1．目的  
本文档将介绍如何使用软件平台开发新的测试项目，并让开发人员专注于测试本身。  
2.定义和缩略语  
AME：先进制造工程  
TE：测试工程  
GUI：图形用户界面  
DUT：被测设备  
证监会：车间控制系统  
TestPlan：它是一个csv文件，用于管理所有测试步骤  
CM：合同制造商  
Kazam：亚马逊测试框架的名称

3．平台架构

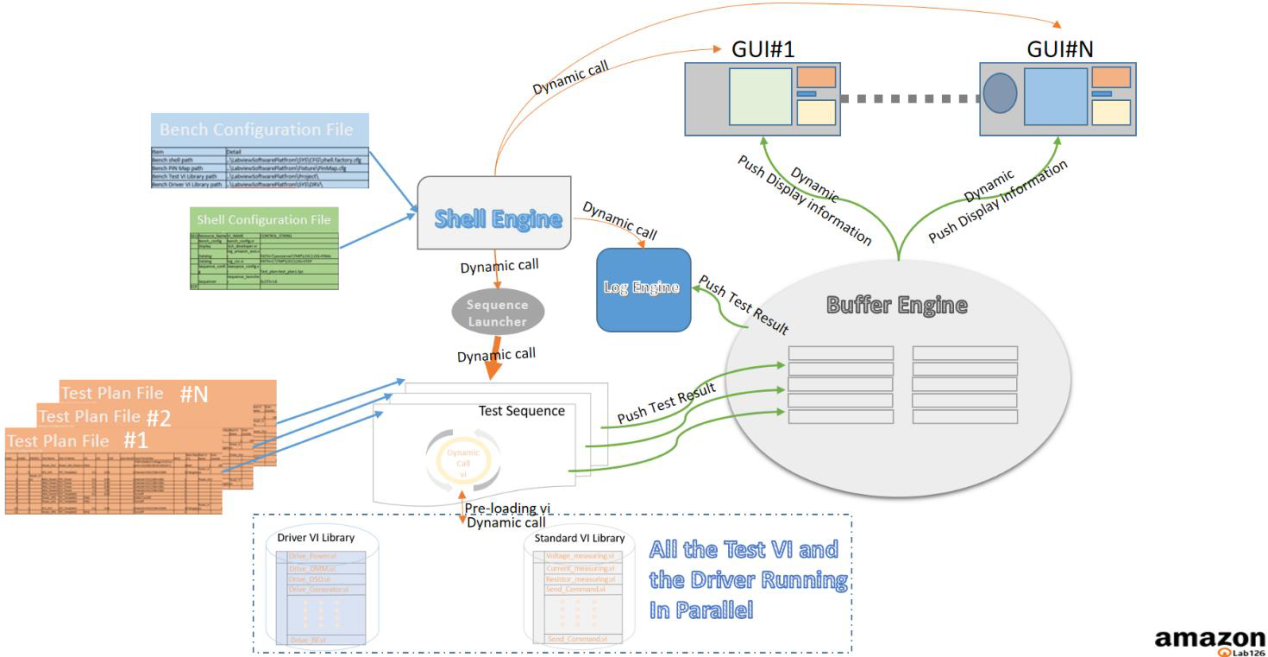


图1

图1是软件平台架构，所有测试vi和驱动程序vi都将通过动态调用，我们可以使用TestPlan来设置调用模式。  
4.特点  
4.1添加/替换/删除功能  
平台可以通过配置shell配置文件来添加/替换/删除功能。

