A simple use case that will help to explain the concept more clearly

Use Case:

Below we have a g_trace that is being collected from the standard output file that we get after running a test.

```
--------------g_trace start------------->

28-June-2021 12:48:24

CHECKSUM = 0x35583303
WRITE OFFSET = 103
=============================================================================================
[  0] TRACE_BOOT                                                  216          0x000000d8
[  1] TRACE_CONFIG_STATE_IDLE                                     1186         0x000004a2
[  2] TRACE_CONFIG_EVENT_BOOT                                     878          0x0000036e
[  3] TRACE_CONFIG_STATE_CONFIGURE                                223          0x000000df
[  4] TRACE_CONFIG_STATE_CONFIGURE                                223          0x000000df

[ 88] TRACE_CONFIG_EVENT_NCONFIG_ASSERT                           427          0x000001ab
[ 89] TRACE_CONFIG_EVENT_PREWIPE                                  428          0x000001ac
[ 90] TRACE_CONFIG_EVENT_INTERRUPTION                             1451         0x000005ab
[ 91] TRACE_BITSTREAM_INTERRUPT                                   250          0x000000fa
[ 92] TRACE_CONFIG_STATE_WIPE                                     970          0x000003ca
[ 93] TRACE_ERROR                                                 384          0x00000180
[ 94] TRACE_ERROR                                                 4026597428   0xf0010034
[ 95] TRACE_ERROR                                                 0            0x00000000
[ 96] TRACE_BIG_HAMMER                                            218          0x000000da
[ 97] TRACE_CONFIG_CNOC_INITIALIZED                               535          0x00000217
[ 98] TRACE_GET_CPU_BLOCK                                         4096         0x00001000
[ 99] TRACE_GET_SKIP_BLOCK                                        0            0x00000000
[100] TRACE_CONFIG_WIPE_DONE                                      919          0x00000397
[101] TRACE_CONFIG_EVENT_NCONFIG_DEASSERT                         78           0x0000004e
[102] TRACE_CONFIG_STATE_CONFIGURE                                1181         0x0000049d

<--------------g_trace end--------------
```

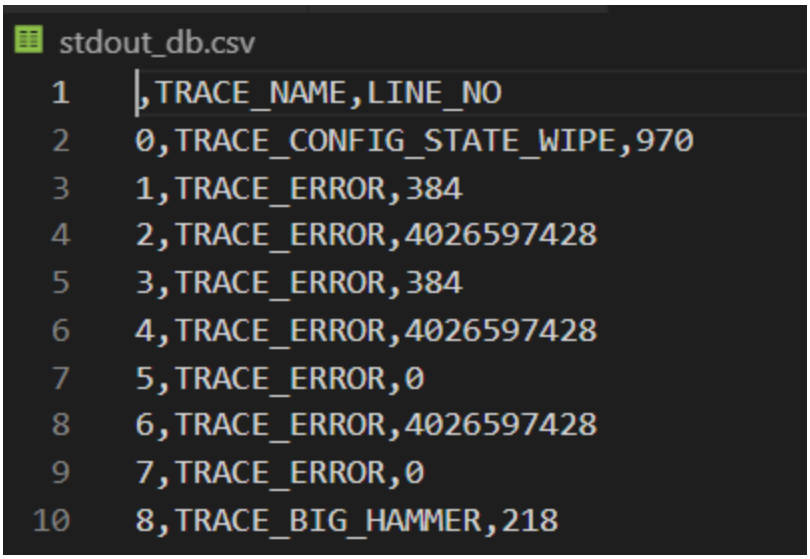
As a developer, I now need to figure out from which file and from which line number the TRACE_ERROR is being generated (highlighted above).

To solve this we can use IFTAT. We can give the tool the standard output file from the test run as input and it would return you the following information.

1. path to the file in which the trace is found
2. function name under which the trace appears
3. trace name
4. line number at which the trace is present in the mentioned file following the file path
5. trace macro which is being called with the target trace name
6. trace info which contains whether the other parameter to the macro is line number

Steps of tool execution:

1. Give the following command on the terminal
   python .\IFTAT_final.py <path_of_standard_output_file_generated_after_test_run>
2. You will be prompted whether you want to build the Main database select yes/no
3. Tool will internally create a pandas table containing only the filtered error traces and a few traces above and below the errored traces (as shown below)



4. In the next step, this Main database would be used as a reference and the tool will search for these errored traces (above) and would return an output file called report.csv that would give you all the required information about the errored traces.

This information is very easy and helpful to get to the exact location of the firmware which is causing the traces to appear in the gtrace and the developer can analyze that particular piece of code of the issue at hand. This information generated by the IFTAT tool not only will help experienced developers but new team members as well, thus reducing the effort to manually search through the codebase.

We have conducted experiments on multiple standard output files generated from different test runs to check the output generated in the report.

We have also experimented with the tool to check the working of Main Database creation, we could observe that if there are some updates in the firmware then our main database can successfully capture those changes wherever required.