**Sherlock Fingerprint Sensor Module Test Flow with Nucleo F446 Board**

# **Test Flow Design**

The module test design flow is implemented into the following test stations for different stages of the manufacturing flow as shown below.

**FLEX, SMT, Bare Module Producing**

**FP Module Vendor**

**Lamination**

**FP Lamination Vendor**

**1**

**4**

**3**

**2**

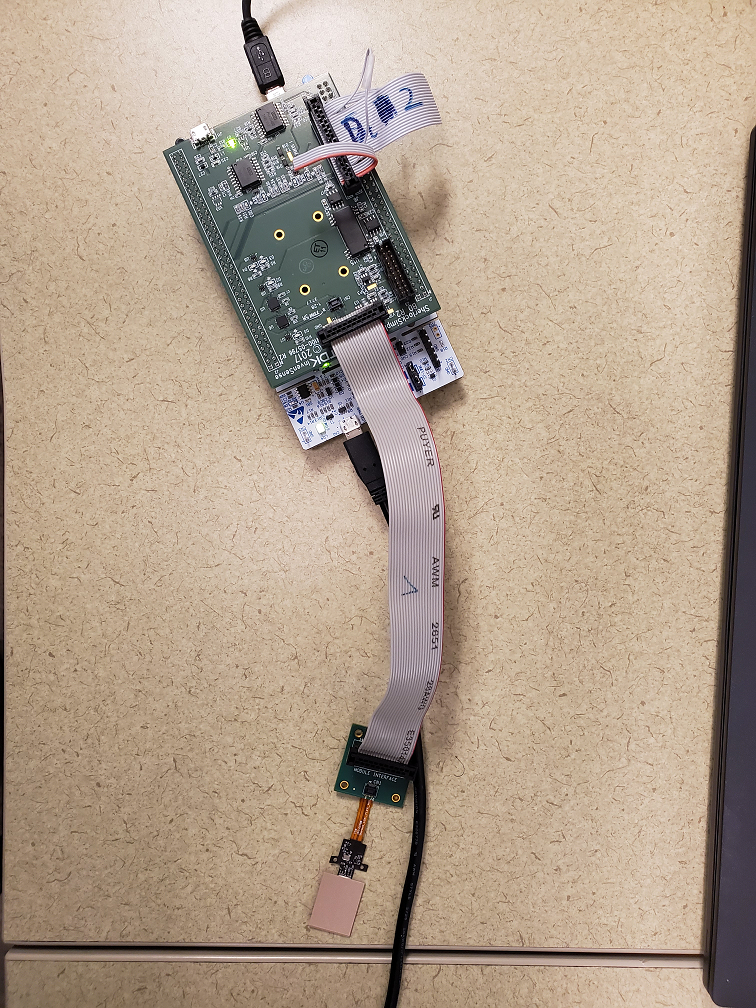
**5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Station** | **Function** | **DUT format** | **Cfg file** |
| 1 | IQC/LGA socket | Sensor | SLK\_Flex-01\_rev\* |
| 2 | FP Module Vendor OQC | Bare sensor/Flex | SLK\_Flex-02\_rev\* |
| 3 | FP Lamination Vendor IQC | Flex\_ Mode | SLK\_Flex-03\_rev\* |
| 4 | FP Lamination Calibration | Module\_Mode | SLK\_Project\_Thickness\_Material\_Glue-04\_rev1 |
| 5 | FP Lamination Vendor OQC | Module\_Mode | SLK\_Project\_Thickness\_Material\_Glue-05\_rev1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Station** | **Function** | **DUT format** | **Tests in Station** | **OTP Write** |
| 1 | IQC/LGA socket | Sensor | SPI, ASIC ID, Calibration, Air Image | NO |
| 2 | FP Module Vendor OQC | Bare sensor/Flex | SPI, ASIC ID, Calibration, Air Image | NO |
| 3 | FP Lamination Vendor IQC | Flex\_ Mode | SPI, ASIC ID, Calibration, Air Image | NO |
| 4 | FP Lamination Calibration | Module\_Mode | SPI, ASIC ID, Calibration, Air & Target Image | **YES** |
| 5 | FP Lamination Vendor OQC | Module\_Mode | SPI, ASIC ID, Air Image, MT pass bit | NO |

# **Test Environment Setup**

## **HW Setup**



Please make sure that the Nucleo FW version is at least 2.3.8 or above.

## **SW Setup**

**Microsoft Visual Studio 2017 Professional version SW**

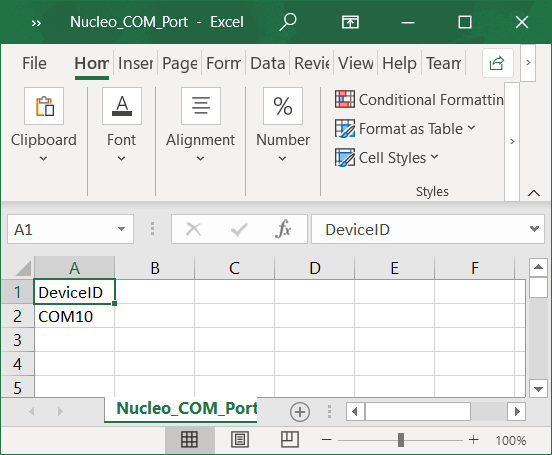
On ***Windows 10 PC w/ 64Bit Operating System***

To be worked with the C++ Reference Codes in the package.

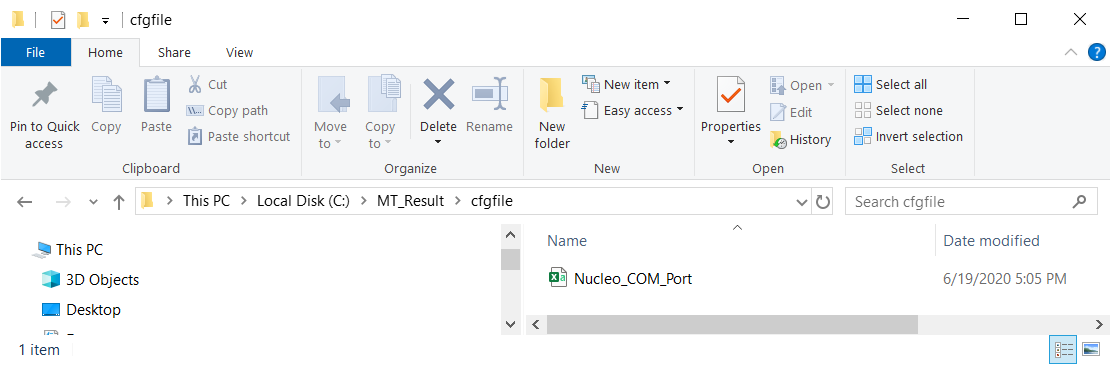
Also, need to check the Virtual COM Port number of the STM32 Nucleo Board from Device Manager.



And update the COM port number into the Nucleo\_COM\_Port.csv in SLK\_MT\_Nucleo\_Rev\*.\*\ folder.



Then copy this Nucleo\_COM\_Port.csv to C:\MT\_Results\cfg folder to be created.



For launching the SLK\_MT\_Nucleo\_Rev\*.\*\Release\SLK\_ModuleTest\_Rev1.exe,

there are two .txt files that would need to be preset under the SLK\_MT\_Nucleo\_Rev\*.\*\Release folder before running the SW.