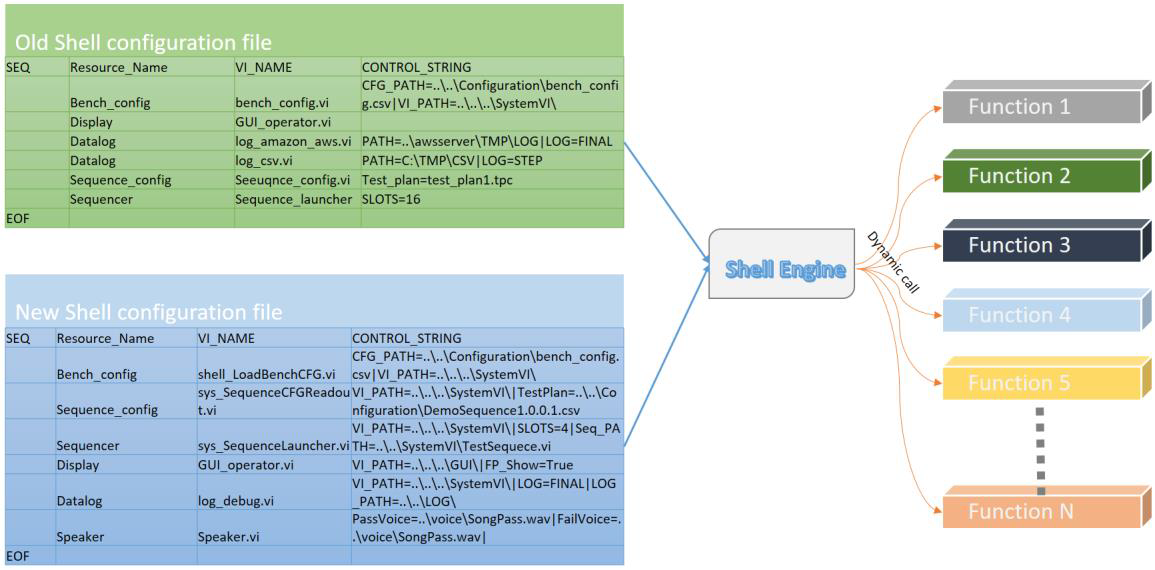


图2

4.2运行不同的测试计划  
平台可以在不同的插槽中运行不同的TestPlan，图3是一个简单的例子。

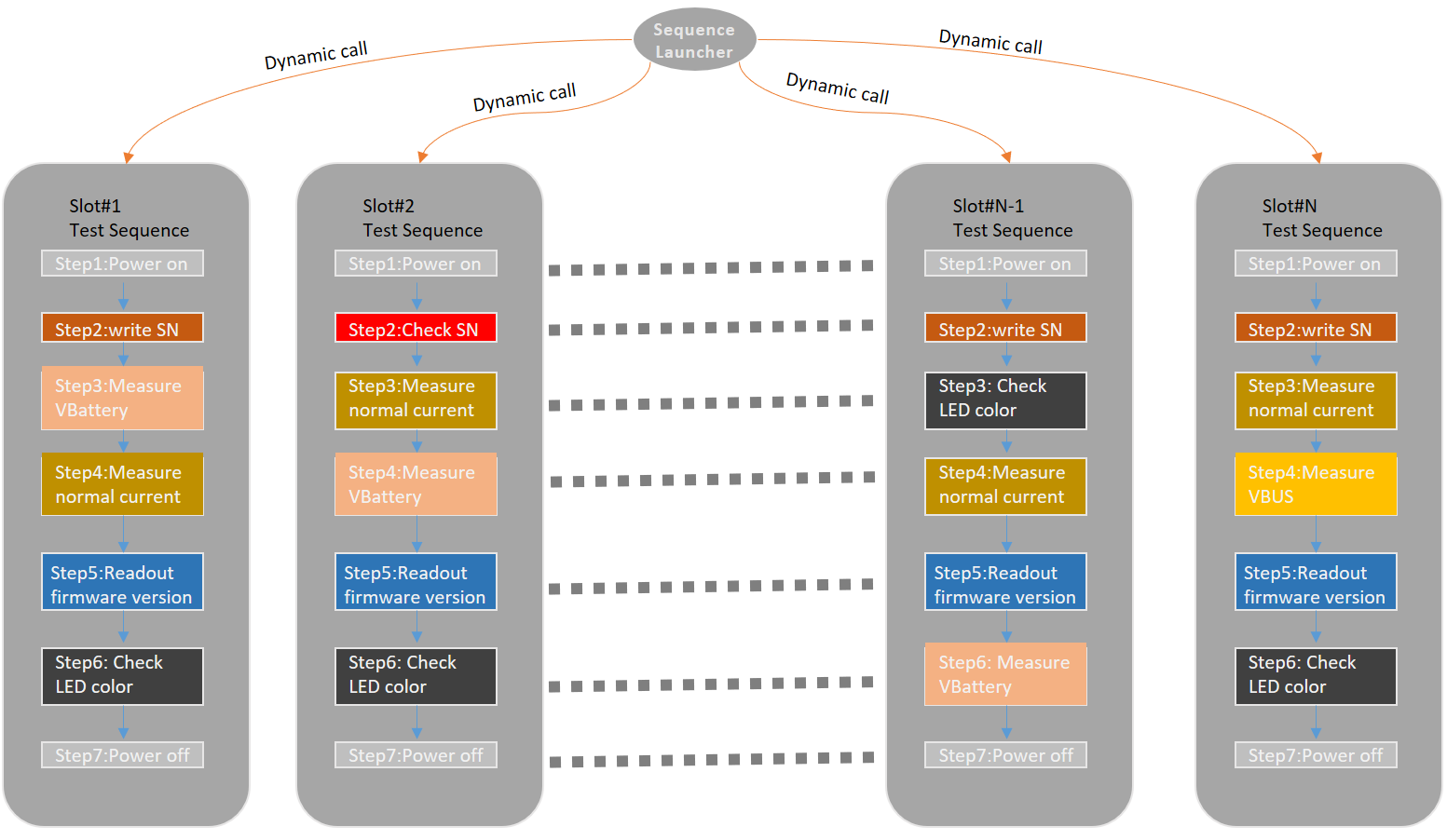
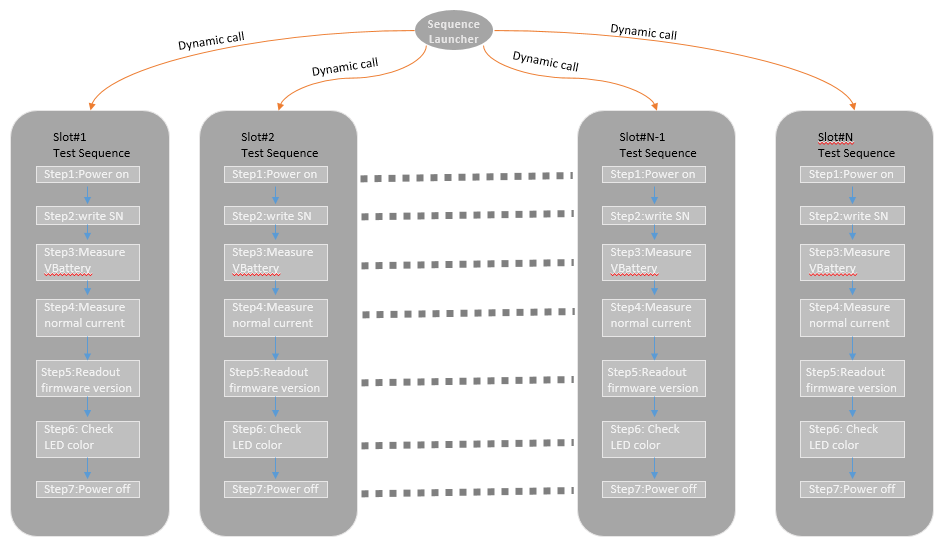
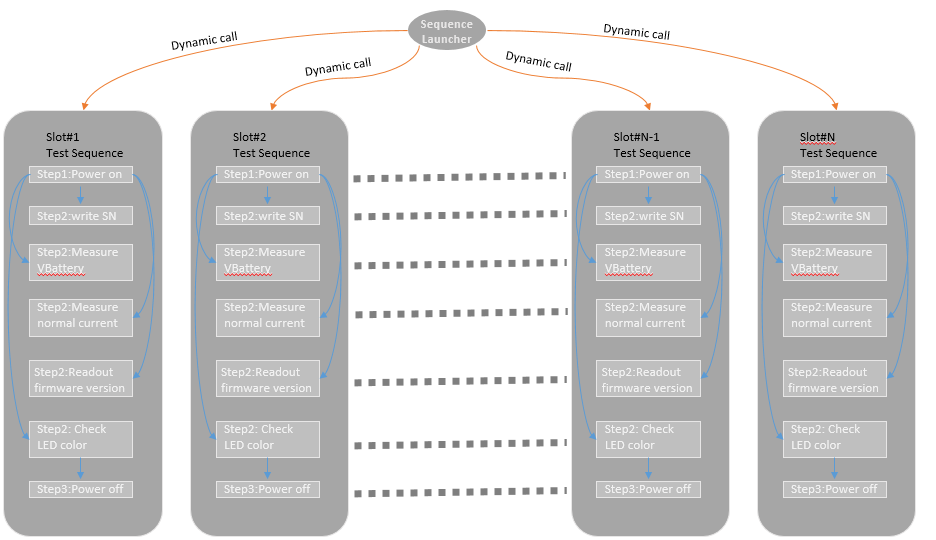


图3

4.3并行运行  
平台可以并行运行所有测试步骤，如图4所示，它是一步一步执行，从每个插槽的步骤1到步骤7。 每个插槽独立于操作。

 图4  
平台可以在每个插槽上运行并行测试步骤。 如图4所示，它有七个步骤，我们可以让步骤2到步骤6同时运行，如图5所示。

图5

4.4自动更新Kazam和Project文件夹  
该平台具有自动更新功能，开发人员可以上传最新的Kazam  
AWS云，然后工厂的服务器将在运行时自动下载它  
Kazam.exe，Kazam会自动检查工厂的更新服务器，如果有的话  
更新，它会自动更新最新的代码。  
有关详细的参数设置，请参阅第8.1节。

