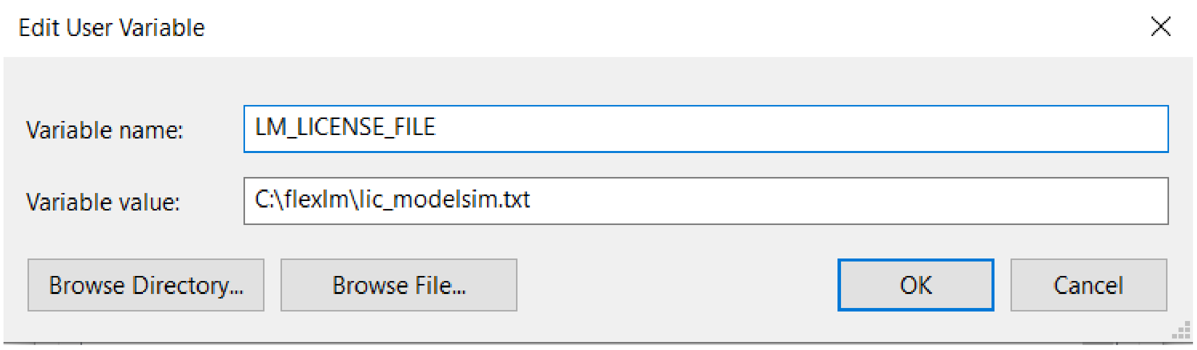
Documentation:

1. If you are wondering what ACSM stands for, its Assembler-Compiler-Simulator-Memcheck 😊
2. I have tried to include everything in this script so that you don't have to do anything other than run the script.
3. Place your ACSM.py in the same directory as your **design files (Verilog files)** and your **Assembler**.
4. Before running, make sure you have your ModelSim **Win64** directory as a Path variable in your system Variable.
   To do this: Search "Edit the system environmental variables" in your search bar-> Environmental Variables -> Double Click path under "system variables" -> New.
   In my case, "C:\modeltech64_10.5\win64" is the NEW path.
5. Make sure ModelSim runs when ran without admin privileges. In case this gives a license error, add the location C:\flexlm (location of ModelSim license file) to User Environmental variables.
   To do this, Control Panel -> User Accounts -> User Accounts -> Change my Environmental Variables (located on the left side of the window) -> Click New under User variables for <USER> ->



Edit User Variable

Variable name: LM_LICENSE_FILE

Variable value: C:\flexlm\lic_modelsim.txt

6. The script will detect all files in the given top directory (whose path is defined "**DIR**" in the script and can be changed by changing the same) and its constituent sub directories. So, **there is no need to move the instruction files separately to another folder**. In case you have any non-instruction files in the same top folder (or in the sub folders), they will get listed under **faulty instruction list** which will be displayed at the end of script execution along with any instruction files the assembler found faulty. If you are too stubborn that you don't want to see the non-instruction files in the list, take your time to add the name (with extension if any) to the **avoid list** which is visible once you open the python script source.
7. For instruction files which contain instructions requiring **read from DM,** make sure to highlight these files by adding an identifier at the starting of the file name. In default, the **identifier assigned is "_"**. You can change it by changing the "**idntfr**" variable.
   For example, if you have an instruction file named "haha" which requires a read from DM, you should rename the file as "_haha" (don't leave behind your extensions!! (if any)).
8. Now, **create** a folder named "**DMrd_files**" whose path is defined through "**DMrdfl_LOCATE**". In there, you should place the pre-created DM files which is required by the instruction files mentioned in point 7. Make sure each DM file placed here has the same name (including extensions) as the instruction file which requires the latter.
   For example, if your instruction file "_hehe.txt" requires a DM file to read from during simulation as mentioned in point 7, pre create this DM file as required and place it in the folder "**DMrd_files**" with the same name as the instruction file, that is, "_hehe.txt".

9. Instruction files with faulty instruction will be displayed (only if there is any) at the end of the python script execution. Display will be as full path.
10. Same goes for any test case that fails MEMCHECK. They will also get displayed at the end of the python script execution separately. Display will be as full path.
11. The DM file of any test case that fails MEMECHECK will get stored in the folder "**MEMfail_files**" who directory is defined by "**MEMfail_LOCATE**". The file's name will be renamed from "dm_file.txt" to "DMfail_<folder of the instruction file is>_<instruction file>" once moved. Every time you run ACSM, the previously stored failed dm files will get cleared from the folder and the folder will be updated with new ones (if any).
12. Please define "**DIR**", "**PM_LOCATE**", "**DM_LOCATE**", "**DMrdfl_LOCATE**" and "**MEMfail_LOCATE**" paths before running ACSM by running the configure_path.py script.
13. If you are kind hearted enough, you can test everyone's test cases by setting "**DIR**" to point to the "Test-cases" folder of your local "GitHub" folder. As mentioned in point 6, you can add "Owner.md" and "Readme.txt" to the "**avoid**" list in the python script source to prevent them from being detected.
14. If any case arises where you had to close the python script before it actually finished execution, make sure that the **$system command in "test_core.v"** and **the logic to reset DM_file in "MEM_top.v"** are **NOT** commented out. This won't affect any runs of ACSM. This is just to ensure proper debugging in case you do so later on.
15. That's all I can remember!!

Configuration of ACSM:

1. To set the configurations of ACSM including paths, identifier and avoid list, run the **configure_path.py** script**.**
2. Place the script in the same directory as ACSM.
3. Update the "**DIR**", "**PM_LOCATE**", "**DM_LOCATE**", "**DMrdfl_LOCATE**" and "**MEMfail_LOCATE**" in the configure_path.py script. Also, the **avoid list** and the **identifier** can be updated through the same. Once the script is updated, run the script to set the ACSM configurations.
4. **No need** to change backward slashes (\) to forward slashes (/) anymore when updating paths in the script. Paths can be updated in the script in the same format as in windows path format.
5. **configure_path.py** when run generates the configuration file "Paths.ini" which will store the ACSM configurations.
6. This makes sure that during each update of ACSM, there is not need to manually change the directory/path/identifier/avoid list configurations again and again when the old ACSM get replaced by the updated version as the configuration file "Path.ini" will remain the same until a change is made in configure_path.py and run again.
7. Even though "Paths.ini" itself can be changed directly as required, its recommended that the changes be made through **configure_path.py** instead.