1. CfgPathInfo.txt

This is to store the location of the .cfg files for the different test stations to be used.

Current default setting is:-

C:\Users\rkwan\Downloads\ModuleTest\SLK\_MT\_Nucleo\_Rev\*.\*\fpsys\_python

And this should be updated to reflect correctly where the \fpsys\_python\\*.cfg are actually stored first.

1. CurTestNo.txt

This file is to store the test station number to be run.

Current default setting is:-

4

And this can be modified between 1 and 5.

**OR**

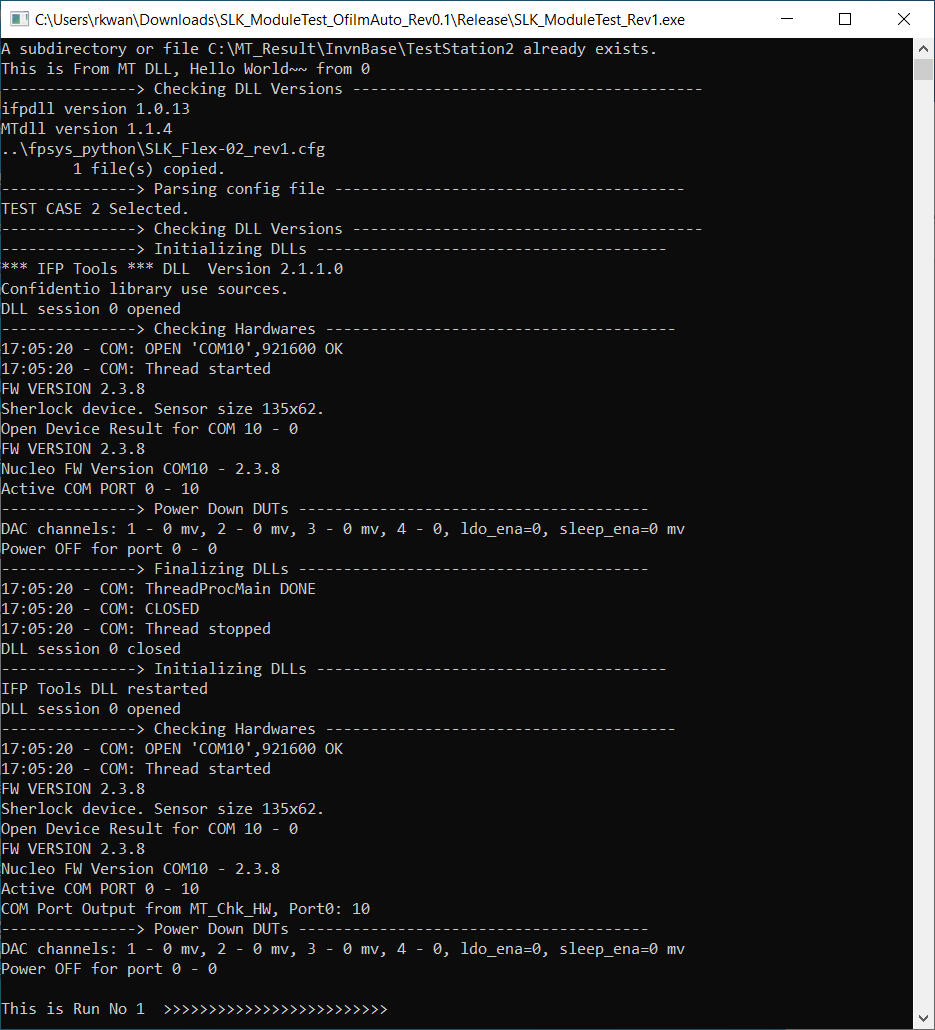
If launching from the project execution inside Microsoft VS2017 Professional version under SLK\_MT\_Nucleo\_Rev\*.\*\SLK\_ModuleTest\_Rev1.sln,

Please update the CfgPathInfo.txt and CurTestNo.txt under the SLK\_MT\_Nucleo\_Rev\*.\*\SLK\_ModuleTest\_Rev1 folder instead.

This is because the VS settings may not be able to look up the ones in \Release folder.

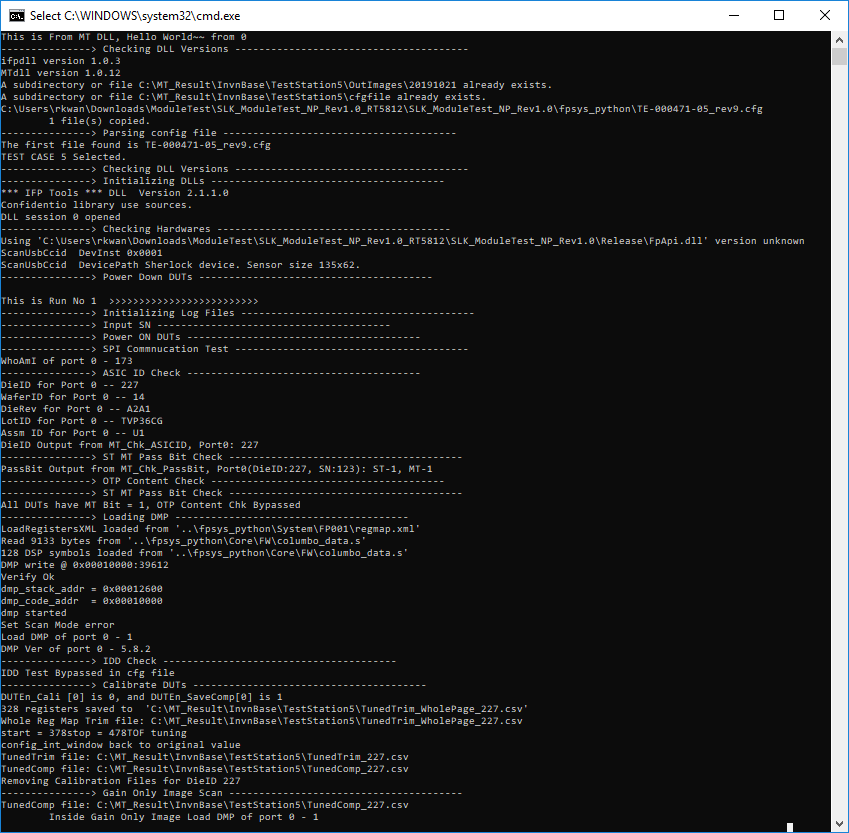
## **Initial Test Setup Inside MT SW**

1. Initialize / Open DLL
2. (Power off DUT in case it was not shut down properly)

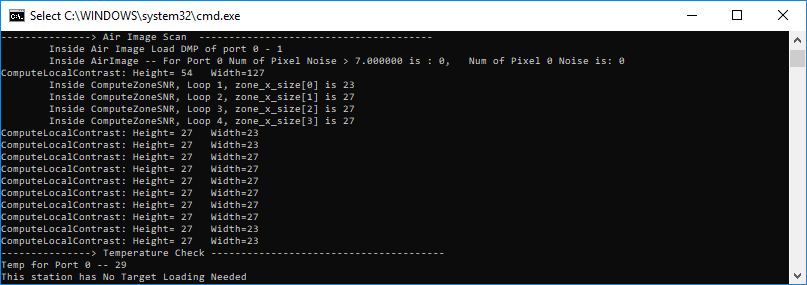


## **Main Test Loop Inside MT SW**

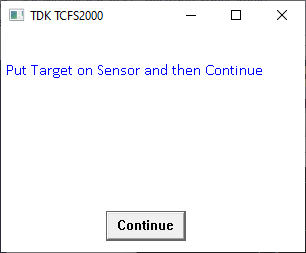
1. Initialize log file
2. Power on DUTs
3. Check SPI communication (Disable DUTs that are not responding)
4. Check ASIC ID information (DieID, LotID, WaferID etc.)
5. Load DMP
6. Check Power rail currents
7. Calibrate DUTs / Load Trim values from OTP
8. Get Gain Only Images