5. GUI介绍  
图6是面板操作员GUI，它用于生产面板测试。 它可以很容易地设置面板的数量。 最多支持32个DUT。

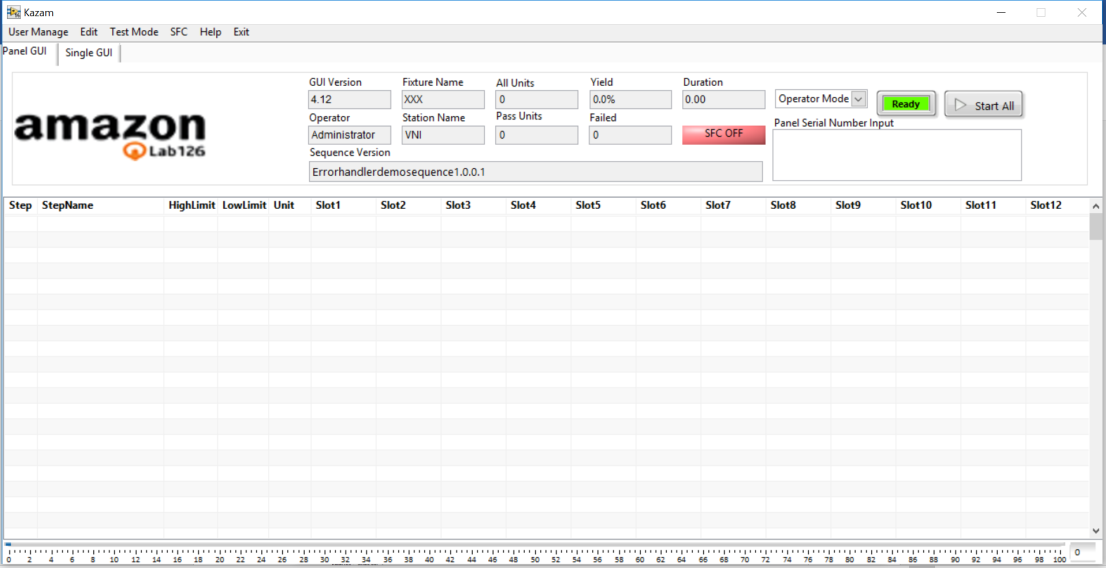


图6

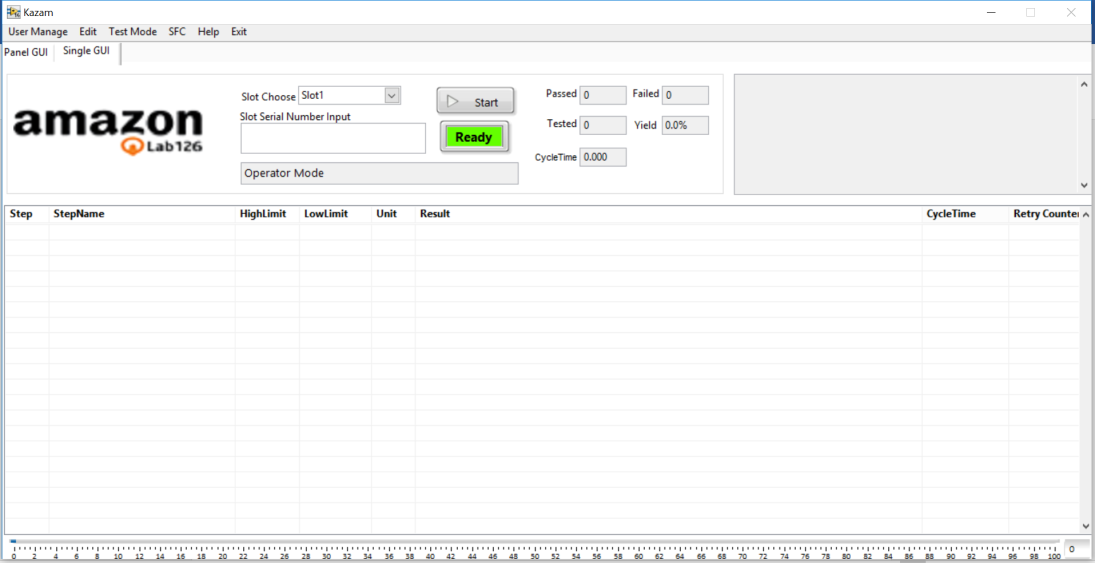
图7是单个GUI，它用于调试或维护。  


图7

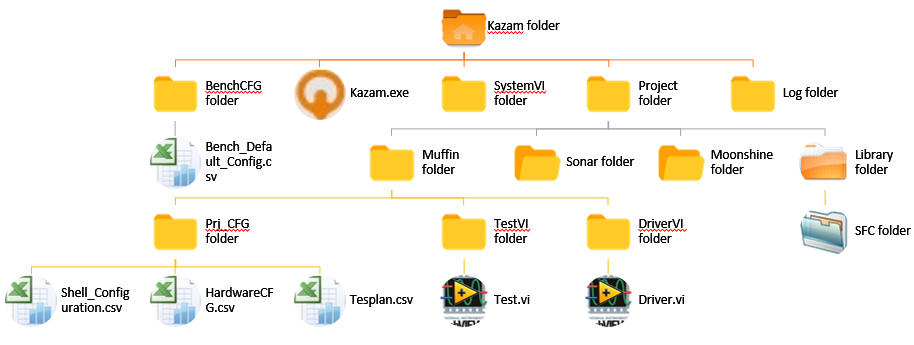
6.如何开发新项目  
6.1 Kazam文件夹  
我们将Shell文件夹发布到供应商或CM，如图8所示，它将具有以下文件或文件夹。

图8

BenchCFG文件夹：它包含工作台配置文件。  
LOG文件夹：它包含所有日志文件  
SystemVI文件夹：它包含所有系统vi文件  
项目文件夹：它包含所有项目文件夹，每个项目都有一个文件夹来管理vi，TestPlan，shell配置文件和硬件配置文件。 项目文件夹中有库文件夹，用于常用库vi，exe和dll。  
SFC文件夹：它包含所有SFC vi，开发人员必须将所有SFC vi放入此文件夹。  
Kazam.exe：它是平台的启动入口，双击启动我们的软件  
6.2项目文件夹  
我们将项目文件夹发布给开发人员，如图9所示。在这个项目文件夹中，它将包括Labview项目，开发人员可以按照项目开发测试vi和驱动程序vi。

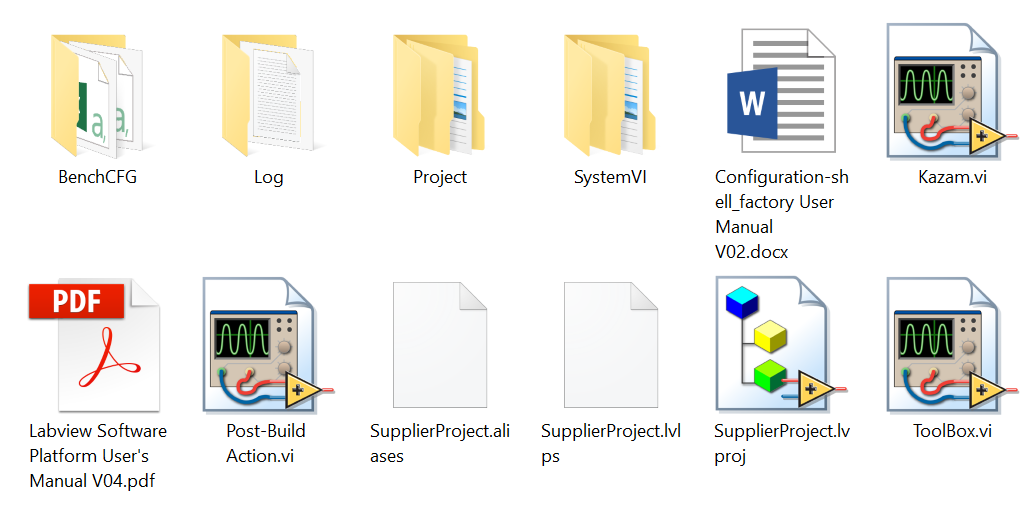
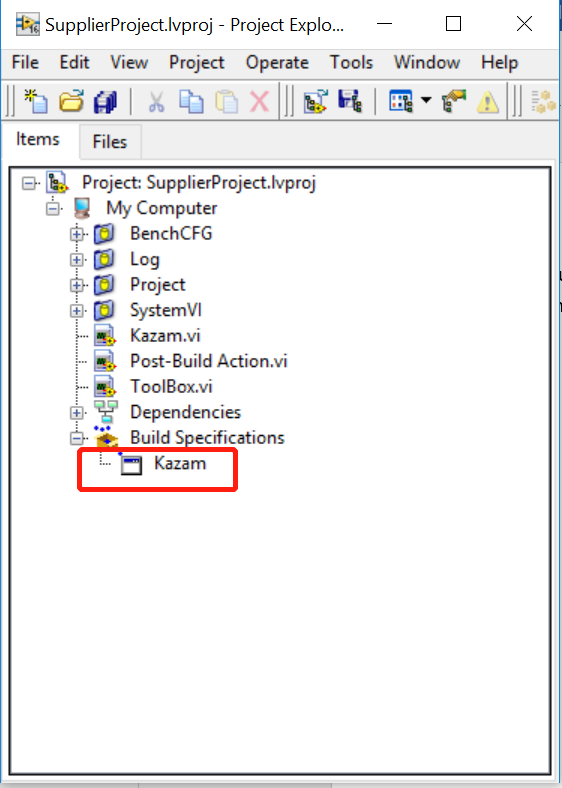


图9

图10是LabVIEW项目，在开发人员完成开发之后，必须编译文件。  
 图10

6.2.1驱动程序VI  
在DriverVI文件夹中，有驱动程序vi模板，其名称为DRV\_template.vi，其面板如图11所示。开发人员可以更改或修改前面板。  
右键单击已编译的文件

