Water Vapor Dial Labview Software

Brad Schoenrock
Robert Stillwell

NCAR
August 3, 2018

Contents

1 Acronyms and Nomenclature **1**

2 Codebase **2**

3 Setup and Configure Files **3**

4 Getting Started **5**

5 Functionality ***needs updating* **7**
5.1 Individual Element Controls 7
5.1.1 MCS 7
5.1.2 Weather Station 7
5.1.3 Laser Locking 7
5.1.4 Housekeeping 8
5.1.5 UPS 8
5.1.6 HSRL Oven 8
5.1.7 Wavemeter 8
5.1.8 Thor 8000 8
5.1.9 Quantum Composer 8
5.1.10 Power Switches 8
5.1.11 NetCDF 8

6 MainOps **9**

7 Data **10**
7.1 Merged Data Output 10

Chapter 1

Acronyms and Nomenclature

- WV DIAL: Water Vapor Differential Absorption Lidar
- MCS: Multi-Channel Scaler
- UPS: Universal Power Supply
- HSRL: High Spectral Resolution Lidar
- DB HSRL: Diode Based High Spectral Resolution Lidar
- T DIAL: Temperature Differential Absorption Lidar
- MFF computer: Micro Form Factor computer

Chapter 2

Codebase

We have used github for our version control. The github repo is located here -

https://github.com/NCAR/WVD-MCSupdate

Instructions for how to install it as well as other software for the operations of the DIAL units is located in google drive here -

https://docs.google.com/document/d/1IzeoFDJbz2wUmzOSmWAWuVJSXAY20Qs0wkWSpMHNQFk/edit

The software should be on each of the DIAL units already, and if those instructions were followed a shortcut to the codebase should be located on the desktop named WVD-MCSupdate. The WVD-MCSupdate directory has 4 sub directores. They are WVDoldSoftware, WVDNewArchitectureUpdate, WVDMCSupdate, and tempShare. WVDoldSoftware has a copy of the old control software. WVDNewArchitectureUpdate has the updated control software. WVDMCSupdate has a copy of the original MCS control software, which was not integrated into final software. tempShare has some small scripts and explorations. The only part of this software that runs for operations is located in WVDNewArchitectureUpdate. Within the WVDNewArchitectureUpdate directory is a docs directory that holds documentation on the deign and operations of the DIAL unit (including this document). The WVD_Architecture_Update directory holds the software for operations. WVDIAL_Main.vi is the way to access the functionality of this software. The other .llb files hold the supporting functionality, but everything needed should be accessable through the main vi. The setup of each of the modes of operations are done through configure files located in the ConfigureFiles directory. The intermediate and final data products are located in Data, which is RSynced to a backup hard drive on the unit as well as to Eldora for collection and processing. The Design documentation located in docs has more detail for development of this software with details of each of the libraries seen here.

Chapter 3

Setup and Configure Files

The software is primarily set up through configure files. Default configure files for most functionality is set up through the ConfigureFiles directory, with the main configure file named Configure_WVDIALMain.txt setting up the main container startup. The Configure_WVDIALPythonNetCDFHeader.txt is a tab delimited configure file to put metadata into the final data products. The file 815nm_841nm_HITRAN_2008.csv is a file that holds absorption lines for calculation of derived products. Each DIAL unit then has a directory to hold configure files named ConfigureFilesDIAL#. By default those config files should be functional, and can be edited to your needs. If some functionality isn't working double check the configure files. That is by far and away the most common problem, and is probably the easiest to fix. Look over each of the entries in the configure files and double check that those settings make sense, have the correct number of entries, that you are editing the config file for the unit you are actually on, etc.... There is a config file for each child that can be called from sub-functions each of which have their own config file.

```
Written By: Robert Stillwell
Written For: NCAR
This file is used to define the initial state of the WVDIAL main function.


Tab Names;;
Main VIs; Testing VIs; System Status; Currently Running VIs;;

Global Variables;;
Global Variable 1; Global Variable 2; Global Variable 3; Global Variable 4;;


File Path Data;;
C:\Users\WVDIAL\Desktop\WVDIALData\Raw\;
C:\Users\WVDIAL\Desktop\WVDIALData\Zipped\;;

Global File paths;;
C:\Users\WVDIAL\Desktop\WVDIALLabview\DataLogs\OperationsLog.txt;
C:\Users\WVDIAL\Desktop\WVDIALLabview\DataLogs\SystemStatusLog.txt;
C:\Users\WVDIAL\Desktop\WVDIALLabview\DataLogs\UserCommentLog.txt;
C:\Users\WVDIAL\Desktop\WVDIALLabview\MainFunctions.llb\SystemWarmUp.vi;
C:\Users\WVDIAL\Desktop\WVDIALLabview\MainFunctions.llb\Operations.vi;
C:\Users\WVDIAL\Desktop\WVDIALLabview\TestingFunctions.llb\MCSTesting;
C:\Users\WVDIAL\Desktop\WVDIALLabview\TestingFunctions.llb\LaserTesting;
C:\Users\WVDIAL\Desktop\WVDIALLabview\TestingFunctions.llb\WavemeterTesting;
C:\Users\WVDIAL\Desktop\WVDIALLabview\TestingFunctions.llb\SwitchTesting;;
```