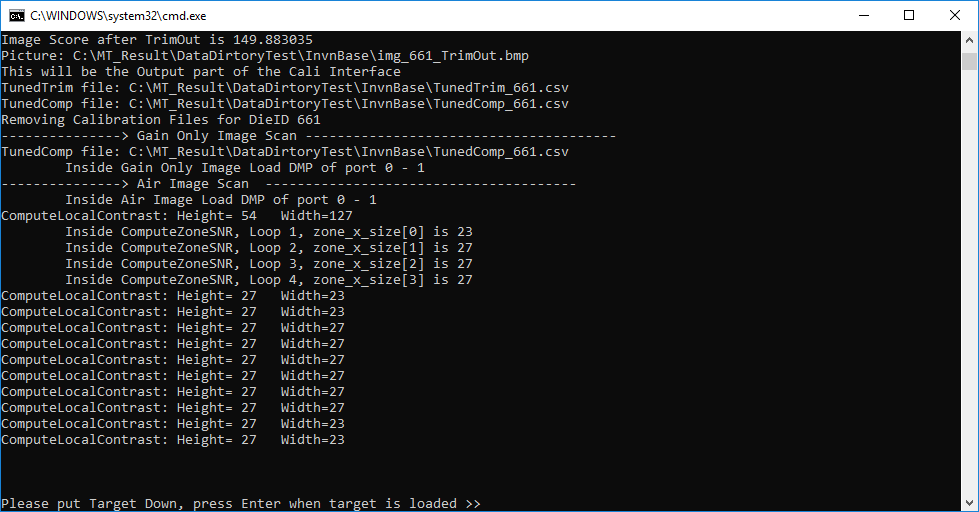


1. Get Air Images
2. Check CMOS temperature



1. Prompt User to Load Target (for Test Station #4 Only, no need for Test Station #5)





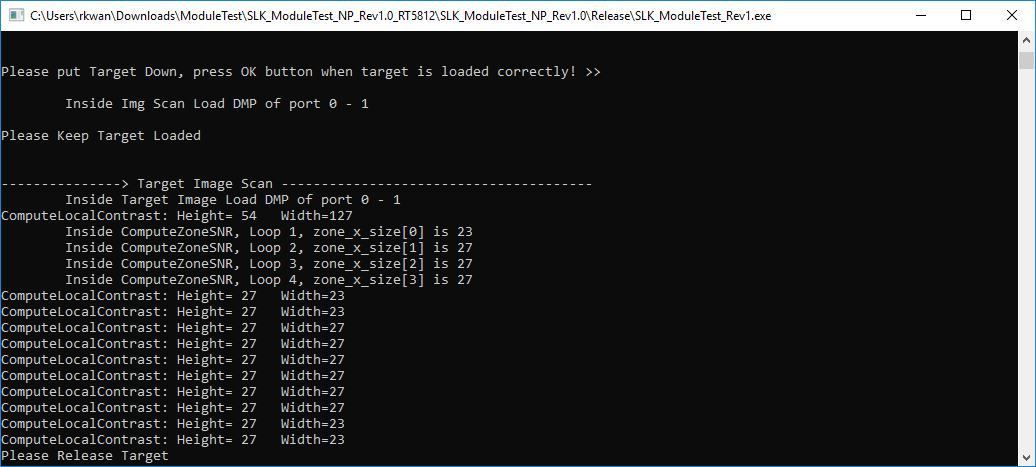
Then it will pop up the fingerprint image BMP file captured in the log folder

and test continues running in the background. No need to press “Continue” button anymore.

**Note:**

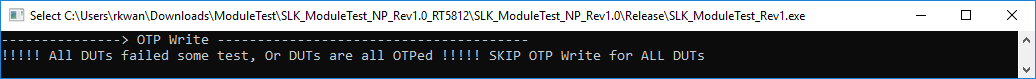
For Test Station 2 Only, Air Image GUI window would pop up for viewing and it would be closed automatically at the end of the test.