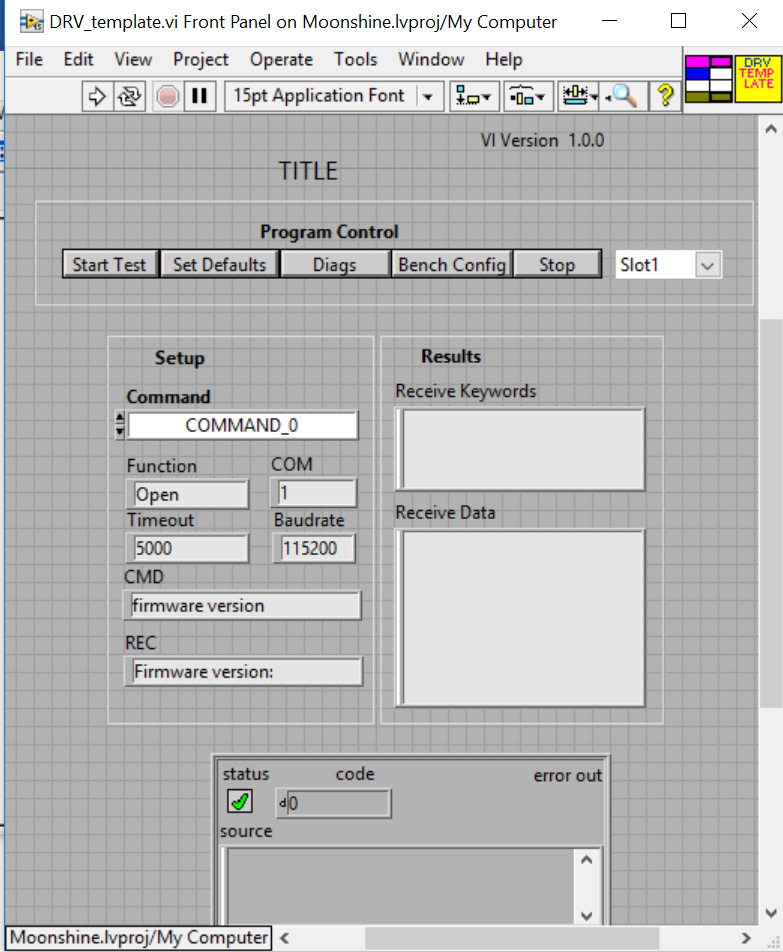


图11

图12是DRV\_template.vi的后面板，开发人员可以在“TEST”情况下添加其代码或修改模板的代码。

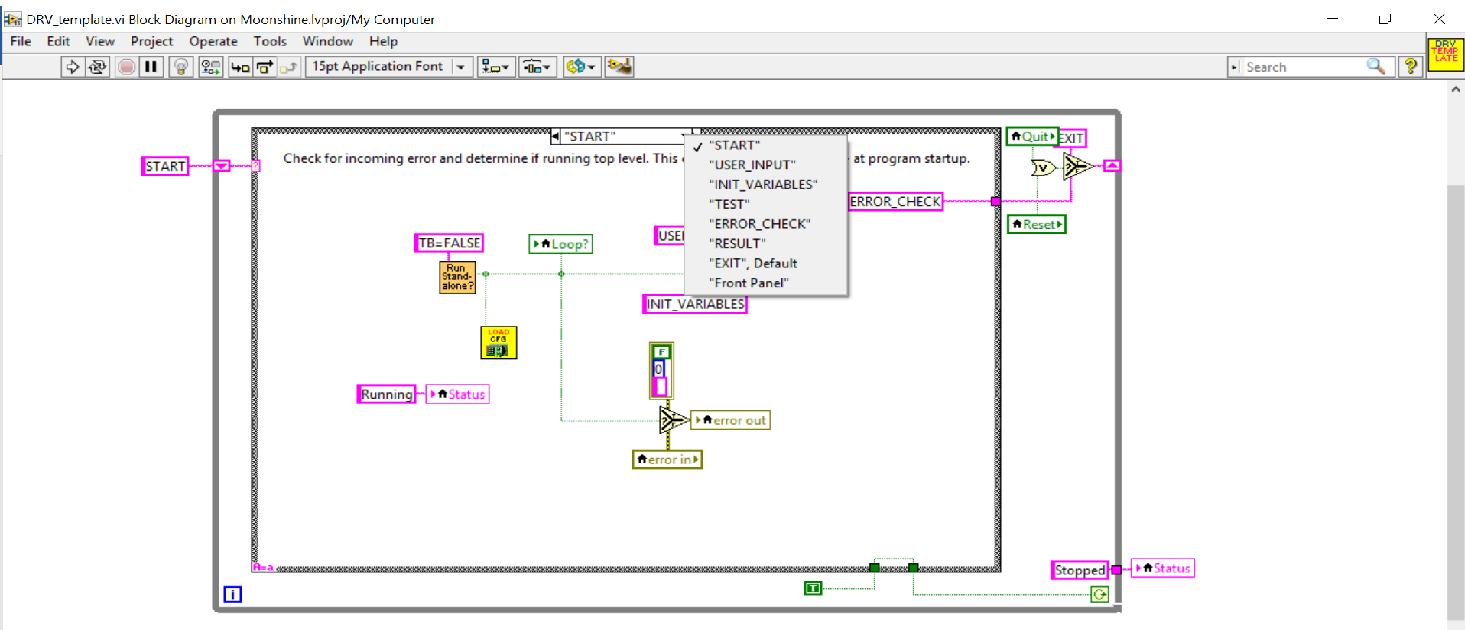


图12

图13是DRV\_template.vi的输入和输出接口，它是默认接口，无法更改。

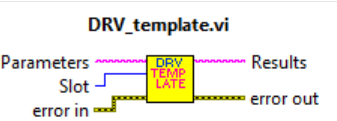


图13

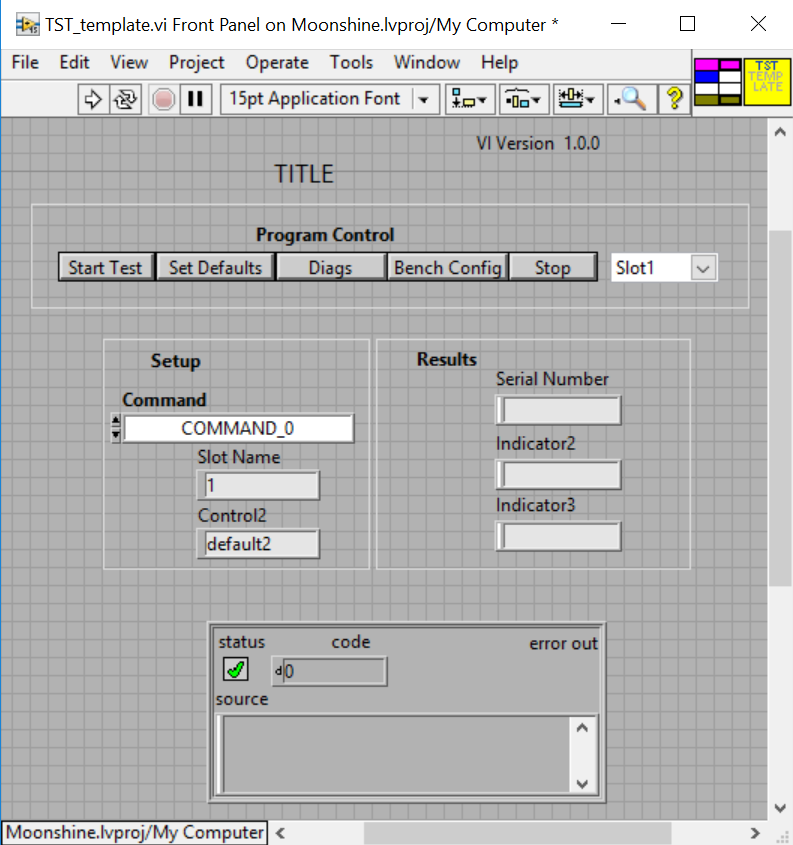
6.2.2测试VI  
在TestVI文件夹中，有test vi模板，其名称为TST\_template.vi，其面板如图14所示。开发人员可以更改或修改前面板。  


图14

图15是TST\_template.vi的后面板，开发人员可以在“TEST”情况下添加其代码或修改模板的代码。

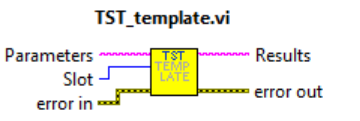
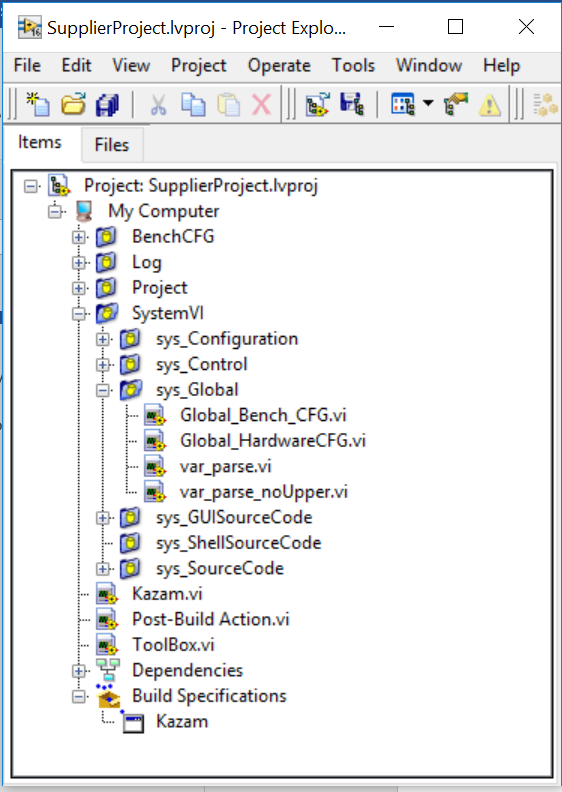
  
