# Extract Data from Files

## Purpose and Use

This program is designed to extract all of the data from the analysis text files that are created during aerodynamic analysis. From the data, it calculates the mean, standard deviation, and coefficient of variation for MFR, SGP, PTP, RL, and SPL for each task for each subject. A new text file is created (default name is DATA) that contains all the extracted data for the folder you select.

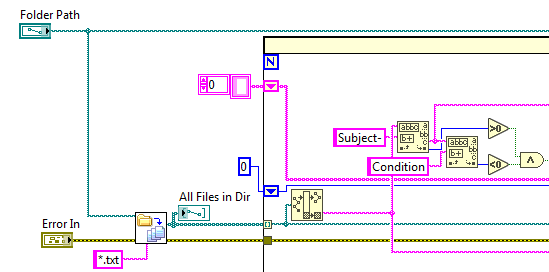
To use, simply click on the “Extract Data from Text Files” button on the main control panel. Then select the folder that contains all of the analysis text files (usually the ‘Subject-#’ folder). You can then save all the extracted data into a new text file. The data is also copied to the clipboard if you wish to add everything to an excel sheet.

## The Block Diagram

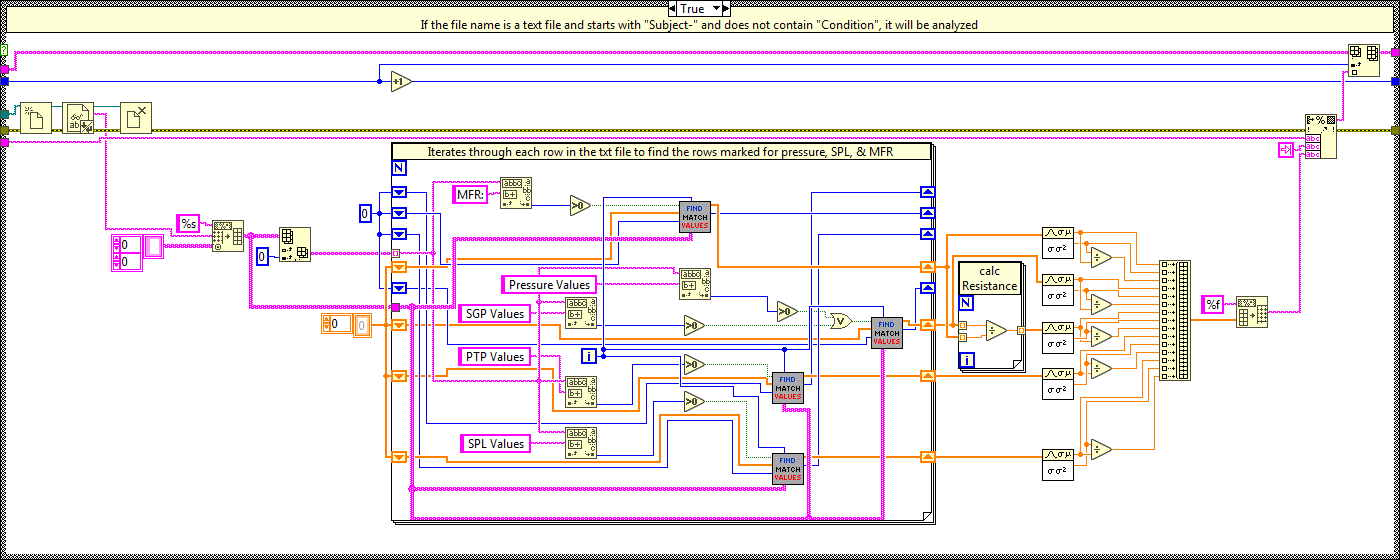
~~This SubVI is named~~ *~~Calc Resistance~~*~~.~~ Its original purpose was to calculate laryngeal resistance from pressure and flow measurements then expanded to extracting all the data from the text files. ~~I just never got around to changing the name.~~ This has been changed to Extract Data.

