

# **Kazam User's Manual**

## **REVISION HISTORY**

<b>REV</b>	<b>CHANGE DESCRIPTION</b>	<b>APPROVED BY</b>	<b>DATE</b>
01	Initial Version	Fei Huang	12/04/17
02	Added sub testplan description Added other language description	Fei Huang	12/16/17
03	Modify folder structure description Added next step control description	Fei Huang	03/16/18
04	Added an external program interface description	Fei Huang	05/03/18
05	Update the Labview Run-time Version	Fei Huang	05/21/18
06	Added case sensitive descriptions	Fei Huang	05/30/18
07	Added a new reserved global variable descriptions	Fei Huang	06/04/18
08	Added important parameters descriptions Corrected test plan descriptions Added the new error handle feature Modified the user manual file name	Fei Huang	07/12/18
09	Enhance "enable" feature	Fei Huang	08/07/18
10	Added "switch...case" for next step control Added some parameters description for GUI Added new methods how to use global variables Added some new parameters in bench configuration file Added auto-update kazam description	Fei Huang	11/06/18
11	Added default error handler feature Modified the log vi name Modified global variables description Added new commands for external API	Fei Huang	04/04/19
12	Add new functionality for "if fail goto", Kazam can handle more jump conditions Add new functionality for "goto", Kazam can handle more jump conditions	Fei Huang	06/04/19

	<p>Add a plug-in to kazam, it is used to synchronize time with the log server</p> <p>Added a new feature for GUI, the SFC menu can be disabled by setting parameters in the shell configuration file</p> <p>Added a new feature, Kazam can load the xlsx file as TestPlan</p> <p>Subtest plan delimiter changed from ":" to "."</p> <p>Updated Kazam development steps</p>		
13	<p>Added parameters for "Enable", the developer can customize "Enable" parameters</p> <p>Add Transmit and Receive property</p> <p>Added a new "Non-blocking" setting in testplan</p> <p>Added the new features for debugging mode</p> <p>Added ResetPlan for debugging mode</p> <p>Added new GUI for Actuator mode, it is named the Actuator GUI</p> <p>Added new reserved global variable</p> <p>Added a new feature to disable start button for all GUI</p>	Fei Huang	08/19/19
14	<p>Added a new feature to call python script directly in "Test Vi Name" column</p> <p>Added a new feature to call bat file directly in "Test Vi Name" column</p> <p>Modified the exe call feature</p> <p>Replaced "Non-blocking" with "RunInParallel"</p> <p>Add "Pause Slot" into "Next Step CTL"</p> <p>Add "Continue Slot" into "Next Step CTL"</p> <p>Add "Stop Slot" into "Next Step CTL"</p> <p>Add "Slots_info" global variable new definition format</p>	Fei Huang	10/25/19
15	Added sequence producer description	Fei Huang	12/05/19
16	Added new multiple single GUI	Fei Huang	01/14/20

	Modified the call description for exe, python, bat file		
17	<p>Added new feature to call c# source code directly in the testplan</p> <p>A new global variable is defined, which is named HD_info, and it's called a hardware variable.</p>	Fei Huang	09/29/20
18	<p>Added a new format of BenchCFG file for centralized management Kazam</p> <p>Added amzFileSync.vi to sync files</p> <p>Added amz_RetentionPolicy.vi to be used for data retention policy on the local computer</p> <p>Added the "OR" feature for result judgement</p> <p>Added a new feature, the sub testplan can be loaded in the 'Function Call' field</p> <p>Rename testplan header</p> <p>Added single log compression feature for amz_log_single.vi</p> <p>Added step by step logging feature for amz_log_single.vi</p> <p>Modified the description of "Route_check" global variable</p> <p>Modified and updated some descriptions of test plan keywords</p>	Fei Huang	07/12/21
19	<p>Added a new tool for CPK analysis</p> <p>Added single log browser</p> <p>Added the "NOT" feature to Kazam core</p> <p>Add a new reserved keyword "Result" to the input parameter</p> <p>A new global variable is defined, which is named Bench_CFG, and it's called a bench config variable</p> <p>Added fixed header sequence to check the fixed column of daily log</p> <p>Added search feature to shortcut menu on Kazam GUI</p> <p>Add some new reserved variables</p>	Fei Huang	11/03/21

REVISION HISTORY .....	2
1. Purpose .....	7
2. Definitions & Acronyms .....	7
3. Platform Architecture .....	8
4. Features .....	8
4.1 Add/replace/delete function .....	8
4.2 Running different test plan .....	9
4.3 Running in parallel .....	9
4.4 Automatically updates Kazam and Project folders .....	11
5. GUI Introduce .....	12
6. How to develop new projects.....	15
6.1 Kazam folder .....	15
6.2 Project folder .....	15
6.2.1 Driver VI .....	16
6.2.2 Test VI.....	18
6.2.3 Log VI.....	19
6.2.4 System configuration .....	22
6.2.5 TestPlan and Hardware configuration .....	25
6.2.6 Reset Plan.....	35
6.2.7 Global variable .....	36
6.3 Development steps .....	42
6.4 Labview Run-time Version .....	45
6.5 Other language or EXE/Python/Bat file support.....	46
6.5.1 EXE/Python/Bat file support .....	46
6.5.2 C/C++ DLL support .....	46
6.5.3 C# source code support .....	47
7. Data log.....	49
7.1 Single Log .....	49
7.2 Time Log.....	49
7.3 Daily Log .....	50
7.3.1 Fixed Header .....	50
8. External API.....	52

8.1 Command summary.....	52
7.1.1 Commands detail description .....	53
9. Important parameters.....	56
9.1 Bench_Default_Config.csv .....	56
9.2 Kazam configuration file .....	59
8.2.1 Common parameter.....	59
8.2.2 Other parameter .....	59
10. Sequence Producer.....	67
9.1 Debugging Tool configuration file.....	67
9.2 Debugging Tool GUI .....	68
9.3 How to configure sequence producer .....	69
11. Kazam Tools.....	71
11.1 Single log browser .....	71
11.2 CPK Analysis Tool .....	75

## **1. Purpose**

This document will introduce how to develop a new test or automation project using the software platform, and let the developers focus on the test or automation itself.

## **2. Definitions & Acronyms**

AME: Advanced Manufacturing Engineering

TE: Test Engineering

GUI: Graphical User Interface

DUT: Device Under Test

SFC: Shop Floor Control System

TestPlan: It is a csv file, using for manage all the test step

CM: Contract Manufacturer

Kazam: The name of the Amazon's test framework

XLSX: It is a file extension name, it created by Microsoft Excel version 2007 and later.

Slot: If a fixture can test several DUTs, each of its DUT test locations is defined as a slot. It is also a thread concept

Actuator: It can be a robot, a cylinder, a rotating motor, a laser, a camera, a 2D/3D scanner and so on, and it could be a combination of them. It is also a thread concept, so we could say that a slot is equal to an actuator

### 3. Platform Architecture

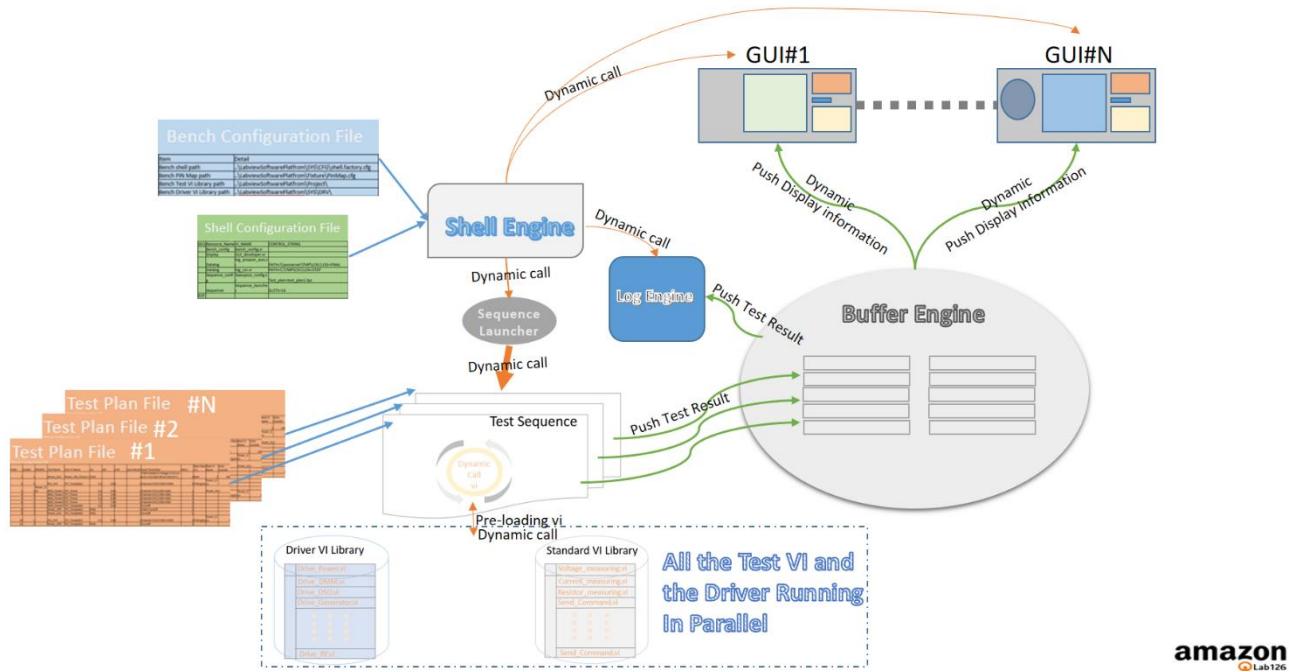


Figure 1

The Figure 1 is the software platform architecture, all the test vi and driver vi will be call by dynamic, we can use the TestPlan to set the call mode.

### 4. Features

#### 4.1 Add/replace/delete function

The platform can add/replace/delete function through configure the shell configuration file.

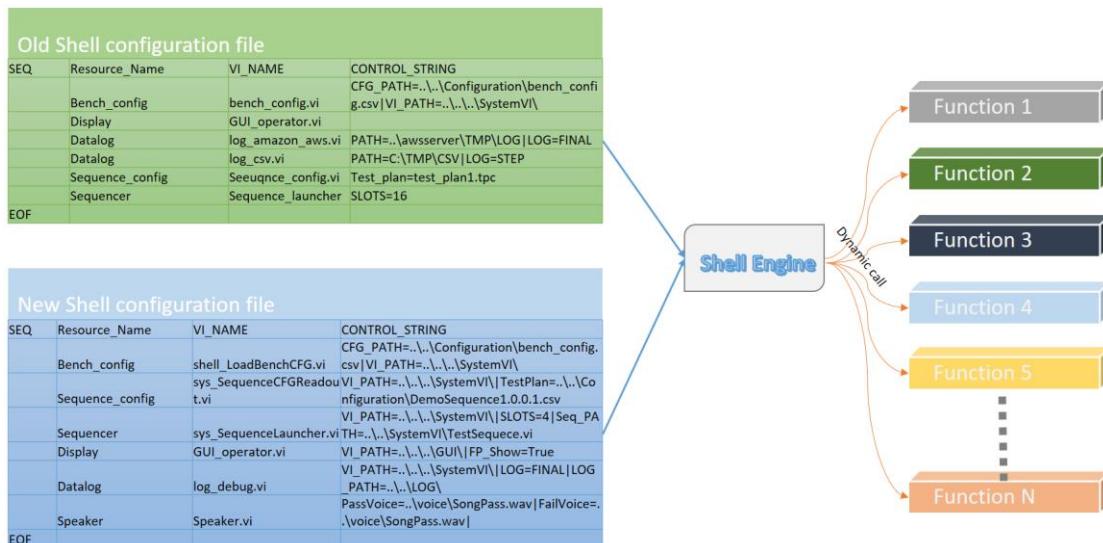


Figure 2

## 4.2 Running different test plan

The platform can run the different TestPlan at different slot, the Figure 3 is a simple example.

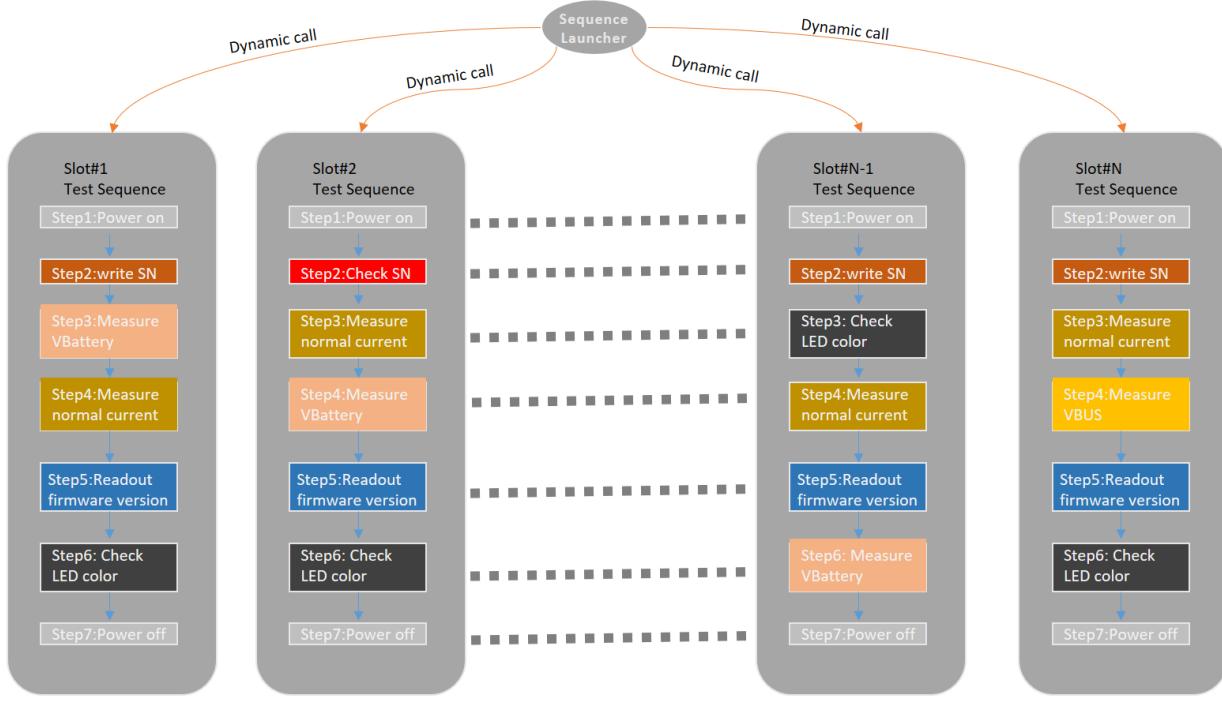


Figure 3

## 4.3 Running in parallel

The platform can run all the test steps in parallel as Figure 4, it is a step by step execution, from step1 to step7 at each slot. Each slot is independent of operation.

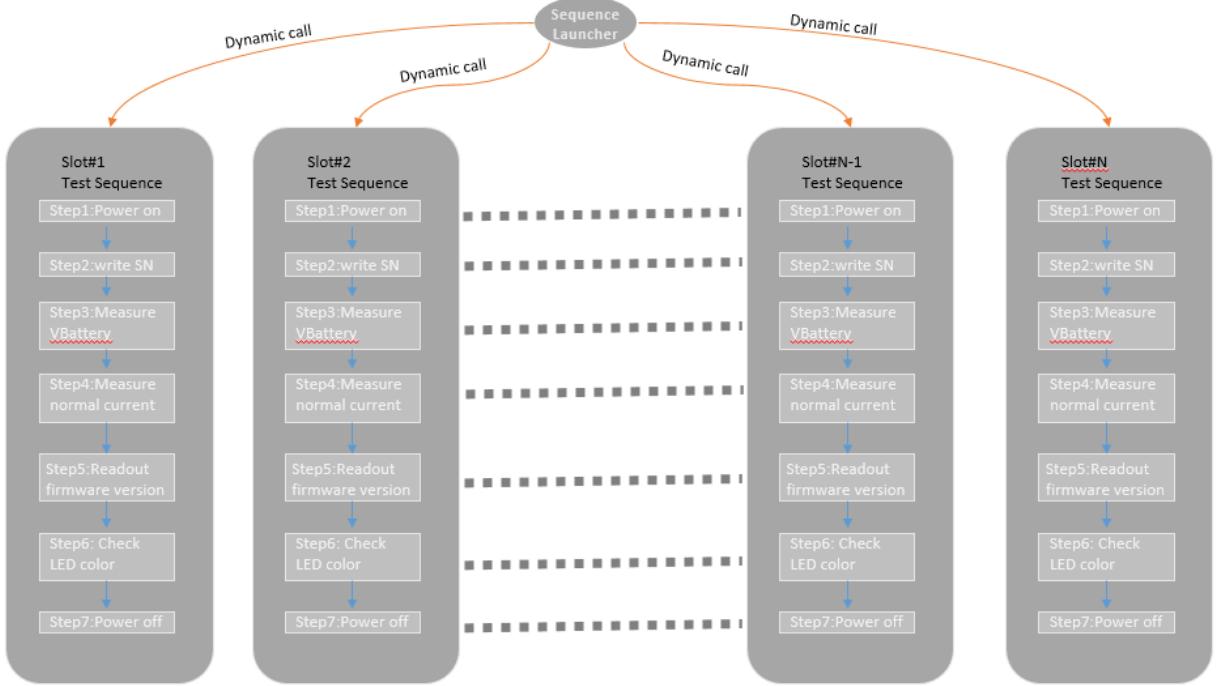


Figure 4

The platform can run parallel test steps at each slot. As Figure 4, it has seven steps, we can let the step 2 to step6 running at the same time, it as Figure 5.

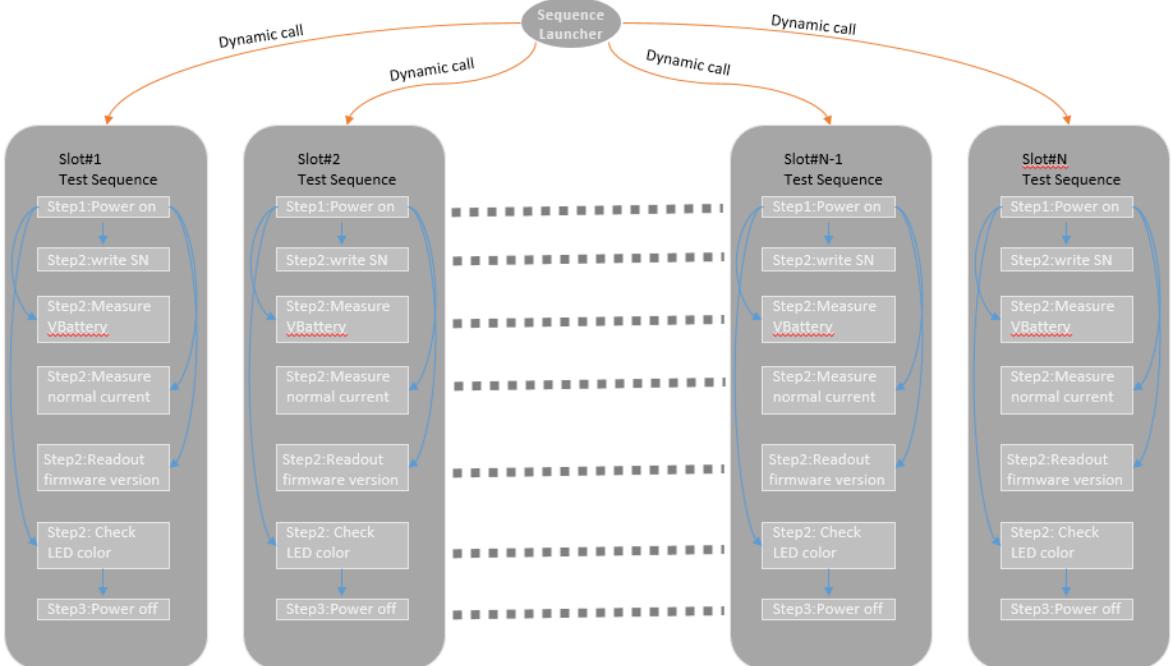


Figure 5

#### 4.4 Automatically updates Kazam and Project folders

The platform has automatic update feature, the developer can upload the latest Kazam to AWS cloud, and then the factory's server will download it automatically, when running the Kazam.exe, Kazam will check the factory's update server automatically, if there is an update, it automatically updates the latest code.

Refer to section [8.1](#) for detailed parameter setting.

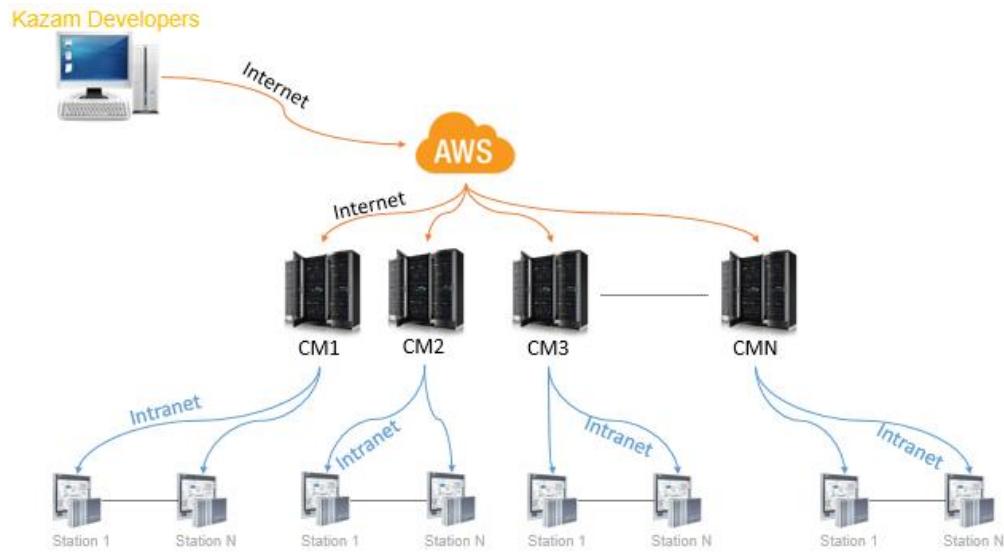


Figure 6

## 5. GUI Introduce

Figure 7 is the panel operator GUI, it used in production for panel test. It can easy to set the number of the panel. Supports up to 32 DUTs.

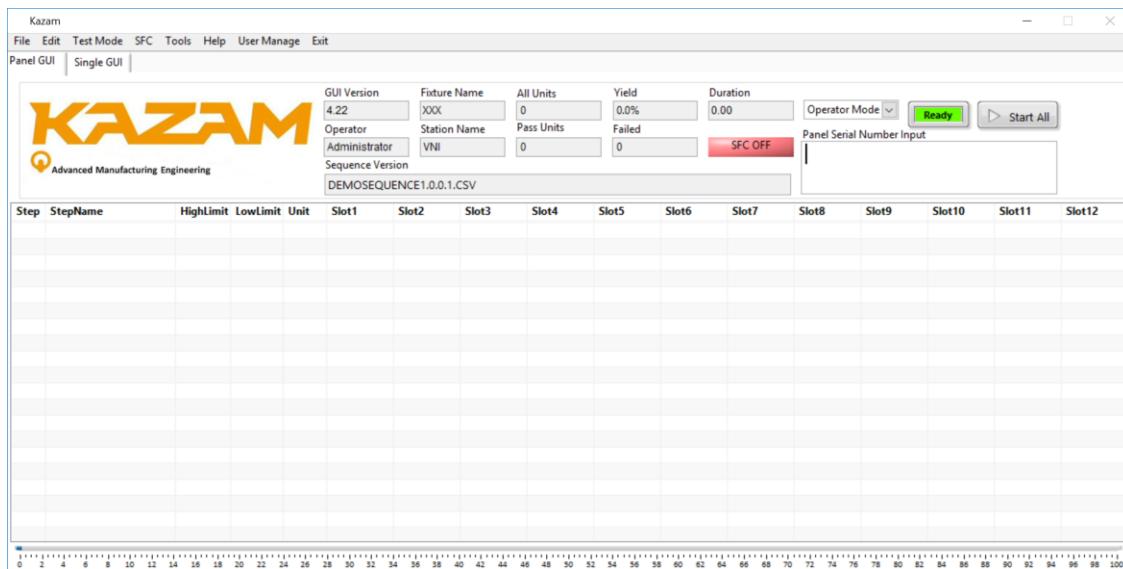


Figure 7

Figure 8 is the panel single GUI, it is used for debugging or maintenance.

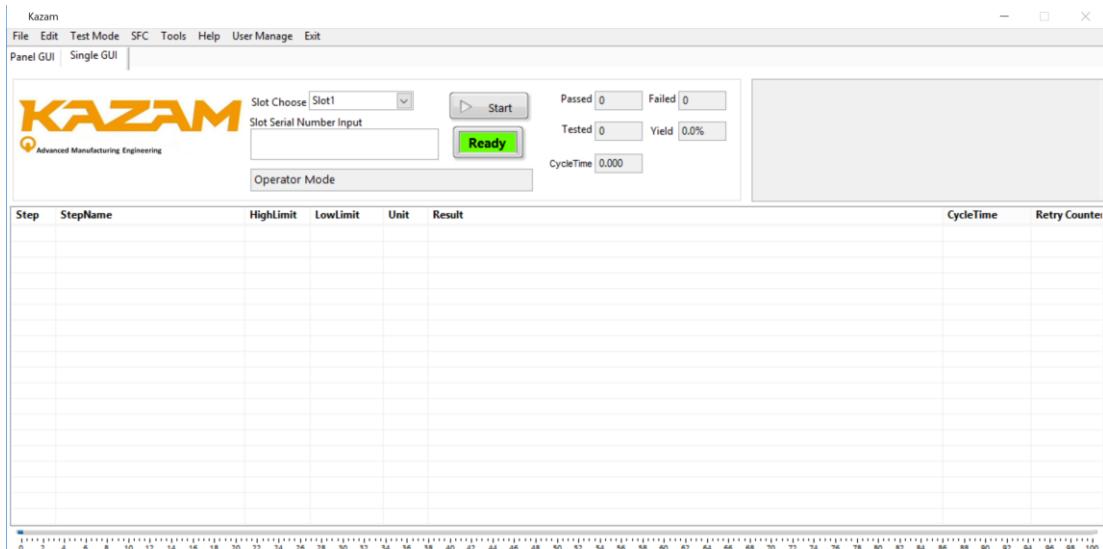


Figure 8

Figure 9 is the multiple single GUI, it is used when each slot needs to run independently.