图16是TST\_template.vi的输入和输出接口，它是默认接口，无法更改。  


图15

图16

6.2.3记录VI

在SystemVI文件夹中，有日志模板，其名称为log\_Template.vi，并且有两个全局变量vi，如图17所示。



这是一个全局变量vi，它存储所有Beach变量

这是一个全局变量vi，它存储所有硬件配置信息

图17

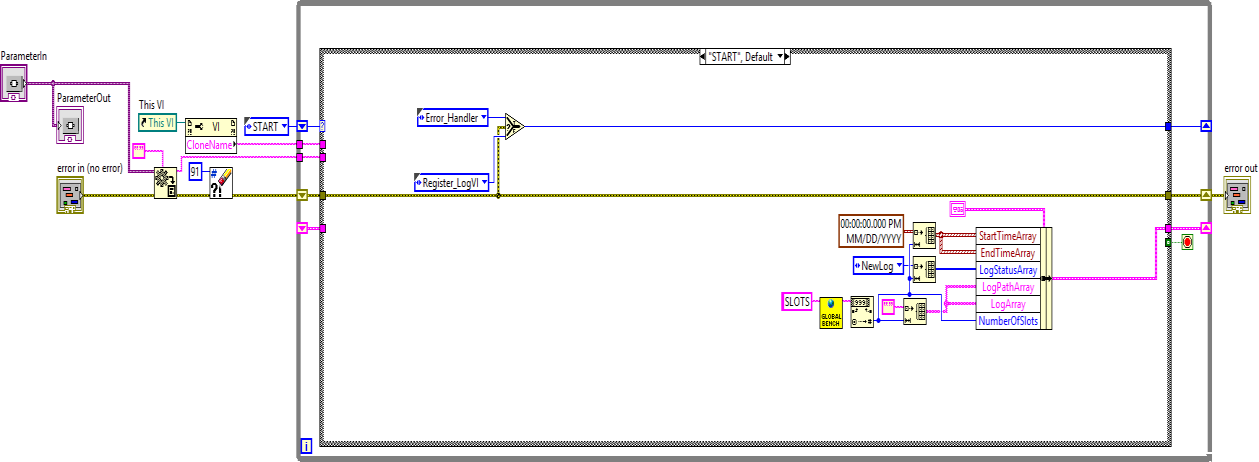
图18至图24是TST\_template.vi的后面板，这里有一些介绍，开发人员可以根据自己的需要自定义特殊的日志文件。

图18

这里有一些临时变量，开发人员可以添加/替换/修改/删除它们。

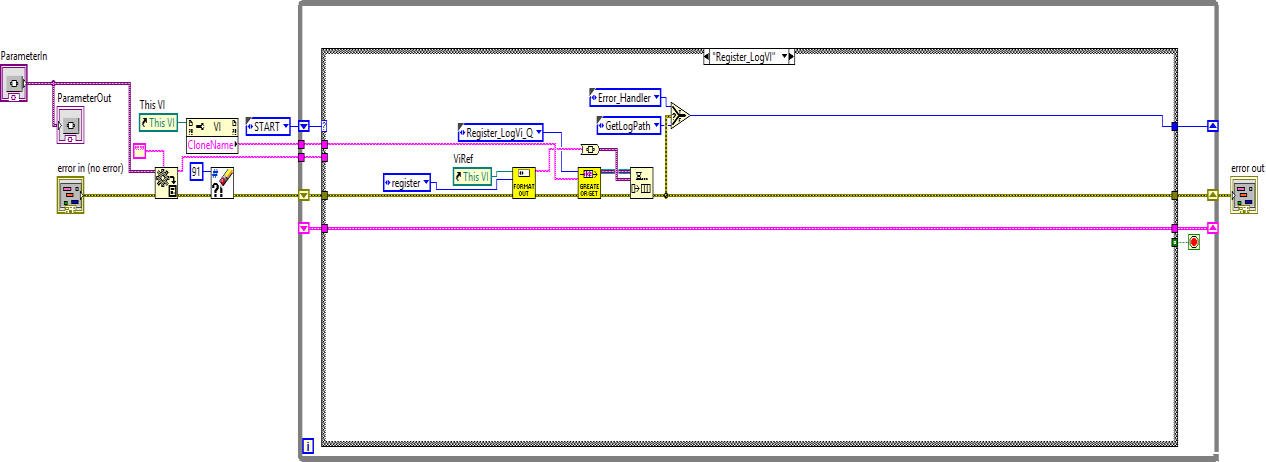
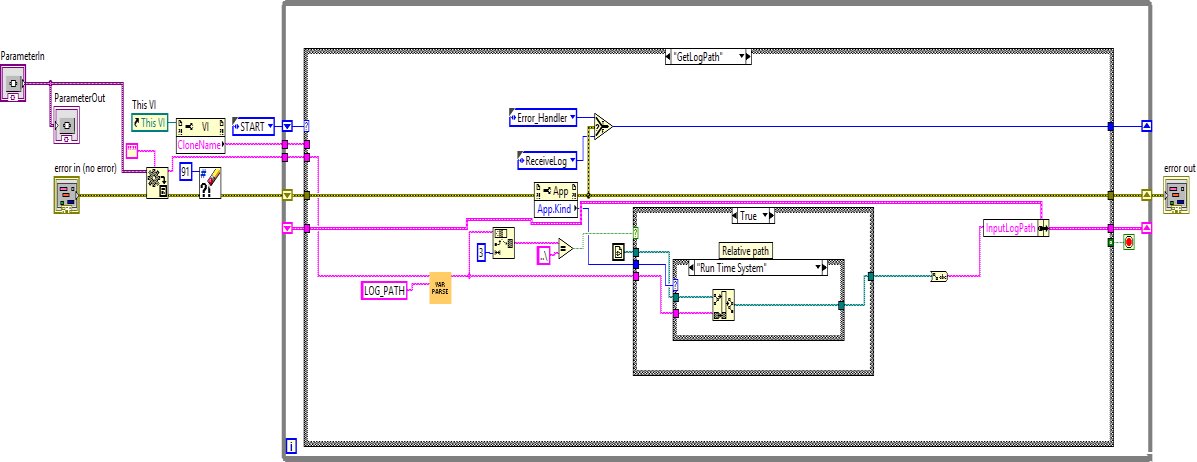