This SubVI is not set up in any special program structure, it is simply some nested for loops and a case structure. If you need to change something on this program, please review this prior to making edits to the LabVIEW files.

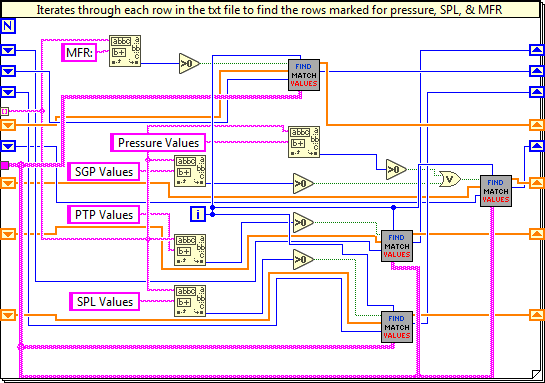
### The outer for loop

A folder selection prompt is located in the appropriate case of the event structure in the *Master* VI and is then passed into the C*alc Resistance* SubVI. Then, the *Recursive File List* VI finds all of the files within the selected folder and outputs the paths into an array. The outer for loop iterates through this array and uses the Match Pattern function to find files that contain “Subject-“ but do not contain “Condition”.

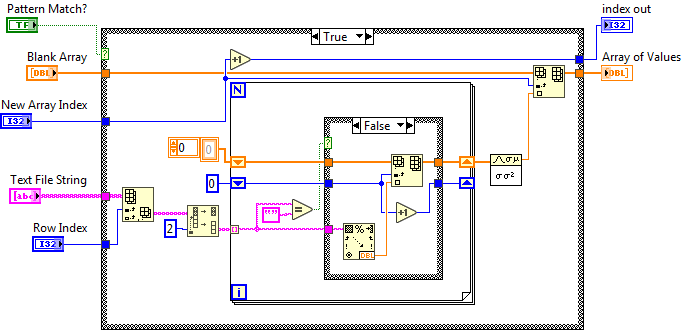
### Main case structure

If a file name matches the search terms, the program enters the True option of the main case structure where it will then search for the lines that contain the different types of data and extract the numbers. If the file name does not match, nothing happens.

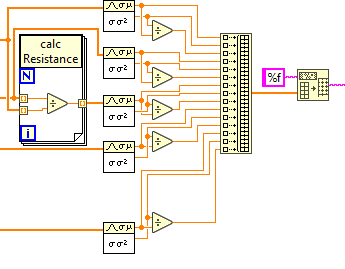
### Inner for loop

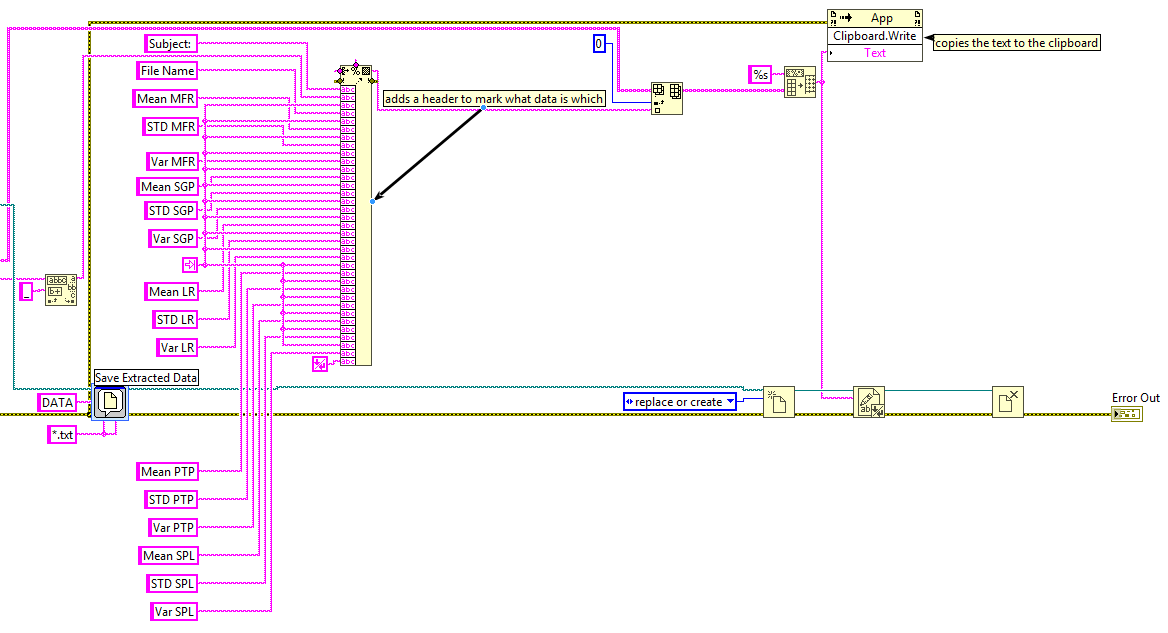
This loop iterates through each row of the text file searching for headers that correspond to the data in that row. Matching rows are passed through the *Find Match Values* SubVI. Here, the data held in the text file is converted to a double and the mean of that row of data is calculated. Thus, the mean of that trial is output and added to an array. Note that only mechanical trials will have a row listed as PTP values. Pressure data extracted from the labial trials will be listed under SGP in the final data file.

