I spent a little time and created a Post-Processor in python. It should work in Windows, Linux, and Mac without issues. I created so you can configure in the Starting Script How often you would like to prime each tool (-1=Always; 0=Never; 1=Once; 2 and Up = Integral), you can also choose if you want to iterate the tool priming cycles off of tool changes, or layer changes. If you elect to for example prime you T3 every 5 layers or tool changes, the tool will always prime the first time it is used to ensure you are good, then it will cycle per your setting. You include the Active temperature so the script knows what they are, and if the tool is not used Simplify reports the active temperature as -1. You can provide your standby temperature for each tool T1-T5, if the tool is not used, you can just use 0, but it really does not matter. You also have an option to include a report that is displayed when your print starts showing you all available settings, including priming, tool changes, layer changes, how often the tool will prime, and so on. I did my best to line up the rows and columns, but the DUET strips all white spaces when using the "M118" information command. I decided to use and underscore "_" so the column headers would line up with the row data. The "M118" pops up the information like with other information, but it is easiest to read it in the ">_ G-Code Console". I like the report option because it give me a verification of what I expected. You will also have to provide tool changing information and returning z_height in the "Tool Change Script".

With the "IncludeReport={1}", it needs to be exactly as shown or the report will not be added, anything else will be ignored, the command are NOT case sensitive. NONE of the commands are case sensitive.

EXAMPLES:

In "Starting Script":

```
;Starting Script

 M141 P1 S0 ;Set Filament Dry Cabinet to 0% Relative Humidity and activate it.

 ;---------------------- For H-Series Post-Processing ----------------------------------

IncludeReport={1}

;Provides Specified Tool Priming Iterations - Integral - and Integrator for Post Processing

ToolPrime={0:0:-1:1:5:1};Example:(ToolPrime={T1:T2:T3:T4:T5:Layer/Tool})

;Cleaning Iterations{-1=Always; 0=Never; 1=Once; 2 and Up = Integral}

;Integrator{0=Layer; 1=Tool}

;Provide Active Temperatures for Post Processing
```

ToolActiveTemperatures={[extruder1_temperature]:[extruder2_temperature]:[extruder3_temperature]:[extruder4_temperature]:[extruder5_temperature]}; Settings form Simplifiy3D

;Provide Standby Temperatures for Post Processing

ToolStandbyTemperatures={0:0:170:171:172};
Example:(ToolStandbyTemperatures={T1:T2:T3:T4:T5})

;---

; home the machine and perform auto bed levelling

G28

G32

In “Tool Change Script”:

; Tool Change Script

; ---------- For H-Series Post-Processing ----------

ToolChanging={[new_tool]}; Provides the Tool Number

G1 Z[current_position_z]; Returns to printing Height after priming.

; ---

Terminal command (Path to where ever you put the script):

python "F:\user\Documents\3D_Printing\Diabase\fff\Python\H-Series_gCode_PostProcessor/H-Series_gCode_PostProcessor_V1.5.py" "[output_filepath]"

In your “Starting Script” these are the important settings and are required to successfully post-process with the exception of the Report Inclusion, commands are NOT case sensitive:

IncludeReport={1}

ToolPrime={0:0:-1:1:5:1}

ToolActiveTemperatures={[extruder1_temperature]:[extruder2_temperature]:[extruder3_temperature]:[extruder4_temperature]:[extruder5_temperature]}

ToolStandbyTemperatures={0:0:170:171:172}

In your "Tool Change Script", and are required to successfully post-process, commands are NOT case sensitive:

ToolChanging={[new_tool]}

G1 Z[current_position_z]

I have also included “H-Series Universal FFF Profile.fff”, You can import it to Simplify3D, I have configured it with a few materials for each Extruder. The Simply3D FFF profile is in XML Version 1.0 and you can open it in text editor and add your own custom materials. I included “fff_Material_Configuration.py”, if you change your settings and give the material a name like “Ninja Flex”, it will create a text file in the script’s directory using the material name, ie. “Ninja Flex.txt with the XML code for T1,T2,T3,T4, and T5 i.e.: “Ninja Flex_(T3)”,

If you open the “H-Series Universal FFF Profile.fff” in a text editor you can locate where the other “Auto-Configure for Materials” are located and delete the ones you do not want, and add the ones you create. Once you have changed your fff profile you will need to delete the current on in Simplify3D and import the one with your changes.

In regards to the post-processing, the script generates 4 file names:

('original_file_name'_(H-Series).gcode) = Final Output File Name

('original_file_name'_(H-Series).TEMP) = Temporary file for post-processing.

('original_file_name'_(H-Series).~temp) = If you ask for report IncludeReport={1}, this file is used to add the report, otherwise it is not used.

('original_file_name'_(H-Series).ERROR) = If there is any error, this file will include information about the error at the bottom and tips if available on how to resolve it.

The script opens the 'original_file_name'.gcode, and creates 'original_file_name'_(H-Series).TEMP. As the script is processing the results are being added to the .TEMP file. Being as there is no visual feedback from a GUI, I used a file naming schema to provide you with feedback as to if the file is processed of not. When the processing is completed, if you asked for the report, then the .TEMP file is closed and renamed to ".~temp", the a new .TEMP file is opened and the report is added. Once completed all file s are closed and ".~temp" is deleted, and ".TEMP" is renamed to ".gcode". If anything goes wrong, I did my best to ensure all errors are handled and the nature of the error is appended to the ".TEMP" file and then it is closed and renamed to ".ERROR".

I created a 23MB print with multiple models and using T3-T5 and the script and the script takes about 2 seconds to process and include the report. A good rule of thumb is it takes as long as it takes Simplify3d to export the gcode. The same file with the Diabase wizard takes about 1 minute.

I included examples and a couple videos showing the script running. I spent the better part of a week developing it and breaking it to make it as relightable as I could make it. I am currently using it for my post-processing, and if I decided something needs to be fixed, I will post updates.

If you know how to python and you see something that can make it better for everyone, I encourage you to do so and I will use your version.