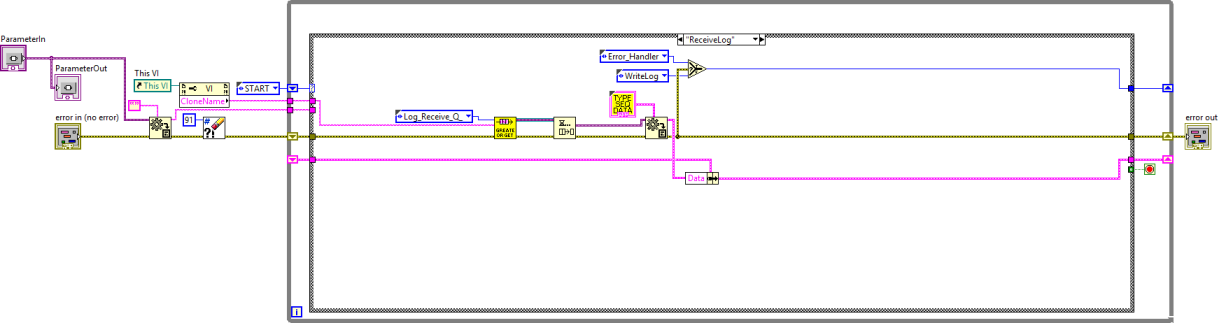
图19

此案例结构用于将日志注册到我们的软件平台。 无法改变



此案例结构用于获取日志文件存储路径。不建议更改，但如果您有特殊需求，可以修改它

图20



此案例结构用于接收日志数据。 无法改变。

图21

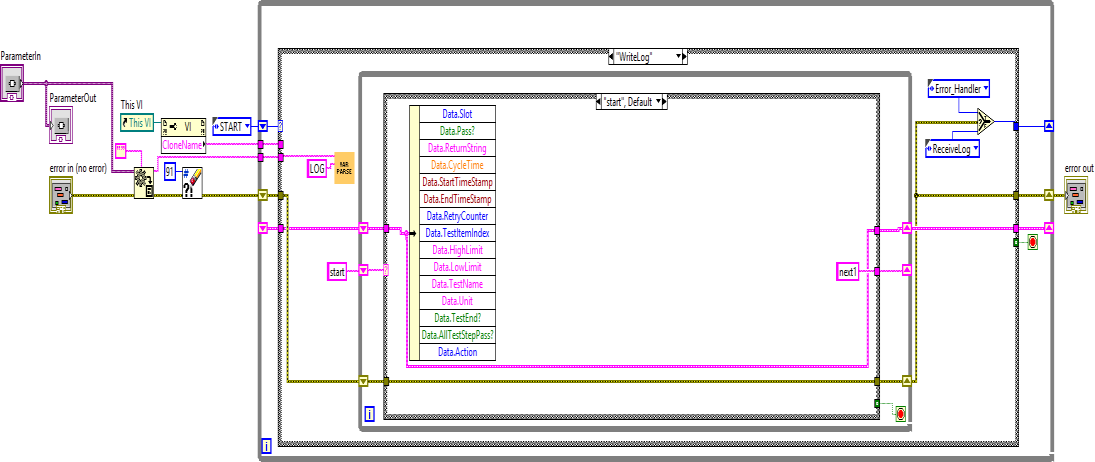
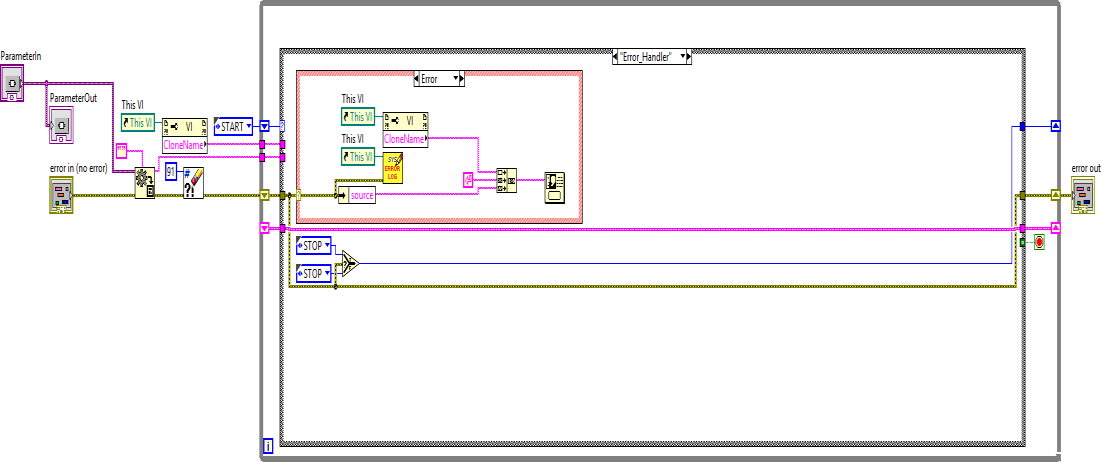


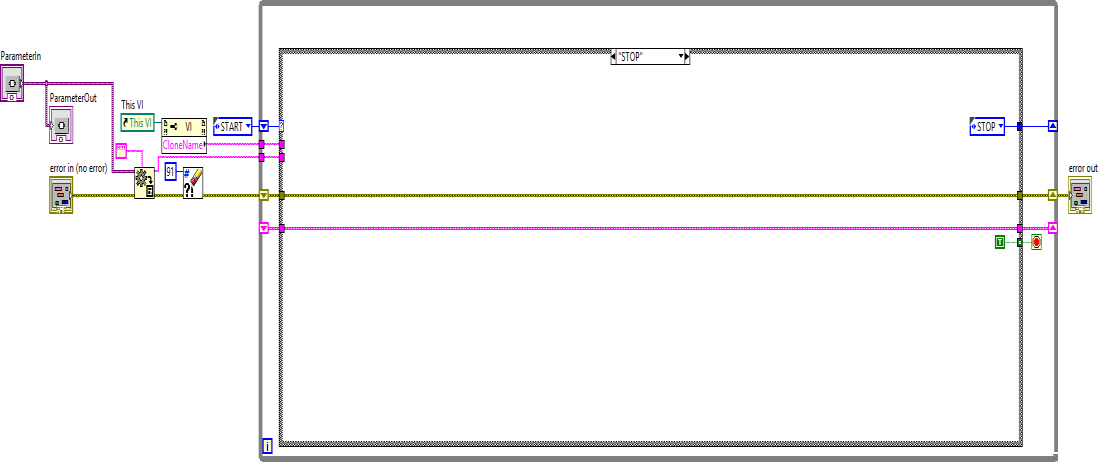
图22

此案例结构用于写入日志数据。开发人员需要编写自己的代码



此案例结构用于将错误信息写入系统日志，如果您有一些特殊需求，可以对其进行修改

图23



这个案例结构用于结束循环，如果你有一些特殊需求，你可以修改它

图24

6.2.4系统配置  
在BenchCFG文件夹中，这是两个文件，如图25所示。  
Bench\_Default\_Config.csv，它用于存储工作台信息，请不要重命名。  
shell\_factory.csv，它是Kazam.exe的配置文件，开发人员需要根据自己的需要进行编辑。

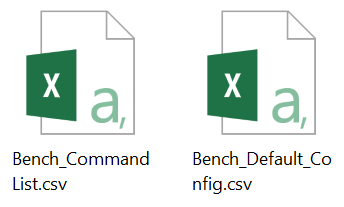


图25

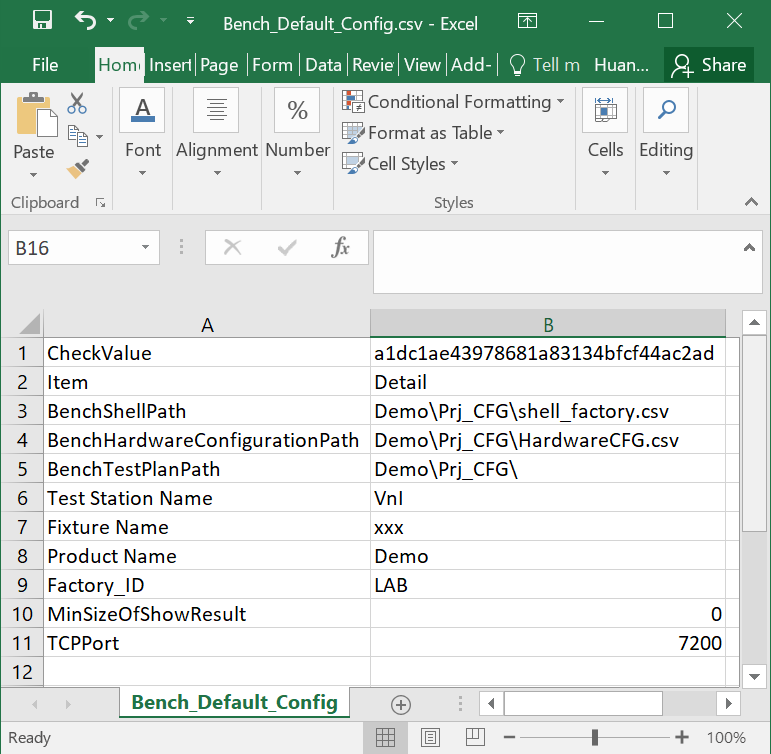
图26是工作台配置文件，在此文件中，有一些关键字，它们无法更改。  
此案例结构用于将错误信息写入系统日志，如果您有一些特殊需求，可以对其进行修改  
这个案例结构用于结束循环，如果你有一些特殊需求，你可以修改它   