### Match Values Subvi

This SubVI extracts the numerical information from the matched row of the analysis text file. It uses a for loop to iterate through the row starting at the second index (LabVIEW is zero-indexed). This is so it starts at the first number, not at the row title. Once through all of the columns through, the average is calculated and added to an array.

### Final calculations and Text file

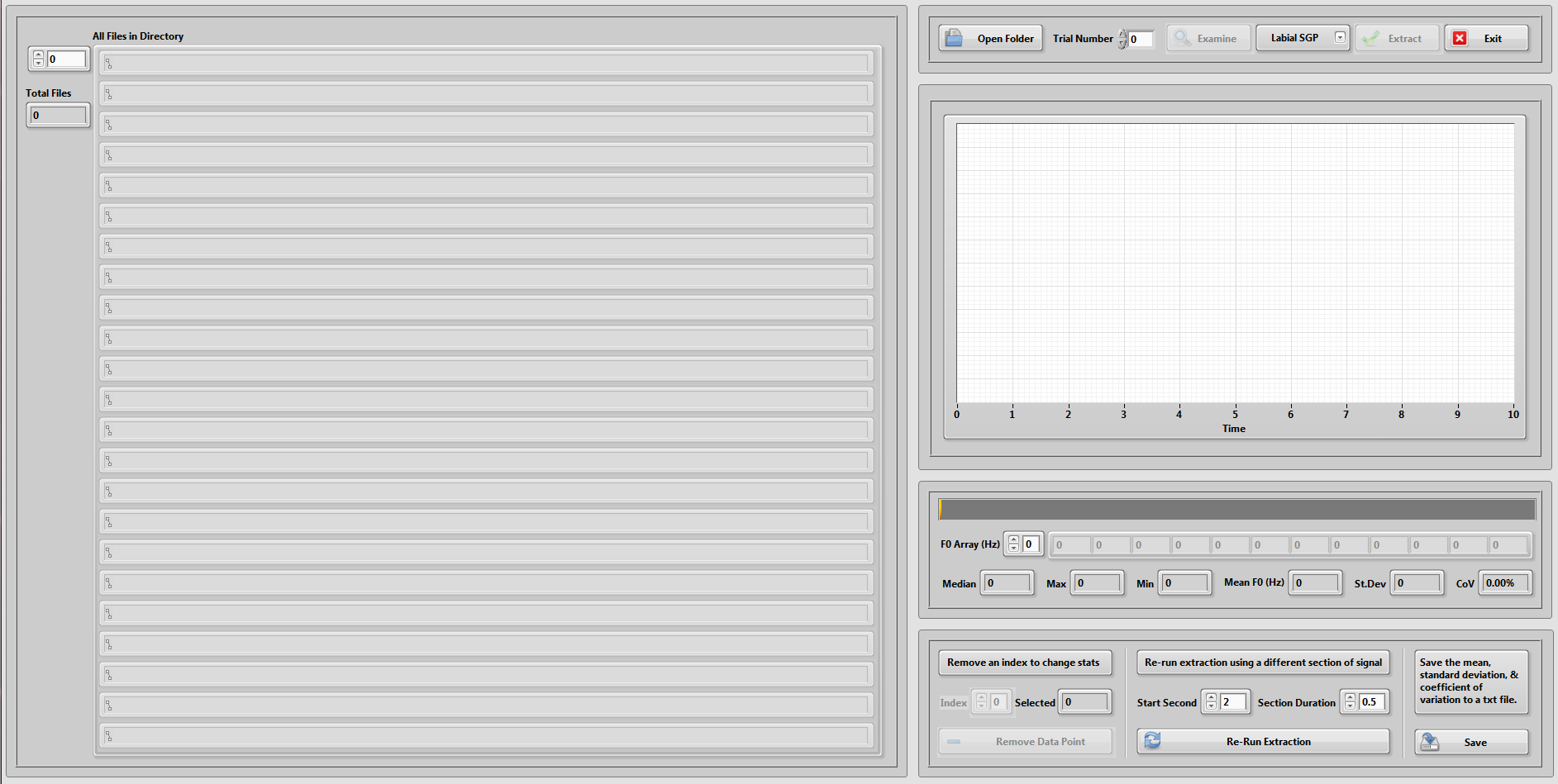
Once all of the rows of the text file are scanned and data is extracted, the mean, standard deviation, and coefficients of variation are calculated. Resistance is also calculated by dividing the SGP values by the MFR values. All of this data is then converted back into a string so it can be input into the new DATA text file. Next, a header is created and added to the top of the new next file. The data collected is also copied to the clipboard here using an invoke node.