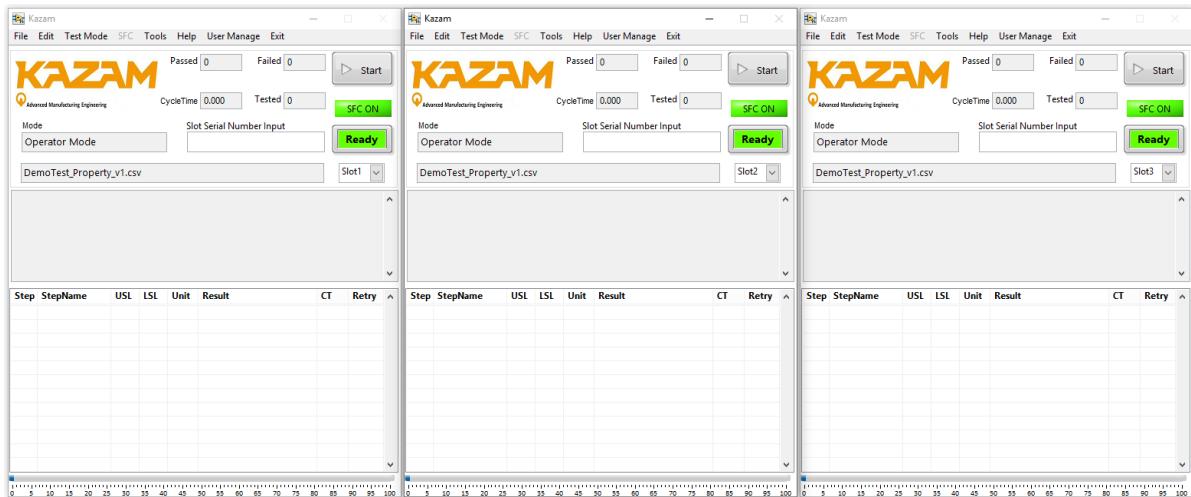


Figure 9

Figure 10 is the actuator GUI, it can be used when multiple actuators run in parallel and all the results of the actuators are displayed at the same time. And the developer can show their front panel of driver vi or test vi on the actuator GUI, the maximum size displayed is **760x600**. More details please reference **6.2.5 TestPlan and Hardware configuration**

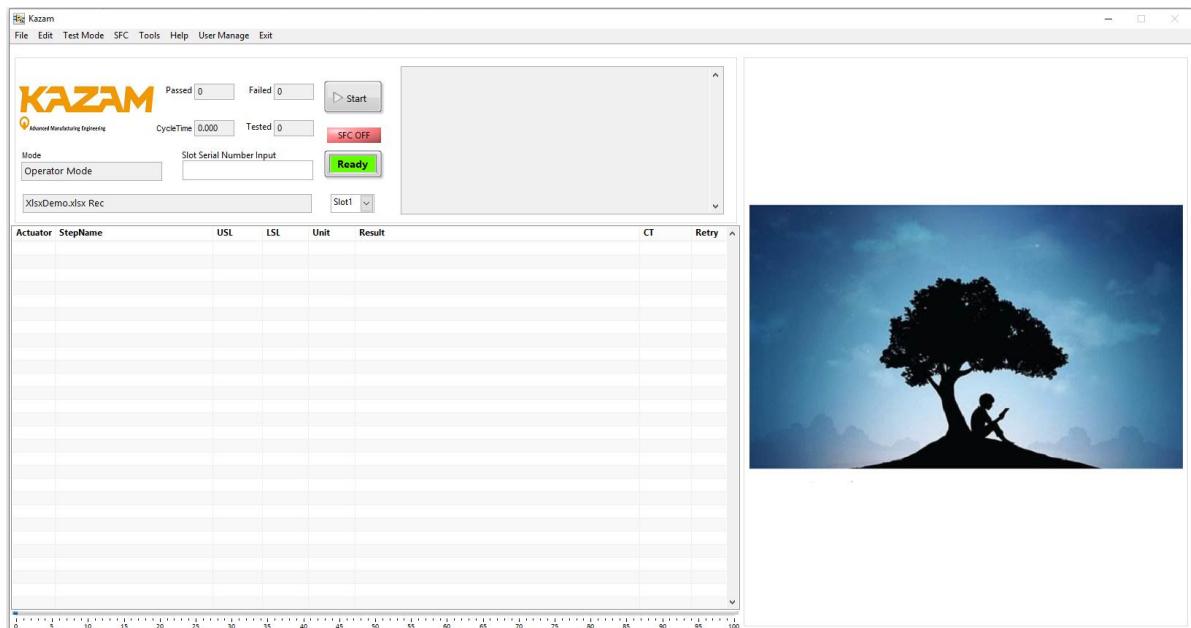


Figure 10

Figure 11 is the multiple single GUI, it is used when more than six slots are running independently.

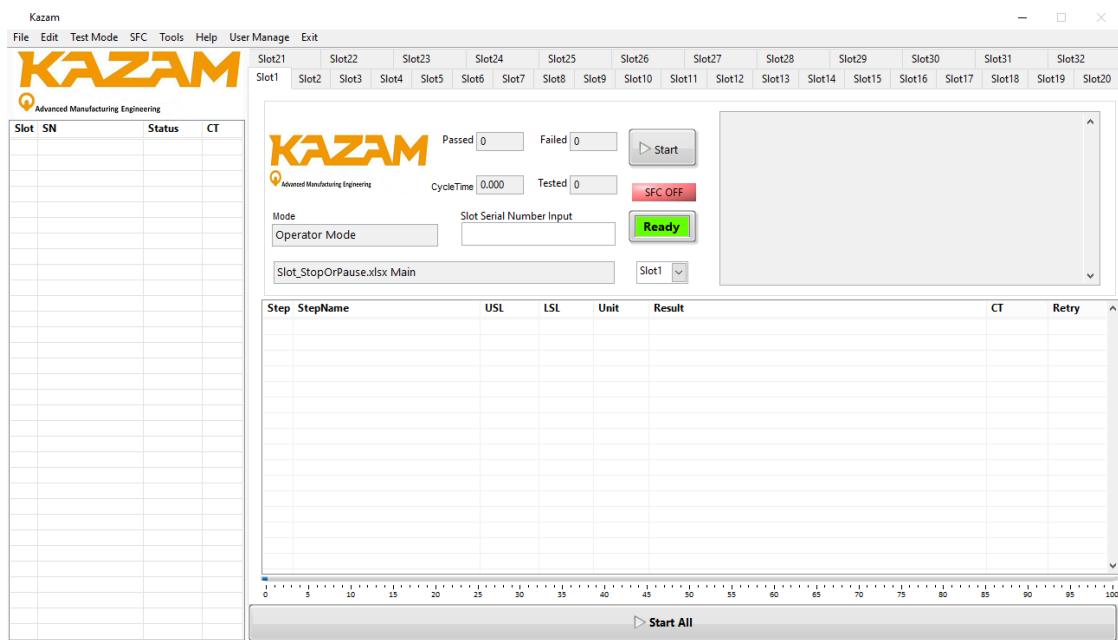


Figure 11

All GUIs support right-click shortcut menus, users can quickly locate the detailed log of the specified steps using the right shortcut menu in the log window. Figure 12 shows how to use the shortcut menu

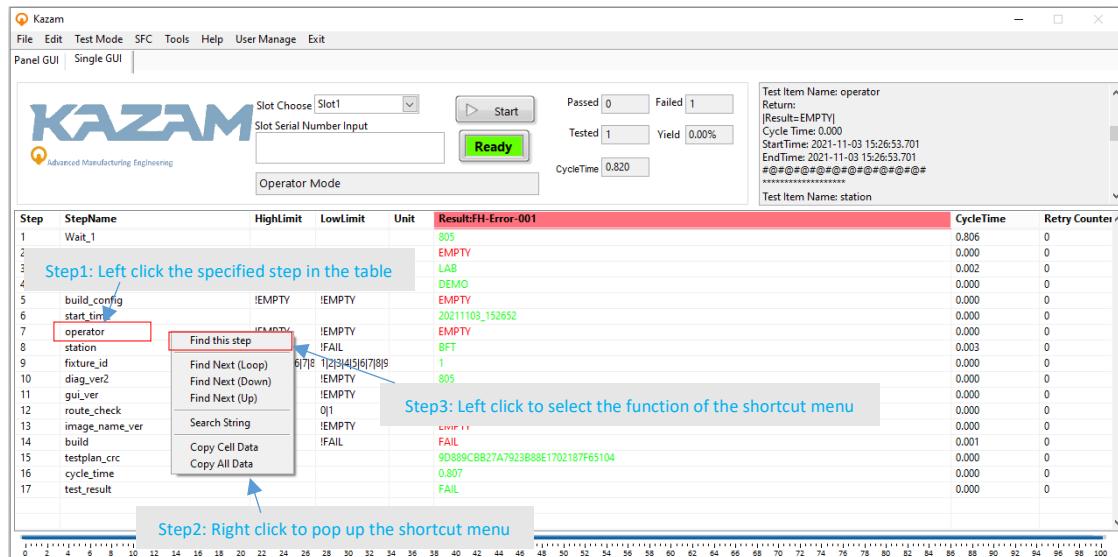


Figure 12

## 6. How to develop new projects

### 6.1 Kazam folder

We will release the Shell folder to supplier or CM, as Figure 13, it will have the following file or folder.

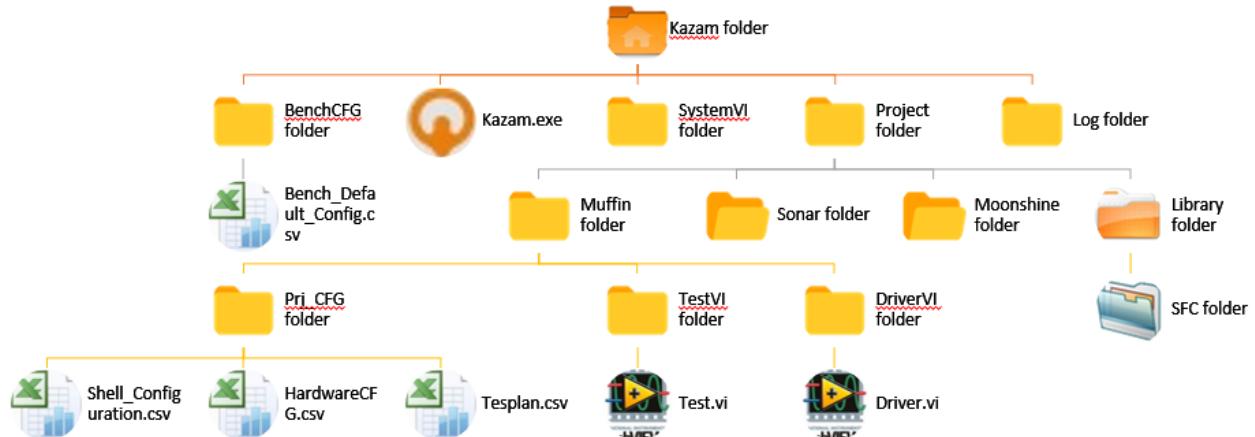


Figure 13

BenchCFG folder: it includes the bench configuration files.

LOG folder: it includes all the log files

SystemVI folder: it includes all the system vi files

Project folder: it includes all the project folder, each project have one folder to manager the vi, TestPlan, shell configuration file and hardware configuration file. And there is library folder in project folder, it is use for common library vi, exe and dll.

SFC folder: it includes all the SFC vi, the developer must to put all the SFC vi into this folder.

Kazam.exe: it is the platform's start-up entrance, double click to start our software

### 6.2 Project folder

We will release the project folder to developer, it as the Figure 14. In this project folder, it will including the Labview project, the developer can follow the project to develop test vi and driver vi.

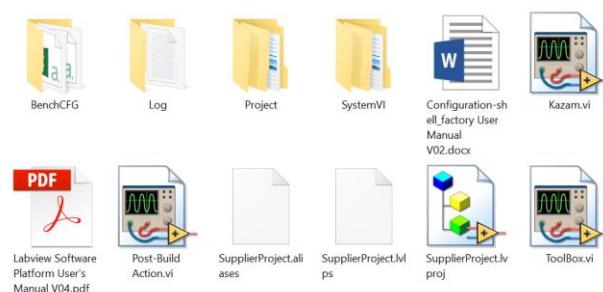


Figure 14

Figure 15 is the LabVIEW project, after the developers complete development, must compile the files.

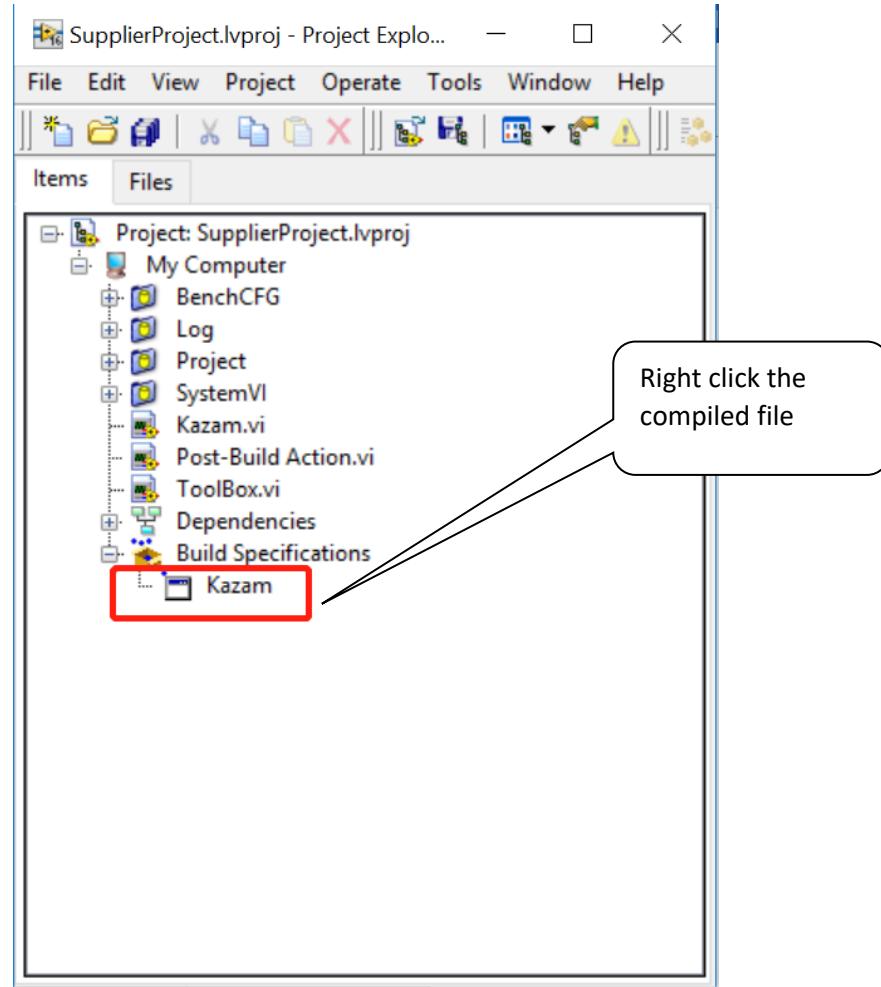


Figure 15

#### 6.2.1 Driver VI

In the DriverVI folder, there is the driver vi template, its name is DRV\_template.vi, its front panel as Figure 16. The developer can change or modify the front panel.

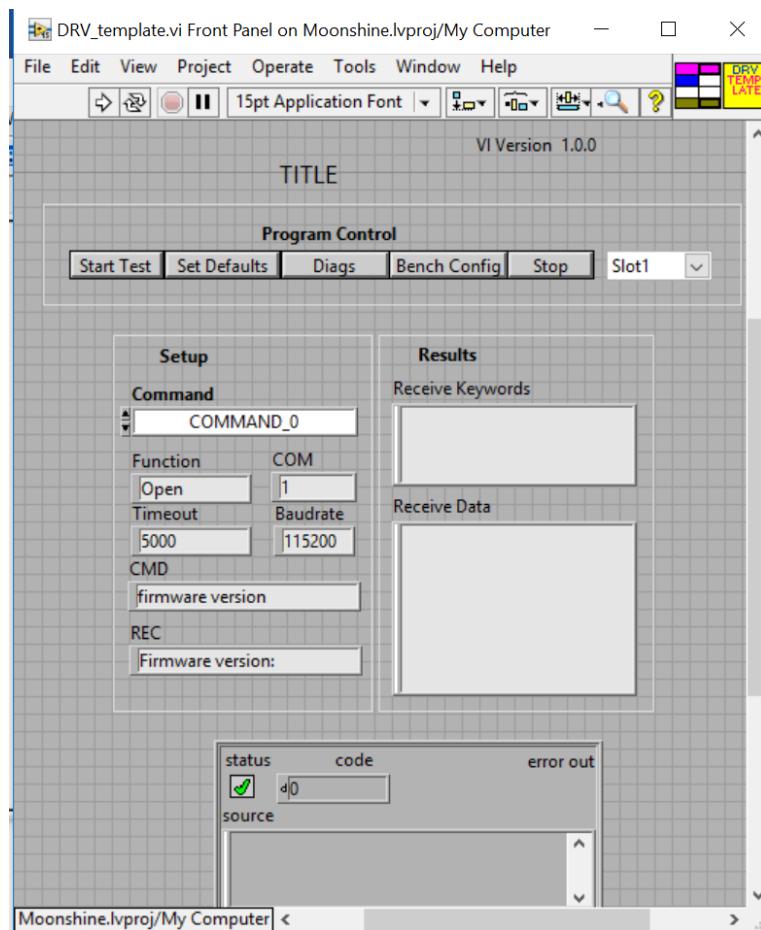


Figure 16

Figure 17 is the back panel of the DRV\_template.vi, the developer can add their code in the “TEST” case or modify the template’s code.

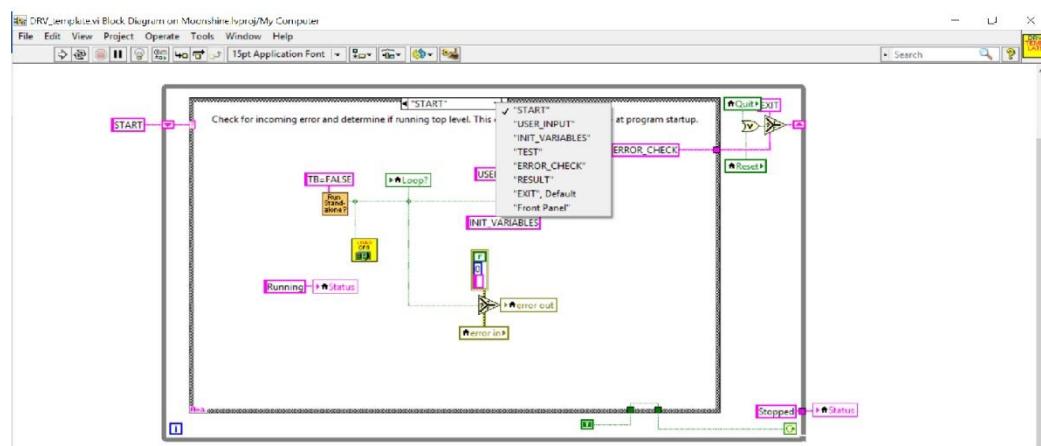


Figure 17

Figure 18 is the input and output interface of the DRV\_template.vi, it is the default interface and cannot be changed.

**DRV\_template.vi**

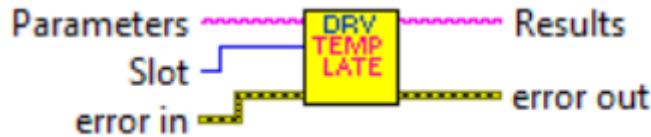


Figure 18

### 6.2.2 Test VI

In the TestVI folder, there is the test vi template, its name is TST\_template.vi, its front panel as Figure 19. The developer can change or modify the front panel.

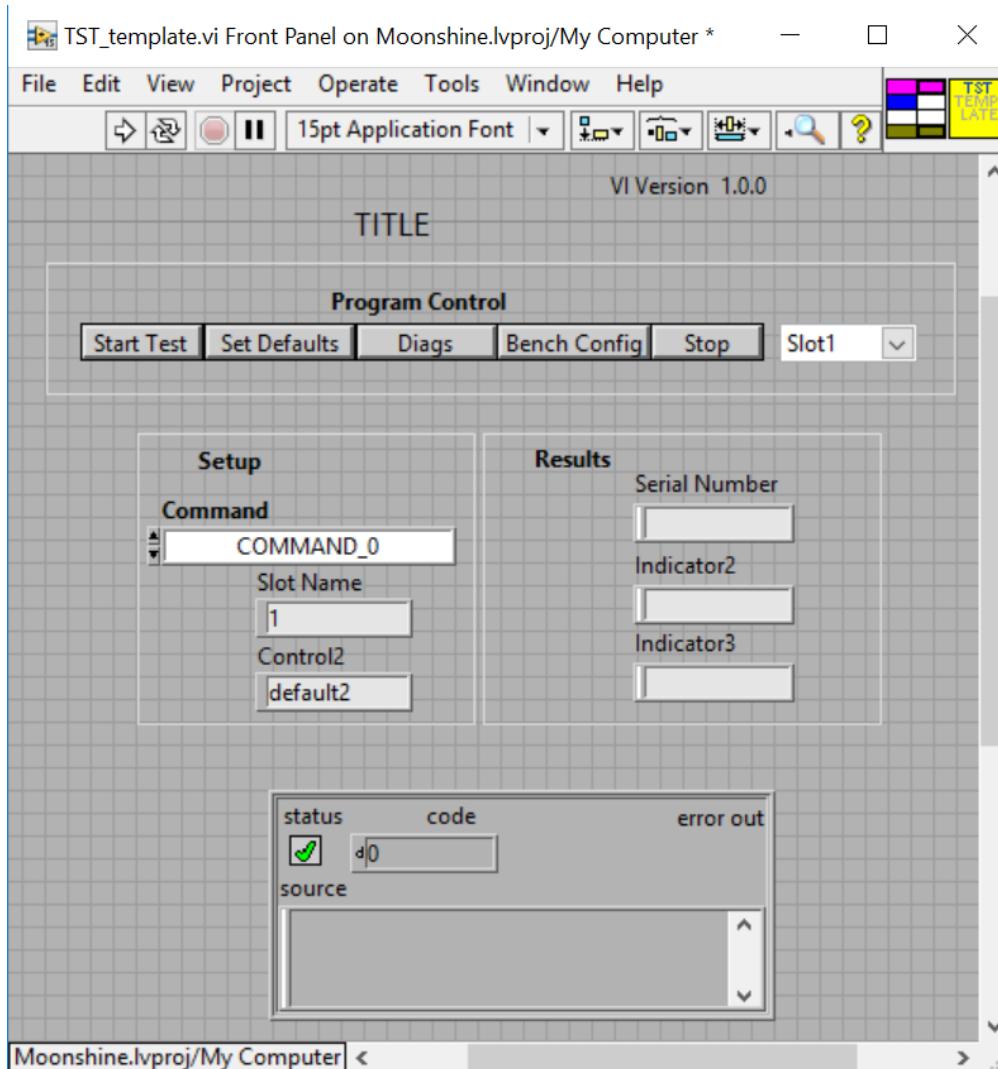


Figure 19

Figure 20 is the back panel of the TST\_template.vi, the developer can add their code in the “TEST” case or modify the template’s code.

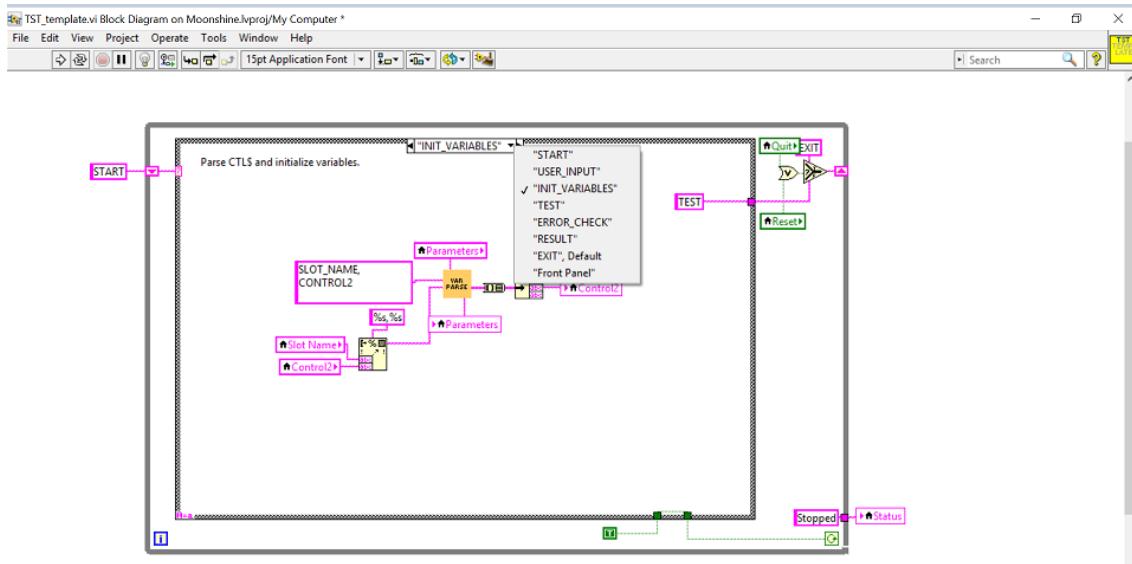


Figure 20

Figure 21 is the input and output interface of the TST\_template.vi, it is the default interface and can not be changed.

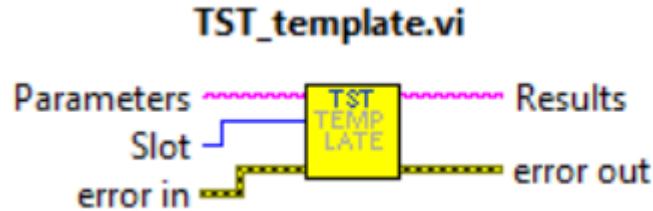


Figure 21

### 6.2.3 Log VI

In the SystemVI folder, there is the log template, its name is log\_Template.vi, and there are two global variables vi, as Figure 22.