1. Get Target Images and calculate image related values (for Test Station #4)

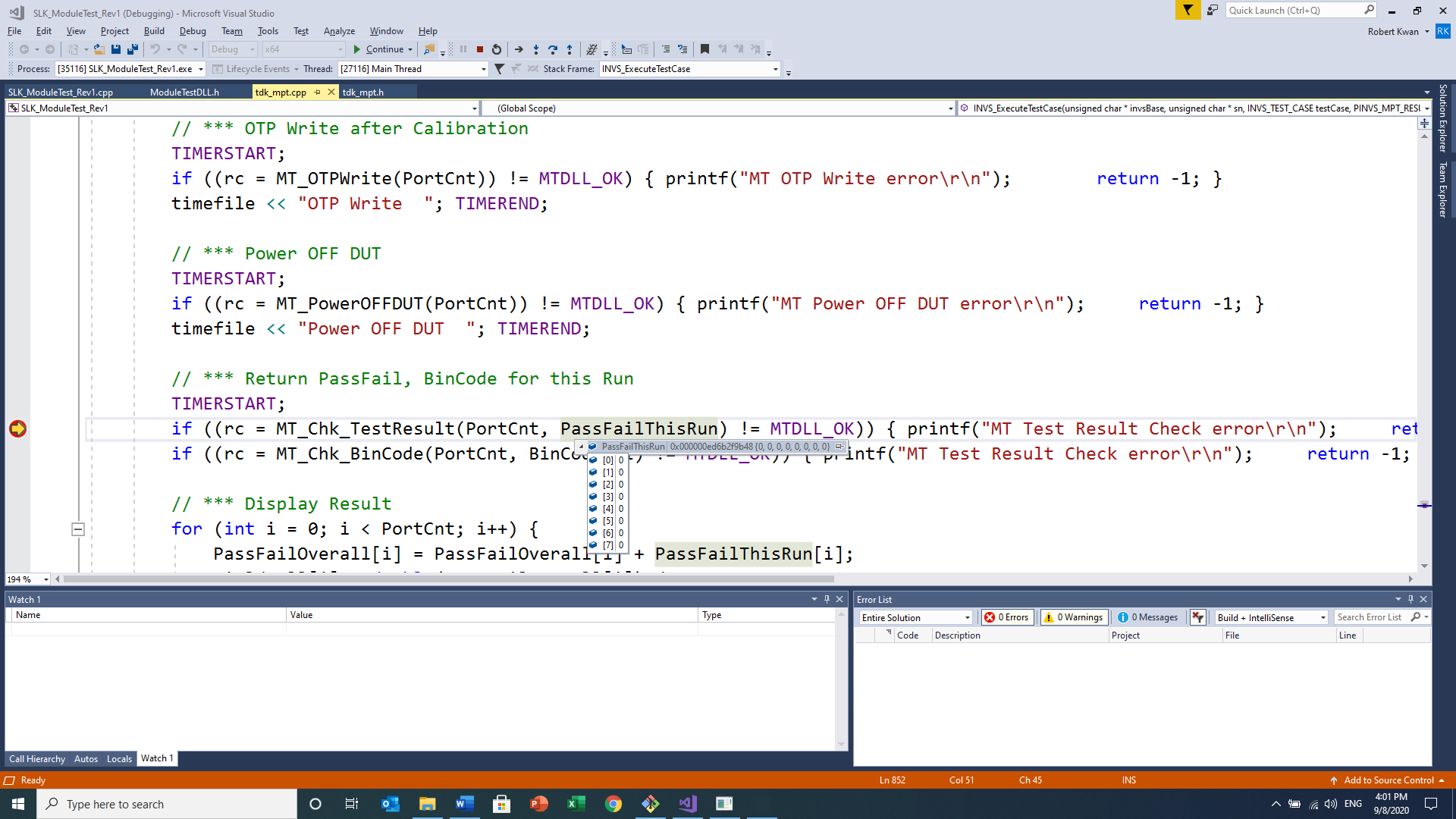


1. Write OTP (Only when OTP\_Write is enabled and All Test Passed for Test Station #4 only)

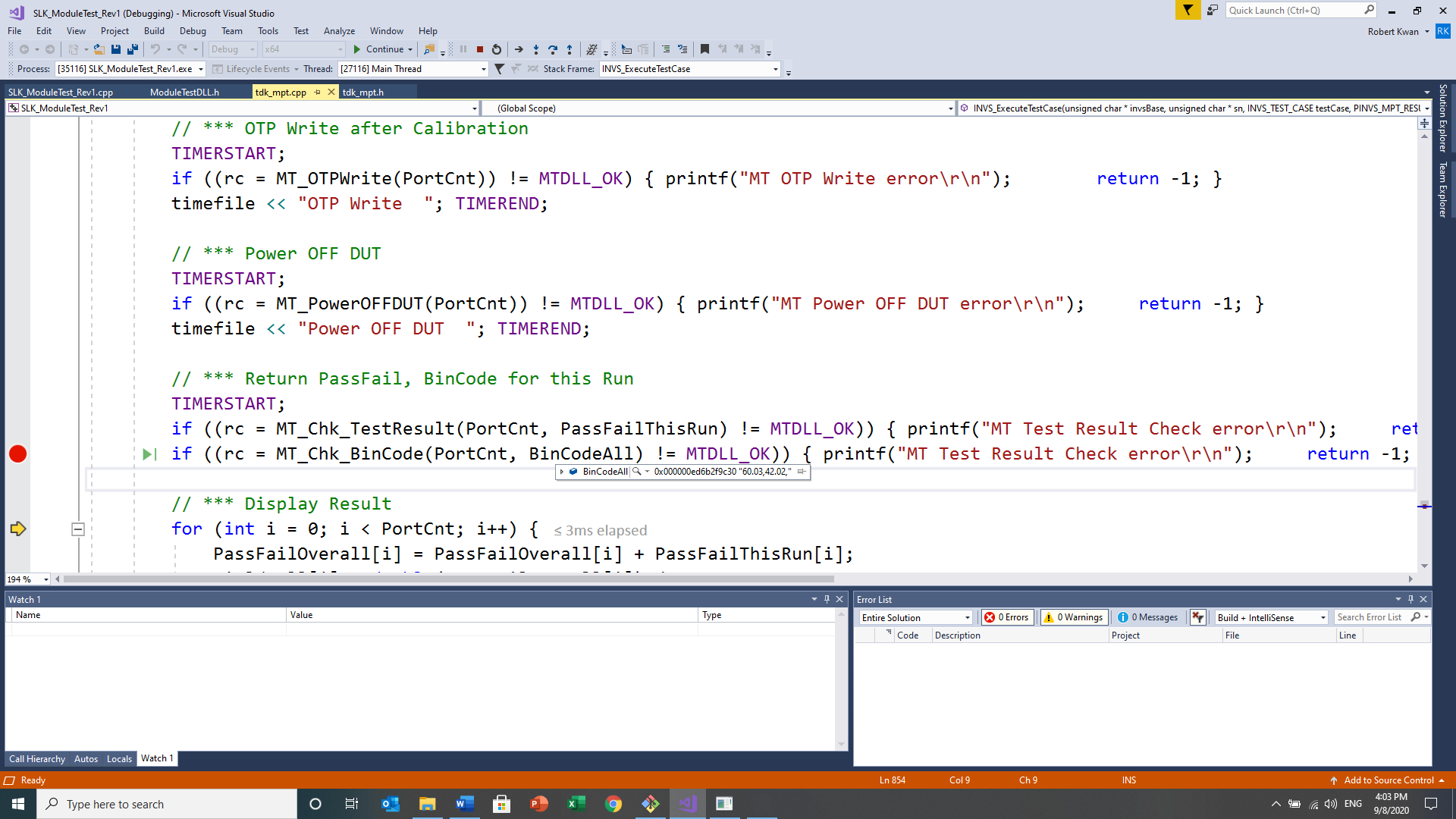
If the module is already OTP-ed, it will show like:-



1. Power off DUT
2. Return PassFail information for this run so user has access to it



This PassFailThisRun variable array will show the PASS[1] or FAIL[0].

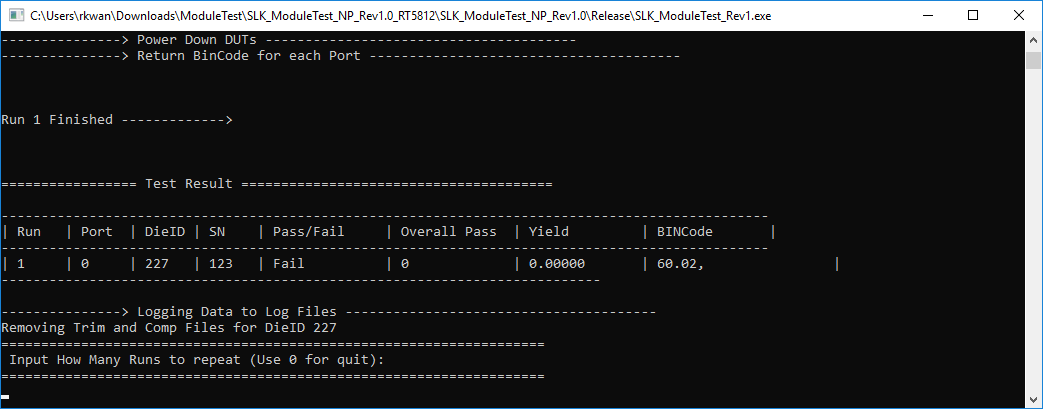


And the BinCodeAll variable would store all the BIN Code Error occurred in this run.