NOTE: Invoke nodes are sort of like property nodes, but for the entire program. Do not worry too much about them. They are rarely used in my programs.

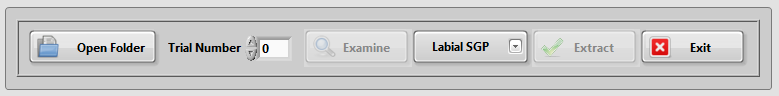
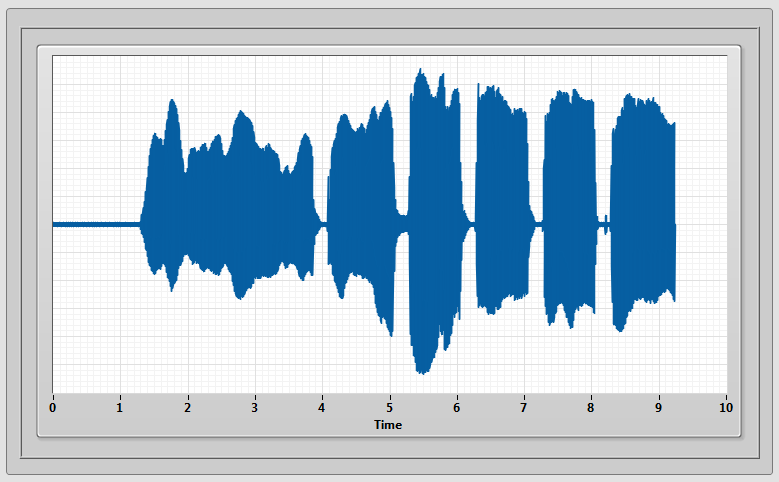
# Extract Fundamental Frequency