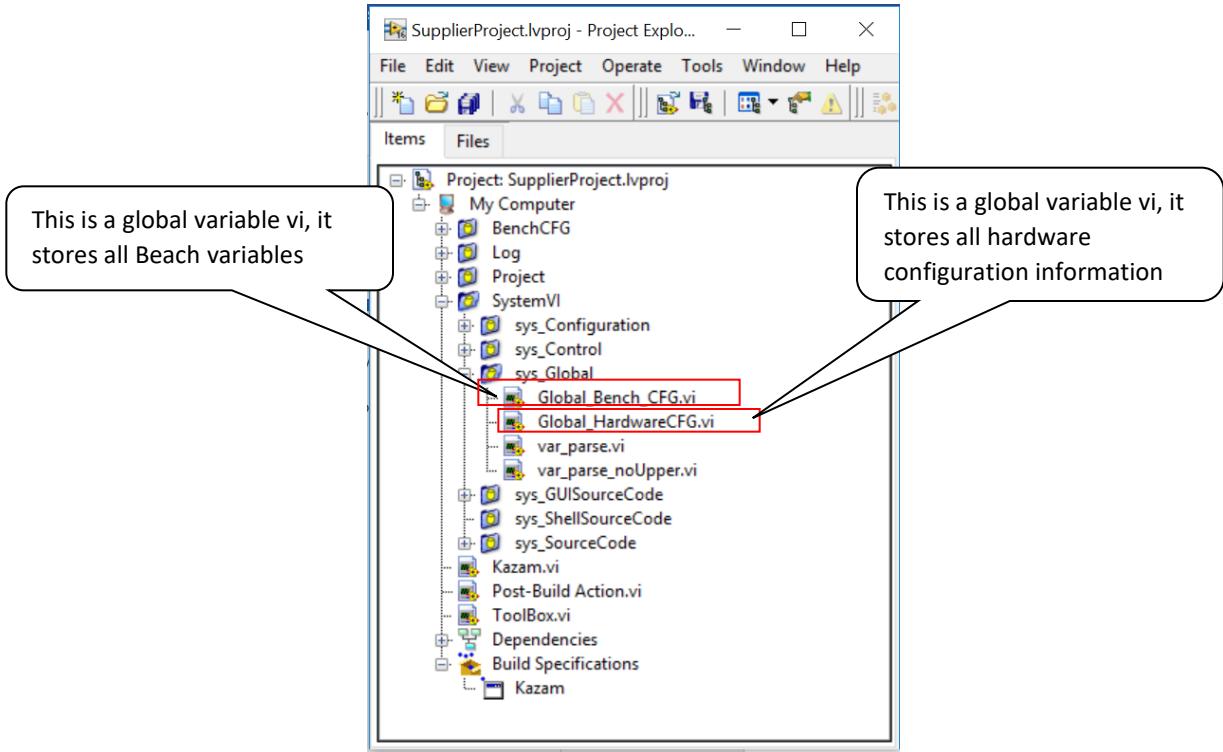


Figure 22

Figure 23 to Figure 29 is the back panel of the log\_template.vi, here are some introduction, the developers can customize special log files according to their needs.

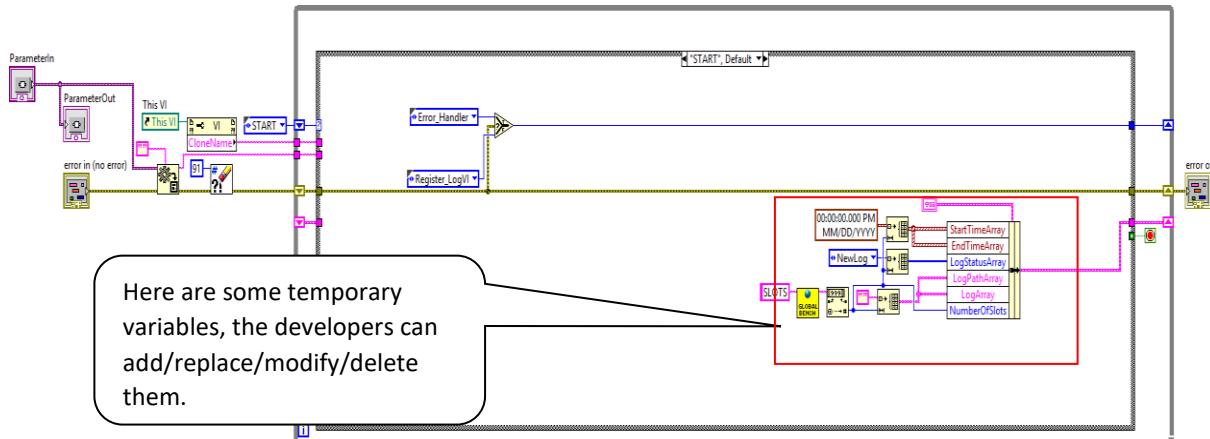
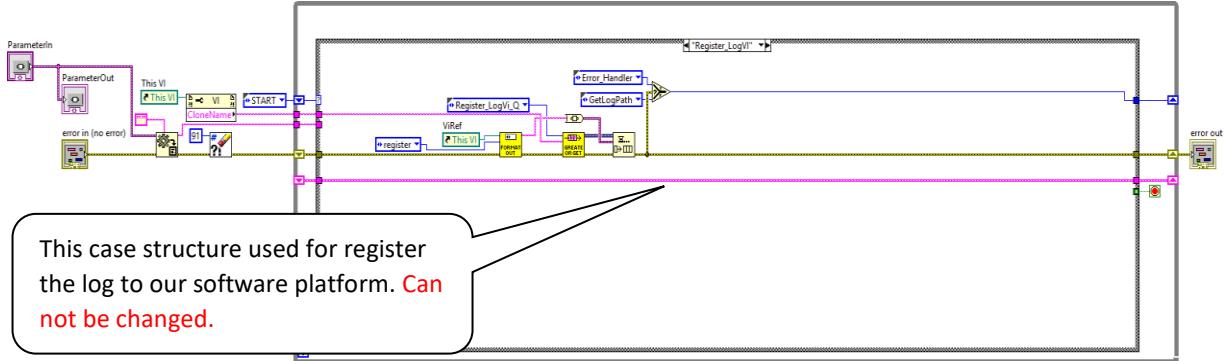
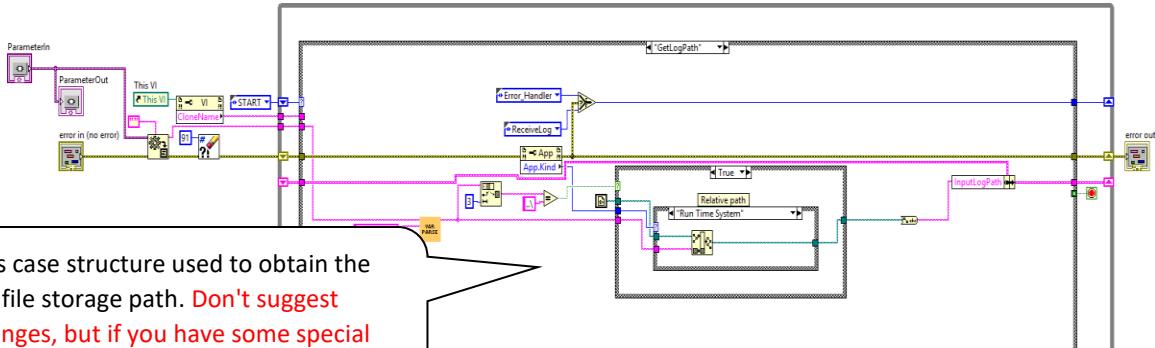


Figure 23



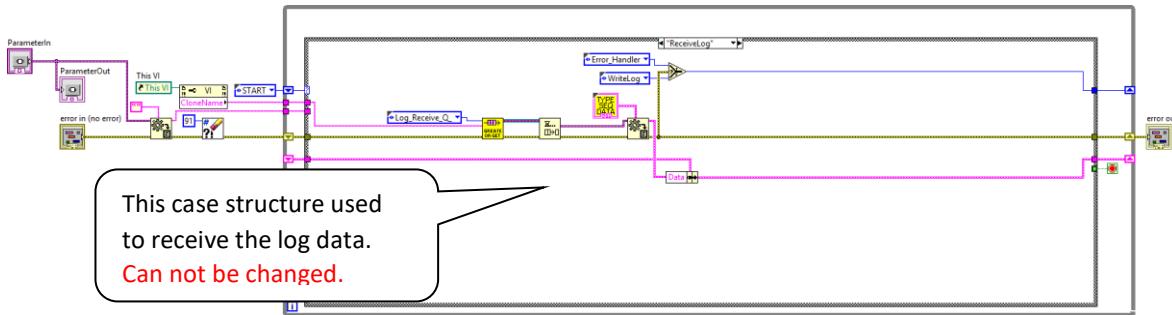
This case structure used for register the log to our software platform. **Can not be changed.**

Figure 24



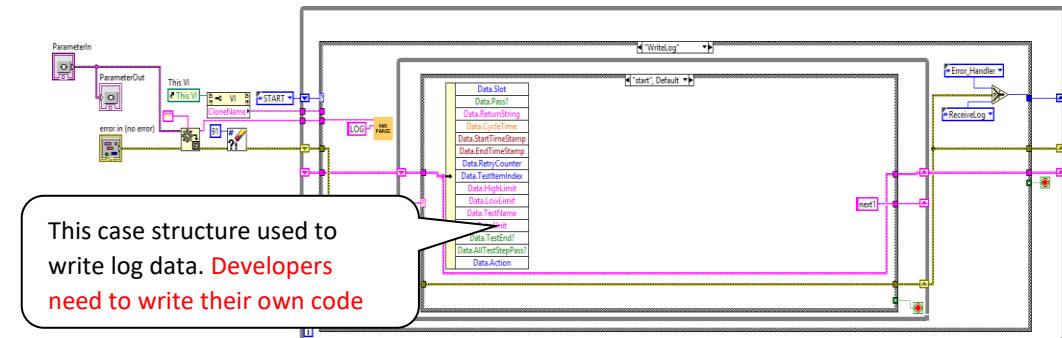
This case structure used to obtain the log file storage path. **Don't suggest changes, but if you have some special needs, you can modify it**

Figure 25



This case structure used to receive the log data. **Can not be changed.**

Figure 26



This case structure used to write log data. **Developers need to write their own code**

Figure 27

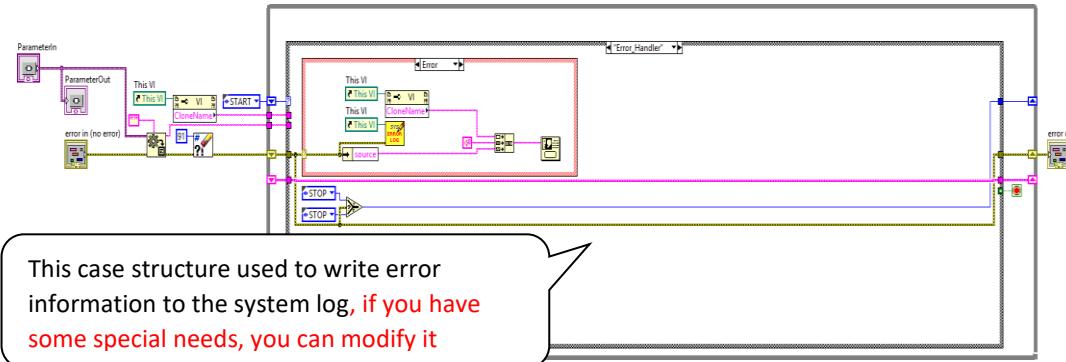


Figure 28

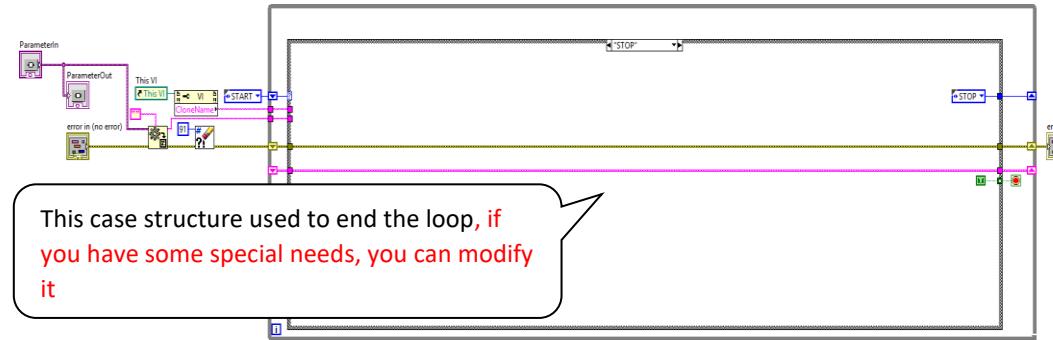


Figure 29

#### 6.2.4 System configuration

In the BenchCFG folder, here are two files, as Figure 30.

Bench\_Default\_Config.csv, it is used for storing bench information, please do not rename it. And there are two kinds of Bench\_Default\_Config.csv files, one is used to start Kazam.exe from local PC, the other is used to start Kazam.exe from the server

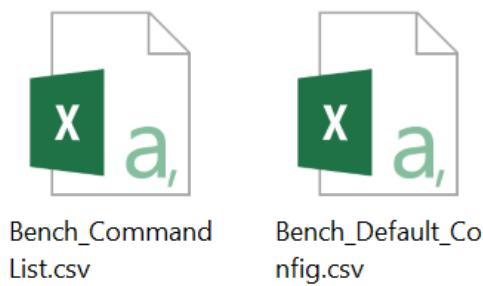


Figure 30

Figure 31 is the bench configuration file, it is used to start Kazam.exe from local PC, in this file, have some keywords, they can not be changed.

A	B
1 CheckValue	f7189e5c21a1e516d93c5ea1fd733fd
2 Item	Detail
3 BenchShellPath	Demo\Prj_CFG\shell_factory.csv
4 BenchHardwareConfigurationPath	Demo\Prj_CFG\HardwareCFG.csv
5 SearchPath	Project\Demo
6 Project Line	1
7 Test Station Name	Vnl
8 Fixture Name	xxx
9 Product Name	Demo
10 Factory_ID	LAB
11 MinSizeOfShowResult	10
12 TCPPort	7200
13 AutoUpdate	
14 Emailcontact	user1@server.com/user2@server.com
15 Textcontact	phonenumber1/phonenumber2
16 LOG_PATH	..\..\LOG\;
17 KazamUpdateServer	
18 ProjectUpdateServer	
19 ProjectUpdatePackageName	BFT
20	
21	

Figure 31

The value of “BenchShellPath” keyword is “Demo\Prj\_CFG\shell\_factory.csv”, it means that the Kazam.exe will run the “..\\Project\\ Demo\\Prj\_CFG\\shell\_factory.csv” file first.

Figure 32 is another bench configuration file, it is used to start Kazam.exe from the server, the developers can use this format of bench configuration file to centralize management of Kazam. It can use computer name or MAC address to specify the BenchCFG parameters of Kazam for each Kazam’s shortcut. If there are several network cards on a PC, the developer can use any MAC address or the computer name to specify BenchCFG parameter

1.	CheckValue	7824c128e0ccdb3bdaea11e04046551																		
2.	ComputerName_or_MACAddress	BenchShellPath	BenchHardwareConfigurationPath	SearchPath	Project	Lir	Test	Stati	Fixture	Ni	Product	Ni	Factory	_1	MinSize0	TCPPort	AutoU	EmailIcon	TextContent	LOG_PATH
3.	52X-52D916ZWP	Demo\Pl_CFG\shell_factory.csv	Demo\Pl_CFG\HardwareCFG.csv	Project\Demo	1	BFT	xxx	Demo	IAB	10	7200	user1@sciphonenu_1_\LOG\LOG	\Kazam;				KazamUp	ProjectUp	ProjectUp	File\plot GUI\Event
4.	00-09-5A-3C-7A-00	Demo\Pl_CFG\shell_factory.csv	Demo\Pl_CFG\HardwareCFG.csv	Project\Demo	1	AFT	xxx	Demo	IAB	10	7200	user1@sciphonenu_1_\LOG\LOG	\Kazam;				2020	2021	2020	2021
5.	D4-38-04-4B-3A-CD	Demo\Pl_CFG\shell_factory.csv	Demo\Pl_CFG\HardwareCFG.csv	Project\Demo	1	AUT	xxx	Demo	IAB	10	7200	user1@sciphonenu_1_\LOG\LOG	\Kazam;				2020	2021	2020	2021

Figure 32

In the shell\_factory.csv file, the developers can add/replace/reduce/modify the function of the software platform.

Note: shell\_factory.csv, it is the configuration file of Kazam.exe, the developers need to edit it according to their own needs.

As Figure 33, it is a demo shell configuration, it has four log functions, one GUI and it redefine the number of the slot. Here's the details of this script:

Resource\_Name: Sequence\_config

VI\_NAME: sys\_SequenceCFGReadout.vi

CONTROL\_STRING Keywords: (Red is the keyword, blue is the value of the keyword)

TestPlan= DemoSequence1.0.0.1.csv

WaitUntilDone=True

This step call the sys\_SequenceCFGReadout.vi from the “..\\..\\SystemVI” path, it will load the TestPlan form the “..\\Project\\xxx\\DemoSequence1.0.0.1.csv” path, this path is configured by the Bench\_Default\_Config.csv file. The TestPlan name is “DemoSequence1.0.0.1.csv”, and here is a keyword, it is called “TestPlan”.

“TestPlan” is the default keyword for all slots, if define “TestPlan1= Seq\_Slot1.csv”, it means that slot 1 will use this “Seq\_Slot1.csv” sequence.

If define “TestPlan2= Seq\_Slot2.csv”, it means that slot 2 will use this “Seq\_Slot2.csv” sequence.

And so on.

For example:

TestPlan= DemoSequence1.0.0.1.csv | TestPlan1= Seq\_Slot1.csv | TestPlan2= Seq\_Slot2.csv

VI_NAME	COMMENTS
CheckValue	TestPlan=DemoSequence1.0.0.1.csv WaitUntilDone=True
RELEASE_DATE	RELEASE_DATE=2011-01-01
SEQ	Resource_Name VI_NAME
Sequence_config	sys.SequenceCFGReadout.vi
Sequencer	sys.SequenceLauncher.vi
Display	GUI_Operator.vi
Datalog	log_Daily.vi
Datalog	log_single.vi
Datalog	log_timelog.vi
Datalog	log_amazon_aws.vi
EOF	
	CONTROL_STRING
	TestPlan=DemoSequence1.0.0.1.csv WaitUntilDone=True
	SLOTS=1 SeqEngineName=TestSequence.vi
	FP_Show=True RefreshTime=500 Real-time=1
	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/155;WTN[3/3]:4RU/919-Else=Muffin Emailcontact=user1@server.com/user
	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/155;WTN[3/3]:4RU/919-Else=Muffin
	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/155;WTN[3/3]:4RU/919-Else=Muffin
	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/155;WTN[3/3]:4RU/919-Else=Muffin
	LOG=STEP LOG_PATH=..\..\LOG

Figure 33

### 6.2.5 TestPlan and Hardware configuration

In the Prj\_CFG folder, here are test plan, Kazam configuration and hardware configuration, the developers can store different TestPlan in this folder. As Figure 34, here are five TestPlans and one hardware configuration file.

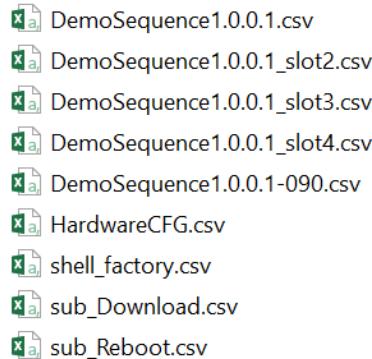


Figure 34

### 6.2.5.1 TestPlan

Kazam can support loading XLSX files as TestPlan, the format is as below:

**TestPlan**=Demo.xlsx Abc Abc is the tab name of xlsx file, the tag names are case-sensitive

Figure 35 is a demo TestPlan, there have some keywords, and the following is a detailed explanation:

Keyword	Description	Example
Index	<p>Step index, it is an integer range 1 to infinity</p> <p>F is a keyword for fixed header sequence, for example F1,F2,F3.....</p>	
Enable	<p>It controls whether running this test step, its value is 0 , 1 ,SFC=ON, SFC=OFF, Loop=ON and Loop=OFF, SFC, LOOP, SFCOFF, LOPOFF</p> <p>If the value is 1, the step will be running</p> <p>If the value is 0, the step will not be running</p> <p>If the value is <b>SFC=ON</b>, when the SFC is OFF, the step will not be running</p> <p>If the value is <b>SFC=OFF</b>, when the SFC is ON, the step will not be running</p> <p>If the value is <b>Loop=ON</b>, When Kazam runs in loop mode, the step will be running</p> <p>If the value is <b>Loop=OFF</b>, When Kazam runs in operator mode, the step will be running</p> <p>If the value is <b>SFC</b>, when the SFC is OFF, the step will not be running</p> <p>If the value is <b>SFCOFF</b>, when the SFC is ON, the step will not be running</p> <p>If the value is <b>LOOP</b>, When Kazam runs in loop mode, the step will be running</p> <p>If the value is <b>LOPOFF</b>, When Kazam runs in operator mode, the step will be running</p> <p>"   " can be used as or condition</p> <p>And the developers can customize enabling parameters in Enable column in Testplan file, and modify its default status in the shell configuration file with the <b>EnableParameter</b> keyword.</p> <p>Please refer to the parameter description of sys_sequencecfgreadout.vi in section 8.2.2 Other parameter</p>	<p>SFC=ON Loop=OFF SFC=OFF Loop=OFF SFC=ON Loop=ON SFC=OFF Loop=ON Loop=ON Loop=OFF SFC=ON SFC=OFF SFC LOPOFF HVT HVT is the developer's own definition</p>
PREREQ	The prerequisite column is used to set the steps to run in parallel. Its value is the name of the previous step of this step	The prerequisite of "OpenPowerSupplyUART" is "Wait all the slot get SN"
RunInParallel	Its value is 0, 1 or empty. Empty and 0 mean that this test step is a blocking step. 1 means that this test step is a non-blocking	

	step and when using this feature, process control syntax only "Next" is valid. This step will launch, and continue with the next step without waiting for a return.	
Step Name	It is the name of this step. <b>All the test name must be unique</b>	GetSN
Function Call	<p>The platform can call dynamic link libraries, executables, python script, batch file and C# source code directly</p> <p>If there is no suffix, it means that will call VI;</p> <p>If this is empty, the result of the previous step is automatically retrieved.</p>	TST_Wait TST_Wait.dll -----6.5.1 EXE/Python/Bat file support TST_Wait.exe -----6.5.1 EXE/Python/Bat file support TST_Wait.py -----6.5.1 EXE/Python/Bat file support TST_Wait.pyc -----6.5.1 EXE/Python/Bat file support TST_Wait.bat -----6.5.1 EXE/Python/Bat file support TST_Wait.cs -----6.5.3 C# source code support
LSL	<p>Low limit, its value can be decimal, hexadecimal, float and string</p> <p>If only one number is defined in LSL, it means that the test result needs to be greater than or equal to LSL</p> <p>If LSL is the same as USL, it means that the test results must be the same as USL and LSL</p> <p>" " can be used as or condition, for example, LSL is "123 456 789", USL is "123 456 789", if the result is 123 or 456 or 789, the test result will pass, otherwise it will fail</p> <p>"!" can be used as Not condition, for example, LSL is "!EMPTY", USL is "!EMPTY", if the result is not empty, the test result will pass, otherwise it will fail, another example, LSL is "!abc", USL is "!abc", if the result is not "abc", the test result will pass, otherwise it will fail,</p> <p><b>Note: when using OR and NOT expressions, the values of USL and LSL must be the same</b></p>	
USL	<p>High limit, its value can be decimal, hexadecimal, float and string</p> <p>If only one number is defined in USL, it means that the test result should be less than or equal to USL</p> <p>If USL is the same as LSL, it means that the test results must be the same as USL and LSL</p> <p>" " can be used as or condition, for example, LSL is "123 456 789", USL is "123 456 789", if the result is 123 or 456 or 789, the test result will pass, otherwise it will fail</p> <p><b>Note: when using or expressions, the values of USL and LSL must be the same</b></p>	
Unit	The limit of the unit, it will be displayed in the GUI	
LimitCheck	If the step has multiple results, this keyword will tell the platform which result is you needs.	If the multiple result is "Vout=3.8 Iout=110", the "LimitCheck" is "Vout", it means that this step result is "Result=3.8"

Save Result	If the value is <b>1</b> , the result of this step will be stored in the daily log;  If the value is <b>0</b> or <b>empty</b> , the result of this step will not be stored in the daily log;	
Casesensitive	If the value is "Yes", the result output and global variable will be case sensitive  If the value is empty or "NO", the result output and global variable will not be case sensitive	
Property	<ul style="list-style-type: none"> <li>➤ Property=<b>normal</b>, it means that when this test vi is called, this test vi will run in parallel</li> <li>➤ Property=<b>Sync</b>, it means that when this test vi is called, this test vi will wait for the other test VI and then start together.</li> <li>➤ Property=<b>One-off</b>, it means that this test vi is executed only once, this property commonly used to initialize instrument</li> <li>➤ Property=<b>Share</b>, it means that this test vi is executed one by one, this property commonly used to share resource, such as instrument</li> <li>➤ Property=<b>Release</b>, it means that when this test vi is called, one of this test vi will run and get some information and then send the information to other slots, such as get the panel SN</li> <li>➤ Property=<b>Once</b>, it means that one of this clone test vi is executed, the other clone will receive the result, this property commonly used to control fixture's lid or other.</li> <li>➤ Property= <b>Transmit</b>, it means that when this slot's vi runs complete, this slot's vi will send some information to other slot, such as a start signal. The receive step name must be specified in the "input parameter" for this step. The keyword is "<b>REC_Step</b>", for example,   <b>REC_Step=Slot2WaitForStart;Slot2WaitForStart;</b>             Slot2WaitForStart and Slot2WaitForStart are the test step name         </li> <li>➤ Property= <b>Receive</b>, it means that before this VI runs, it will wait for information from the transmitter VI.</li> </ul>	
Dynamic	If the value is 1, this test step will be dynamic call;  If the value is 0 or empty, this test step will be preload;	
Input Parameter	The parameter of this step. The format is as below:  Keyword=value  " " is the delimiter	Function=Open COM=PSCOM Baudrate=115200  ShowFP=YES