图26

开发人员可以在此处添加新关键字。 并且所有新关键字将自动存储到Beach全局变量中。

以下是关键字的值。 开发人员可以根据需要进行编辑

它们是关键字，无法更改

“BenchShellPath”关键字的值是“Demo \ Prj\_CFG \ shell\_factory.csv”，这意味着Kazam.exe将首先运行“.. \ Project \ Demo \ Prj\_CFG \ shell\_factory.csv”文件。  
在shell\_factory.csv文件中，开发人员可以添加/替换/减少/修改软件平台的功能。  
如图27所示，它是一个演示shell配置，它有两个日志功能，一个GUI，它重新定义了插槽的编号。以下是此脚本的详细信息：  
Resource\_Name：Sequence\_config  
VI\_NAME：sys\_SequenceCFGReadout.vi  
CONTROL\_STRING关键字:(红色是关键字，蓝色是关键字的值）  
 TestPlan = DemoSequence1.0.0.1.csv  
 WaitUntilDone =真  
此步骤从“.. \ .. \ .. \ SystemVI \”路径调用sys\_SequenceCFGReadout.vi，它将加载TestPlan表单“.. \ Project \ xxx \ DemoSequence1.0.0.1.csv”路径，这个path由Bench\_Default\_Config.csv文件配置。 TestPlan名称是“DemoSequence1.0.0.1.csv”，这是一个关键字，它被称为“TestPlan”。

“TestPlan”是所有插槽的默认关键字，如果定义“TestPlan1 = Seq\_Slot1.csv”，则表示插槽1将使用此“Seq\_Slot1.csv”序列。  
如果定义“TestPlan2 = Seq\_Slot2.csv”，则意味着插槽2将使用此“Seq\_Slot2.csv”序列等等。  
例如：  
 TestPlan = DemoSequence1.0.0.1.csv | TestPlan1 = Seq\_Slot1.csv | TestPlan2 = Seq\_Slot2.csv

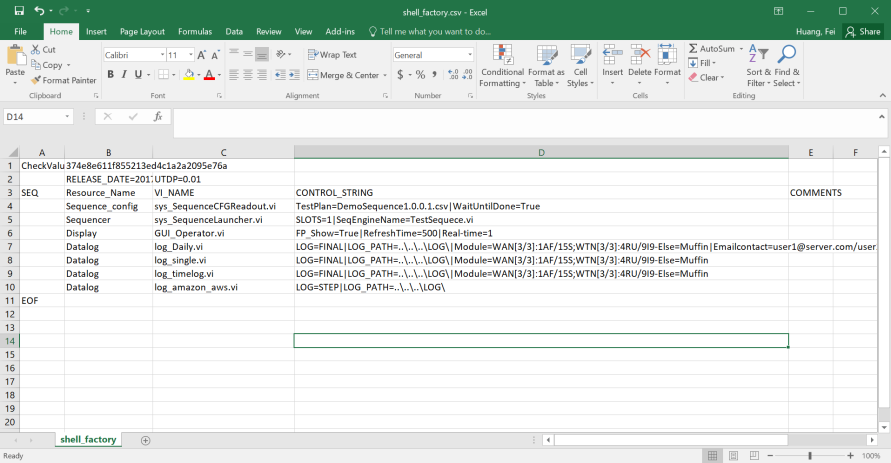


图27

6.2.5 TestPlan和硬件配置  
在Prj\_CFG文件夹中，这里有测试计划，Kazam配置和硬件配置，开发人员可以在此文件夹中存储不同的TestPlan。 如图28所示，这里有四个TestPlans和一个硬件配置文件。

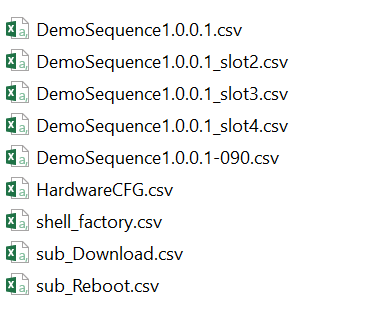


图28

6.2.5.1 测试计划  
图29是一个演示测试计划，有一些关键字，以下是详细说明：

|  |  |  |
| --- | --- | --- |
| 关键词 | 描述 | 例 |
| 指数 | 步测试索引，它是1到无穷大的整数范围 |  |
| 启用 | 它控制是否运行此测试步骤，其值为0,1，SFC = ON，SFC = OFF，Loop = ON和Loop = OFF。 如果值为1，则该步骤将运行 如果值为0，则该步骤将不会运行 如果值为SFC = ON，则当SFC为OFF时，该步骤将不会运行 如果值为SFC = OFF，则当SFC为ON时，该步骤将不会运行 如果值为Loop = ON，则当Kazam以循环模式运行时，该步骤将运行 如果值为Loop = OFF，当Kazam在操作员模式下运行时，该步骤将运行 “|”可以用作或条件 | SFC = ON |循环= OFF SFC = OFF |循环= OFF SFC = ON |循环= ON SFC = OFF |循环= ON 循环= ON 循环= OFF SFC = ON SFC = OFF 以上是所有条件 |
| PREREQ | 先决条件，该步骤的名称是此测试步骤的上一步骤 | “OpenPowerSupplyUART”的先决条件是“等待所有插槽获取SN” |
| 测试名称 | 测试步骤的名称。所有测试名称必须是唯一的 | GetSN |
| 测试Vi名称 | 测试vi的名称，如果没有后缀，则表示将调用VI;  该平台可以调用动态链接库和可执行文件 如果为空，则自动检索上一步的结果。 | TST\_Wait TST\_Wait.dll TST\_Wait.exe |
| LSL | 下限 |  |
| USL | 上限 |  |
| 单元 | 单位的限制，它将显示在GUI中 |  |
| LimitCheck | 如果该步骤有多个结果，则此关键字将告诉平台您需要哪个结果。 | 如果多重结果为“Vout = 3.8 | Iout = 110”，则“LimitCheck”为“Vout”，表示此步骤结果为“Result = 3.8” |
| 保存结果 | 如果值为1，则测试步骤的结果将存储在每日日志中;  如果该值为0，则测试步骤的结果将不会存储在每日日志中; |  |
| 区分大小写 | 如果值为“是”，则结果输出和全局变量将区分大小写 |  |
| 属性 | 属性=正常，这意味着当调用此测试vi时，此测试vi将并行运行  属性 = 同步，这意味着当调用此测试vi时，此测试vi将等待另一个测试VI，然后一起启动。 属性=一次性，这意味着这个测试vi只执行一次，这个属性常用于初始化仪器 属性= 共享，表示此测试vi是逐个执行的，此属性常用于共享资源，如工具 属性=发布，这意味着当调用此测试vi时，其中一个测试vi将运行并获取一些信息，然后将信息发送到其他插槽，例如获取面板SN 属性=一次，表示执行此克隆测试vi之一，另一个克隆将接收结果，此属性通常用于控制夹具的盖子或其他。 |  |
| 动态 | 如果值为1，则此测试步骤将为动态调用; 如果值为0，则此测试步骤将为预加载; |  |
| 输入参数 | 此步骤的参数。格式如下： 为keyword = value “|”是分隔符 | 功能=打开| COM = PSCOM |波特率= 115200 |
| 重试 | 如果该值为0，则此步骤失败，此步骤不会重试; 如果该值不为0，则此步骤失败，此步骤将重试; |  |
| 下一步CTL | 接下来，这意味着当完成此测试步骤时，下一步将是下一步 如果goto失败，则表示当此测试步骤失败时，下一个将运行指定的步骤 如果传递goto，则表示当通过此测试步骤时，下一个将运行指定的步骤 goto，这意味着当这个测试步骤完成后，下一个将运行指定的步骤 强制通过，表示当此测试步骤完成时，强制执行此步骤 强制失败，这意味着当此测试步骤完成时，强制执行此步骤失败 如果失败则停止，这意味着当此测试步骤失败时，测试序列将停止 如果......其他，则表示此步骤不是测试项目，其结果不会记录到最终结果，这是一个真或假的步骤。 |  |
| 下一个测试名称 | 当前步骤后将运行的下一步的名称 | 当我们使用“If ... else”来控制步骤流程时，我们将使用“|”来分隔真正的步骤和假步骤，例如：阅读固件版本|清理 |
| 转到柜台 | goto跳转语句的数量  如果该值为空，则表示该步骤将始终跳转 |  |
| 错误处理 | 开发人员可以为每个测试步骤定义错误处理程序 错误处理条目格式定义如下：  错误处理：测试步骤名称 如果错误处理程序为空，则将停止测试 | 0：EH\_0| 1：EH\_1|2：EH\_2| 3：EH\_3|4：EH\_4 |
| 错误代码 | 如果此步骤失败，则会将错误代码记录到日志中 |  |
| 注意 | 开发人员可以根据自己的需要编辑一些注释 |  |