1. Display result in command line
2. Write this run’s information to log file

If Pass OK, it would show:-



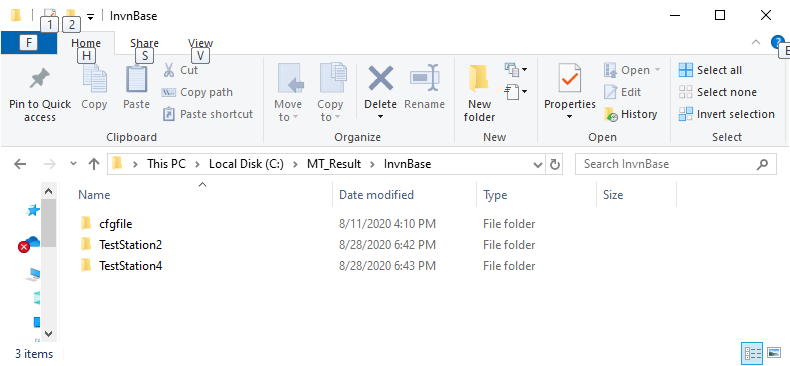
If Fail somwhere, it would show the BIN Code error Information:-

## **Clean up (after all main loop runs are complete)**

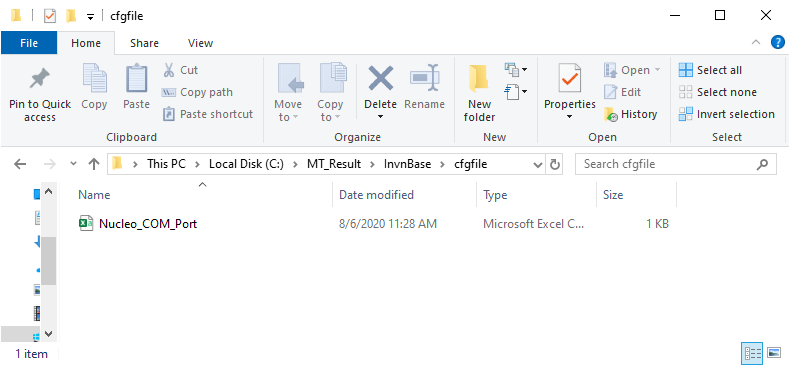
1. Finalized / Close DLL

## **Logging Outputs**

Log data will be saved under C:\MT\_Result\InvnBase folder like below.

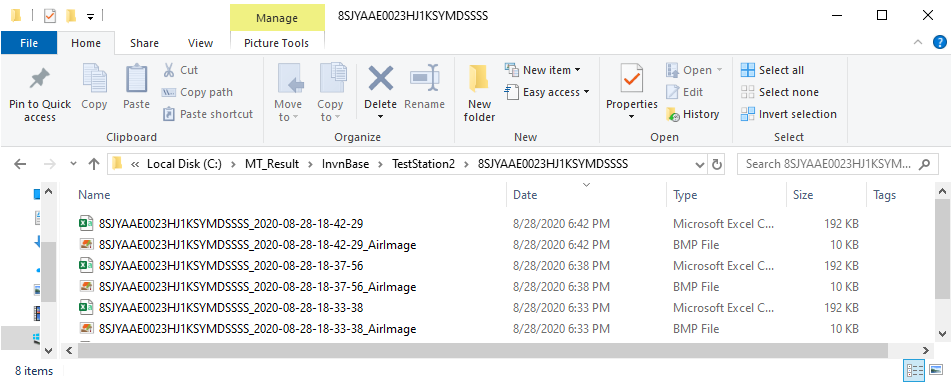


Where \cfgfile folder has the Nucleo\_COM\_Port.csv.



TestStation2 folder would have:





Where ModuleTestDate-YYYYMMDD\_Port0.csv would have all the sample logs in one single big file.

This ModuleTestDate-YYYYMMDD\_Port0.csv relies on the use of the MT\_WriteLOG() API to capture the logging into the file.

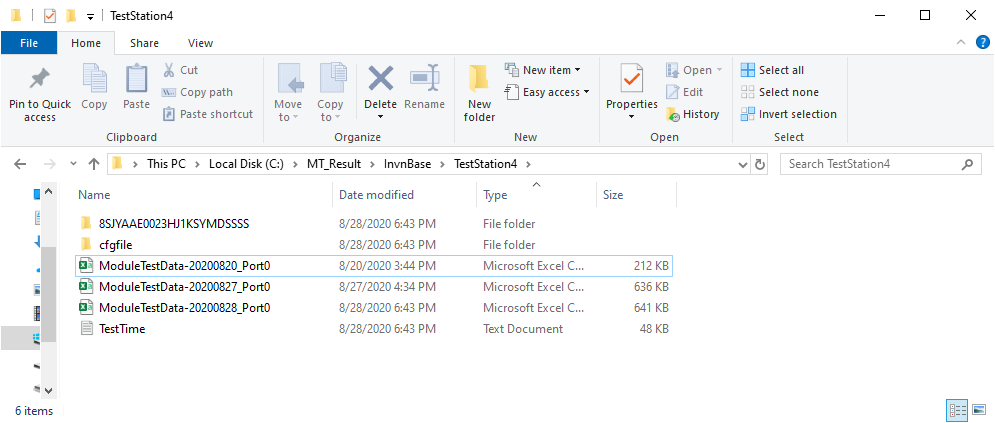
Each individual sample with the SN would also have individual log file based on the format:

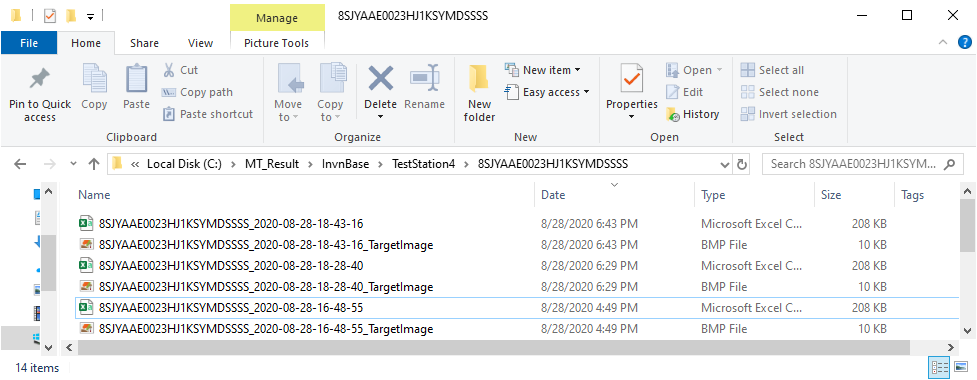
23byte-SN\_YYYY\_MM\_DD\_HH\_MM\_SS

To show the runtime of the module test and the corresponding image as well.

For the individual log file, it would rely on the use of the MT\_WriteSNLOG() API to capture the logging into the file.