	<p>The developer can use "ShowFP" keyword to show their front panel of driver vi or test vi on the actuator GUI, the maximum size displayed is 760x600</p> <p>The value of "ShowFP" is YES or NO, its default value is NO</p> <p>"Result" keyword is used to assign a value to the current step, for example, "Result=123", it means that the result of this step is 123, another example, "Result=&lt;&lt;Slots_info.SN&gt;&gt;", it means that the result of this step is the value of Slots_info.SN</p>	
Retry	<p>If the value is 0 or empty, this step is failed, this step does not retry;</p> <p>If the value is greater than or equal to 1, this step is failed, this step will retry;</p>	
Next Step CTL	<ul style="list-style-type: none"> <li>➤ <b>Next</b>, it means that when this test step is completed, the next will be the next step</li> <li>➤ <b>if fail goto</b>, it means that when this test step is failed, the next will be run the specified step</li> <li>➤ <b>if pass goto</b>, it means that when this test step is passed, the next will be run the specified step</li> <li>➤ <b>goto</b>, it means that when this test step is completed, the next will be run the specified step</li> <li>➤ <b>force pass</b>, it means that when this test step is completed, forced this step pass</li> <li>➤ <b>force fail</b>, it means that when this test step is completed, forced this step fail</li> <li>➤ <b>Stop if fail</b>, it means that when this test step is failed, the test sequence will be stopped</li> <li>➤ <b>If...else</b>, it means that this step is not a test item, its result will not record to the final result, it is a true or false step.</li> <li>➤ <b>Switch...case</b>, it means that this step is not a test item, its result will return the next step name, and it is a true or false step. When Kazam receives the next step name, kazam will jump to this step</li> <li>➤ <b>Pause Slot</b>, it means that after running this step, the specified slot is paused, use <b>SlotNum</b> to set the specified slot in the "Input Parameter"</li> <li>➤ <b>Continue Slot</b>, it means that after running this step, the specified slot will continue to run, use <b>SlotNum</b> to set the specified slot in the "Input Parameter"</li> <li>➤ <b>Stop Slot</b>, it means that after running this step, the specified slot will stop running, use <b>SlotNum</b> to set the specified slot in the "Input Parameter"</li> </ul>	<p><b>SlotNum=1;2;3</b> , it's means you specify slot 1, slot2 and slot3</p> <p><b>SlotNum=Other</b>, it's going to pause/continue/stop all the other slots except for itself</p> <p><b>SlotNum&gt;All</b>, it's going to pause/continue/stop all the slots</p>
Next Step Name	<p>The name of the next step which will be running after current step</p> <p>There are three kind of expressions:</p>	<p>When we use the "If fail goto" to control the step flow, Kazam can handle more jump conditions.</p> <p>TestEnd---it means that if this step fail, it will go to "TestEnd"</p>

	<ul style="list-style-type: none"> <li>➤ Test name of the next step, for example: TestEnd</li> <li>➤ Jump by number of cycles, for example: RandomNum Fail:TestEnd <b>2</b>:GetResult <b>default:Random Fail:TestEnd 3:GetResult</b> "Fail" and "default" are the keywords "2" and "3" are cycles number</li> <li>➤ True or false jumps, for example: Read firmware version cleanup</li> </ul>	<p>RandomNum Fail:TestEnd <b>3</b>:GetResult---it means that if this step fail, the first time fail will go to "RandomNum", the second time fail will go to "RandomNum", the third time fail will go to "GetResult", if the goto count is reached, this step is still failed, it will go to "TestEnd"</p> <p>default:Random Fail:TestEnd <b>3</b>:GetResult---it means that if this step fail, the first time fail will go to "Random", the second time fail will go to "Random", the third time fail will go to "GetResult", if the goto count is reached, this step is still failed, it will go to "TestEnd"</p> <p>When we use the "If...else" to control the step flow, we will use " " to separate the true step and false step, for example:</p> <p><b>Read firmware version cleanup</b></p>
Goto Counter	The number of goto jump statement  If the value is empty, it means that the step will always jump	
ErrorHandle	The developer can define the error handler for each test step  The error handling entry format is defined as follows:  ErrorCode:Test Step Name  If the error handler is empty, the test will be stopped  If the ErrorCode is "default", the test will go to the default step, for example: default:TestEnd	0:EH_0 1:EH_1 2:EH_2 3:EH_3 4:EH_4 default:TestEnd
ErrorCode	If this step fails, will be recorded the error code into the log	
Note	Developers can according to their own need to edit some notes	

File Home Insert Page Layout Formulas Data Review View Add-ins Tell me what you want to do... Huang, Fei Share

Cut Copy Format Painter Clipboard Font Alignment Number Cells Editing

DemoSequence1.0.0.1.slot2.csv - Excel

A B C D E F G H I J K L M N O P Q R S

1 Check'8a600c16f57d57166e3dc898273473d4

2 Index Enable PREREQ Step Name Function Call LSL USL Unit LimitCI Save Resu Property Dynamic Input Para Retry Next Step CTL Next Step Name Goto Cour ErrorCode Note

3 1 1 GetSN TST\_GetSN 1 release 0 Length=5 0 Next Error-001

4 2 1 Wait all the slot get SN TST\_Rendezvous 0 sync 0 Timeout= 0 Next Error-002

5 3 1 Wait all the OpenPowerSupplyUART DRV\_SerialPort 0 normal 0 Function= 0 Next Error-003

6 4 1 Wait all the OpenGPIOUART DRV\_SerialPort 0 normal 0 Function= 0 Next Error-004

7 5 1 Wait all the OpenDUTUART DRV\_SerialPort 0 normal 0 Function= 0 Next Error-005

8 6 1 GPIO Init DRV\_GPIO 0 one-off 0 COM=GPK 0 Next Error-006

9 7 1 Wait Colse the Lid TST\_Wait.dll 0 0 COM=GPK 0 if fail goto Wait Colse the Li 20 Error-008

10 8 1 Read Lid Status DRV\_GPIO Low Low H/L Ch1 0 once 0 COM=GPK 0 if fail goto Cleanup Start Error-009

11 9 1 Check fixture is closed? TST\_FlowControl LOW LOW H/L 0 0 SYS\_Info= 0 if fail goto Cleanup Start Error-010

12 10 1 Cylinder Downward DRV\_GPIO 0 once 0 Function= 0 Next Error-011

13 11 1 Wait Cylinder Down TST\_Wait 0 0 Wait=50 0 Next Error-012

14 12 1 Read Cylinder Status DRV\_GPIO Low Low H/L Ch1 0 once 0 COM=GPK 0 if fail goto Wait Cylinder Do 20 Error-012

15 13 1 Check Cylinder is ok? TST\_FlowControl LOW LOW H/L 0 0 SYS\_Info= 0 if fail goto Cleanup Start Error-013

16 14 1 Power\_On DRV\_PowerSupply 0 0 Function= 0 if fail goto Cleanup Start Error-014

17 15 1 Wait DUT Power up TST\_Wait 0 0 Wait=10 0 Next Error-015

18 16 1 Read Normal Current DRV\_PowerSupply 100 650 mA Iout 1 0 Function= 3 Next Error-016

19 16 1 Read Ship Current DRV\_PowerSupply 200 550 mA Iout 1 0 Function= 3 Next Error-016

20 17 1 Wait DUT P Read firmware version DRV\_SerialPort 23.98.0.3 23.98.0.3 1 0 Function= 0 Next Error-017

21 17 1 Wait DUT P Read Hardware version DRV\_SerialPort 23.98.0.3 23.98.0.3 1 0 Function= 0 Next Error-017

22 18 1 Cleanup Start TST\_Rendezvous 0 sync 0 Timeout= 0 Next Error-018

23 19 1 Power\_Off DRV\_PowerSupply 0 normal 0 Function= 0 Next Error-019

24 20 1 Wait for discharge TST\_Wait 0 0 Wait=100 0 Next Error-020

25 21 1 Cylinder Upward DRV\_GPIO 0 once 0 Function= 0 Next Error-021

26 22 1 ClosePowerSupplyUART DRV\_SerialPort 0 0 Function= 0 Next Error-022

27 23 1 CloseGPIOUART DRV\_SerialPort 0 0 Function= 0 Next Error-023

28 24 1 CloseDUTUART DRV\_SerialPort 0 0 Function= 0 Next Error-024

29 DemoSequence1.0.0.1.slot2

Figure 35

### 6.2.5.2 Sub TestPlan

The developer can define the sub TestPlan to replace the repeat test step. There are two ways to call the sub testplan.

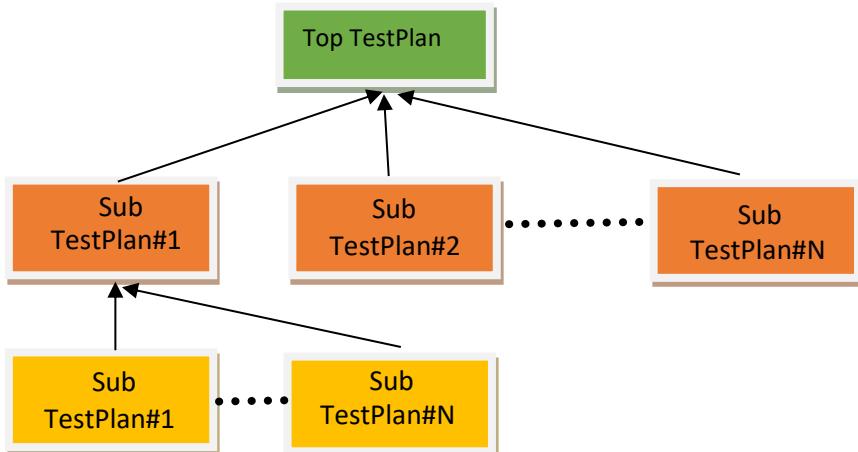


Figure 36

As Figure 36, the developer can use the `%xxxxxx%` format to tell the platform that it is the sub testplan, `xxxxxx` is the name of sub tesplan, “%” is the separator.

DemoSequence1.0.0.csv - Excel																	
Index	Enable	PREREQ	Step Name	Function Call	LSL	USL	Unit	LimitCheck	Save Property	Dynamic	Input Parameter	Retry	Next Step	Next Step Goto	Cour	Error Code	Note
1	CheckVal	a600c1657d57166e3d898273473d4															
2																	
3	1	1	GetSN	TST_GetSN				1	release	0	Length=5	0	Next		Error-001		
4	1	1	Wait all the slot get SN	TST_Rendezvous				0	sync	0	Timeout=10000	0	Next		Error-002		
5	3	1	Wait all the OpenPowerSupplyUART	DRV_SerialPort				0	normal	0	Function=Open COM=PSC01	0	Next		Error-003		
6	4	1	Wait all the OpenGPIOUART	DRV_SerialPort				0	normal	0	Function=Open COM=SPID0	0	Next		Error-004		
7	5	1	Wait all the OpenDUTUART	DRV_SerialPort				0	normal	0	Function=Open COM=DUT01	0	Next		Error-005		
8	6	1	GPIO Init	DRV_GPIO				0	one-off	0	COM=GPIO COM Function=C	0	Next		Error-006		
9	7	1	Power_On	DRV_PowerSupply				0		0	Function=PowerOn COM=PS	0	if fail goto Cleanup Start		Error-007		
10	8	1	Wait DUT Power up	TST_Wait				0		0	Wait=10	0	Next				
11			Read Normal Current	DRV_PowerSupply	100	650 mA	iout	1		0	Function=Read COM=PSC01	0	Next				
12			Send ship Command	DRV_ADB				0		0	Function=Read COM=SC01	0	Next				
13			Wait to enter ship mode	TST_Wait				0		0	Wait=2000	0	Next				
14			Read Ship Current	DRV_PowerSupply	10	35 uA	iout	1		0	Function=Read COM=PS01	0	Next				
15			Reboot					0		0	FileName=sub_Reboot.csv	0	Next				
16			Measure VBAT	DRV_DMM	3.7	3.9 V		1		0	Function=ReadVoltage COM	0	Next		Error-014		
17			Cleanup Start	TST_Rendezvous				0	sync	0	Timeout=20000	0	Next		Error-015		
18			Power_Off	DRV_PowerSupply				0		0	Function=PowerOff COM=P	0	Next		Error-016		
19			Wait for discharge	TST_Wait				0		0	Wait=100	0	Next		Error-017		
20			ClosePowerSupplyUART	DRV_SerialPort				0		0	Function=Close COM=PSC01	0	Next		Error-018		
21			CloseGPIOUART	DRV_SerialPort				0		0	Function=Close COM=SPID0	0	Next		Error-019		
22			CloseDUTUART	DRV_SerialPort				0		0	Function=Close COM=DUT01	0	Next		Error-020		

sub_Reboot.csv - Excel																	
Index	Enable	PREREQ	Step Name	Function (LSL)	USL	Unit	LimitCheck	Save Resu	Property	Dynan	Input Paramre	Retry	Next Step CTL	Next Step Name	Goto Cour	Err	Note
1	CheckVal	b2c60982df9894ad0165fecb3a5f															
2																	
3	1	1	Power_Off	Random\$	30	100		0	normal	0	max=100	0	if...else				
4	2	1	Wait for discharge	TST_Wait				0		0	Wait=100	0	Next				
5	3	1	Power_On	TST_Wait				0		0	Function=Pov	0	if fail goto				

Figure 37

As Figure 38, the sub testplan can be loaded in the ‘Function Call’ field. If the format of sub testplan is xlsx, the developer must specify the name of the sheet to be called. If the format of sub testplan is csv, the developers only need to fill in the name of the calling CSV file

**Sub testplan name**

**The file name of Sub testplan**

**sheet**

**The sheet name of the sub testplan**

**"\$" is the separator, it means that this step is the top testplan step**

Index	Enable	PREREQ	Step Name	Function Call	LSL	USL	Unit	LimitChec	Save Property	Dynamic	Input Parameter	Retry	Next Step	Next Step Name	Goto Cou	Error Code	Note
1	CheckVal:8a6000c16f57d571663dc98273473d4		GetSN	TST_GetSN				1 release	0 Length=5	0		0 Next				Error-001	
2	Index	1	Wait all the slot get SN	TST_Rendezvous				0 sync	0 Timeout=10000	0		0 Next				Error-002	
3	2	1	Wait all the OpenPowerSupplyUART	DRV_SerialPort				0 normal	0 Function=Open COM=PSC01	0		0 Next				Error-003	
4	3	1	Wait all the OpenGPIOUART	DRV_SerialPort				0 normal	0 Functions=Open COM=GPIOC	0		0 Next				Error-004	
5	4	1	Wait all the OpenDUTUART	DRV_SerialPort				0 normal	0 Function=Open COM=DUT01	0		0 Next				Error-005	
6	5	1	GPIO Init	DRV_GPIO				0 one-off	0 COM=GPIOCOM Function=C	0		0 Next				Error-006	
7	6	1	Power_On	DRV_PowerSupply				0	0 Function=PowerOn COM=PS01	0		0 if fail goto Cleanup Start				Error-007	
8	7	1	Wait DUT Power up	TST_Wait				0	0 Wait=10	0		0 Next				Error-008	
9	8	1	Read Normal Current	DRV_PowerSupply	100	650 mA	Iout	1	0 Function=Read COM=PSC01	0		0 Next				Error-009	
10	9	1	Send ship Command	DRV_ADB				0	0 Function=Read COM=PSC01	0		0 Next				Error-010	
11	10	1	Wait to enter ship mode	TST_Wait				0	0 Wait=2000	0		0 Next				Error-011	
12	11	1	Read Ship Current	DRV_PowerSupply	10	35 uA	Iout	1	0 Function=Read COM=PSC01	0		0 Next				Error-012	
13	12	1	Reboot	sub_Reboot.xlsx sheet				3.7	3.7	1		0				Error-013	
14	13	1	Measure VBAT	DRV_DMM								0				Error-014	
15	14	1	Cleanup Start	TST_Rendezvous								0				Error-015	
16	15	1	Power_Off	DRV_PowerSupply								0				Error-016	
17	16	1	Wait for discharge	TST_Wait								0				Error-017	
18	17	1	ClosePowerSupplyUART	DRV_SerialPort								0				Error-018	
19	18	1	CloseGPIOUART	DRV_SerialPort								0				Error-019	
20	19	1	CloseDUTUART	DRV_SerialPort								0				Error-020	
21	20	1															
22	21	1															

Index	Enable	PREREQ	Step Name	Function Call	LSL	USL	Unit	LimitChec	Save Resu	Property	Dynamic	Input Para	Retry	Next Step	Next Step Name	Goto Cou	Error Code	Note
1	CheckVal:b2c6098cb2df9894dad0165fecba3a5f																	
2	Index	1	Power_Off	RandomSt	30	100	Random	0 normal	0 max=100	0 if...else	0	0 Wait=100	0 Next	Wait for discharge	Wait for Power_014	Error-014		
3	2	1	Wait for discharge	TST_Wait				0	0	0	0	0 Wait=100	0 Next	Power_On				
4	3	1	Power_On	TST_Wait				0	0	0	0	0 Function=	0 if fail goto \$Cleanup Start\$					
5	4	1																

Figure 38

### 6.2.5.3 Hardware configuration

Figure 39 is a demo hardware configuration file.

In the red box is the name of the resource, the developers can customize.

In the blue box is the slot name, they are the keywords cannot be changed.

In the green box is the value of the resource, the developers can customize.

A	B	C	D	E	F	G	H	I	J	K	L	M	
1	CheckValue	7f9021a4849fd6816ae3434607b917c2											
2		Slot1	Slot2	Slot3	Slot4	Slot5	Slot6	Slot7	Slot8	Slot9	Slot10	Slot11	Slot12
3	DUTCOM	1	2	3	4	5	6	7	8	9	10	11	12
4	PSCOM	13	14	15	16	17	18	19	20	21	22	23	24
5	GPIOCOM	25	26	27	28	29	30	31	32	33	34	35	36
6	Lid	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1	P9.1
7	Cylinder	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1	P0.1
8	PowerKey	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7	P1.0	P1.1	P1.2	P1.3	P1.4	P1.5
9	USBData	P1.6	P1.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P3.0	P3.1
10	USBPower	P3.2	P3.3	P3.4	P3.5	P3.6	P3.7	P4.0	P4.1	P4.2	P4.3	P4.4	P4.5
11	CylinderSensor	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2	P9.2
12													
13													
14													

Figure 39

### 6.2.6 Reset Plan

The reset plan is a TestPlan to restore the automation equipment to its original state, it means that when an exception occurs and the developer wants to reset the equipment to its original correct state, the developer can use this feature. This feature only supports running in debug mode, as shown in Figure 40, developers can enable this feature in the menu, after the reset plan is loaded into slot1, click the start button on the GUI to execute the reset plan

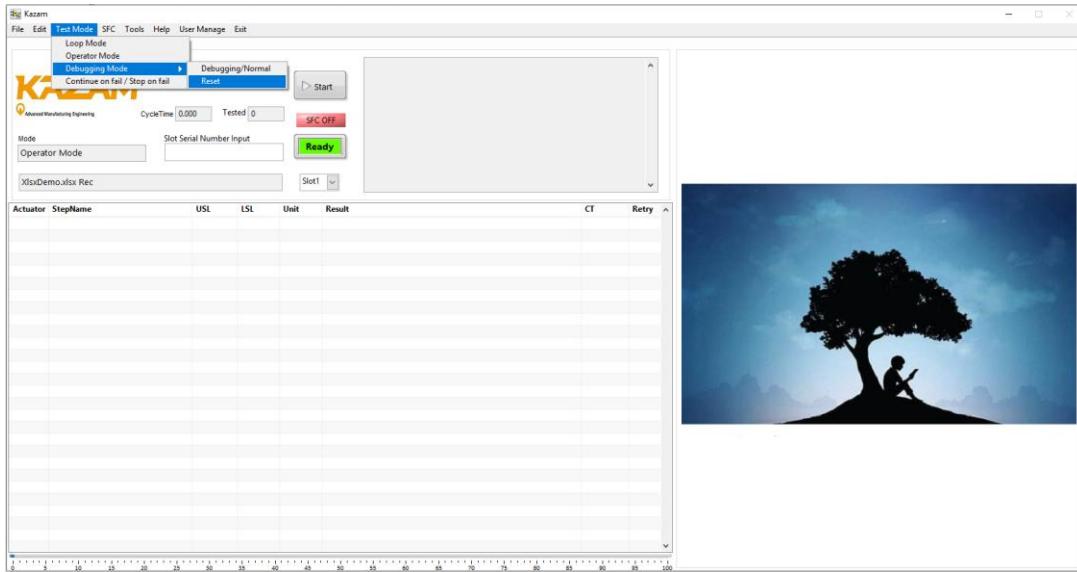


Figure 40

The reset plan is defined using the "ResetPlan" keyword in shell configuration file.

### 6.2.7 Global variable

In the TestPlan, the developers can define four kind of global variable.

- System global variable, it can be defining as below:

`Store=XXX>SYS_info.YYY`

`Store` is the keyword, cannot be changed.

`XXX` is the keyword of the test step result, the developers can customize.

`SYS_info` is the keyword, cannot be changed.

`YYY` is the name of the global variable, the developers can customize.

For example, if the result of test step is “Ch1=Low”, the developers want to store this information, it is defined as follows:

`Store=Ch1>SYS_info.LidStatus`

The developers can store multiple variables into the system global variable, the format is as follows:

`Store=XXX>SYS_info.YYY;ZZZ>SYS_info.NNN;TTT>SYS_info.QQQ`

- Slot global variable, **the storage space of each slot is separate**, it can be defining as below:

`Store=XXX>Slots_info.YYY`

The default global store is ‘`Slots_info`’. The above can also be represented by

`Store=XXX>.YYY` which functions the same as `Store=XXX>Slots_info.YYY`

`Store` is the keyword, cannot be changed.

`XXX` is the keyword of the test step result, the developers can customize.

`Slots_info` is the keyword, cannot be changed.

`YYY` is the name of the global variable, the developers can customize.

For example, if the result of test step is “serialnumber=123456”, the developers want to store this information, it is defined as follows:

`Store= serialnumber >Slots_info.SN`

The developers can store multiple variables into the system global variable, the format is as follows:

`Store=XXX>Slots_info.YYY;ZZZ>SYS_info.NNN;TTT>SYS_info.QQQ`

or

**Store=XXX>.YYY;ZZZ>SYS\_info.NNN;TTT>SYS\_info.QQQ**

➤ How to obtain the global variable value?

1. The developers need to define as below, it means telling the platform which the variable you want

**SYS\_info=LidStatus%**

2. In the vi, the developers need to search the keyword of “**SYS\_info.LidStatus**” at the input parameter

The developers can also get multiple variable values, the format is as follows:

**Slots\_info=SN%CylinderStatus%|SYS\_info=PanelSN%DUTCom%**

➤ How to use the global variables in test plan directly?

The “Test Name”, “Input Parameter”, “LSL”, “USL”, “Unit” and “ErrorCode” can contain global variable in the TestPlan file, and use the << and >> symbol to mark as follows:

**<< Slots\_info.SN >> or << SYS\_info.PanelSN >>**

or

**<< .SN >> or << SYS\_info.PanelSN >>**

For example, if the “Input Parameter” is CMD=idme serial <<Slots\_info.SN>>, the value of Slots\_info.SN is P012345678, so the “Input Parameter” will be replaced by CMD=idme serial P012345678.

➤ Reserved variable name

**PanelSN** is the reserved name of system global variable, the developers can get the panel serial number from this global variable

**SFC** is the reserved name of system global variable, the developers can get the status of SFC from this global variable, the value of SFC is ON and OFF

**SN** is the reserved name of slot global variable, the developers can get the slot serial number from this global variable, and it can modified by developers during the testing.

**Operator** is the reserved name of system global variable, the developers can get the operator ID from this global variable, and it can modified by developers during the testing.

`Diag_Ver` is the reserved name of slot global variable, the developers can get the slot diag version from this global variable, and it can modified by developers during the testing.

`Build_Config` is the reserved name of slot global variable, the developers can get the slot build config information from this global variable, and it can modified by developers during the testing. It may be need to read from SFC server and then write into this global variable. It is Diag software version.

`RunMode` is the reserved name of slot global variable, the developers can get the slot running mode information from this global variable. There are only two modes, operator mode and loop mode, for example: `RunMode=LOOP` or `RunMode=OPERATOR`

`Image_Name_Ver` is the reserved name of slot global variable, it can modified by developers during the testing. It is Device software image bundle name and version.

`Amazon_HW_Ver` is the reserved name of slot global variable, it can modified by developers during the testing. It is Hardware version e.g. HVT2.

`Route_check` is the reserved name of slot global variable, it can modified by developers during the testing. It is used to determine whether DUT should be tested at the current test station, If DUT is ready for testing on current station, its value must be set to 1. Otherwise, its value must be set to 0. Figure 41 shows a SMT test flow, there are three stations, and the test flow of DUT1 is ISP -> BFT -> BRF, so the value of `Route_check` is 1 for DUT1. The test flow of DUT2 is ISP -> BRF, maybe due to the operator's wrong operation, BFT was skipped, and so when DUT2 tries to start testing at BRF, the test sequence should fail and prompt the operator that DUT2 is not ready to test at BRF station, because it skipped the BFT station, the value of `Route_check` should be set to 0.

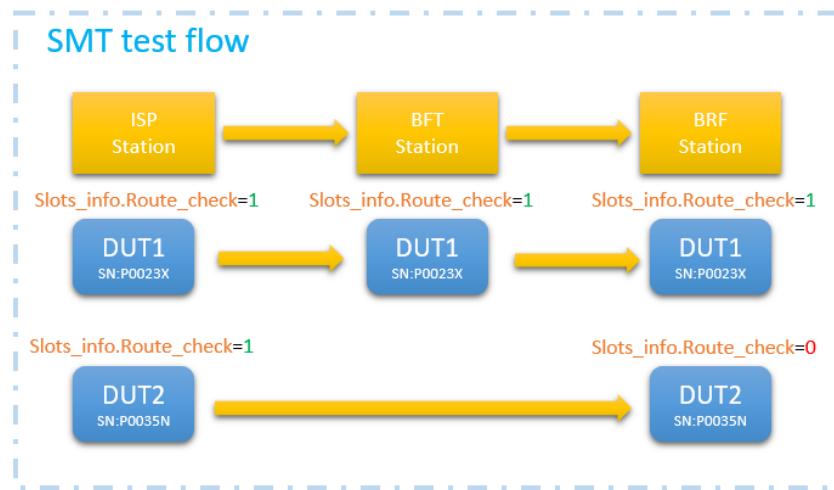


Figure 41

`TestPlan_CRC` is the reserved name of slot global variable, its value is MD5 value of the TestPlan, it is read-only

`TestPlan_ip` is the reserved name of slot global variable, its value is IP percentage of the TestPlan, it is read-only

`Current_CT` is the reserved name of slot global variable, its value is the time when Kazam ran to this step, it is read-only

`OverallResult` is the reserved name of slot global variable, its value is the overall result before this step, it is read-only, and it is used to get the status of the current test

`Result.XXX` is the reserved name of slot global variable, `XXX` is the step name, the developer need to replace it with the real step name, it can be used to get the results of a specified step

`Value.XXX` is the reserved name of slot global variable, `XXX` is the step name, the developer need to replace it with the real step name, it can be used to get the return string of a specified step

- Hardware global variable, this variable can be used to get the information defined by the developer in the hardware configuration file, it is read-only. It can be defining as below:

`HD_info.XXX`

`HD_info` is the keyword, cannot be changed.