## Purpose and Use

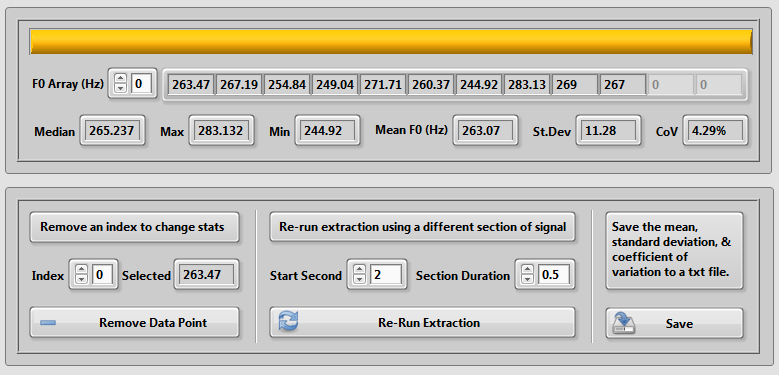
This program was designed to calculate the fundamental frequency from the acoustic traces of subjects collected prior to January 28th, 2019. Prior to this date, we did not normally extract frequency during our normal analysis. This functionality has been added to the Labial and Mechanical analysis.

Unlike my other extraction program, this one has a front panel so the user can see the data being extracted and manipulate it if needed.

On the left-hand side of the front panel is an indicator for the array of files found in a selected directory. On the right are controls for selecting the folder, choosing the test type to be analyzed, and options for data manipulation.

The upper right toolbar allows the user to select a folder to open. Once a folder is selected, the file paths will populate the directory indicator and the Examine button will be enabled. Clicking this will pull the audio data from the selected trial number of the selected test type (assuming only a single subject folder is selected). The audio data will appear in the graph.

Once at least one trial is examined, the user can then extract the fundamental frequencies of all the trials by hitting the Extract button.

F0 is extracted from a section specified in the bottom toolbar. The default is to start at 2 seconds and have a ½ second section. Once all the frequencies are calculated, they are displayed in an array. The median, maximum, minimum, mean, standard deviation, and coefficient of variation of that array are displayed. If one of the extracted frequencies seems abnormal, the user can examine that trial and determine whether or not to remove it from the analysis. This can be done by selecting the index of the desired value and clicking the Remove Data Point button.

Once the user is satisfied with the array of frequencies they have extracted, they can click the save button to save the mean, standard deviation, and coefficient of variation to a text file.

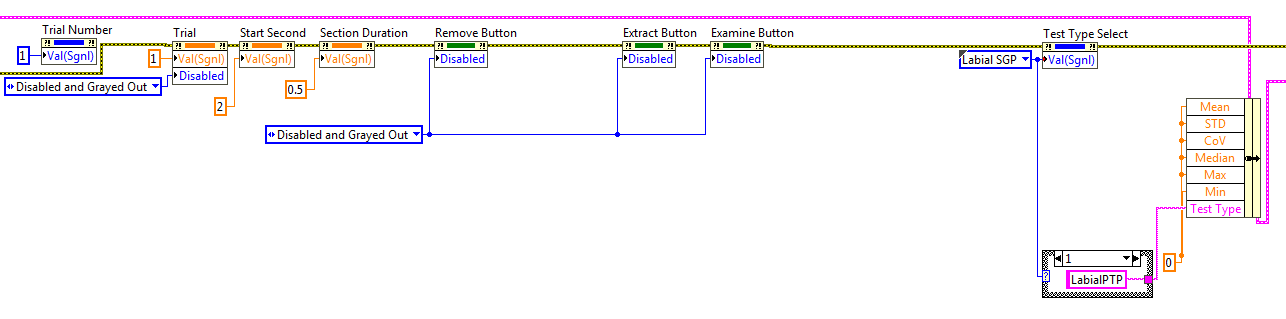
## Block Diagram

This SubVI is organized into a state machine pattern. The different states/cases are as follows:

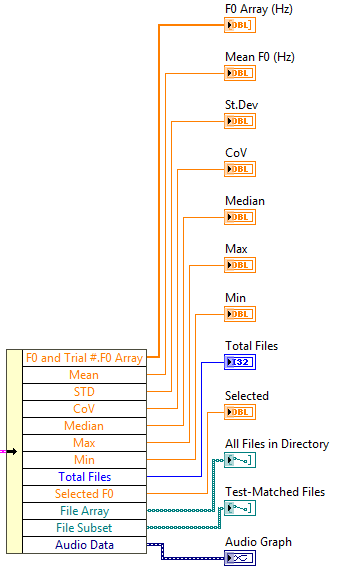
* ***Initialize*** *–* Initializes front panel controls then calls the "update" case.
* ***Update*** – Updates the front panel objects then calls the “wait” case.
  + This case is called after all other cases except exit and save.
* ***Wait*** – Contains an event structure that waits for a value to change on the front panel.
* ***Open Folder*** *–* Prompts the user to select a folder. They should choose a subject folder.
* ***Examine*** – Opens the selected trial of the selected type and displays it on the graph.
* ***Extract*** *–* Calculates the fundamental frequency of each trial at the specified section.
* ***Remove*** – Deletes the selected index form the F0 array.
* ***Save*** – Saves the mean, standard deviation, and coefficient of variation of the F0 array.
  + The trial number and corresponding F0 values are also saved – Update 2/1/2019
* ***Exit*** – Exits the program.

Note that most of the data transferred between cases is done with a cluster.

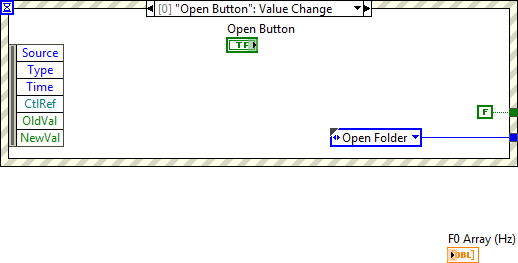
### Initialize

Like most “Initialize” cases, this is where we set the defaults for the front panel objects. All of the statistical values are set to 0 and the Remove, Extract, and Examine buttons are all disabled. The graph is also cleared and the directory is set to empty.

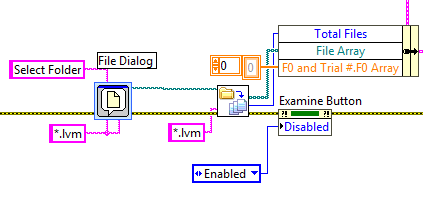
### update

All of the front panel indicators (except for the progress bar) are updated from new values that have been bundled into the data cluster.

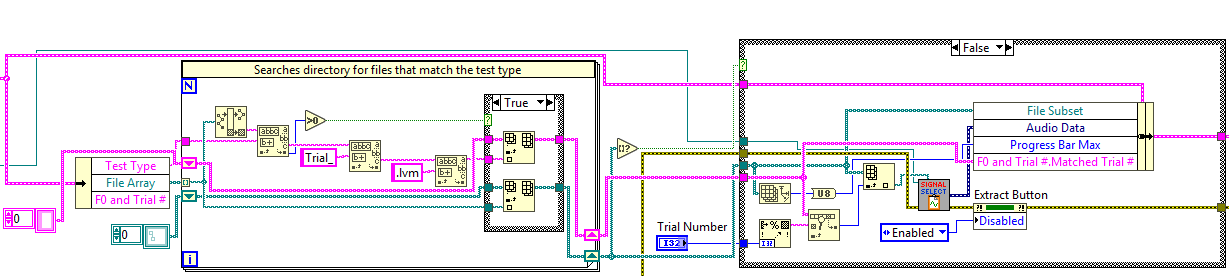
### Wait

This simply contains an event structure that waits for buttons to be pressed, the index selector to change, or the application to close. Each case calls up the next state depending on which button is pressed. The block diagram object for each of the front panel controls is also

