - Verify a few known bytes in the UMB return packet
' ---------------------------------------------------------------------------
 If cfgRSSProtocol(RSSCount) = "UMBV1"
    Call VerifySOHandETX (ErrPacket)
 EndIf

' ---------------------------------------------------------------------------
' H) Validity Check - Checksum the return packet compared to it's own checksum
' ---------------------------------------------------------------------------
 If (ErrPacket = "N")
    Call VerifyCheckSum (ErrCheckSum)
 EndIf

' ---------------------------------------------------------------------------
' I) Validity Check - Checking for an error response from the RSS. Byte 11
'       contains the command response status. A good response is '0h'.
' ---------------------------------------------------------------------------
 If (hexcheck(11) <> "0")
    Call LogRSSError ("BadCmdStatus")
 EndIf

' Only if all verifications are cleared.

 If ErrCheckSum = "N" AND ErrPacket = "N" AND (hexcheck(11) = "0")

---------------------------------------------------------------------------
J) Response from RSS looks valid, parse the data packet for UMB protocol
---------------------------------------------------------------------------
 If cfgRSSProtocol(RSSCount) = "UMBV1"
    Call ParseUMBDataPacket
 EndIf

---------------------------------------------------------------------------
K) Now we have data from RSS, validity check data and load NTCIP variables
---------------------------------------------------------------------------
 Call NTCIP_Pavement_Set (RSSCount)
```

Appendix A

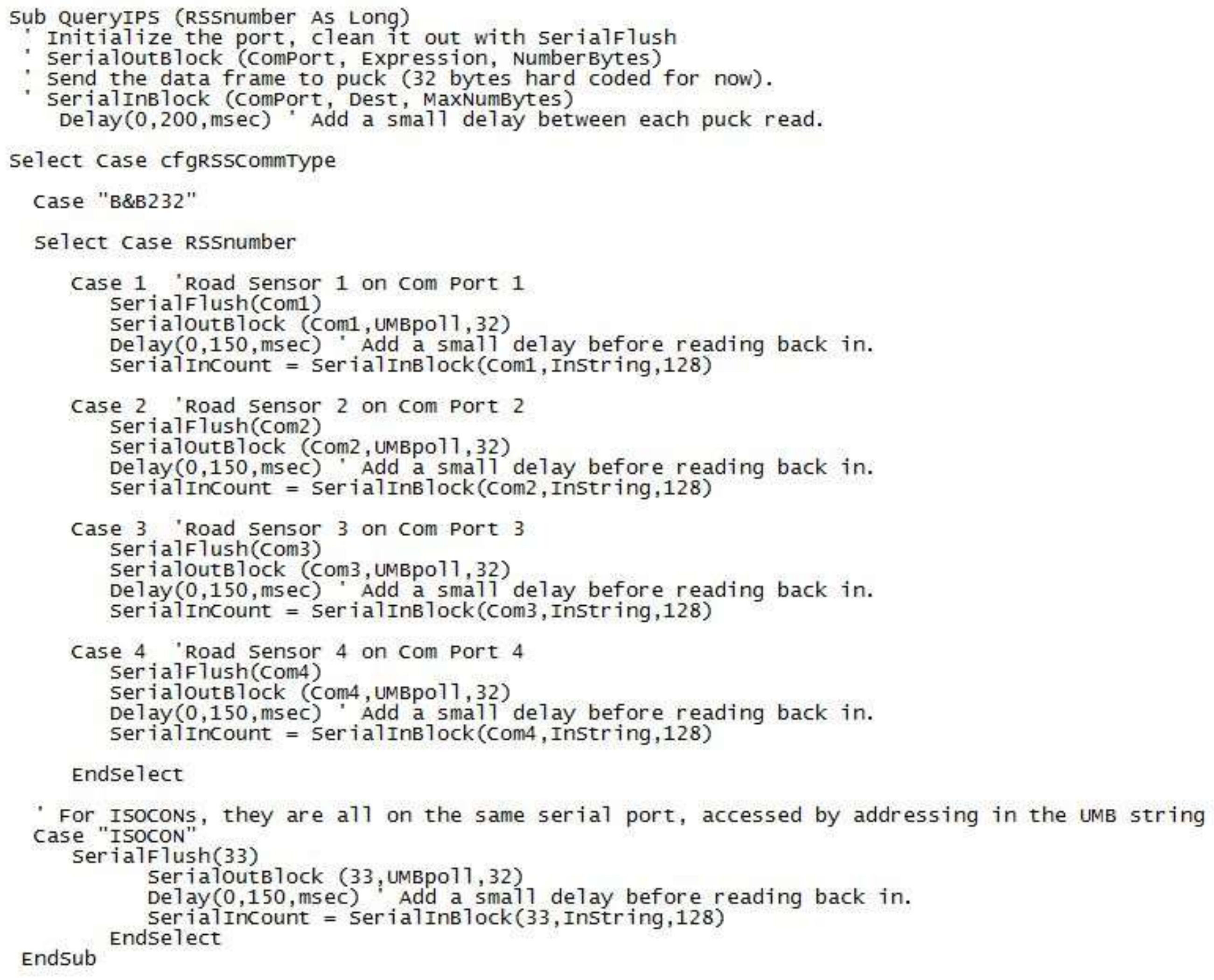