`XXX` is the name of the global variable, it should be defined in the hardware configuration file, the developers can customize.

For example, the developer defined the value of DUTCOM in Slot1 as 3 in HardwareCFG file, if there is the "Function=Open |COM=<<HD\_info.DUTCOM>> |BaudRate=115200" in the "input parameter" of testplan, Kazam will replace <<HD\_info.DUTCOM>> to 3, so the "input parameter" of testplan will be changed to "Function=Open |COM=3 |BaudRate=115200" when the driver/test vi received it.

And using hardware variables to fill "Test Vi Name", it can assign a different driver VI to each slot. The following is example.

The screenshot shows two Microsoft Excel spreadsheets side-by-side.

**Top Spreadsheet: HardwareCFG.csv**

	A	B	C	D	E	F	G
1	CheckValue	82697951840d2691b9cf2fca3dee95ef					
2		Slot1	Slot2	Slot3	Slot4	Slot5	Slot6
3	DUTCOM		3	41	47	45	
4	PSCOM		3	40	46	44	
5	GPIOCOM		3	40	46	44	
6	FAN_CTL	P0.0	P0.0	P0.0	P0.0		
57	Robot	TST_Wait	TST_Inout				
58							
59							
60							

**Bottom Spreadsheet: DemoSequence1.0.1.csv**

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	CheckVal	2af0ff55cae1bd9fease1dabfd4abd																							
2	Index	Enable	PREREQ	Test Name	VI Name	LSL	USL	Unit	Limit	Chek	Save	Resu	Property	Dynamic	Casesensi	Input	Parameter	Retry	Next Step	Next Test	Next Goto	Cur ErrorCode	Note		
26	6	1	wait14	<>HD_info.Robot>>	80	120		1	0	YES			wait>100 slots_info>n% sys_infopanel					0	Next			Error-010			
27																									
28																									
29																									

Figure 42

- Bench configuration global variable, this variable can be used to get the information defined by the developer in the bench configuration file, it is read-only. It can be defining as below:

Bench\_CFG.XXX

Bench\_CFG is the keyword, cannot be changed.

XXX is the name of the global variable, it should be defined in the bench configuration file, the developers can customize.

For example, the developer defined the value of Factory\_ID in Bench\_Default\_Config.csv file, if there is the "Function=GetName|FactoryName=<>Bench\_CFG.Factory\_ID>>" in the "input parameter" of testplan, Kazam will replace <>Bench\_CFG.Factory\_ID>> to LAB, so the "input parameter" of testplan will be changed to "Function=GetName|FactoryName=LAB" when the driver/test vi received it.

	A	B	C
1	CheckValue	4c3b2095caee309b4c2a6e22b9890285	
2	Item	Detail	
3	BenchShellPath	Demo\Prj_CFG\shell_factory.csv	
4	BenchHardwareConfigurationPath	Demo\Prj_CFG\HardwareCFG.csv	
5	SearchPath	Project\Demo	
6	Project Line		1
7	Test Station Name	BFT	
8	Fixture Name		1
9	Product Name	Demo	
10	Factory_ID	LAB	
11	MinSizeOfShowResult		2
12	TCPPort		7200
13	AutoUpdate		
14	Emailcontact	user1@server.com/user2@server.com	
15	Textcontact	phonenumber1/phonenumber2	
16	LOG_PATH	..\..\LOG\C\Kazam;	
17	KazamUpdateServer		
18	ProjectUpdateServer		
19	ProjectUpdatePackageName	BFT	
20	FileUpload_Path	..\..\LOG\\192.168.0.1\TestLog	
21	GUIEvent		2020
22	GUIData		2021
23	SOP_PATH		
24	SequenceUploadPath	C:\Users\feih\Downloads\Ktest	
25			
26			

Figure 43

### 6.3 Development steps

1. Develop the driver vi, using the DRV\_template.vi
2. Develop the test vi, using the TST\_template.vi
3. Develop the log vi, using the log\_Template.vi (If developers need special log)
4. Edit hardware configuration file
5. Edit TestPlan
6. Edit shell configuration file
7. Edit bench configuration file
8. Preview your project before build, if there are any errors here, it will tell you, as Figure 44
9. Build the Labview project, as Figure 45
10. Select the source folder, it is the compilation path for the supplier project, as Figure 46
11. Select the target folder, it is the path of Kazam.exe folder, as Figure 47

Note: All CSV and XLSX files need to be closed before compiling, such as TestPlan, shell configuration file, etc

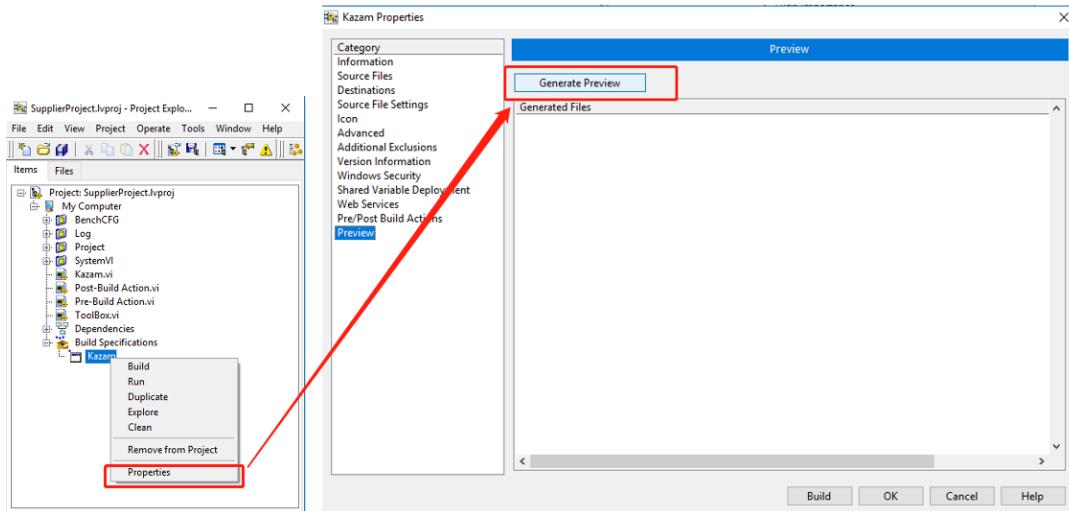


Figure 44

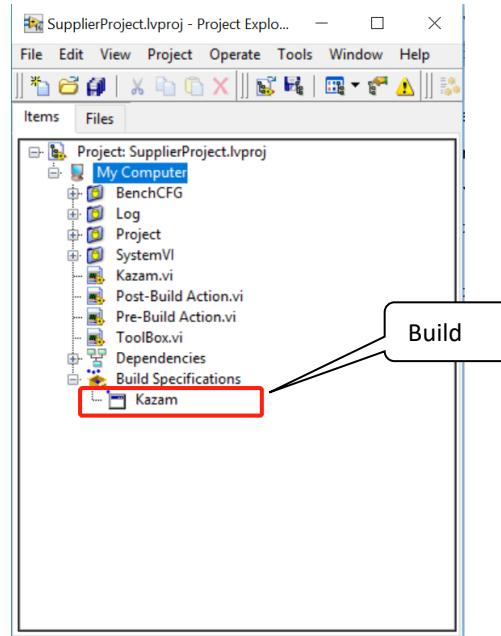


Figure 45

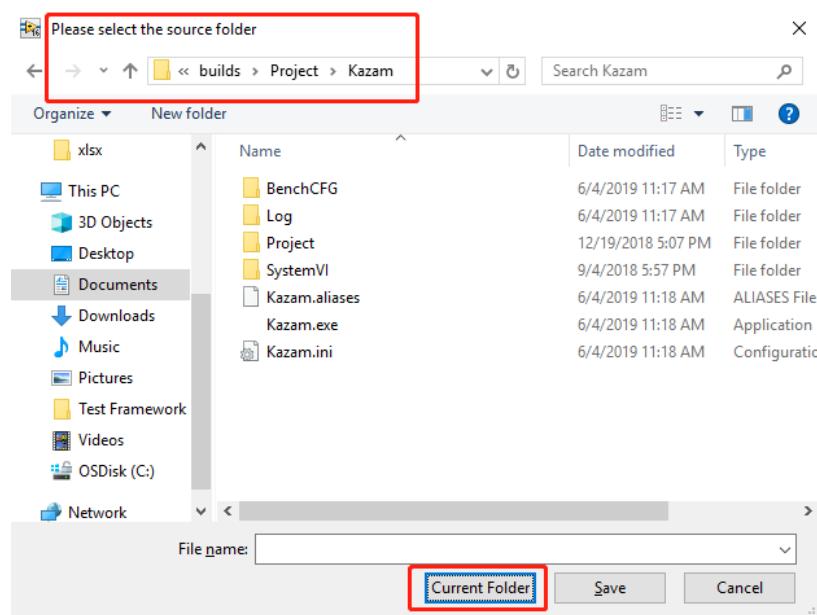


Figure 46

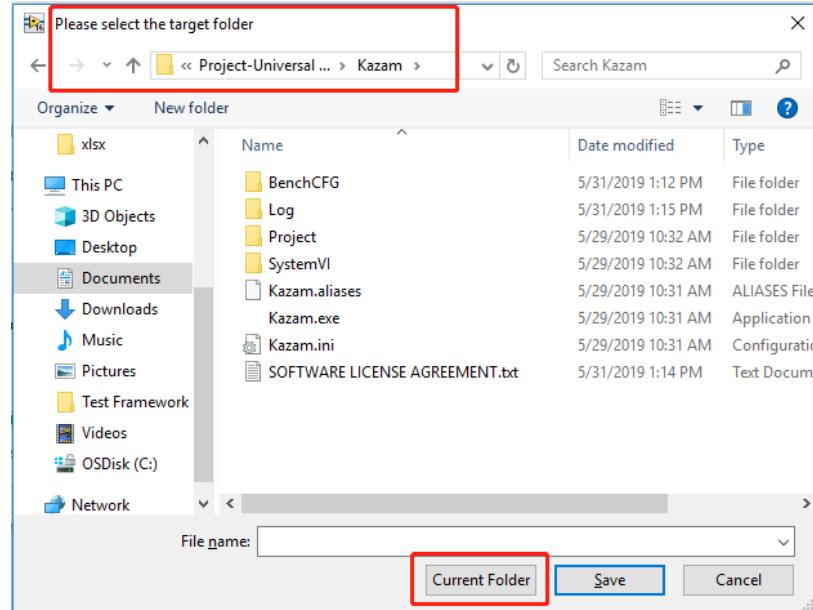


Figure 47

## 6.4 Labview Run-time Version

The developers need to download the “LabVIEW Run-Time Engine 2016 - (64-bit) – Windows” from the following website:

<http://www.ni.com/download/labview-run-time-engine-2016/6067/en/>

The screenshot shows a software download page for the LabVIEW Run-Time Engine 2016. At the top, it displays the product name: "LabVIEW Run-Time Engine 2016 - (64-bit) - Windows - Windows 10/8.1 64-bit/7 (SP1) 64-bit/Server 2012 R2 64-bit/Server 2008 R2 64-bit". Below this, there are rating details: "49 Ratings | 3.86 out of 5 | Print". A section titled "Available Downloads:" contains a "Browser Download" button. Underneath, a "Download Link" is provided: "LVRT2016\_f5Patch-64std.exe". A "To get started:" list includes three steps: clicking the download link, launching the installer, and following onscreen prompts. File size information ("Filesize: 293.18 MB") and checksum details ("Checksum (MD5): 6ebf6aa18d09aef0ddf60a3aeb706351") are also shown. At the bottom, there's a note about updates and notifications.

**LabVIEW Run-Time Engine 2016 - (64-bit) - Windows - Windows 10/8.1 64-bit/7 (SP1) 64-bit/Server 2012 R2 64-bit/Server 2008 R2 64-bit**

49 Ratings | 3.86 out of 5 | [Print](#)

**Available Downloads:**

**Browser Download**

Download Link: [LVRT2016\\_f5Patch-64std.exe](#)

**To get started:**

- Click the **Download Link** link above.
- Your browser will begin downloading the standalone installer for your software.
- Once the standalone installer has been downloaded, launch the executable and follow the onscreen prompts to complete the installation of your software.

**Filesize:** 293.18 MB  
**Checksum (MD5):** 6ebf6aa18d09aef0ddf60a3aeb706351

**Updates and Notifications:**  
Critical Updates and Security Notifications are posted on ni.com. Before downloading, click [here](#) to review this information.

Figure 48

## 6.5 Other language or EXE/Python/Bat file support

### 6.5.1 EXE/Python/Bat file support

The platform uses the Windows command line to call the executive file, py, pyc and bat file. Kazam will call it in the following format:

```
cmd /c python \found_path_2_file\testfunction.py %args%|Timeout=30000|WorkingDir= found_path_2_file|wait4return=1  
cmd /c python \found_path_2_file\testfunction.pyc %args%|Timeout=30000|WorkingDir= found_path_2_file|wait4return=1  
cmd /c python \found_path_2_file\testfunction.exe %args%|Timeout=30000|WorkingDir= found_path_2_file|wait4return=1  
cmd /c python \found_path_2_file\testfunction.bat %args%|Timeout=30000|WorkingDir= found_path_2_file|wait4return=1
```

%args% is used to set parameters, and it can be wrote in the input parameters, its keyword is args, for example: args= --port com4 --mode 0

Timeout is used to set a timeout to wait for a return, the default value is 30000 millisecond, and it can be overridden in the input parameters

WorkingDir is used to set the script path, the default path is the path of the project folder, and it can be overridden in the input parameters

wait4return is used to set whether to wait for a return, if this step is a non-blocking step, the default value is 0, if this step is a blocking step, the default value is 1, 1 means that it will wait for return, 0 means that it will not wait for return, and it can be overridden in the input parameters

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U			
1	CheckVal:752a7bc52dcbcd1585412999ed5cf07																						
2	Index	Enable	PREREQ	RunInParas	Test Name	Test Vi Name	LSL	USL	Unit	Limit	Ches	Save	Resu	Property	Dynamic	Casesensi	Input Parameter	Retry	Next Step	Next Test	Goto Cour	Error Code	Note
3	1	1			Delay_1	TST_delay.pyC				0	normal	0	Yes					args=Delay=2000	0	next		Error-001	
4	1	1			Delay_2	TST_delay.py				0	normal	0	Yes					args=Delay=1000 WorkingDir= D:\Kazam wait4return=1	0	next		Error-002	
5	1	1			Wait_1	TST_Wait.exe				0	normal	0	Yes					args=Wait=1000	0	next		Error-003	
6	1	1			Wait_2	TST_Wait.bat				0	normal	0	Yes					args=Wait=1000 Timeout=30000 wait4return=1	0	next		Error-004	

Figure 49

As Figure 49, this is demo testplan, the developers can directly call exe, python, bat files in "Test Vi Name", and set their parameters in "Input Parameter"

### 6.5.2 C/C++ DLL support

The platform can call the C or C++ DLL directly. The DLL function define as below:

```
CStr AME_Function(const CStr ParametersIn);
```

AME\_Function is the function name, it can't be changed, and the type of function return value is string, the format is same as executive file return value.

const CStr ParametersIn is the input of the function, the parameter name can't be changed, and the type of input parameters is string, the format is same as executive file return value

### 6.5.3 C# source code support

The platform can call the C# source code in testplan directly. And the developer can use Microsoft Visual Studio to debug their source code directly. The following is a C# code demo:

```
using System;
/*this name can be changed as you defined*/
namespace AAA
{
    /*Class name cannot be changed*/
    public class AME_TestScripture
    {
        /*Function name cannot be changed*/
        public string AME_Function(string para1, int para2)
        {
            string feedback = "";
            try
            {
                feedback ="Result=" + para1 + " | para2=" + para2.ToString();
            }
            catch (Exception ex)
            {
                throw(ex);
            }
            return feedback;
        }
    }
}
```

AAA is the namespace, it can be changed by the developer.

AME\_TestScripture is the class name, it cannot be changed.

AME\_Function is the function name, it cannot be changed. Its parameters can be defined by developer, Kazam supports common parameter types, such as string, integer, floating point, etc.

para1 and para2 are the input parameters of AME\_Function, the developer can add, delete, and modify the input parameter by themselves. And the corresponding parameters must be defined in the “input parameter” of the testplan. Figure 49 is a demo, the testplan calls a c# soure code, and there are two parameters in the “input parameter” of the testplan, Kazam will automatically find para1 and para2 and assign the function parameters to AME\_Function.

The screenshot shows a dual-pane development interface. On the left is a code editor with a C# file named `TST_SampleScripture.cs`. The code defines a class `AAA` with a public method `AME_Function` that takes two parameters and returns a string. On the right is an Excel spreadsheet titled `GlobalVariableTestingDemo.xlsx`. The spreadsheet contains a single sheet with columns labeled A through V. Row 2 contains the header row with column titles: `CheckVal`, `Index`, `Test Name`, `Test VI Name`, `LSL`, `USL`, `Unit`, `LimitChec`, `Save Result`, `Property`, `Dynamic`, `Casesensi`, `Input Parameter`, `Retry`, `Next`, `Next Got Error`, and `Error Code Note`. Row 3 contains data: Index 1, Test Name `Random2`, Test VI Name `TST_SampleScripture.cs`, LSL 0, USL 0, Unit normal, LimitChec 0, Save Result Yes, Property 0, Dynamic Yes, Casesensi 0, Input Parameter `para1\abcde|para2\123456`, Retry 0, Next next, Next Got Error 0, and Error Code Note Error-011. The cell containing the value `para1\abcde|para2\123456` in the `Input Parameter` column is highlighted with a red border.

```

1  using System;
2
3  /*this name can be changed as you defined*/
4  namespace AAA
5  {
6      /*Class name cannot be changed*/
7      public class AME_TestScripture
8      {
9          /*Function name cannot be changed*/
10         public string AME_Function(string para1, int para2)
11         {
12             string feedback = "";
13             try
14             {
15                 feedback = "Result=" + para1 + "|" + para2.ToString();
16             }
17             catch (Exception ex)
18             {
19                 throw(ex);
20             }
21             return feedback;
22         }
23     }
24 }

```

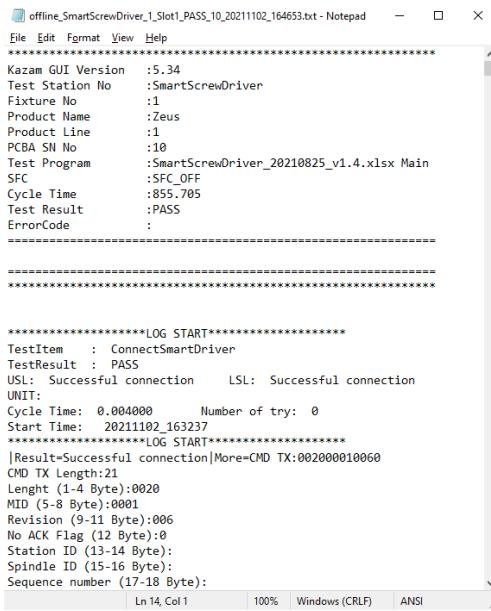
Figure 50

## 7. Data log

The platform contains three types of standard log formats, single log, time log and daily log. As mentioned in the previous chapter, the platform supports custom log formats, the developers can follow the log interface specification to develop their own log format.

### 7.1 Single Log

Single log is used to record information for all steps, it is in TXT format. The following is an example of a single log:



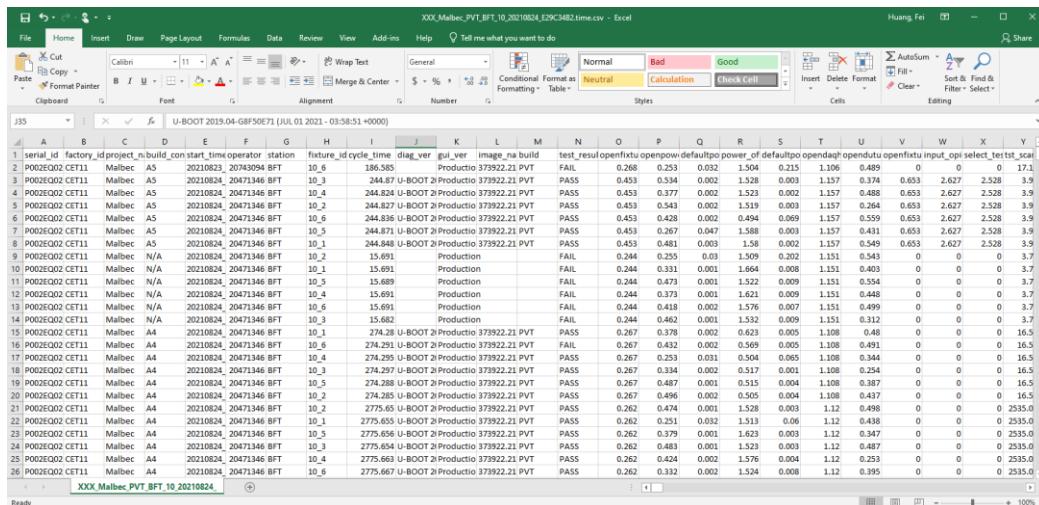
```
offline_SmartScrewDriver_1_Slot1_PASS_10_20211102_164653.txt - Notepad
*****
Kazam GUI Version :5.34
Test Station No :SmartScrewDriver
Fixture No :1
Product Name :zeus
Product Line :1
PCBA SN No :10
Test Program :SmartScrewDriver_20210825_v1.4.xlsx Main
SFC :SFC_OFF
Cycle Time :855.705
Test Result :PASS
ErrorCode :

*****
*****LOG START*****
TestItem : ConnectSmartDriver
TestResult : PASS
USL: Successful connection LSL: Successful connection
UNIT:
Cycle Time: 0.004000 Number of try: 0
Start Time: 20211102_163237
*****LOG START*****
|Result=Successful connection|More=CMD TX:002000010060
CMD TX Length:21
Length (1-4 Byte):0020
MID (5-8 Byte):0001
Revision (9-11 Byte):006
No ACK Flag (12 Byte):0
Station ID (13-14 Byte):
Spindle ID (15-16 Byte):
Sequence number (17-18 Byte):
Ln 14 Col 1 100% Windows (CRLF) ANSI
```

Figure 51

### 7.2 Time Log

Time log is used to record cycle time for all steps, it is in csv format. The following is an example of a time log:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	
1	serial	factory_id	project_n	build	conn_start_time	operator	station	fixture_id	cycle_time	diag_ver	gua_ver	image_no	build	test_res	openfxtu	openpovn	defaultpov	power_of	defaultpov	openqaq	openpudtu	openfntu	input_opei	select	test	sca
2	P002E002	CET11	Malbec	A5	20210823	20473094	BFT	10_6	186.585	Product0	37932.21	PVT	FAIL	0.26	0.253	0.032	1.504	0.215	1.106	0.489	0	0	0	0	17.1	
3	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_6	15.691	Product0	37932.21	PVT	PASS	0.26	0.533	0.001	1.504	0.215	1.106	0.489	0.001	0.001	0.001	0.001	3.9	
4	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_4	244.824	U-BOOT 3	Product0	37932.21	PVT	PASS	0.453	0.377	0.001	1.520	0.001	1.157	0.489	0.051	2.037	2.528	0	0
5	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_2	244.827	U-BOOT 3	Product0	37932.21	PVT	PASS	0.453	0.543	0.002	1.519	0.001	1.157	0.264	0.053	2.037	2.528	0	0
6	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_6	244.836	U-BOOT 3	Product0	37932.21	PVT	PASS	0.453	0.428	0.002	0.494	0.069	1.157	0.559	0.053	2.037	2.528	0	0
7	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_5	244.871	U-BOOT 3	Product0	37932.21	PVT	PASS	0.453	0.267	0.047	1.588	0.003	1.157	0.411	0.053	2.037	2.528	0	0
8	P002E002	CET11	Malbec	A5	20210824	20473346	BFT	10_1	244.848	U-BOOT 3	Product0	37932.21	PVT	PASS	0.453	0.481	0.003	1.58	0.002	1.157	0.549	0.053	2.037	2.528	0	0
9	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_2	15.691	Production			FAIL	0.244	0.255	0.03	1.509	0.202	1.151	0.543	0	0	0	0	3.7	
10	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_1	15.691	Production			FAIL	0.244	0.331	0.001	1.664	0.008	1.151	0.403	0	0	0	0	3.7	
11	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_5	15.688	Production			FAIL	0.244	0.473	0.001	1.522	0.009	1.151	0.554	0	0	0	0	3.7	
12	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_4	15.691	Production			FAIL	0.244	0.373	0.001	1.621	0.009	1.151	0.448	0	0	0	0	3.7	
13	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_6	15.691	Production			FAIL	0.244	0.418	0.001	1.507	0.007	1.151	0.409	0	0	0	0	3.7	
14	P002E002	CET11	Malbec	N/A	20210824	20473346	BFT	10_3	15.691	Production			FAIL	0.244	0.404	0.001	1.510	0.009	1.151	0.512	0	0	0	0	3.7	
15	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_1	274.284	U-BOOT 3	Product0	37932.21	PVT	PASS	0.367	0.378	0.002	0.603	0.001	1.108	0.490	0	0	0	0	16.5
16	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_6	274.291	U-BOOT 3	Product0	37932.21	PVT	FAIL	0.367	0.432	0.002	0.569	0.005	1.108	0.491	0	0	0	0	16.5
17	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_4	274.295	U-BOOT 3	Product0	37932.21	PVT	PASS	0.267	0.253	0.011	0.504	0.065	1.108	0.344	0	0	0	0	16.5
18	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_3	274.297	U-BOOT 3	Product0	37932.21	PVT	PASS	0.267	0.334	0.002	0.517	0.001	1.108	0.254	0	0	0	0	16.5
19	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_5	274.288	U-BOOT 3	Product0	37932.21	PVT	PASS	0.267	0.487	0.001	0.515	0.004	1.108	0.387	0	0	0	0	16.5
20	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_2	274.285	U-BOOT 3	Product0	37932.21	PVT	PASS	0.267	0.496	0.002	0.505	0.004	1.108	0.437	0	0	0	0	16.5
21	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_2	277.65	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.474	0.001	1.528	0.003	1.12	0.498	0	0	0	0	235.0
22	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_1	277.651	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.251	0.032	1.513	0.006	1.12	0.438	0	0	0	0	235.0
23	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_3	277.651	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.379	0.001	1.623	0.003	1.12	0.347	0	0	0	0	235.0
24	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_3	277.669	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.463	0.001	1.509	0.004	1.12	0.487	0	0	0	0	235.0
25	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_4	277.669	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.424	0.002	1.576	0.004	1.12	0.253	0	0	0	0	235.0
26	P002E002	CET11	Malbec	A4	20210824	20473346	BFT	10_6	277.667	U-BOOT 3	Product0	37932.21	PVT	PASS	0.262	0.332	0.002	1.524	0.008	1.12	0.395	0	0	0	0	235.0

Figure 52

## 7.3 Daily Log

Daily log is used to record all data in a day, it is in csv format. Figure 53 is an example of a daily log, the part marked by the red box is called the fixed header.