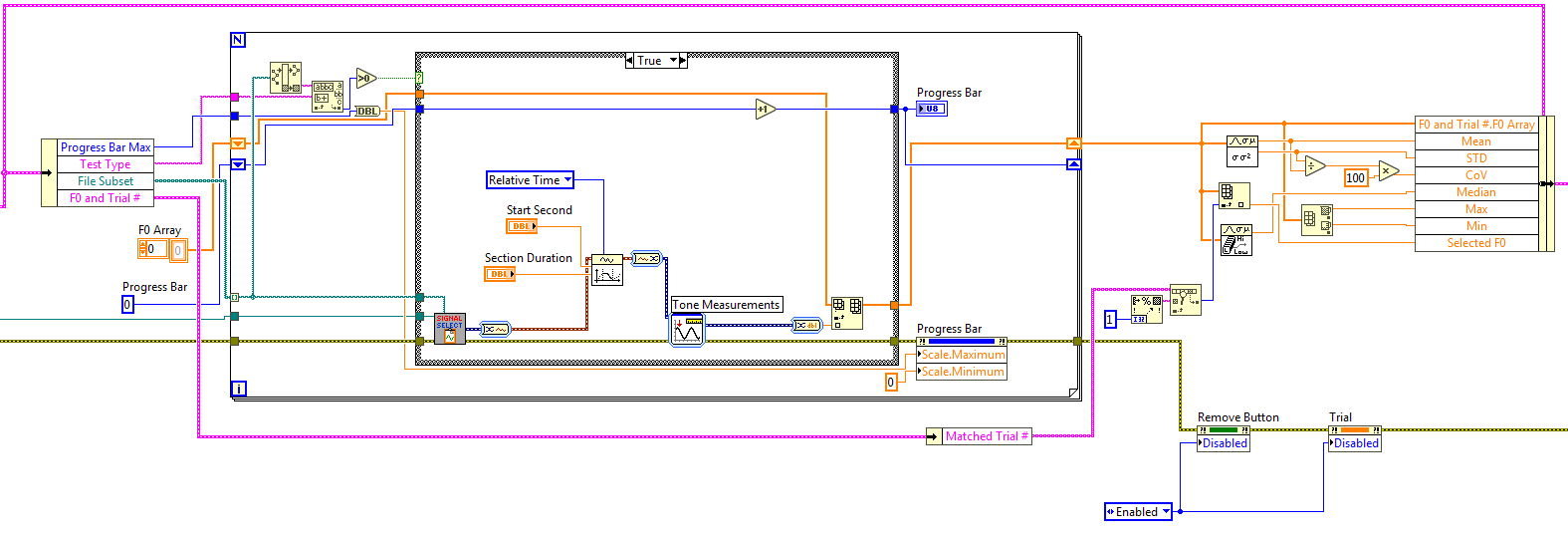
### Open Folder

Here, the program prompts the user to select a folder. It then searches for all the files that have the extension .lvm and builds an array of those file paths. These are sent to be displayed in the directory array on the front panel. The Examine button is enabled here. Whenever this case is called, the F0 Array is also reset. From here, the Update case is called so the file array can be displayed.

### Examine

To pick the file to be displayed, we search through the folder directory using a for loop and a match pattern function to find all of the files that match the selected test type. These are then added to an array that is saved as “File Subset” within the data cluster. Another array is created that contains the trial numbers associated with each matched file. From there, the file is indexed based on the trial number from the file subset array and passed to the signal splitter SubVI. You have seen this SubVI used elsewhere in this binder. From there, the audio trace is displayed and the Extract button is enabled. If no files are found to match the selected test type, a one-button dialog notifies the user.

### Extract

Similar to our other extraction program, we search through the directory for the files matching our selected test type. When a match is found, the audio data is extracted, a section of the data is located through the Get Waveform Subset VI, and the fundamental frequency is calculated for that section. Each frequency is then added to an array. Once we have run through all the files, statistics are calculated on the array and the data cluster is updated.

### Remove

If there is a data point that seems out of place and can be confirmed to be abnormal through examining the audio file, the user can delete that value from the array. When the button is pressed, we use the Delete From Array function to remove the selected trial number along with the corresponding number in the string array. Statistics are then recalculated.

### Save

Once the Save button is clicked, a default file name is generated through concatenating strings and extracting information from the first of the text-matched files. The data is then formatted into a spreadsheet string and written to a text file.