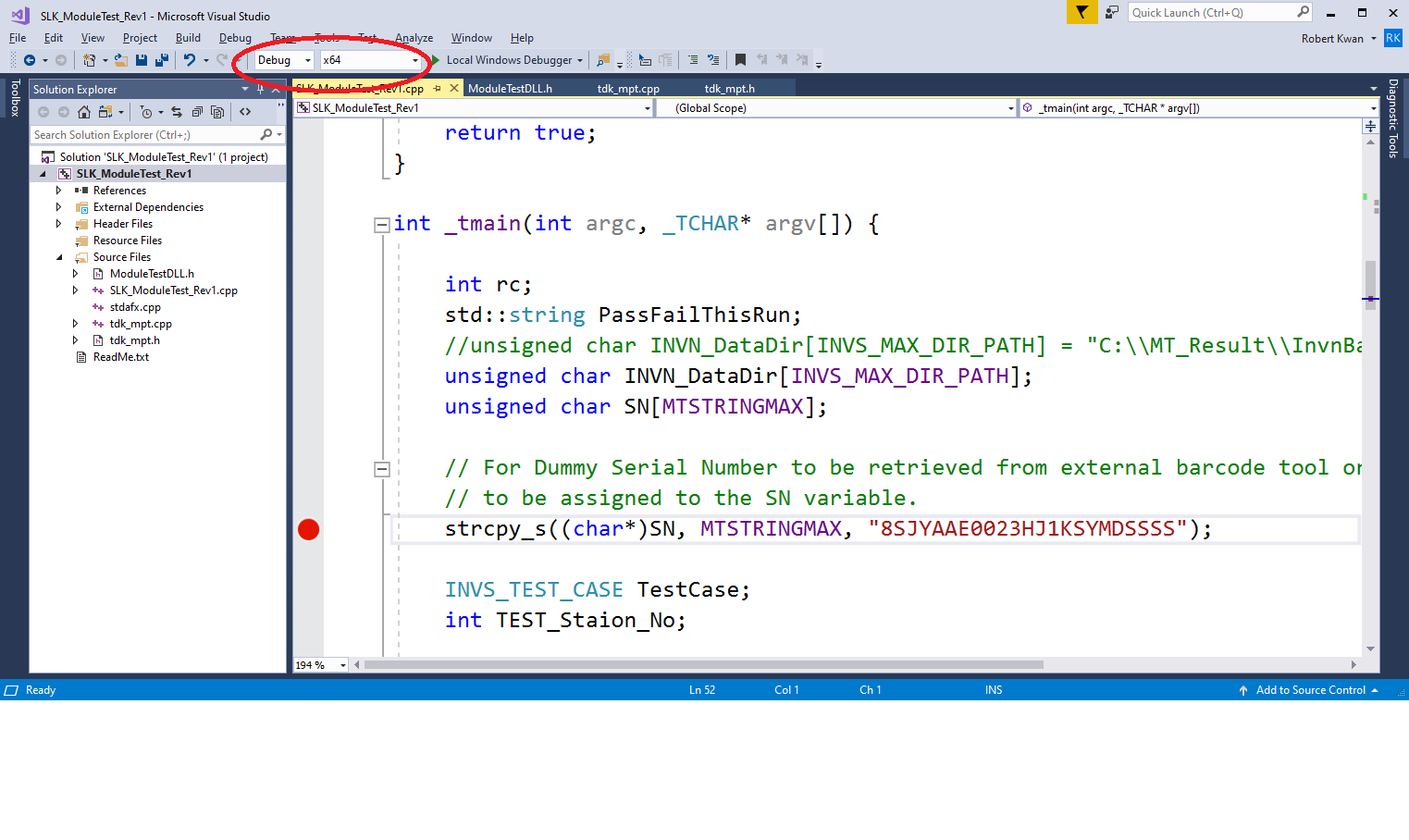
Similarly, TestStation 4 folder would have:-



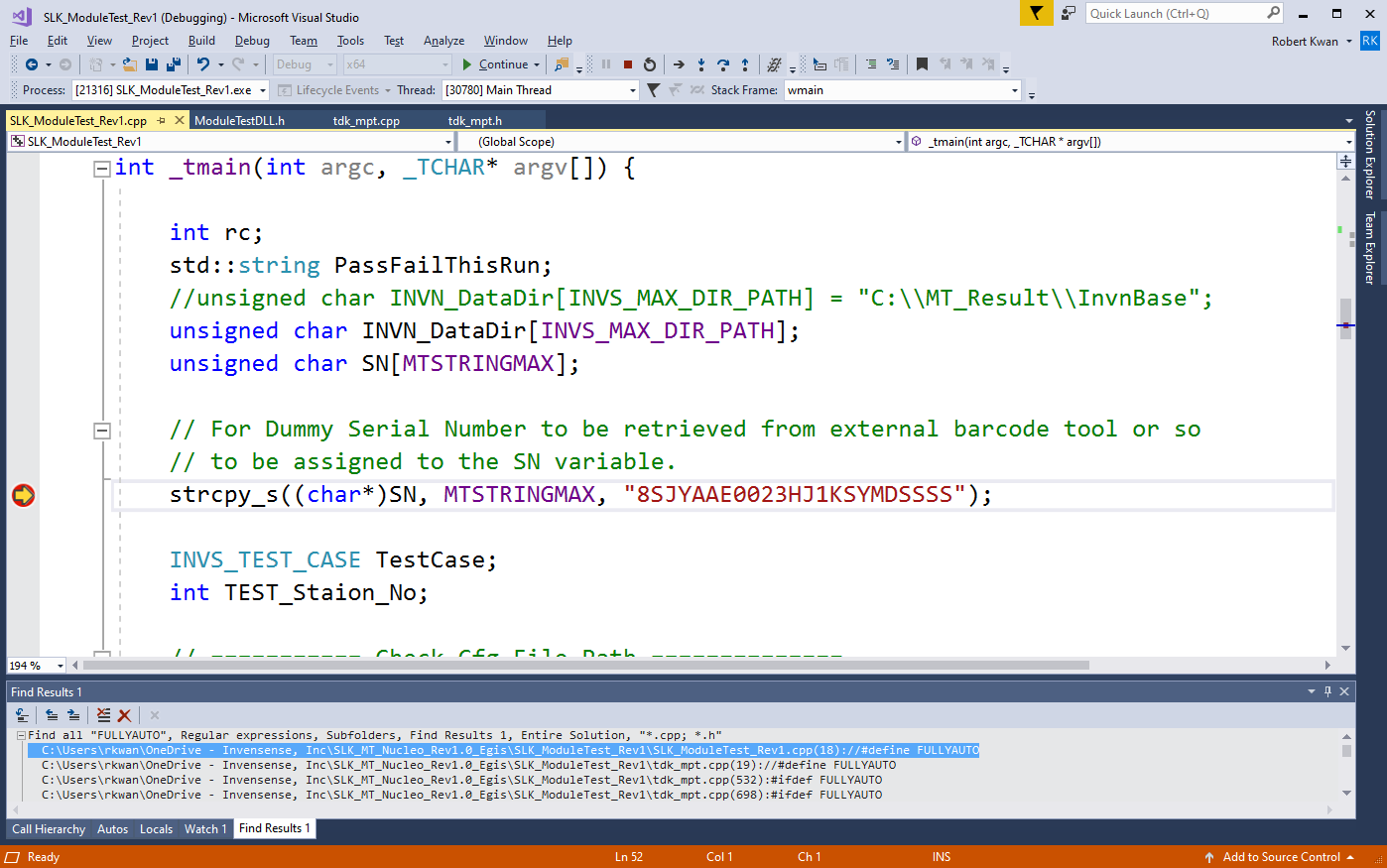


# **Appendix I: Debugging/Tracing the reference C++ Codes**

The SLK\_ModuleTest\_Rev1.sln project can be compiled into \Debug configuration for x64 platform as shown below.



Once set a breakpoint at the beginning of the main() and select “Start Debugging” in the \Debug tab menu, one can step through the codes and do debugging easily with the inline comments in the codes too.