1	Emailcontact	:user1@server.com	/user2@server.com Textcontact	:phonenumber1/	phonenumber2																	
2	Errorcode																					
3	Upperlimit																					
4	Lowerlimit																					
5	Units																					
6	serial_id	factory_id	project_nibuild	conn_start_time	operator	station	fixture_id	cycle_time	diag_ver	gui_ver	route_chassis	image_na	build	test_result	get_sn	sfc_check	sfc_show	sfc_show_sfcshow	sfc.show_sfcshow	sfc.show_sfcshow_sfc_pc		
7	P002EQ02_CET11	Malbec	N/A	20210821	20743094	BFT	10_5	13443.2	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C482:80F7C489 13CE5F5H/N/A	N/A	N/A	2.1.5.31	2.1.5.
8	P002EQ02_CET11	Malbec	N/A	20210821	20743094	BFT	10_1	13443.2	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBB99:EC2EBE80 QLBDSO75/N/A	N/A	N/A	2.1.5.31	2.1.5.
9	P002EQ02_CET11	Malbec	N/A	20210821	20743094	BFT	10_3	13443.22	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBBDC:EC2EBE807 VXON075/N/A	N/A	N/A	2.1.5.31	2.1.5.
10	P002EQ02_CET11	Malbec	N/A	20210821	20743094	BFT	10_4	13443.22	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBB99:EC2EBE87A 6ZQ9T7AF/N/A	N/A	N/A	2.1.5.31	2.1.5.
11	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_6	13443.2	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	0CA43F96E:80F7CD0 OTPB2ZYII/N/A	N/A	N/A	2.1.5.31	2.1.5.
12	P002EQ02_CET11	Malbec	N/A	20210821	20743094	BFT	10_2	13443.2	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C4FF:FC2E8BBA 21NV27PN/N/A	N/A	N/A	2.1.5.31	2.1.5.
13	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_3	270.412	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C006:EC2EBE880 9XB9V4W/N/A	N/A	N/A	2.1.5.31	2.1.5.
14	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_4	270.406	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBBCC:EC2EBE800 PRAXTRFK/N/A	N/A	N/A	2.1.5.31	2.1.5.
15	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_2	270.411	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBB5D:EC2EBE826 M721YDE0I/N/A	N/A	N/A	2.1.5.31	2.1.5.
16	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_5	270.41	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBE0D:80F7CAFJ FXUUOXGU/N/A	N/A	N/A	2.1.5.31	2.1.5.
17	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_1	270.422	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C4AB:043F98L 9FMHZSTT/N/A	N/A	N/A	2.1.5.31	2.1.5.
18	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_6	270.413	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	806D7153:EC2EBE843 R200RF90/N/A	N/A	N/A	2.1.5.31	2.1.5.
19	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_5	4708.419	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C4FE:FC2E8BBA 21NV27PN/N/A	N/A	N/A	2.1.5.31	2.1.5.
20	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_4	4708.432	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	A0D2B1EE:84E45422 PW4DK18/N/A	N/A	N/A	2.1.5.31	2.1.5.
21	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_6	4708.417	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C4D4:EC2EBE800 ZVWA4DF/N/A	N/A	N/A	2.1.5.31	2.1.5.
22	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_3	4708.429	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBE09:EC2EBE8DE GX52F611/N/A	N/A	N/A	2.1.5.31	2.1.5.
23	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_1	4708.442	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	EC2EBE96:AO0D2B194 0GN352V/N/A	N/A	N/A	2.1.5.31	2.1.5.
24	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_2	4708.416	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	806D7194:80F7C478 MT57OKQ/N/A	N/A	N/A	2.1.5.31	2.1.5.
25	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_6	276.839	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	B0F7C43C:EC2EBE8AC G6YH0G2/N/A	N/A	N/A	2.1.5.31	2.1.5.
26	P002EQ02_CET11	Malbec	N/A	20210822	20743094	BFT	10_5	276.863	U-BOOT	2i	Productio	1	37992.21	PVT	PASS	P002EQ02	PASS	806D7136:B0F7C435 12E9YST0C/N/A	N/A	N/A	2.1.5.31	2.1.5.

Figure 53

### 7.3.1 Fixed Header

The fixed header can be customized by the developer, and it can be defined in shell configuration file, Figure 54 shows how to define the developer's own fixed header file, it is similar to define a testplan.

FixedHeader is the keyword for the default fixed header sequence, if define “FixedHeader1= BBB.csv”, it means that slot 1 will use this “BBB.csv” as its fixed header sequence

If define “FixedHeader2= CCC.csv”, it means that slot 2 will use this “CCC.csv” as its fixed header sequence

And so on.

3	SEQ	Resource_Nam	VI_NAME	CONTROL_STRING	D	COMMENTS
4		Sequence	sys_SequenceCFGReadout.vi	FixedHeader=AAA.csv FixedHeader1=BBB.csv TestPlan=GlobalVariableTestingDemo.xlsx ABDE TestPlan1=Loop_Test.csv WaitUntilDone=True EnableParameter=HVT;REL;		
5		Sequencer	sys_SequenceLauncher.vi			
6		Display	GUI_Single1024x768.vi	[FP_Show=True RefreshTime=500 Real-time=1 SFC_menu=enable SFC=Off StartButton=On InitPicture=SystemVI sys_GUISourceCode GU_Act		
7		Datalog	amz_log_Daily.vi	LOG_FINAL LOG_PATH=..\..\LOG\ Module=WAN[3/3]:AF[155]:WTN[3/3]:4RU/99-Else=WIFI  Emailcontact:user1@server.com Textcontact:phonenumber1/		
8		Datalog	amz_log_single.vi	LOG_FINAL LOG_PATH=..\..\LOG\ Module=WAN[3/3]:AF[155]:WTN[3/3]:4RU/99-Else=WIFI		
9		Datalog	amz_log_timeLog.vi	LOG_FINAL LOG_PATH=..\..\LOG\ Module=WAN[3/3]:AF[155]:WTN[3/3]:4RU/99-Else=WIFI		
10	EOF					
11						
12						

It is a user-defined default fixed header file      It is a user-defined fixed header file for slot1

Figure 54

If the developer does not define the fixed header file in shell configuration file, Kazam will automatically call the default fixed header file for each slot, its default path is Kazam\SystemVI\sys\_Configuration\KazamFixedHeader.csv

Figure 55 shows the default fixed header sequence, its format is the same as other sequences, the only difference is the index definition, the platform uses F to identify that this row is a fixed header row, F1 means that the name of this step is the name of the first fixed column in daily log, F2 means that the name of this step is the name of the second fixed column in daily log, F3 means that the name of this step is the name of the third fixed column in daily log ,and so on.

1	CheckVal:1bc5a35945118b03fe36b01789fadf5	PR	Test Name	Function	(L,SL	USL	Unit	LimitChec	Save Resu	Property	Dynamic	Casesensi	Input Parameter											
2	Index	Enable										0	0	Result=<>Slots_info.SN>>										
3	F1	1	serial_id	EMPTY	EMPTY							0	0	Function<CheckFactory> StringInput=<>Bench_CFG.Factory_ID>>										
4	F2	1	factory_id	TST_Form	IFAIL	IFAIL						0	0	Result=<>Bench_CFG.Product Name>>										
5	F3	1	project_name	EMPTY	EMPTY							0	0	Result=<>Slots_info.Build_Config>>										
6	F4	1	build_config	EMPTY	EMPTY							0	0	Result=<>Slots_info.start_time>>										
7	F5	1	start_time									0	0	Result=<>SYS_info.Operator>>										
8	F6	1	operator	EMPTY	EMPTY							0	0	Function<CheckStation> StringInput=<>Bench_CFG.Test Station Name>> RegExp= ignoreCase=TRUE										
9	F7	1	station	TST_Form	IFAIL	IFAIL						0	0	Result=<>Bench_CFG.Fixture Name>>										
10	F8	1	fixture_id	[1 2 3 4 5 1 2 3 4 5 6 7 8 9 10 11 12								0	0	Result=<>Slots_info.Diag_Ver>>										
11	F10	1	diag_ver	EMPTY	EMPTY							0	0	Result=<>Bench_CFG.Kazam_Core_Ver>>										
12	F11	1	gui_ver	EMPTY	EMPTY							0	0	Result=<>Slots_info.Route_check>>										
13	F12	1	route_check	0 1	0 1							0	0	Result=<>Slots_info.Image_Name_Ver>>										
14	F13	1	image_name_ver	EMPTY	EMPTY							0	0	Function<CheckBuild> StringInput=<>Slots_info.Amazon_HW_Ver>> RegExp=										
15	F14	1	build	TST_Form	IFAIL	IFAIL						0	0	Result=<>Slots_info.testplan_CRC>>										
16	F16	1	testplan_crc									0	0	Result=<>Slots_info.testplan_IP>>										
17	F17	1	testplan_ip									0	0	Result=<>Slots_info.current_ct>>										
18	F9	1	cycle_time									0	0	Result=<>Slots_info.overallresult>>										
19	F15	1	test_result									0	0											
20																								
21																								

Figure 55

## 8. External API

The platform communicates with external software using TCP/IP, The default port is 7200, and it can be modified by developer in Bench\_Default\_Config.csv file. Kazam work in server mode, external software need to access using TCP clients.

### 8.1 Command summary

The command format is as follows:

`CMD=XXX | RequiredParameter=YYY | OptionalParameter=ZZZ\r\n`

`CMD` is keyword of an external command, it can't be changed.

`XXX` is a command, can find them in the command columns of the table below.

`RequiredParameter` is a must keyword, can find them in the required parameters columns of the table below, some commands do not require this parameter.

`YYY` is the value of the required parameter.

`OptionalParameter` is the keyword of optional parameter, the developers can decide whether to need them.

`ZZZ` is the value of the optional parameter.

`\r\n` means CR (carriage return) followed by a LF (linefeed), it is end char.

**Note1:** All commands and parameters are not case sensitive.

**Note2:** There is already a vi to refer, and its name is DRV\_EAPI.vi

Command	Required parameters	Optional Parameter	Description
StartAll		PanelSN	Start all the tests
StartOne	Slot	SN	Start the test of one slot
SetMode	Mode=Operator Mode=Loop		Set the test mode
SetSFC	SFC=ON SFC=OFF		Set the SFC
SetLoopCounter	LoopCounter=100		Set the number of loops.

ForceExit			Forced exit Kazam.exe
ReadSlotResults	Slot		Read all the results of a slot and Kazam running status
ReadStepResults	Slot, TestName		Read the result of the specified step and Kazam running status

For example:

External software send the following commands:

**CMD= StartAll|PanelSN=30975**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

### 7.1.1 Commands detail description

**StartAll:** This command will make all the slot to begin testing.

External software send the following commands:

**CMD= StartAll|PanelSN=30975**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

**PanelSN** is used to tell the platform the panel serial number.

---

**StartOne:** This command will make one slot to begin testing.

External software send the following commands:

**CMD= StartOne|Slot=1|SN=12345678**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

**Slot** is used to tell the platform which slot is selected. Its range is from **1** to **32**.

**SN** is used to tell the platform the serial number of this slot.

---

**SetMode**: This command sets the running mode of the platform, the loop mode, or the operator mode.

External software send the following commands:

**CMD= SetMode|Mode=Operator**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

**Mode** is used to tell the platform which mode is selected. Its value is **loop** or **operator**.

---

**SetSFC**: This command will set the state of the SFC, online or offline.

External software send the following commands:

**CMD= SetSFC|SFC=ON**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

**SFC** is used to tell the platform whether to test online or offline. Its value is **ON** or **OFF**.

---

**SetLoopCounter**: This command will set the number of loop.

External software send the following commands:

**CMD= SetLoopCounter|LoopCounter=100**

If no error, the platform will return back “Successful”

If there is error, the platform will return back the error information

**LoopCounter** is used to tell the platform how many times it needs to loop. Its value is **-1** or **1~2147483648**, “**-1**” means forever.

---

**ForceExit**: This command is used to force an exit from the platform.

External software send the following commands:

**CMD=ForceExit**

This command does not return a value.

---

**ReadSlotResults:** This command will read all the results of a slot and Kazam running status.

External software send the following commands:

**CMD= ReadSlotResults|Slot=1**

If no error, the platform will return back all the results of this slot, and it is **JSON string**, the developer can use “Unflatten From JSON” ( ) to parse it, or use other methods, more details, please reference Kazam\Project\Library\EAPI\EAPI\_GetTestResults.vi

If there is error, the platform will return back the error information

**Slot** is used to tell the platform which slot is selected. Its range is from **1** to **32**.

---

**ReadStepResults:** This command will read the result of the specified step and Kazam running status.

External software send the following commands:

**CMD= ReadStepResults|Slot=1|TestName=WaitGetSN**

If no error, the platform will return back all the results of this slot, and it is **JSON string**, the developer can use “Unflatten From JSON” ( ) to parse it, or use other methods, more details, please reference Kazam\Project\Library\EAPI\EAPI\_GetStepResults.vi

If there is error, the platform will return back the error information

**Slot** is used to tell the platform which slot is selected. Its range is from **1** to **32**.

**TestName** is used to tell the platform which test step is selected.

## 9. Important parameters

### 9.1 Bench\_Default\_Config.csv

Some important parameters are predefined in the Bench\_Default\_Config.csv file

A	B
1 CheckValue	f7189e5c21a1e516d93c5ea1fd733fdd
2 Item	Detail
3 BenchShellPath	Demo\Prj_CFG\shell_factory.csv
4 BenchHardwareConfigurationPath	Demo\Prj_CFG\HardwareCFG.csv
5 SearchPath	Project\Demo
6 Project Line	1
7 Test Station Name	Vnl
8 Fixture Name	xxx
9 Product Name	Demo
10 Factory_ID	LAB
11 MinSizeOfShowResult	10
12 TCPPort	7200
13 AutoUpdate	user1@server.com/user2@server.com
14 Emailcontact	phonenumber1/phonenumber2
15 Textcontact	..\..\LOG\;
16 LOG_PATH	BFT
17 KazamUpdateServer	
18 ProjectUpdateServer	
19 ProjectUpdatePackageName	
20	
21	

Figure 56

**Project Line:** This is used to define the name of project line. It will be used to define the file name of log.

**AutoUpdate:** This is used to set whether to automatically update the project folder. Its value is **ENABLE** or **disable**.

**Note:** Kazam will automatically check the “CheckValue” of each file when enabling the project folder to update

**ProjectUpdateServer:** This is used to define the project server path, when Kazam is running in exe mode, it automatically checks to see if there is an update in the update server.

**ProjectUpdatePackageName:** This is used to define the name of update package, when Kazam is running in exe mode, it automatically checks the name of update package to see if there is an update in the update server.

**KazamUpdateServer**: This is used to define the kazam server path, when Kazam is running in exe mode, it automatically checks to see if there is an update in the update server.

**Emailcontact**: It is used to set the mailbox for the contact. It will be writhed into the daily log.

**Textcontact**: It is used to set the phone number for the contact. It will be writhed into the daily log.

**LOG\_PATH**: This is used to set the storage path for log. It can set multiple paths, and isolate each path with a semicolon.

**BenchShellPath**: It is used to define the shell configuration file path.

**BenchHardwareConfigurationPath**: It is used to define the hardware configuration file path.

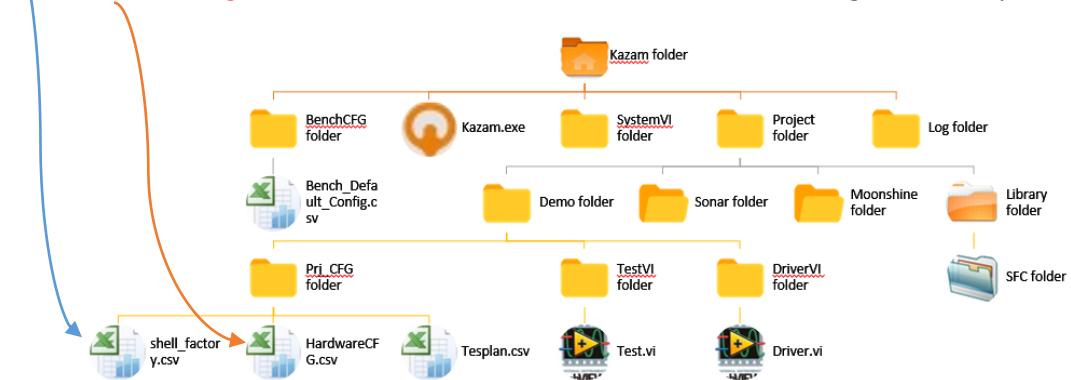


Figure 57

**SearchPath**: It is used to define search paths, it can be multiple paths, it allows developer to enter both the absolute path and the relative path, and isolate each path with a semicolon, for example:

SearchPath = Project\Demo;C:\kazam;\192.168.0.32\AFT\Code;

In this case, Kazma will first search for the "Project\Demo" path under the kazam.exe path, if kazam find a specific test plan, kazam will stop the search, otherwise will continue to search next path. The second step, kazam will start from the first search path search the driver and test vi.

**Test Station Name**: This is used to define the type of test station, such as AFT, Vnl, MT, PT, and so on. It will be writhed into the daily log

**Fixture Name**: This is used to define the fixture name, such as BFT01, BFT02, AFT01, and so on. It will be writhed into the daily log and will be used to define the file name of log.

**Product Name**: This is used to define the product name, such as Moonshine, Muffin, Sonar, and so on. It will be writhed into the daily log and will be used to define the file name of log.

**Factory\_ID**: This is used to define the factory name. Maybe in the future will be new rules. It will be writhed into the daily log.

**MinSizeOfShowResult**: This is used to define the length of show results. As Figure 58, there are 11 results need to show on GUI. So the length of show results is 11.

If the developer wants to control the “next step” of each show result, then **MinSizeOfShowResult** needs to be set to greater than 11, such as 12.

If the developer does not want to control the “next step” of each show result, the **MinSizeOfShowResult** needs to be set to less than 11, such as 10.

**MinSizeOfShowResult** is a very important parameter, please be careful to set it, otherwise the test time may become very long

DemoSequence1.0.0.1.csv - Excel															Huang, Fei Share					
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	CheckValuef8646e267c7497e8c6d435d96dd4685d																			
2	Index	Enable	PREREQ	Test Name	VI Name	LSL	USL	Unit	LimitCheck	SaveR	PrcDy	InputPara	Retry	Next Step	Next Test	Goto	Cour	ErrorCode	Note	
3	0	1	Random1	RandomSeed.dll		80	80		Random	0	0	max=100	0	Next						
4	1	1	Random1	GetResult:TST_LongArray		50	100 mA			0	0	number=2	0	Next				Error-999		
5	2	1	Show R1			1	100 mA		R1	1	0		0	Next				Error-1001		
6	3	1	Show R2			1	100 mA		R2	1	0		0	Next				Error-1002		
7	4	1	Show R3			1	100 mA		R3	1	0		0	Next				Error-1003		
8	5	1	Show R4			1	100 mA		R4	1	0		0	Next				Error-1004		
9	6	1	Show R5			1	100 mA		R5	1	0		0	Next				Error-1005		
10	7	1	Show R6			1	100 mA		R6	1	0		0	Next				Error-1006		
11	8	1	Show R7			1	100 mA		R7	1	0		0	Next				Error-1007		
12	9	1	Show R8			1	100 mA		R8	1	0		0	Next				Error-1007		
13	10	1	Show R9			1	100 mA		R9	1	0		0	Next				Error-1007		
14	11	1	Show R10			1	100 mA		R10	1	0		0	Next				Error-1007		
15	12	1	Show R11			1	100 mA		R11	1	0		0	Next				Error-1007		
16	13	1	Random1	GetResult:TST_LongArray		50	100 mA			0	0	number=2	0	Next				Error-999		
17	14	1	Show R1_1			1	100 mA		R1	1	0		0	Next				Error-1001		
18	15	1	Show R1_2			1	100 mA		R2	1	0		0	Next				Error-1002		
19	16	1	Show R1_3			1	100 mA		R3	1	0		0	Next				Error-1003		
20	17	1	Show R1_4			1	100 mA		R4	1	0		0	Next				Error-1004		
21	18	1	Show R1_5			1	100 mA		R5	1	0		0	Next				Error-1005		

Figure 58

**TCPPort:** This is used to define the TCP port. Its default port is 7200.

## 9.2 Kazam configuration file

Kazam configuration file is used to define the feature of Kazam, such as what log vi is used, what GUI is used, and so on. As Figure 59, it is a Kazam configuration file, see below for parameter definitions:

1	CheckValu	374e8e611f855213ed4c1a2a2095e76a		
2	RELEASE_DATE	=20170906	UTDP=0.01	
3	SEQ	Resource_Name	VI_NAME	CONTROL_STRING
4		Sequence_config	sys_SequenceCFGReadout.vi	TestPlan=DemoSequence1.0.0.1.csv WaitUntilDone=True
5		Sequencer	sys_SequenceLauncher.vi	SLOTS=4 SeqEngineName=TestSequence.vi WaitUntilDone=True
6		Display	GUI_Operator.vi	FP_Show=True RefreshTime=10000 Real-time=0 Minimizable=Disable SFC=ON delay=3000
7		Access Permissions	log_Get_Access_Permissions.vi	Server_PATH=\10.117.119.235\shareDoc\TestLog\Moonshine Name=Moonshine
8		Datalog	log_Daily.vi	LOG=FINAL LOG_PATH=\10.117.119.235\shareDoc\TestLog\Moonshine Module=WAN[3/3]:1AF/15
9		Datalog	log_xmilog.vi	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/15\$;WTN[3/3]:4RU/9I9-Else=Muffin
10		Datalog	log_single.vi	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/15\$;WTN[3/3]:4RU/9I9-Else=Muffin
11		Datalog	log_timelog.vi	LOG=FINAL LOG_PATH=..\..\LOG\Module=WAN[3/3]:1AF/15\$;WTN[3/3]:4RU/9I9-Else=Muffin
12		EOF		
13				

Figure 59

### 8.2.1 Common parameter

Common parameters can be used everywhere in Kazam configuration file.

Common parameter name	Description	Example
WaitUntilDone	<p>It is used to tell the platform to wait for the completion of this vi execution  <b>WaitUntilDone=True</b>, it means that the platform need to wait for the completion of this vi execution  Empty, it means that no need to wait for execution to end</p>	<b>WaitUntilDone=True</b>

### 8.2.2 Other parameter

For each function, the developer can define their parameter. Below is the parameters that the platform has defined

parameter name	Description	Example
sys_SequenceLauncher.vi		
SLOTS	It is used to tell the platform how many slots will be used	<b>SLOTS=4</b>
SeqEngineName	It can't be modified	<b>SeqEngineName=TestSequence.vi</b>

<b>sys_SequenceCFGReadout.vi</b> parameter name	Description	Example
TestPlan	It is used to tell the platform which testplan you want to use. <b>“TestPlan” is the default keyword for all slots</b>	<b>TestPlan</b> =DemoSequence1.0.0.1.csv
TestPlan1	It is used to tell the platform that the slot1 will use this testplan.	<b>TestPlan1</b> =Seq_Slot1.csv <b>TestPlan1</b> =Seq_Slot1.xlsx Sheet1
TestPlan2	It is used to tell the platform that the slot2 will use this testplan.	<b>TestPlan2</b> =Seq_Slot2.csv <b>TestPlan2</b> =Seq_Slot2.xlsx myTab
.....	.....	.....
TestPlan32	It is used to tell the platform that the slot32 will use this testplan.	<b>TestPlan32</b> =Seq_Slot32.csv
ResetPlan	It is used to tell the platform which reset plan you want to use.	<b>ResetPlan</b> =Seq_reset.csv
EnableParameter	It is used to set the enabling parameters that need to be enabled by default	<b>EnableParameter</b> =HVT;DVT;

<b>GUI_Operator.vi</b> parameter name	Description	Example
FP_Show	It is used to tell the platform that this GUI needs to be displayed	<b>FP_Show</b> =True
RefreshTime	It is used to set the refresh time of GUI, unit is a millisecond <b>The refresh time must set to 10 seconds for mass production</b>	<b>RefreshTime</b> =10000
Real-time	It is used to set whether test results are displayed in real time, its value is 0 or 1 <b>The Real-time must set to 0 for mass production</b>	<b>Real-time</b> =0
Minimizable	It is used to disable the GUI minimized, its value is Disable or Enable	<b>Minimizable</b> =Enable
SFC	It is used to tell the platform whether the SFC is ON or OFF by default	<b>SFC</b> =OFF
delay	In loop mode, it is used to set the delay between this run and the next	<b>delay</b> =3000

AutoScrolling	It is used to enable automatic scrolling, its value is Disable or Enable	<code>AutoScrolling=Enable</code>
ToolPath	It is used to specify the path of the developer's own tool vi	<code>ToolPath=Demo\TesterVI\TST_tool.vi</code> <code>ToolPath=C:\Tools\TST_tool.vi</code>
SFC_Menu	It is used to disabled the SFC menu, its value is Disable or Enable, default is enable	<code>SFC_Menu=Enable</code>
StartButton	It is used to disabled the start button, its value is Disable or Enable, default is enable	<code>StartButton=Enable</code>
DebuggingGUIPath	It is used to specify the path of the debugging tool vi	<code>DebuggingGUIPath=DebuggingTool.vi</code> <code>DebuggingGUIPath=C:\Tools\DebuggingTool.vi</code>
DebuggingToolCFG	It is used to specify the path of the debugging configuration file	<code>DebuggingToolCFG= DebuggingToolCFG.xlsx</code> <code>DebuggingToolCFG =C:\Tools\</code> <code>DebuggingToolCFG.xlsx</code>

<code>GUI_Single.vi</code> <code>GUI_Single1024x768.vi</code> <code>GUI_Single512x768.vi</code> parameter name	Description	Example
FP_Show	It is used to tell the platform that this GUI needs to be displayed	<code>FP_Show=True</code>
RefreshTime	It is used to set the refresh time of GUI, unit is a millisecond  <b>The refresh time must set to 10 seconds for mass production</b>	<code>RefreshTime=10000</code>
Real-time	It is used to set whether test results are displayed in real time, its value is 0 or 1  <b>The Real-time must set to 0 for mass production</b>	<code>Real-time=0</code>
Minimizable	It is used to disable the GUI minimized, its value is Disable or Enable	<code>Minimizable=Enable</code>
SFC	It is used to tell the platform whether the SFC is ON or OFF by default	<code>SFC=OFF</code>
delay	In loop mode, it is used to set the delay between this run and the next	<code>delay=3000</code>
SlotPosition	It is used to tell the platform the slot number for this GUI, its value is 1,2,3,...32	<code>SlotPosition=1</code>