Figure 3.1: *An example of the structure of a configure file. Variables are identified by a single name followed by two semi-colons. The next line has the required values delimitated by a single semi-colon with the end of line denoted by a double semi-colon.*

If a configure file has been edited and a revert to a default state is required, local changes to the configure file can be discarded and replaced with the defaults from the github repository. This is accomplished from the git bash terminal by navigating to the ConfigureFiles directory for the unit and running a command like

-

git checkout ⟨ConfigureFileToCheckoutFromGithub⟩

If you want to replace the configure file in the github repo with a currently functioning configure file, then run these commands from the git bash after navigating to the ConfigureFiles directory for the unit -

git add ⟨ConfigureFileToAddToGithub⟩

git commit -m “comment describing the commit”

git push

This assumes that your local repo is up to date with the remote repo. Much, much more can be learned through a google search about Git/Github including branches, releases, etc... than i will be able to describe here.

Chapter 4

Getting Started

Now that the config files are together we can get started. Open The vi WVDIAL_Main.vi and start the main container. It should look like figure 4.1. All functionality of the container is within a tabular control. The first tab holds what you see now, and when an operation mode is selected the functionality associated with that mode will open up in tabs that were defined in the configure files associated with that mode. The main container also has some readout information near the bottom of the screen as well as three buttons for permissions, comments, and an abort. For relampago deployment the permissions and comments are not hooked up to anything, but the abort button is hooked up and will stop the labview program. When you want to stop labview operations make sure to stop the current running mode (usually main operations) wait for it to fully stop (all the tabs will go away and the buttons on the main tab will return) and then hit the abort button. Another feature that you will notice when entering an operational mode is a child responsivness display on the right hand side of the screen shown in figure 4.2. That display will show the children that are currently being used, and will confirm their responsivness to the main container.