**Noted:**

All the needed .dll, .lib and related files are already placed under the respective \Debug and \Release folders for the SLK\_ModuleTest\_Rev1.exe to be executable. Please do not delete them away without consideration.

# **Appendix II: Serial Number Setting**

At the beginning of the reference code’s main(),

// For Dummy Serial Number to be retrieved from external barcode tool or so

// to be assigned to the SN variable.

strcpy\_s((char\*)SN, MTSTRINGMAX, "8SJYAAE0023HJ1KSYMDSSSS");

a dummy 23-byte SN is assigned based on the feedback from another customer as we do not work on any barcode software to retrieve it.

However, this piece of SN information would allow us to format the filename of each sample easily then.

The way that one customer did is to scan barcode and then reset this dummy variable SN into the value they want to generate the logs and they used paper labels to mark down the sample without any need to OTP it. Current OTP has only 8 bytes available for the SN and so it also would not meet the demand of this customer.

# **Appendix III: USE\_CONSOLE\_MODE vs USE\_IMAGE\_GUI**

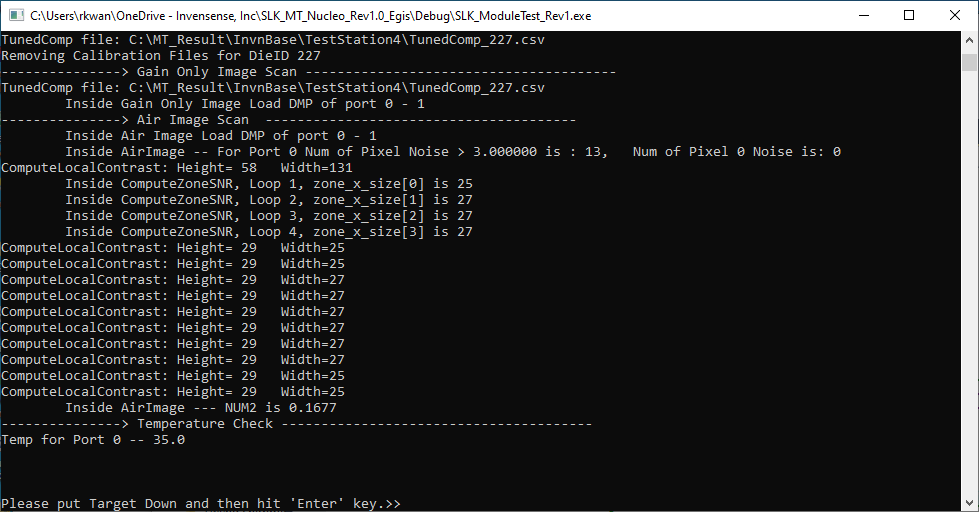
Before the use of

#define USE\_IMAGE\_GUI 1,

the reference code used

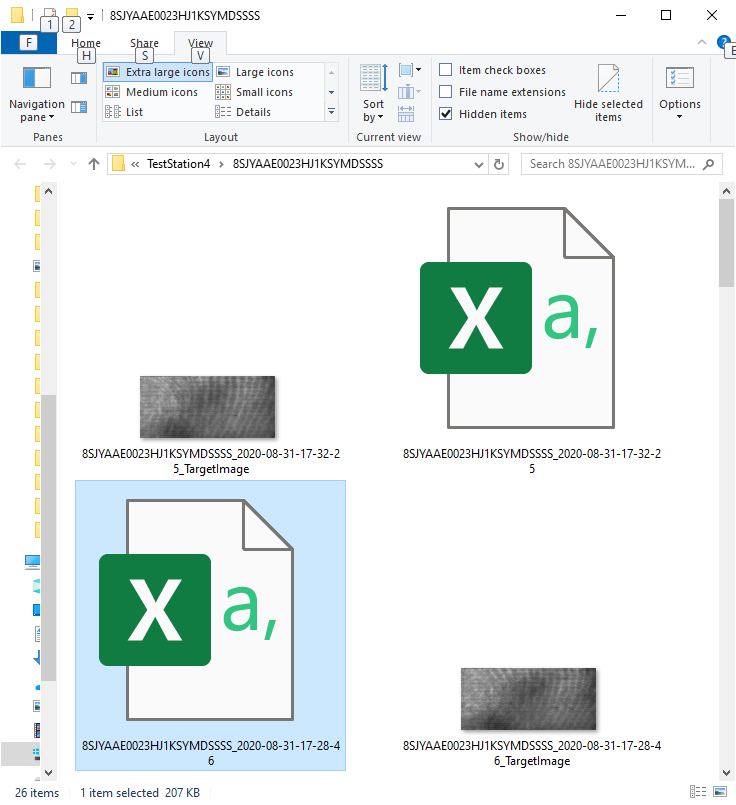
#define USE\_CONSOLE\_MODE 1

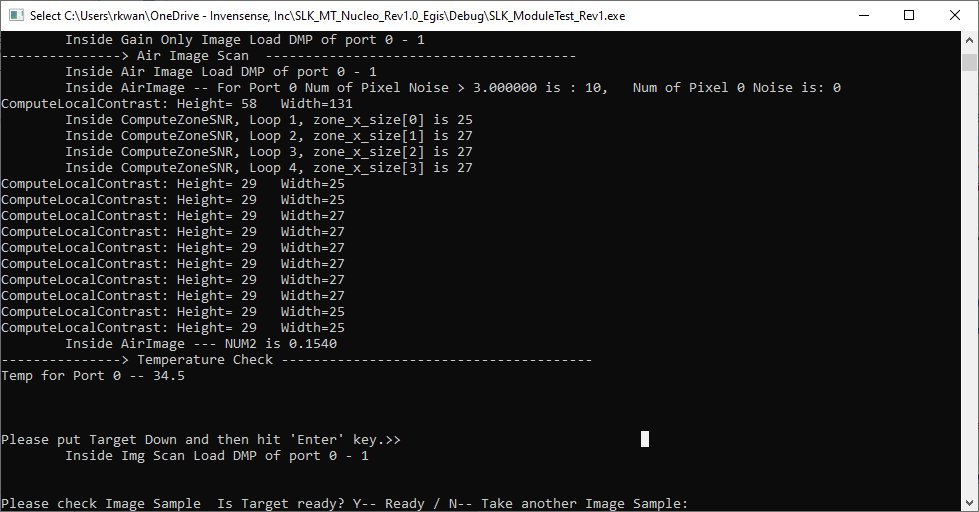
To view the captured image file from the corresponding log folder in Test Station 2 or 4 as shown below.



Then the corresponding image file folder will pop up in another Window Explorer

And set it into “Extra Large Icon” view mode to view the BMP file.





If the image in the BMP file is OK, press “Y” to continue and keep loading the target.

If the image in the BMP file is not OK, press “N” to retake another image again.

