# Dev Update 001

## 02/19/2025

Committed by Spencer Rohwer

Uvsim\_gui.py Changes:

* Relocated UVSim import from after GUI construction to before, making simulator instance accessible to the App Build functions.
* Deprecated the Display Log button. Note that the code is still there, just commented out, in case we want it later.
* Multiple File Select changes. The input field now has focus on App launch, allowing for instant keyboard inputs.
* Additionally, allowing further file selection after one file was partial run caused instruction counter issues. So, to prevent this, only one file is allowed to be accessed per App instance. This means successful file selection:
  + Input field becomes disabled to prevent further input
  + Input field clears the input file name and has background text of “File loaded successfully!”
  + The File Select button becomes disabled.
* All state control buttons (execute, step, and save) are non-functional while the File Select input field is still active, instead outputting a message that instructs the user to input a file.
* Save button now functions, however for simplicity, it no longer asks the user what to call the output file. It just names it uvsim\_output.txt.
  + Save also does not terminate the program like initially designed. This allows multiple saves during stepping. Nice for debugging too.
* Quit button now functions, fully exiting the program.
* Console input field is now disabled by default, only becoming active when specific logic calls for user input. Previously, any entered input in this field would cause logic interruptions and start new wordProcess() calls with crazy and unpredictable input, regardless of the state of the program, leading to a plethora of errors that would not get caught by the program. Better refactoring and state checks could potentially solve this problem, but a disabled field seemed like the easiest fix.
  + Input field enabling is handled by focus\_console\_input() and is disabled in input\_text\_handler() whenever input is entered.
* Console output now starts with a welcome message on startup that has instructions for the user on how to proceed.
* Changed the vertical size of the entries in the memory table by 30%

Uvsim.py Changes:

* Added return values to fileInputToMemory() to validate success or failure.
* Added Save and Quit instructions to Program Halted message.
* Fixed process\_input() to handle invalid input. Previously, invalid input would lead to an infinite loop (explained in the change made to accumulator.py) or simply skipping the input instruction all together. Now valid/invalid info is returned by the accumulator and handled here. Valid proceeds as normal, while invalid sends a message to the console output, sets the instruction counter back one, and calls for the user input instruction to be run again.
* Fixed stepping console output. Previously, stepping would display the last operation run, rather than the one that is currently running. This caused the Accumulator value and the displayed step to be out of sync, and sometimes outright wrong. To fix this, stepProgram() is now check for at the end of both process\_input() and wordProcess() which only displays the current instruction.
* Fixed an issue where Step would not stop the program from advancing after any user-input-related instructions. Step will now cause a console prompt for user input and upon receiving valid input it will immediately display the related instruction, but will not advance to the next instruction until Step is clicked again.
* Made a small logic check in getAccumulator() to ensure that there is a value in the Accumulator before retrieving one.
* saveMemory() now uses a try/except block to keep from crashing on a bad file open. The function now also returns success/fail values so the GUI can validate the save appropriately.
* Added a quit() function that the quit button accesses to wipe the window, close it properly, and exit the program.

Accumulator.py Changes:

* The read() function was still only printing a message to the console on invalid input, leading to an infinite loop that waited for better input that would never come. The function now returns 0/-1 on success/fail so the results can be validated and handled elsewhere.