```
' ------------------------------------------------------------------------
' L) This helps us manage more than one subsurface sensor if there.
' ------------------------------------------------------------------------
    If ChanStatus(2) = 0 ' We have a sub surface temp from Lufft Sensor
       If SubSurface1Temp <> NAN AND ErrSubTemp <> "Y"
          Call NTCIP_Subsurface_Set (SubSurfaceCount, RSSCount)
          SubSurfaceCount = SubSurfaceCount + 1
       EndIf
    EndIf
'-------------------------------------------------------------------------
    EndIf ' Checksum and other RSS error filters

       Else
          Call LogRSSError ("NoResponse")
       EndIf ' Got data from the road sensor

   If ErrCheckSum = "Y" OR SerialInCount = 0 OR ErrPacket = "Y"
     Call NTCIP_Pavement_Init (RSSCount)    ' There was an error, so initialize to Null values.
   EndIf

  ' This logic for a manually disabled puck (RSSDisable variable = Y)
   Else
     Call NTCIP_Pavement_Init (RSSCount)    ' Manually disabled, so initialize to Null values
   EndIf
EndSub
```

End of the “ReadIPS” subroutine.

Appendix A

This subroutine is used in step D of the previous subroutine. These are the CR Basic statements that send data to the COM ports, and read data back.



```
Sub QueryIPS (RSSnumber As Long)
 ' Initialize the port, clean it out with SerialFlush
 ' SerialOutBlock (ComPort, Expression, NumberBytes)
 ' Send the data frame to puck (32 bytes hard coded for now).
 ' SerialInBlock (ComPort, Dest, MaxNumBytes)
    Delay(0,200,msec) ' Add a small delay between each puck read.

Select Case cfgRSSCommType

  Case "B&B232"

  Select Case RSSnumber

     Case 1  'Road Sensor 1 on Com Port 1
        SerialFlush(Com1)
        SerialOutBlock (Com1,UMBpoll,32)
        Delay(0,150,msec) ' Add a small delay before reading back in.
        SerialInCount = SerialInBlock(Com1,InString,128)

     Case 2  'Road Sensor 2 on Com Port 2
        SerialFlush(Com2)
        SerialOutBlock (Com2,UMBpoll,32)
        Delay(0,150,msec) ' Add a small delay before reading back in.
        SerialInCount = SerialInBlock(Com2,InString,128)

     Case 3  'Road Sensor 3 on Com Port 3
        SerialFlush(Com3)
        SerialOutBlock (Com3,UMBpoll,32)
        Delay(0,150,msec) ' Add a small delay before reading back in.
        SerialInCount = SerialInBlock(Com3,InString,128)

     Case 4  'Road Sensor 4 on Com Port 4
        SerialFlush(Com4)
        SerialOutBlock (Com4,UMBpoll,32)
        Delay(0,150,msec) ' Add a small delay before reading back in.
        SerialInCount = SerialInBlock(Com4,InString,128)

     EndSelect

  ' For ISOCONs, they are all on the same serial port, accessed by addressing in the UMB string
  Case "ISOCON"
     SerialFlush(33)
           SerialOutBlock (33,UMBpoll,32)
           Delay(0,150,msec) ' Add a small delay before reading back in.
           SerialInCount = SerialInBlock(33,InString,128)
        EndSelect
 EndSub
```

Appendix B – RWIS Station V2 – Design BOM, 2016 (Current)

Sensor Instruments;

- RM Young 5103 Wind Sensor
- *Texas Electronics TR-525M Heated Rain Guage*
- HC2S3 Rotronic HygroClip2 Temperature/RH Probe
- Lufft *IRS31Pro* Intelligent RSS with subsurface probe.
- *Non-Invasive Pavement Sensing* added/replacing In-Pavement sensors.

* *Blue italics indicate a change from original V1.0 Design*

Appendix B – RWIS Station V2 – Design BOM, 2016 (Current)

RPU Components;

- Campbell Scientific CR1000-Extended Temp.
- Campbell Scientific NL116 Network Interface-Extended Temp.
- (1) DC 12V Power Supply, *(1) DC 24V Power Supply*
- Aluminum Fabricated Backboard, Alodine coated. *Updated for more flexibility in layouts and accommodation of future datalogger.*
- (48) Surge Suppression blocks, one for each external wired connection.
- Campbell Scientific SIO1 SDM module (serial comm). Fewer needed.
- *(4) Lufft ISOCON RS485 isolated power and data.*
- Miscellaneous wiring and terminal blocks.

* *Blue italics indicate a change from original V1.0 Design*