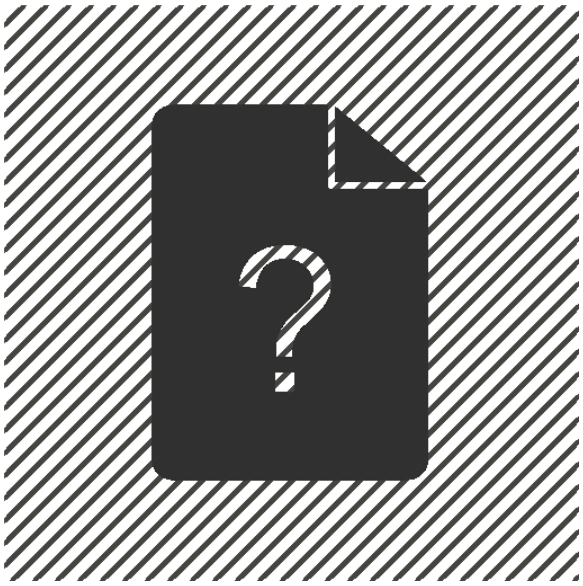

Figure 4.1: *The main container on startup.*

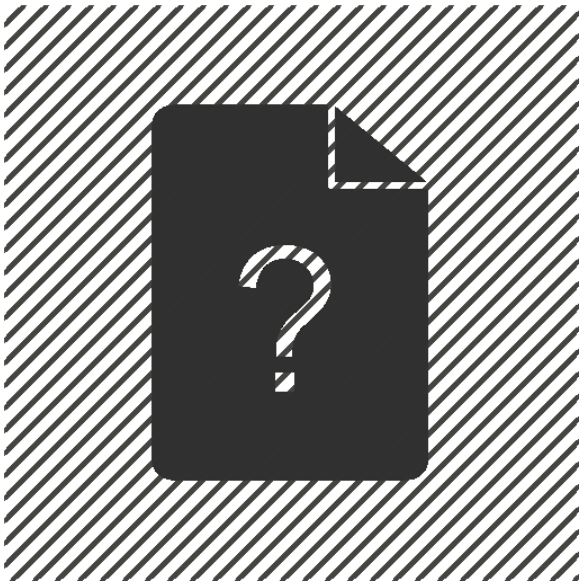

Figure 4.2: *The display showing children as responsive.*

Chapter 5

Functionality ***needs updating***

These are the things you can do with the software, represented by buttons in the main container. The operational modes that are called by the buttons on the main panel are:

1. Warm up sub-function that brings all hardware to operational status. This includes things like warming the lasers and warming the etalons, which needs to be done before high quality data can be taken.
2. Main operations sub-function that performs all the mission critical hardware communication during data collection. This is discussed further in Chapter 6
3. A template sub function which brings up an empty child with minimal functionality.
4. Switches sub-function which tests our ability to control the switches.
5. Temp. Scan sub-function which sweeps through temperatures to test the lasers.
6. Testing sub-functions for individusal controls to check operational status of hardware pieces such as the wavemeter (laser locking), the MCS operation, or the weather station.

5.1 Individual Element Controls

The proposed software update parses the main hardware control function into sub-functions. These sub-functions serve to control individual elements of the WVDIAL, serve as simplified routines to warm up elements of the WVDIAL, or are to test out specific functionality in isolation of the rest of the unit.

5.1.1 MCS

A sub-function that brings up two children. One does the communications via UDP to read the MCS, while the other is a set of controls to change the state of the MCS. These were split into two functions in order to prioritize the UDP communication so photon counting data was always running without interuption, and so that while the child was reading the UDP port the controls would continue to feel responsive. In a previous version of the MCS software putting the UDP communications in the same VI as the controls would lead to delays in the responsivness of the front panel due to the translation from a series of controls into a 32 bit hex word and back that was needed for MCS communications.

5.1.2 Weather Station

A sub-function that brings up the weather station child to monitor surface level temperature, pressure, relative humidity, and absolute humidity.

5.1.3 Laser Locking

A sub-function that brings up the laser locking routines that controls laser wavelengths and the etalons.

5.1.4 Housekeeping

A sub-function that brings up one child whose responsibility is to relay information about the temperature of the container. Thermocouples are placed within the container in various positions which can be specified for writing into the data in the Configure_WVDIALPythonNetCDFHeader.txt. This is primarily to help ensure that the climate control for the unit is functioning properly.

5.1.5 UPS

A sub-function that calls the UPS child to monitor the state of the UPS Battery and power to the unit. The UPS child has a subroutine to automatically send out an email when the UPS Battery gets too low.

5.1.6 HSRL Oven

A sub-function that warms up the HSRL. This is not currently built, but the button on the front panel is there for the addition of the feature in the future.

5.1.7 Wavemeter

A sub-function that brings up the wavemeter to read the wavelengths of the lasers.

5.1.8 Thor 8000

A sub-function that controls the Thor 8000 laser diode current control module.

5.1.9 Quantum Composer

A sub-function that controls the Quantum Composer timing unit. For the Relampago release the only functionality is to write, the read function does not work. When writing to the QC you may have to click through a couple pop ups in order to sucessfully set the state of the QC.

5.1.10 Power Switches

A sub-function that controls the power switches.

5.1.11 NetCDF

A sub-function that brings up the NetCDF writer for reprocessing of data files.

Chapter 6

MainOps

This is the most fundamental part of the software. While deployed in the field this is the mode that the software will be running in for a large majority of the campaign. To begin main operations click the MainOps button. It will take a moment for all children to start but when they do you should see all children as responsive on the right hand side. The tab names are configurable via the Configure_WVDIALMainOps.txt configure file. I will refer to them by their default names.

