**Backing up the Code**

**Background**

This should only be necessary if you made performance-impacting changes to any of the below files:

* DriveSelect
* VFD\_Template
* M\_Drives
* Motor\_BOM\_Tool
* Repair for M-Line Macros
* Setup for M-Line Macros
* Test
* Detailed Operation Instructions - Drive Selection
* How it Works
* Manual Entry Procedure
* Troubleshooting
* Updating a Lookup Table
* Visual Operation Instructions - Drive Selection
* Visual Operation Instructions - Email Template Generation
* DriveSelect - No PDF Converter
* Emailing\_Saving Macros
* Parse Macros
* VFD\_Template – Backup

It is most likely that you will only ever need to make changes to the yellow highlighted files. All the others are considered infrequent, and you should be very sure before changing anything about the file. Changes made to the red highlighted files will **irreversibly** change the files, and this is very risky considering they are backup files. There are very few instances in which this should be done.

**Instructions**

1. Navigate to [**https://github.com/Bwmosher22/VFD\_Parse**](https://github.com/Bwmosher22/VFD_Parse)(There’s also a link in the **Backup** folder).This is the cloud repository for handling version control. You will have access to create new versions, or “commits” of new files. It’s licensed copy-left, with GNU, so it allows just about everything you’d ever need or want to do to the code
2. The most user-friendly method to do this is by a simple file upload. Begin by noting the file that you changed and its parent folder, and then navigate to this folder in the web repository.
3. In the upper right-hand corner of the repository details page, click the dropdown for **Add file** and then click **Upload files.**
4. Choose the **file** you just changed, enter a **description of the change**, and press the green button near the bottom named **Commit changes.** Make sure your commit will **replace** the existing file in the web repository. In other words, at no point should you commit a file outside of the folder to which it belongs.
   1. This will commit the changes to the repository, and create a **new version** of the specified file. If there is not a file change, GitHub will recognize this and not upload the file to avoid redundancy.
   2. If you ever need to access an old file version for the sake of functionality, feel free to click on the file in question, and look at **History** for any one file. Clicking **View at this point in history** allows you to download the file at any point it was uploaded here, and you can locally restore original functionality to the **S:\** drive folder in the off chance functionality is impeded.
   3. You can never commit too many file versions. GitHub logs it all. If you’ve made even possibly performance-impacting changes to the files specified above, it is highly recommended that you commit the new version to the GitHub repository. It’s the best way to ensure this macro set functions for years to come.