AutoScrolling	It is used to enable automatic scrolling, its value is Disable or Enable	<b>AutoScrolling=Enable</b>
ToolPath	It is used to specify the path of the developer's own tool vi	<b>ToolPath=Demo\TesterVI\TST_tool.vi</b> <b>ToolPath=C:\Tools\TST_tool.vi</b>
SFC_Menu	It is used to disabled the SFC menu, its value is Disable or Enable, default is enable	<b>SFC_Menu=Enable</b>
StartButton	It is used to disabled the start button, its value is Disable or Enable, default is enable	<b>StartButton= Enable</b>
DebuggingGUIPath	It is used to specify the path of the debugging tool vi	<b>DebuggingGUIPath=DebuggingTool.vi</b> <b>DebuggingGUIPath=C:\Tools\DebuggingTool.vi</b>
DebuggingToolCFG	It is used to specify the path of the debugging configuration file	<b>DebuggingToolCFG=</b> <b>DebuggingToolCFG.xlsx</b> <b>DebuggingToolCFG =C:\Tools\DebuggingToolCFG.xlsx</b>

<b>GUI_MultiSingle.vi</b> parameter name	Description	Example
FP_Show	It is used to tell the platform that this GUI needs to be displayed	<b>FP_Show=True</b>
RefreshTime	It is used to set the refresh time of GUI, unit is a millisecond <b>The refresh time must set to 10 seconds for mass production</b>	<b>RefreshTime=10000</b>
Real-time	It is used to set whether test results are displayed in real time, its value is 0 or 1 <b>The Real-time must set to 0 for mass production</b>	<b>Real-time=0</b>
SFC	It is used to tell the platform whether the SFC is ON or OFF by default	<b>SFC=OFF</b>
AutoScrolling	It is used to enable automatic scrolling, its value is Disable or Enable	<b>AutoScrolling=Enable</b>
ToolPath	It is used to specify the path of the developer's own tool vi	<b>ToolPath=Demo\TesterVI\TST_tool.vi</b> <b>ToolPath=C:\Tools\TST_tool.vi</b>
DebuggingGUIPath	It is used to specify the path of the debugging tool vi	<b>DebuggingGUIPath=DebuggingTool.vi</b> <b>DebuggingGUIPath=C:\Tools\DebuggingTool.vi</b>
DebuggingToolCFG	It is used to specify the path of the debugging configuration file	<b>DebuggingToolCFG=</b> <b>DebuggingToolCFG.xlsx</b> <b>DebuggingToolCFG =C:\Tools\DebuggingToolCFG.xlsx</b>

<b>GUI_Actuator1620x800.vi</b> parameter name	Description	Example
FP_Show	It is used to tell the platform that this GUI needs to be displayed	FP_Show=True
RefreshTime	It is used to set the refresh time of GUI, unit is a millisecond <b>The refresh time must set to 10 seconds for mass production</b>	RefreshTime=10000
Real-time	It is used to set whether test results are displayed in real time, its value is 0 or 1 <b>The Real-time must set to 0 for mass production</b>	Real-time=0
Minimizable	It is used to disable the GUI minimized, its value is Disable or Enable	Minimizable=Enable
SFC	It is used to tell the platform whether the SFC is ON or OFF by default	SFC=OFF
delay	In loop mode, it is used to set the delay between this run and the next	delay=3000
SlotPosition	It is used to tell the platform which slot details information you want to print in the GUI, its value is 1,2,3,...32	SlotPosition=1
AutoScrolling	It is used to enable automatic scrolling, its value is Disable or Enable	AutoScrolling=Enable
ToolPath	It is used to specify the path of the developer's own tool vi	ToolPath=Demo\TesterVI\TST_tool.vi ToolPath=C:\Tools\TST_tool.vi
SFC_Menu	It is used to disabled the SFC menu, its value is Disable or Enable, default is enable	SFC_Menu=Enable
InitPicture	It is used to set the default image	InitPicture=SystemVI\sys_GUISourceCode\GUI_Actuator\GUI_Actuator_InitPicture.vi InitPicture=C:\Tools\InitPicture.vi
StartButton	It is used to disabled the start button, its value is Disable or Enable, default is enable	StartButton=Enable
DebuggingGUIPath	It is used to specify the path of the debugging tool vi	DebuggingGUIPath=DebuggingTool.vi DebuggingGUIPath=C:\Tools\DebuggingTool.vi
DebuggingToolCFG	It is used to specify the path of the debugging configuration file	DebuggingToolCFG= DebuggingToolCFG.xlsx

		DebuggingToolCFG =C:\Tools\DebuggingToolCFG.xlsx
--	--	--

amz_log_Daily.vi parameter name	Description	Example
LOG_PATH	<p>It is used to set the log store paths It can be multiple paths, separated by semicolons</p>	<code>LOG_PATH=\\"10.117.119.235\Mooshine\PVT\SMT\BFT\BFT03;..\..\..\LOG\</code>
Module	<p>It is used to define how to distinguish the different modules from serial number. Its format is as follows: <b>Module=ModuleName[Offset to search string/Length of special string]: Special string</b> For more module name, user can use ';' to separate different module, and use ' <b>Else=ModuleName'</b> for the rest</p>	<code>Module=WAN[3/3]:1AF/15S/1EA/1EQ/1F4/1F1/;WTN[3/3]:4RU/9I9-Else=WIFI</code>

amz_log_single.vi parameter name	Description	Example
LOG_PATH	<p>It is used to set the log store paths It can be multiple paths, separated by semicolons</p>	<code>LOG_PATH=\\"10.117.119.235\Mooshine\PVT\SMT\BFT\BFT03;..\..\..\LOG\</code>
Module	<p>It is used to define how to distinguish the different modules from serial number. Its format is as follows: <b>Module=ModuleName[Offset to search string/Length of special string]: Special string</b> For more module name, user can use ';' to separate different module, and use ' <b>Else=ModuleName'</b> for the rest</p>	<code>Module=WAN[3/3]:1AF/15S/1EA/1EQ/1F4/1F1/;WTN[3/3]:4RU/9I9-Else=WIFI</code>
LOG	<p>It is used to set the logging mode, its value is STEP or Final, its default value is Final. If LOG=STEP, the single log vi is able to record the log line by line If LOG=Final, the single log vi will record the log after the test is completed</p>	<code>LOG=STEP</code>

ZIPTXT	It is used to enable the log file compression feature, its value is TRUE or FALSE, its default value is TRUE	ZIPTXT=FALSE
--------	--	--------------

amz_log_timelog.vi parameter name	Description	Example
LOG_PATH	It is used to set the log store paths It can be multiple paths, separated by semicolons	LOG_PATH=\10.117.119.235\Mooshine\PVT\SMT\BFT\BFT03;..\..\LOG\
Module	It is used to define how to distinguish the different modules from serial number. Its format is as follows: <b>Module=ModuleName[Offset to search string/Length of special string]: Special string</b> For more module name, user can use ';' to separate different module, and use ' <b>Else=ModuleName'</b> for the rest	Module=WAN[3/3]:1AF/15S/1EA/1EQ/1F4/1F1;/WTN[3/3]:4RU/9I9- <b>Else=WIFI</b>

amzFileSync.vi parameter name	Description	Example
exePath	It is used to specify the path of exe, its default value is "C:\Windows\System32\Robocopy.exe"	exePath=C:\Windows\System32\Robocopy.exe
WorkingDirectory	It is used to specify the working directory, its default value is "C:\Windows\System32"	WorkingDirectory=C:\Windows\System32
Source	It is used to define the source directory that needs to be synchronized, it is the absolute path.	Source= C:\Kazam\Log
Dest	It is used to define the destination directory	Dest=\10.117.119.235\Log
Options	It is used to define the optional parameters of Robocopy.exe, its default value is "/mot:1 /mon:1 /e"	Options=/mot:1 /mon:1 /e
Log	It is used to define whether to generate log or not. Its value is enable or disable, its default value is disable	Log=enable

<b>amz_RetentionPolicy.vi</b> parameter name	Description	Example
Interval	It is used to define the interval between scanning file information, its unit is second, its default value is 3600	Interval=3600
days	It is used to define how many days to retain, and delete files older than n day, its unit is day. It can be a number less than 1, such as 0.1 day. Its default value is 14	days=14
Source	It is used to define the source directory that needs to be deleted, it is the absolute path.	Source=C:\Kazam\Log
Recursive	It is used to define whether to delete subdirectory files, its value is true or false and the default vale is true	Recursive=True
Log	It is used to define whether to generate log or not. Its value is enable or disable, its default value is disable	Log=enable

amz\_Plug-in\_SyncSystemTime.vi only sync up the time with server one time when Kazam boot up.

<b>amz_Plug-in_SyncSystemTime.vi</b> parameter name	Description	Example
Server_PATH	It is used to set the time server path	Server_PATH=\10.117.119.235

log\_SFC.vi need to develop by CM or supplier, and this vi need to save to the SFC folder, the developer need to create a new folder for their factory.

<b>log_SFC.vi</b> parameter name	Description	Example
LOG_PATH	It is used to set the log store paths It can be multiple paths, separated by semicolons	LOG_PATH=\10.117.119.235\Moonshine\PVT\SMT\BFT\BFT03;..\..\LOG\
VI_PATH	It is used to set the log vi path	VI_PATH=..\Project\SFC\Compal\BFT\

## 10. Sequence Producer

The sequence producer is a component of Kazam that is used to generate new sequences and develop unified debugging tools. Its architecture is as follows:

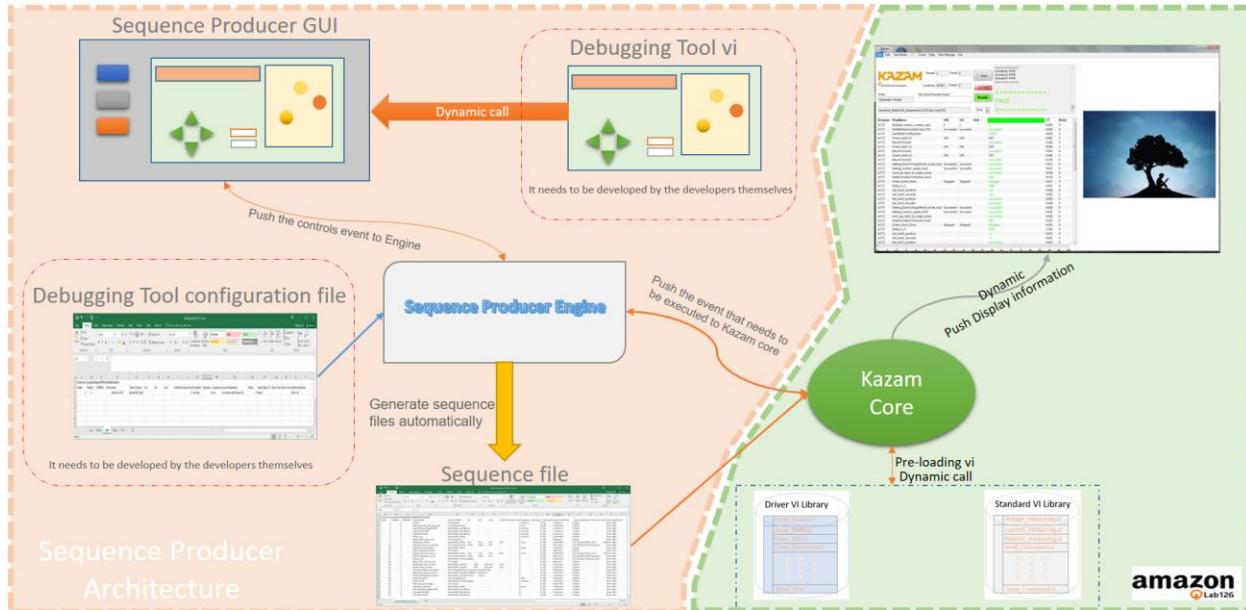


Figure 60

If the developer has completed the development of drive vi and test vi, then the developer only needs to complete the following two steps to complete the sequence producer development:

- 1, Create a debug tool profile in XLSX format
- 2, Develop the debugging tool GUI

### 9.1 Debugging Tool configuration file

The debugging tool configuration file in XLSX format, its contents the same as a normal TestPlan. The developer needs to manually create their own sequence on each sheet, and the sheet name will bind to the control in the debugging tool GUI.

The screenshot shows an Excel spreadsheet with the following data in the first few rows:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	CheckVal:727d9e42ea4777a5677ba37ce8af45bf																			
2	Index	Enable	PREREQ	Test Name	Test VI Name	LSL	USL	Unit	Limit	Save Result	Property	Dynamic	Casesensitive	Input Parameter	Retry	Next Step	Next Test	Goto Cour ErrorCode Note		
3	1	1		Downward_movement	DemoDRV_Ball					0 normal	0	Yes		Function=Down Step=10 Event_Input=Step Event_output=Ball:event	0	Next		Error-10		
4																				
5																				
6																				
7																				
8																				

A callout box points to the 'Down' button in the bottom-left corner of the Excel ribbon, with the text: "The sheet name will bind to the control in the debugging tool GUI".

Figure 61

The developers can use the following two keywords to define input and output controls in “input parameters”:

Event\_input=Name1;Name2;...Name N

Event\_input is the keyword, cannot be changed.

Name1/Name2/Name N is the control’s label, the developers can customize, and it must have the same name as the control in the debugging tool GUI, it has to be unique.

Event\_output=Name:event or Event\_output=Name

Event\_output is the keyword, cannot be changed.

Name is the control’s label, the developers can customize, and it must have the same name as the control in the debugging tool GUI, it has to be unique.

event is the type that defines the output control, and there are two types of output: value output and event output, the default is value output, which need not be specified.

## 9.2 Debugging Tool GUI

The developers need to develop their own debugging tool GUI, the controls name in this GUI are automatically bound to the sheet name in the configuration file, so if the developer needs to bind a control, the label name of the control must be matched to the sheet name of the configuration file. If the control does not need to be bound, it can name the control whatever you want.

Here is an example of a debugging tool GUI:

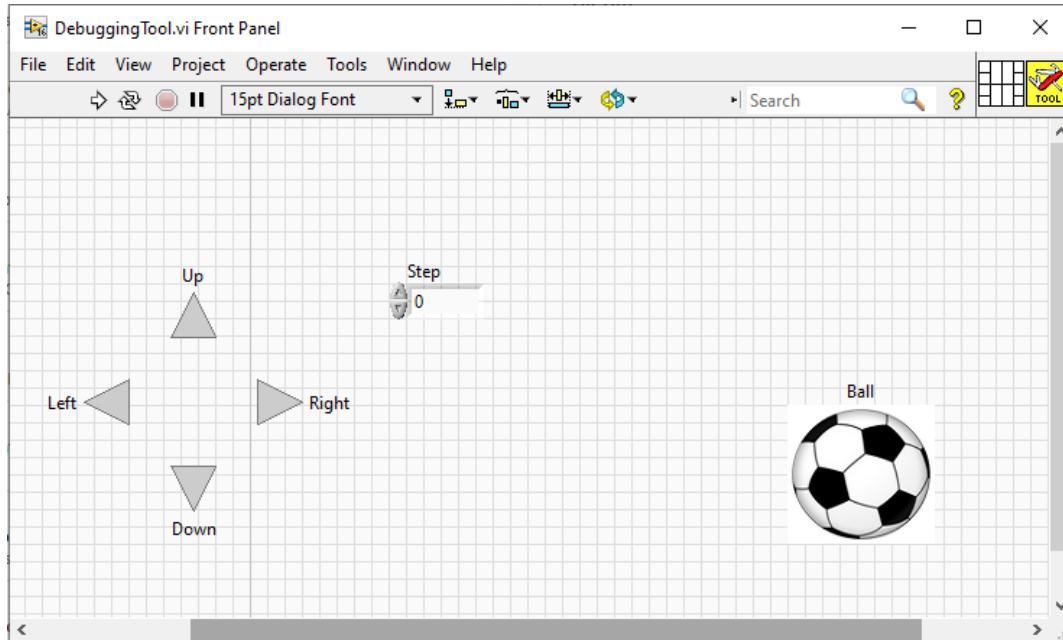


Figure 62

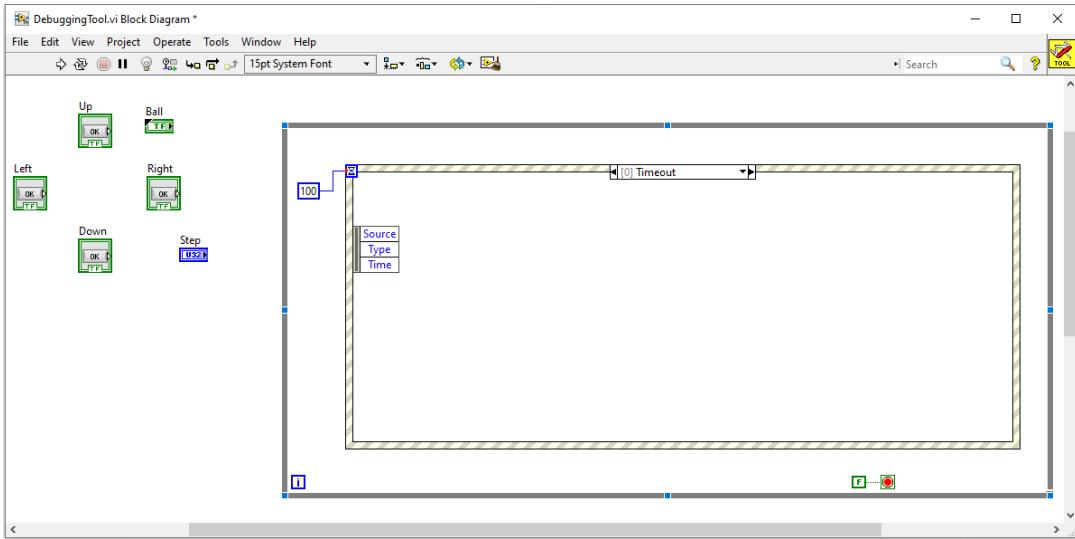


Figure 63

From Figure 62, there are only six controls are placed in the front panel. They are "up", "down", "right", "left", "Step" and "Ball".

From Figure 63, the "up", "down", "right", "left", "Step" and "Ball" are not connected anywhere. And

There is a blank event structure in the while loop, it ensures that this vi is always running when it is called.

### 9.3 How to configure sequence producer

The developer must place the debugging tools configuration file and debugging tools GUI in the project folder, Figure 64 shows the directory structure.

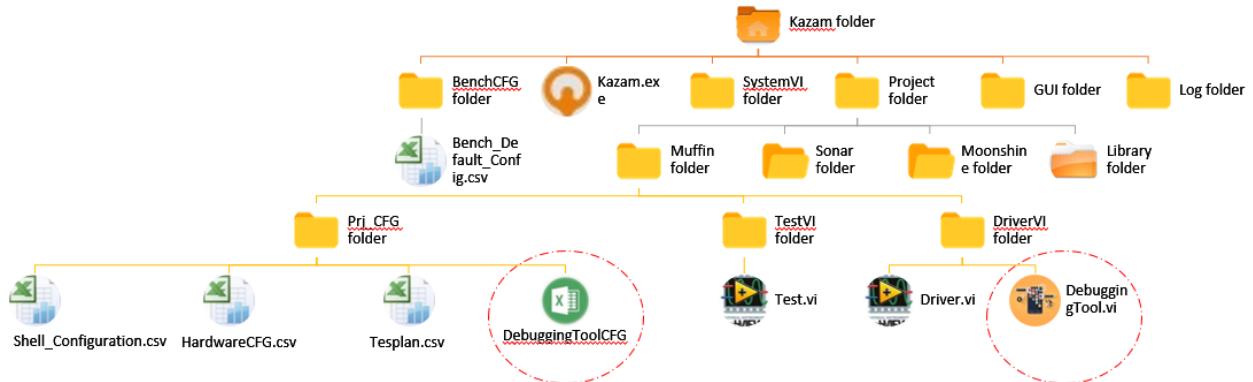


Figure 64

The developers can configure different debugging tools and their corresponding configuration files in each shell configuration file. Refer to [8.2.2 Other parameter](#) for a description of display parameters

A	B	C	D
1	CheckVal:4295704283\$af50160be79c6a4a		
2	RELEASE_UTDPhi:0.01		
3	SEQ Resource_VI_NAME	CONTROL_STRING	
4	Sequence sys_SequenceCFGReadout.vi	TestPlan2>Slot_StopOrPause.xlsx Main TestPlan2>Slot_StopOrPause.xlsx Second TestPlan3>Slot_StopOrPause.xlsx BG ResetPlan>DemoSequence1.0.0.2.csv WaitUntilDone=True EnableParameter=HVT;REL;	
5	Sequence sys_SequenceLauncher.vi	SlotStopOrPause TestSequence.vi	
6	DataLog GUI_Actuator162x800.vi	LOG=FINAL[LOG_PATH=\_\_\_\LOG1]Module=WAN[3/3];1AF/155;WTN[3/3];4RU/998-Else=WIFI EmailContact=user1@server.com/user2@server.com TextContact:phonenumber1/phonenumbe	
7	DataLog amz_log_Daily.vi	LOG=FINAL[LOG_PATH=\_\_\_\LOG1]Module=WAN[3/3];1AF/155;WTN[3/3];4RU/998-Else=WIFI	
8	DataLog amz_log_TimedLog.vi	LOG=FINAL[LOG_PATH=\_\_\_\LOG1]Module=WAN[3/3];1AF/155;WTN[3/3];4RU/998-Else=WIFI	
9	DataLog amz_log_UploadFile.vi	LOG=FINAL[LOG_PATH=\_\_\_\LOG1]Module=WAN[3/3];1AF/155;WTN[3/3];4RU/998-Else=WIFI	
10	DataLog EAPI_Autostart.vi		
11			
12	EOF		

Figure 65

The user can open the sequence producer from the menu, Tool>Sequence Producer, it shows as Figure 67

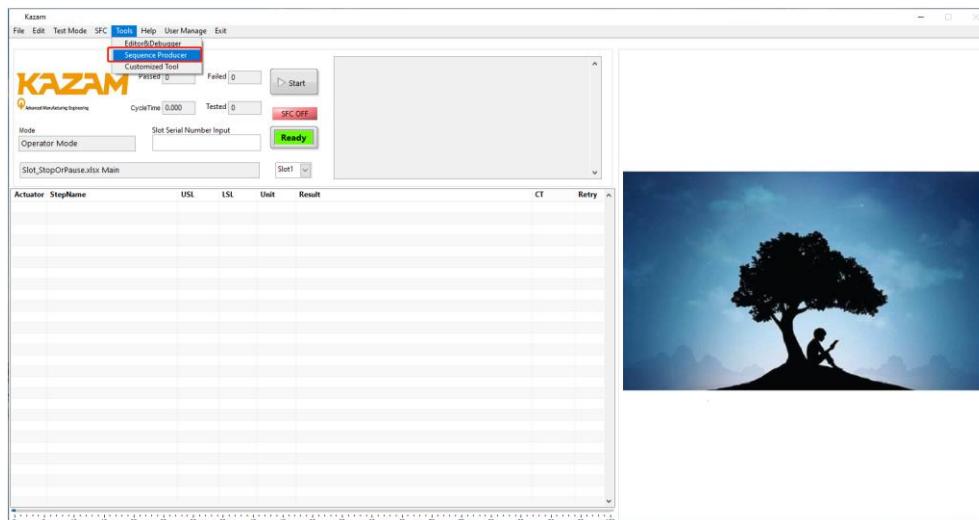


Figure 66

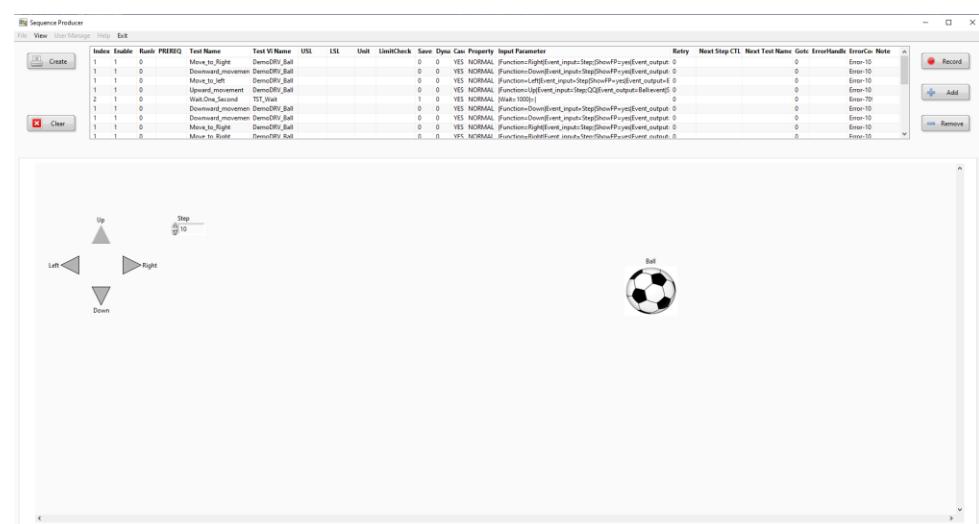


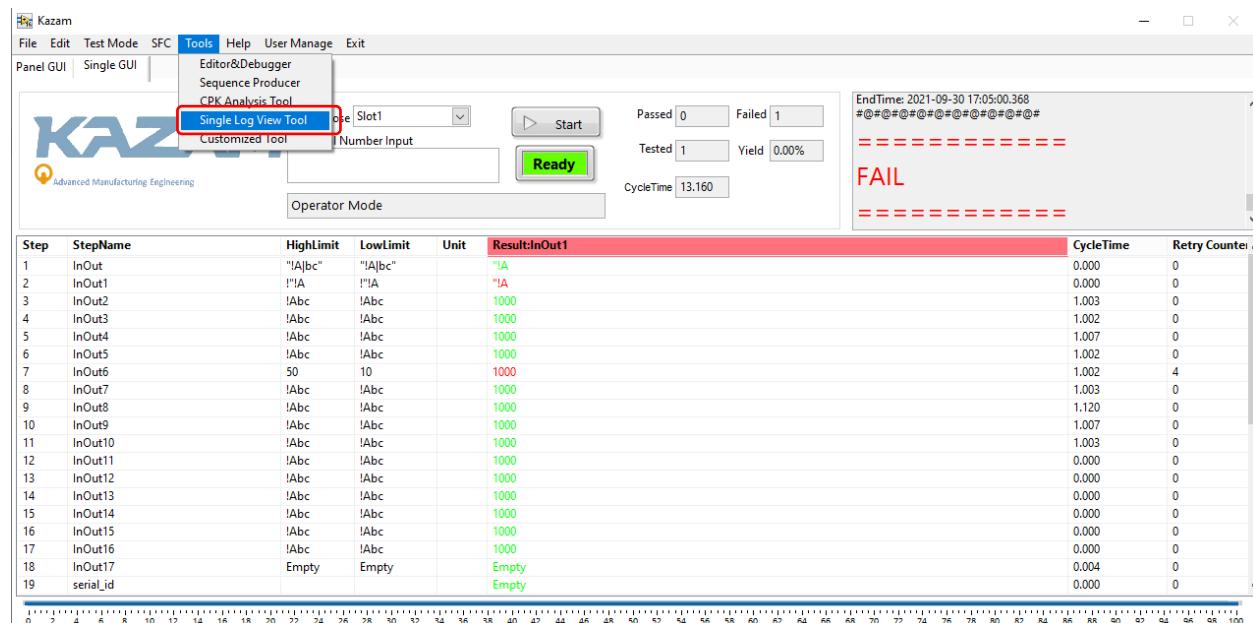
Figure 67

## 11. Kazam Tools

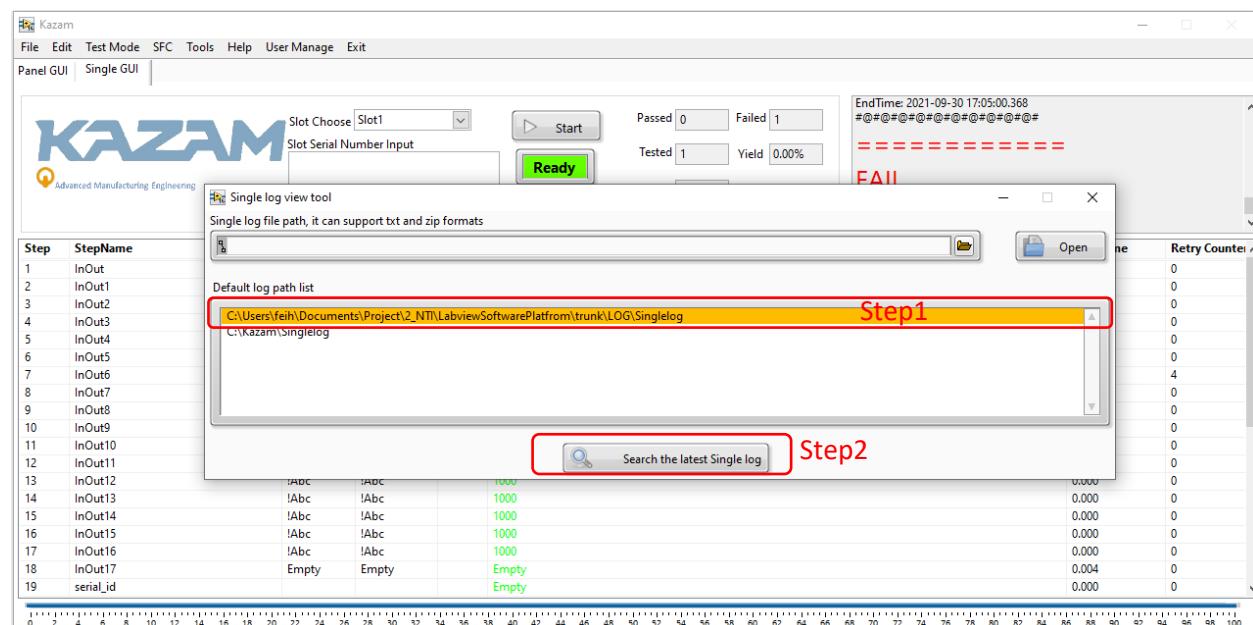
### 11.1 Single log browser

The single log browser can more efficiently view logs on the production line, it can support txt and zip formats, and support parallel browsing of multiple logs. The following is the instructions:

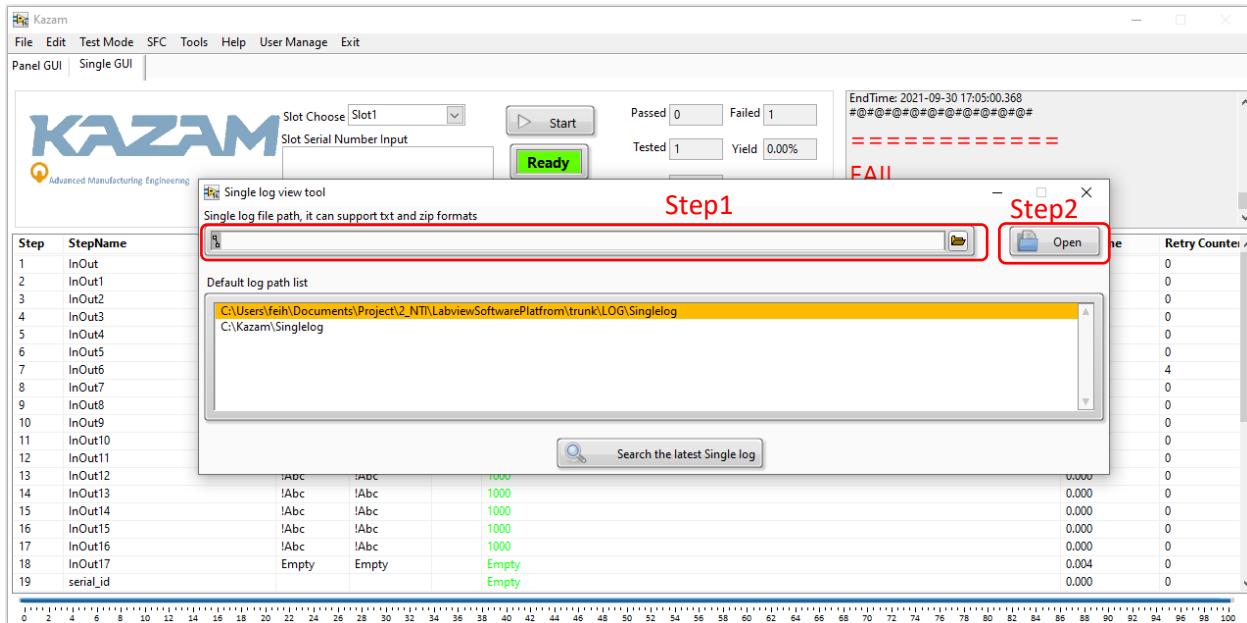
1, Open the log viewing tool from the menu



2, Select the default path to open the latest single log



### 3, Select a specified single log file, it can support txt and zip formats



### 4, A single log browser will be opened

Step	StepName	USL	LSL	Unit	Result	CT	NoT	StartTime	EndTime
1	InOut	"IA bc"	"IA bc"		Result= "IA bc"	0.000	0	170447.190_20210930_0447.190_20210	
2	InOut1	"!IA	"!IA		Result= "IA"	0.000	0	170447.190_20210	170447.190_20210
3	InOut2	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.003	0	170447.191_20210	170448.194_20210
4	InOut3	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.002	0	170448.195_20210	170449.197_20210
5	InOut4	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.007	0	170449.197_20210	170450.205_20210
6	InOut5	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.000	0	170450.205_20210	170451.206_20210
7	InOut6	50	10		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.003	0	170451.206_20210	170452.209_20210
8	InOut7	50	10		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.003	1	170452.209_20210	170453.212_20210
9	InOut8	50	10		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.002	2	170453.212_20210	170454.214_20210
10	InOut9	50	10		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.002	3	170454.214_20210	170455.216_20210
11	InOut10	50	10		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.000	4	170455.216_20210	170456.218_20210
12	InOut11	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.003	0	170456.218_20210	170457.221_20210
13	InOut12	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.120	0	170457.221_20210	170458.341_20210
14	InOut13	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.007	0	170458.341_20210	170459.349_20210
15	InOut14	!Abc	!Abc		Result= 1000 =1000	0.000	0	170500.352_20210	170500.353_20210
16	InOut15	!Abc	!Abc		Result= 1000 =1000	0.000	0	170500.352_20210	170500.353_20210
17	InOut16	!Abc	!Abc		Result= 1000 vi_version=1.0 UserStopped=FALSE	1.003	0	170500.353_20210	170500.356_20210
18	InOut17	Empty	Empty		Result= Empty vi_version=1.0	0.004	0	170500.356_20210	170500.357_20210
19	serial_id				Result= Empty	0.000	0	170500.357_20210	170500.357_20210

5, The following is an introduction to the log browser

The screenshot shows the Log Browser interface with the following components:

- Log path:** C:\Users\feih\Documents\Project\2\_NTI\LabviewSoftwarePlatform\trunk\LOG\Singlelog\InOut1\offline\_Vnl\_xxx\_Slot1\_FAIL\_\_20210930\_170502.txt
- Pass or fail:** FAIL offline
- Log summary:** A box containing test parameters like Kazam GUI Version, Test Station No., Fixture No., etc.
- Search box:** A string-based search box for step names.
- Detailed information box:** A large box containing a table of step details. It highlights failed steps (e.g., InOut1) with red background color.
- Color setting box:** A small box with a blue arrow pointing to the detailed information box, indicating it's used to set row colors.

**Detailed Step Log Table:**

Step	StepName	USL	LSL	Unit	Result	CT	NoT	StartTime	EndTime
InOut	"!Abc"	"!Abc"			[Result="!A In= !A PreserveCase=TRUE USL= '!Abc"]	0.000	0	170447.190_20210930_0447.190_20210	
InOut1	!`IA	!`IA			[Result= !A]	1.003	0	170447.190_20210 170447.190_20210	
InOut2	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	1	170447.191_20210 170448.194_20210	
InOut3	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	2	170448.195_20210 170449.197_20210	
InOut4	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.007	0	170449.197_20210 170450.205_20210	
InOut5	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.002	0	170450.205_20210 170451.206_20210	
InOut6	50	10			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	0	170451.206_20210 170452.209_20210	
InOut6	50	10			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	1	170452.209_20210 170453.212_20210	
InOut6	50	10			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.002	2	170453.212_20210 170454.214_20210	
InOut6	50	10			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	3	170454.214_20210 170455.216_20210	
InOut6	50	10			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.004	4	170455.216_20210 170456.218_20210	
InOut7	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.003	0	170456.218_20210 170457.221_20210	
InOut8	!Abc	!Abc			[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.120	0	170457.221_20210 170458.341_20210	
					[Result= 1000 v_version=1.0 UserStopped= FALSE]	1.007	0	170458.341_20210 170459.349_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.000	0	170500.352_20210 170500.353_20210	
					Result= 1000 =1000	0.003	0	170459.349_20210 170500.352_20210	
					[Result= Empty v_version=1.0 ]	0.004	0	170500.352_20210 170500.356_20210	
					[Result= Empty ]	0.000	0	170500.357_20210 170500.357_20210	
					[Result= LAB ]	0.000	0	170500.357_20210 170500.357_20210	
					[Result= Demo ]	0.000	0	170500.358_20210 170500.358_20210	
					[Result= eCDef ]	0.003	0	170500.358_20210 170500.361_20210	

6, The following describes how to set/clear row or cell colors

**Step1**

Step	StepName	USL	LSL	Unit	Result	CT	NoT	StartTime	EndTime
1	InOut1	"IA"	LSL!"A	USL!"A	[Result="IA"]	0.000	0	170447.190_20210	170447.190_2021C
2	InOut2	"IA"	LSL!"A	USL!"A	[Result="IA"]	0.000	0	170447.190_20210	170447.190_2021C
3	InOut3	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170447.191_20210	170448.194_2021C
4	InOut4	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170448.195_20210	170448.197_2021C
5	InOut5	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170448.197_20210	170450.205_2021C
6	InOut6	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170450.205_20210	170451.206_2021C
7	InOut7	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170451.206_20210	170452.209_2021C
8	InOut8	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	1	170452.209_20210	170453.212_2021C
9	InOut9	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	2	170453.212_20210	170454.214_2021C
10	InOut10	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	4	170455.216_20210	170456.218_2021C
11	InOut11	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170456.218_20210	170457.221_2021C
12	InOut12	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.120	0	170457.221_20210	170458.341_2021C
13	InOut13	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170458.341_20210	170459.349_2021C
14	InOut14	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170459.349_20210	170500.352_2021C
15	InOut15	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.000	0	170500.352_20210	170500.353_2021C
16	InOut16	IAbc	LSL!"A	USL!"A	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170500.352_20210	170500.353_2021C
17	InOut17	Empty	LSL!"A	USL!"A	[Result="Emptyvi_version:1.0.0"]	0.004	0	170500.352_20210	170500.356_2021C
18	serial_id				[Result="Empty"]	0.000	0	170500.357_20210	170500.357_2021C
19	factory_id				[Result="LAB"]	0.000	0	170500.357_20210	170500.357_2021C
20	project_name				[Result="Demo"]	0.000	0	170500.358_20210	170500.358_2021C
21	build_config				[Result="eCDef"]	0.003	0	170500.358_20210	170500.361_2021C

**FAIL offline**

**Step2, left click the cell**

Step	StepName	USL	LSL	Unit	Result	CT	NoT	StartTime	EndTime
1	InOut1	"IA bc"	"IA bc"	IAbc	[Result="IA bc"]	0.000	0	170447.190_20210	170447.190_2021C
2	InOut1	"IA"	"IA"	IAbc	[Result="IA"]	0.000	0	170447.190_20210	170447.190_2021C
3	InOut2	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170447.191_20210	170448.194_2021C
4	InOut3	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170448.195_20210	170448.197_2021C
5	InOut4	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170448.197_20210	170450.205_2021C
6	InOut5	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170450.205_20210	170451.206_2021C
7	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170451.206_20210	170452.209_2021C
8	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	1	170452.209_20210	170453.212_2021C
9	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	2	170453.212_20210	170454.214_2021C
10	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	3	170454.214_20210	170455.216_2021C
11	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	4	170455.216_20210	170456.218_2021C
12	InOut11	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170456.218_20210	170457.221_2021C
13	InOut12	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.120	0	170457.221_20210	170458.341_2021C
14	InOut13	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170458.341_20210	170459.349_2021C
15	InOut14	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170459.349_20210	170500.352_2021C
16	InOut15	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.000	0	170500.352_20210	170500.353_2021C
17	InOut16	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170500.352_20210	170500.353_2021C
18	InOut17	Empty	Empty	IAbc	[Result="Emptyvi_version:1.0.0"]	0.004	0	170500.357_20210	170500.357_2021C
19	serial_id				[Result="Empty"]	0.000	0	170500.357_20210	170500.357_2021C
20	factory_id				[Result="LAB"]	0.000	0	170500.357_20210	170500.357_2021C
21	project_name				[Result="Demo"]	0.000	0	170500.358_20210	170500.358_2021C
22	build_config				[Result="eCDef"]	0.003	0	170500.358_20210	170500.361_2021C

**FAIL offline**

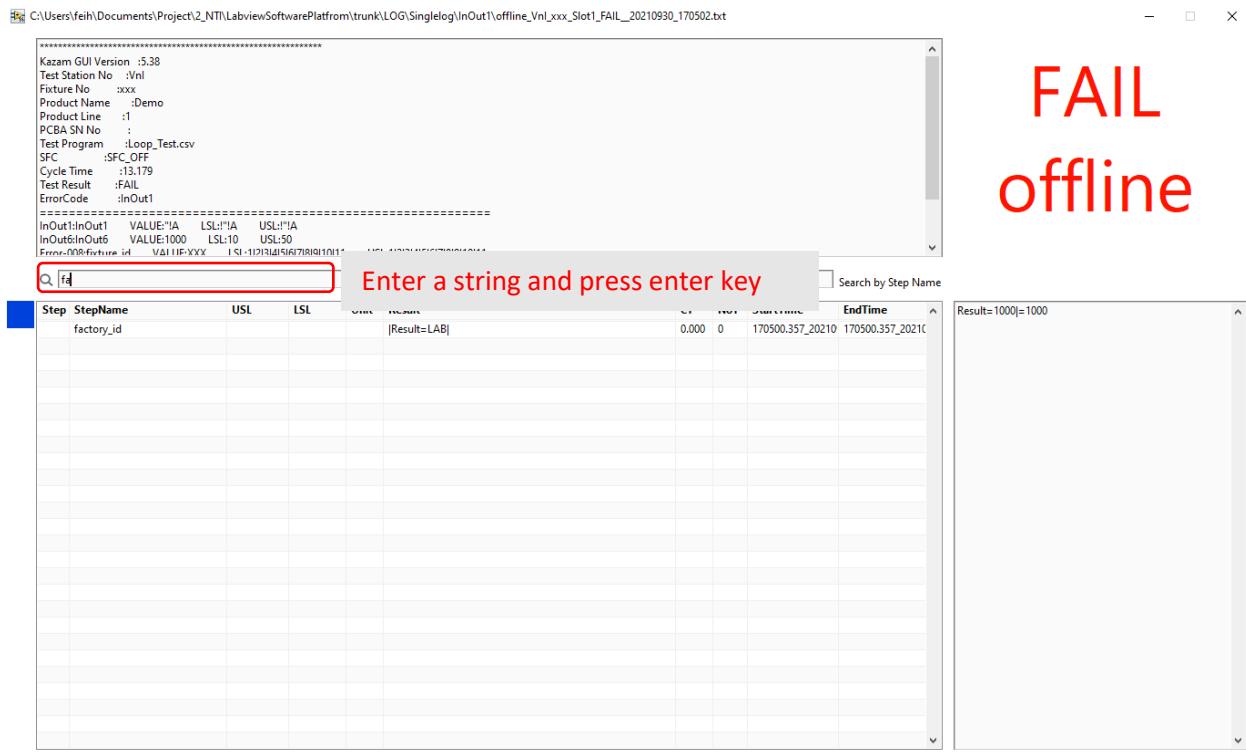
  

**Step3, right click to pop up the shortcut menu**

Step	StepName	USL	LSL	Unit	Result	CT	NoT	StartTime	EndTime
1	InOut1	"IA bc"	"IA bc"	IAbc	[Result="IA bc"]	0.000	0	170447.190_20210	170447.190_2021C
2	InOut1	"IA"	"IA"	IAbc	[Result="IA"]	0.000	0	170447.190_20210	170447.190_2021C
3	InOut2	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170447.191_20210	170448.194_2021C
4	InOut3	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170448.195_20210	170448.197_2021C
5	InOut4	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170448.197_20210	170450.205_2021C
6	InOut5	IAbc	"IA"	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170450.205_20210	170451.206_2021C
7	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170451.206_20210	170452.209_2021C
8	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	1	170452.209_20210	170453.212_2021C
9	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	2	170453.212_20210	170454.214_2021C
10	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	3	170454.214_20210	170455.216_2021C
11	InOut6	50	10	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	4	170455.216_20210	170456.218_2021C
12	InOut11	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170456.218_20210	170457.221_2021C
13	InOut12	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.120	0	170457.221_20210	170458.341_2021C
14	InOut13	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.007	0	170458.341_20210	170459.349_2021C
15	InOut14	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.002	0	170459.349_20210	170500.352_2021C
16	InOut15	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.000	0	170500.352_20210	170500.353_2021C
17	InOut16	IAbc	IAbc	IAbc	[Result="1000vi_version:1.0.0>UserStopped=False"]	1.003	0	170500.352_20210	170500.353_2021C
18	InOut17	Empty	Empty	IAbc	[Result="Emptyvi_version:1.0.0"]	0.004	0	170500.357_20210	170500.357_2021C
19	serial_id				[Result="Empty"]	0.000	0	170500.357_20210	170500.357_2021C
20	factory_id				[Result="LAB"]	0.000	0	170500.357_20210	170500.357_2021C
21	project_name				[Result="Demo"]	0.000	0	170500.358_20210	170500.358_2021C
22	build_config				[Result="eCDef"]	0.003	0	170500.358_20210	170500.361_2021C

**FAIL offline**

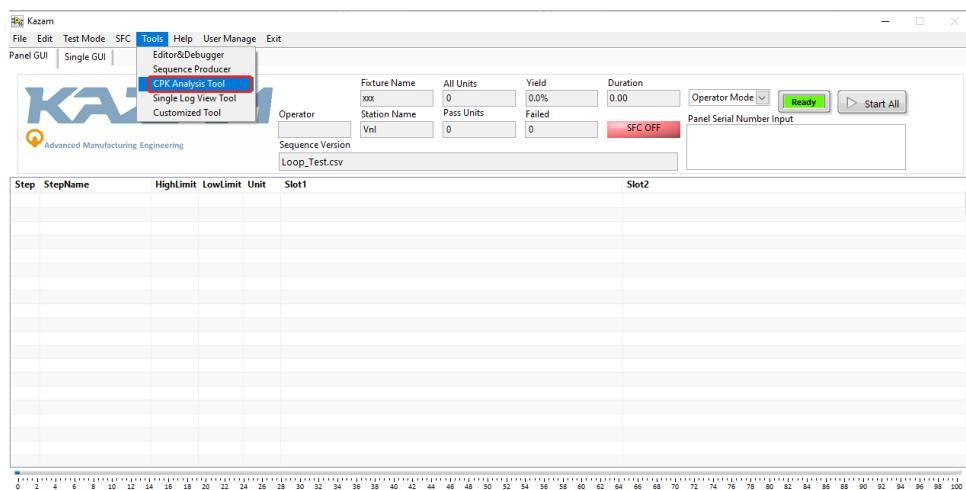
7, The user can use the search string feature to search for steps that contain this string in step name



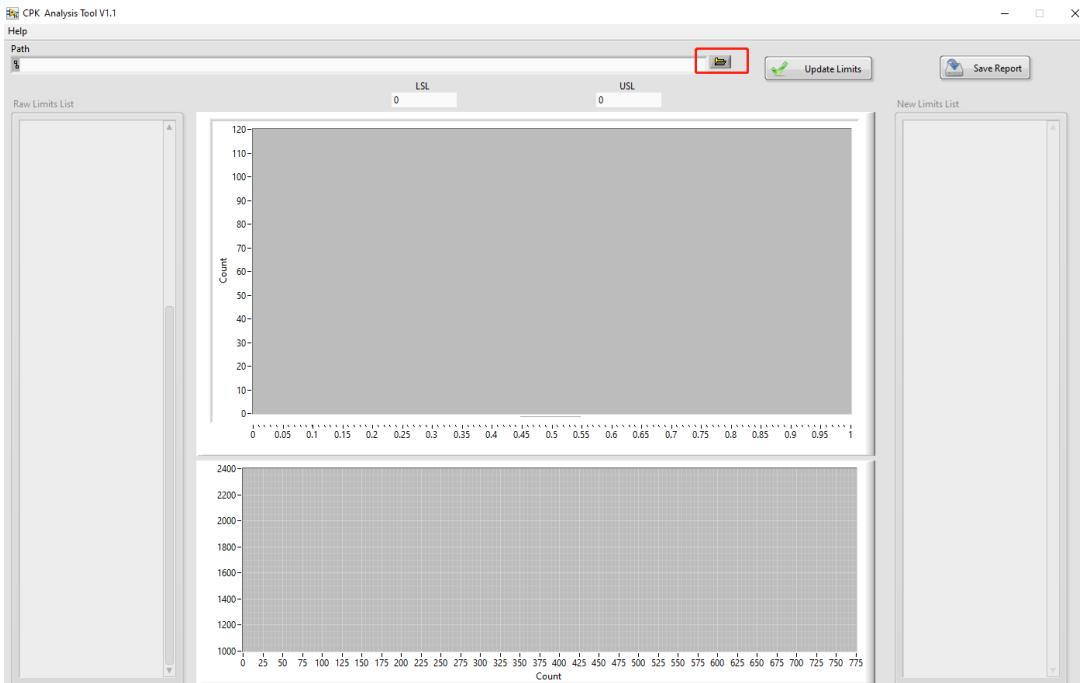
## 11.2 CPK Analysis Tool

The CPK analysis tool can help users quickly generate histograms, it can support direct import of daily logs to analyze CPK. And the users can also use it to generate a new limits adjustment suggestion report.

1, Open the CPK analysis tool from the menu



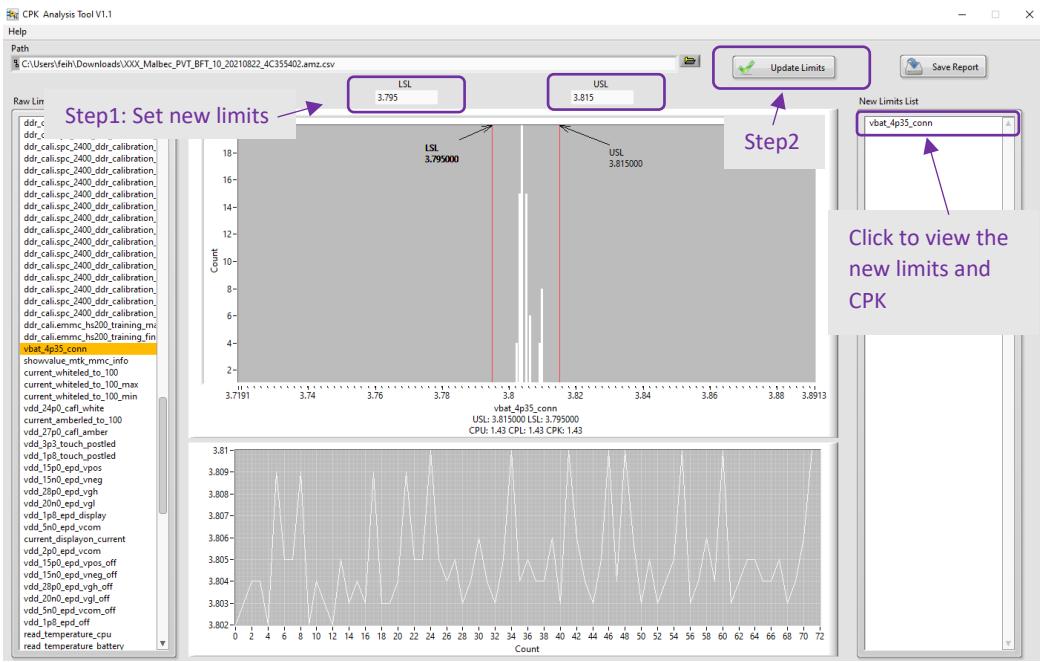
2, Select a daily log



### 3, Select a step name to show its CPK and histogram



### 4, The users can adjust limits according to the following steps



5, Click save report button, CPK tool will automatically generate CPK report to the same level directory of log file, and users can open folders under the same level directory to find reports in HTML format