The first tab is always Main Controls, which holds the controls to begin modes of operation. The next two are Raw Data and MCS Controls which are used to communicate with the MCS and gather photon counting and power data. The Laser Locking tab controls the wavemeter and is used to control the lasers and etalons. The UPS monitors the state of power to the unit and the UPS battery. The Adv. Visualization tab is used to display composite variables, but does not write any data. This tab could completely crash and would not affect the final data products in any way. The NetCDF tab is used to control the python scripts that write out the final and intermediate data products. Lastly the Weather Station tab controls the weather station.

When you start main operations be sure that all children are reporting as responsive to the main container, then flip through the tabs to be sure each is responding as you would expect. For the Raw Data tab make sure that photon returns look reasonable, that peaks are showing up where clouds are present, that the data is changing on the expected interval (1/2 Hz by default), etc.... Go into the Laser Locking tab and make sure the wavemeter is responsive. There is a known problem with the wavemeter that causes it to return a default value instead of actual wavelengths, so make sure to give this 5-15 seconds for the wavemeter to respond with real values. If the unit is running not on the battery then check that the UPS is reporting a full battery, and if it is not then make sure the battery is charging and report the UPS behaviour. Check the Weather Station tab to ensure that the weather station is responsive with physically sensable values, the default frequency for weather station returns is 1/10 Hz so don't be suprised if it looks like it isn't updating. Lastly the NetCDF tab will begin by running the python on startup. This python code can take a minute to run so if the output and error boxes are empty then check that the PythonRunning light is lit. When the python is done running the output and error fields should be filled. If the python runs sucessfully the end of the output box should end with a Goodnight World line that has a time stamp. The error field is not empty even in the event of sucess. Seeing output in this field is not a problem or a bug, it should look like a series of paths to output files. If there is any error reporting in this field that is not a path to an output file, then that is an error, and needs attention. Attempts were made to log errors and warnings and the location of those error and warning files is printed at the top of the output.

The data should be written to the Data directory, as well as being RSynced to an external hard drive connected to the unit, and lastly it should be RSynced to Eldora. Make sure that the data product is actually being written to each of these locations.

If you've looked at all these tabs and checked the data output then congratulations, you're done. You can walk away and let the DIAL unit run for as long as needed. You can collect your final data products as described in chapter 7.

Chapter 7

Data

Data is saved in three locations, in the Data directory embedded within the software, on an external hard drive conneted to the DIAL unit, and on Eldora. The Data directory has several directories containing data in various stages. The final form of the data is the merged CFRadial files which is in the Data/CFRadialOutput directory, more on those below. Raw data is stored in two forms, NetCDFOutput which contains raw data in a NetCDF format, and in either binary or text format (child dependant) which is stored in seperate directories for each child. The merged CFRadial files should be the files you are interfacing with by default, but if for some reason those were not able to be written don't panic. As long as the binary and text files are being written then you have your data, and raw NetCDF files as well as the merged CFRadial files can be re-processed after the fact if nessicary. Raw NetCDF files are derived from the binary and text files, and merged CFRadial files are derived from the raw NetCDF files.

7.1 Merged Data Output

Merged files can be checked with a quick print script which is located in tempShare. That function is NetCDFQuickPrint.py. It can be called from the command line as -

python NetCDFQuickPrint.py ¡pathToMergedFile¿MergedFiles130000.nc WVOffline

Once you get the path to the merged file correct you can pick a data product that you want to plot. The options are (if they are present in the file) [WVOnline, WVOffline, HSRLCombined, HSRLMolecular, O2Online, O2Offline]. For Relampago there is no HSRL or O2 lasers so those selections are expected to error out if you are using this quick print script. Before it errors out it should show you the variables and dimentions so you can get an idea of what is in the file. The option of what to plot is currently limited to a photon counting selection, but hopefully this example script allows you to start reading the merged CFRadial files in python.

When attempting to read these files in with Matlab, be sure to use the hdf5read functionality instead of the ncread functionality. This is a consequence of having chosen NetCDF4 as the format of the files and Matlab's interaction with NetCDF4's chosen string formatting. This is a known bug that has been submitted to Matlab, it is unknown when it will be addressed. Example Matlab code to read in these files is on Eldora.