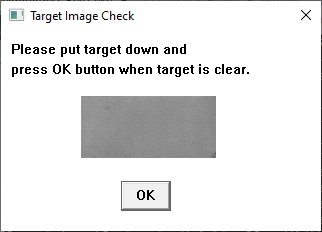
If the other GUI option is used,

#define USE\_IMAGE\_GUI\_II 1,

The GUI window display would be controlled by the underlying MT DLL by setting the parameter to true in the MT\_ImageScan() API:

rc = MT\_ImageScan(PortCnt, (char \*)SNLogFile, "TargetImage", **true**, 1);

And then the GUI Window will pop up like this:-



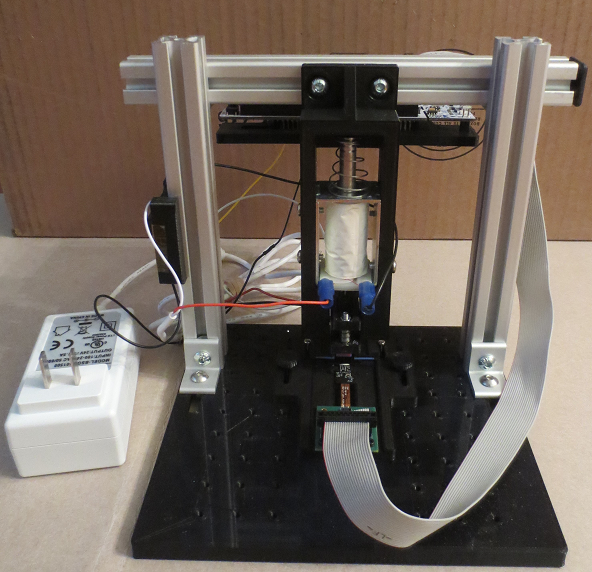
# **Appendix IV: Automation Flag: FULLYAUTO**

In order to facilitate the automation of fake finger press in Test Station 4, GPIO pin toggling can be used to trigger the hardware to load or release the fake finger target.

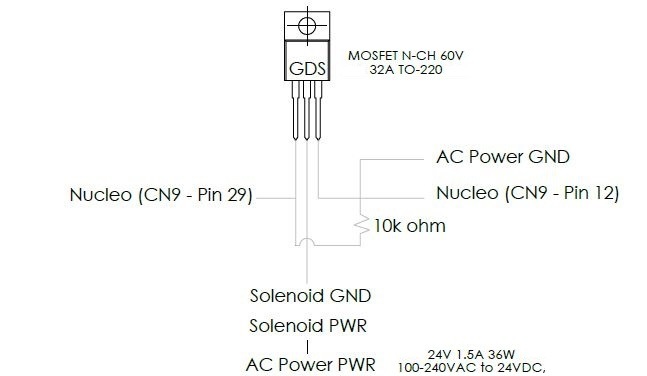
In the reference code,

//#define FULLYAUTO 1

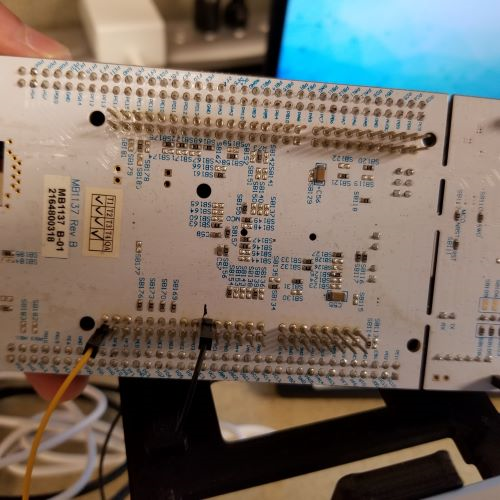
Is used for the reference codes in the case of using the Nucleo board to control such an automation fixture as shown below.



And the solenoid used in this automation fixture is:



Hence, the wire connection to the Nucleo board can be done from the bottom of the board.



Two points to pay attention to when trying to set up automation fixture:

1. Whether the output of the GPIO pin would be sufficient to drive the power of the automation fixture.

For example, Nucleo board can give the 3.3V output from the GPIO pin and so it is good enough to drive the solenoid as shown above.

If the GPIO pin can give only 3.0V output, then it will not work.

1. Depending on the MCU board used, it may not have any extra GPIO pin nor high enough GPIO output voltage to drive the automation fixture.

In this case, a USB GPIO module may be used instead.

One example is the **Numato Lab 8 Channel USB GPIO Module** would give **5V** GPIO output.

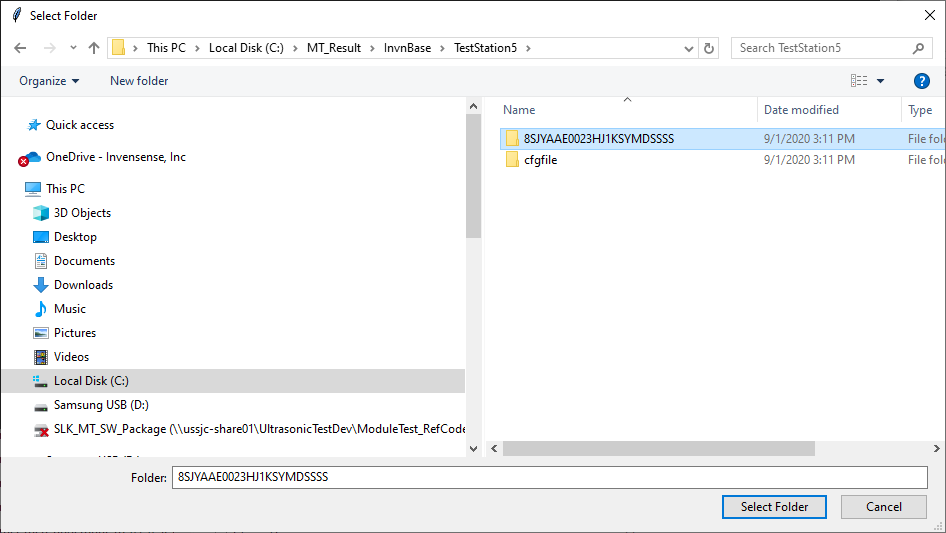
In addition, GUI is not triggered when the FULLYAUTO is enabled in current reference code. If needed, one can modify to show the GUI display as well.

# **Appendix V: Log Post-Processing Script**

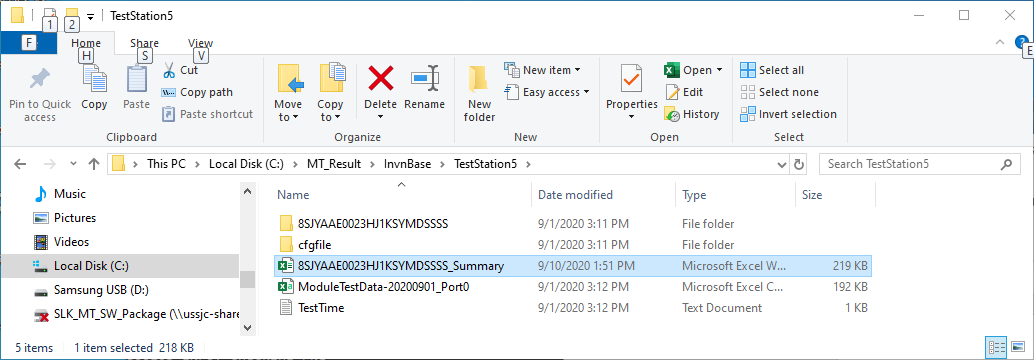
Below is the reference Python script based on Python 3.7 to post-process the logs to give the relevant information for log analysis when there is any issue with the sample.



Once it is launched, it would ask the folder with the logs to be processed.



And when the post-processing is done, the corresponding foldername\_Summary.csv would be generated.



And the foldername\_Summary.csv file would have the Images worksheet and Summary worksheet as shown below.