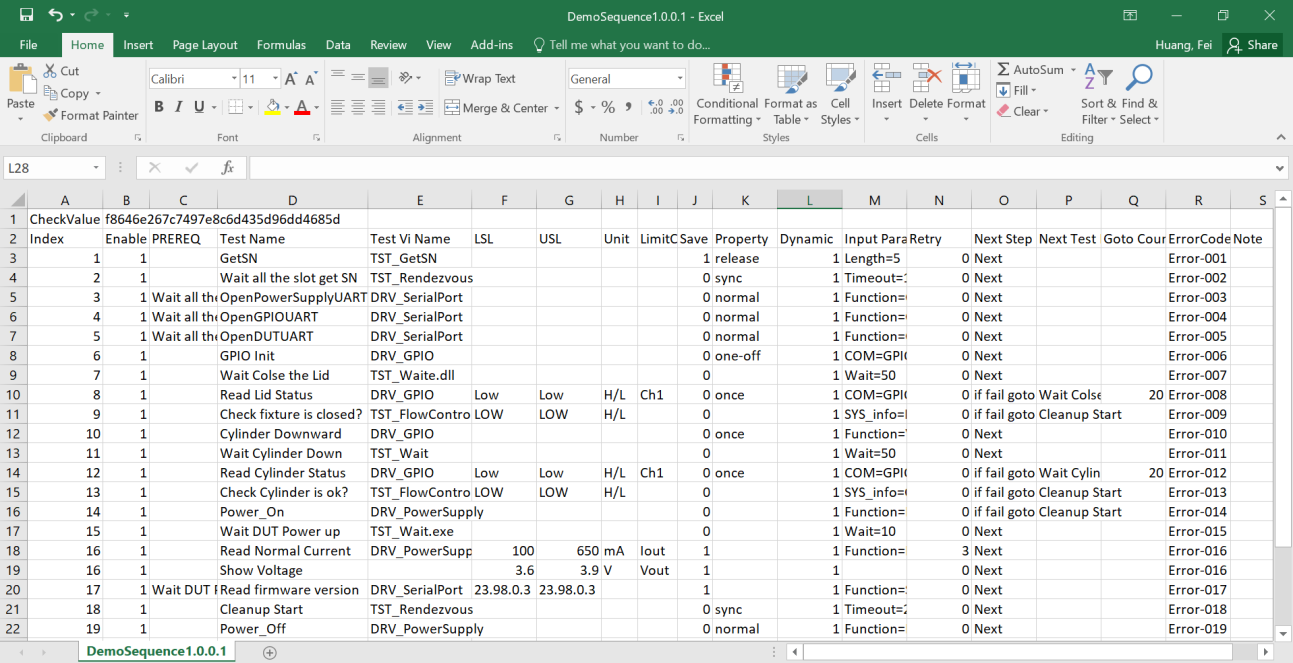
****

图29

6.2.5.2 Sub TestPlan  
开发人员可以定义子测试计划来替换重复测试步骤。

子测试计划

#1TestPlan# …

子测试计划

#2TestPlan#... TestPlan#1TestPlan#

子测试计划

#1TestPlan# …

测试计划

子测试计划

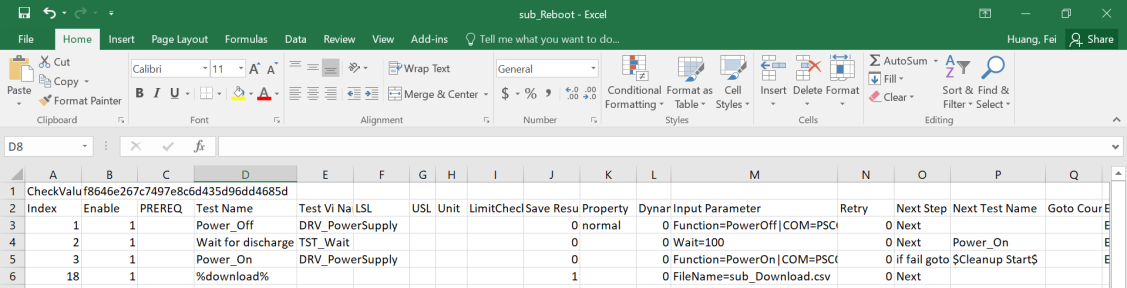
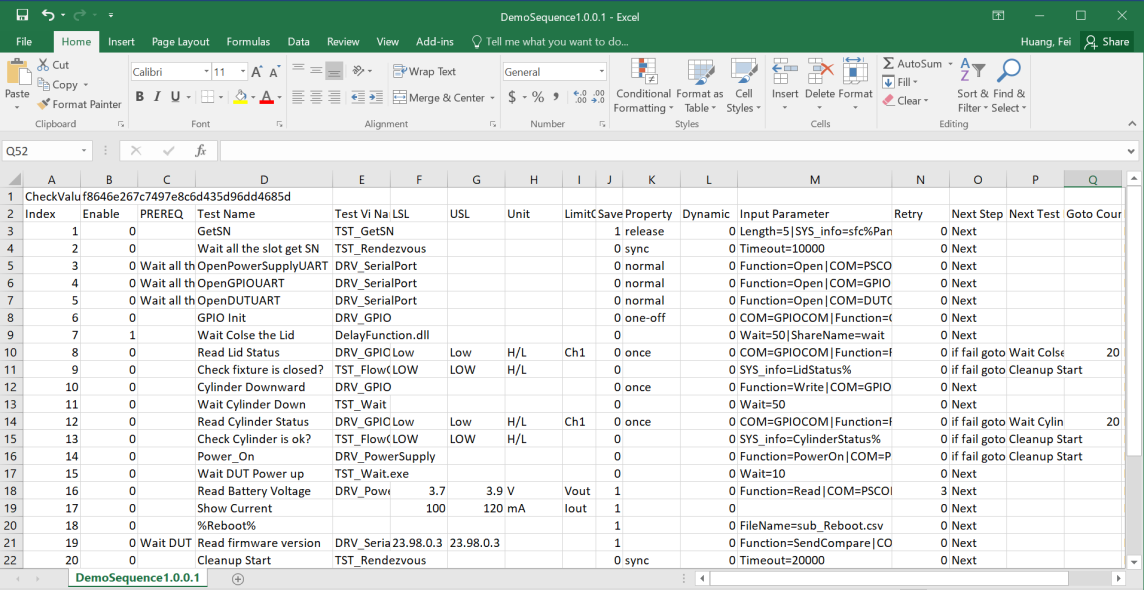
#NTestPlan#...

子测试计划

#NTestPlan#...

图30

如图31所示，开发人员可以使用％xxxxxx％格式告诉平台它是子测试计划，xxxxxx是子测试计划的名称，“％”是分隔符。



子测试计划名称

$”是分隔符，这意味着此步骤是最佳测试计划步骤称

子测试计划的文件名

子测试计划名称

子测试计划的文件名

图31

6.2.5.3硬件配置  
图32是演示硬件配置文件。  
在红色框中是资源的名称，开发人员可以自定义。  
在蓝色框中是插槽名称，它们是无法更改的关键字。  
在绿色框中是资源的价值，开发人员可以自定义

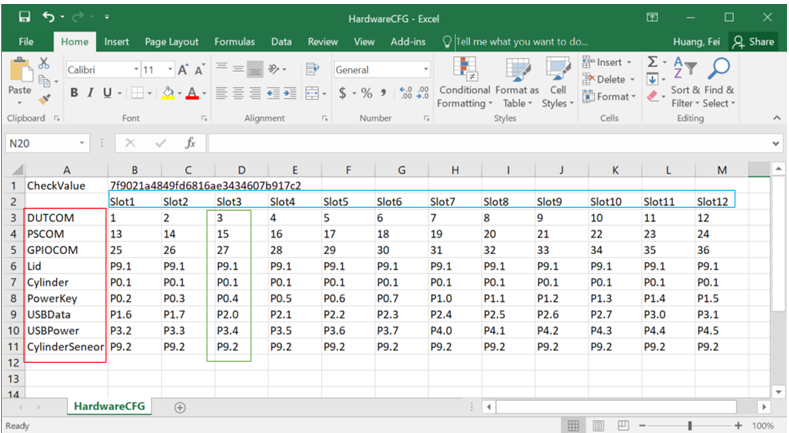


图32

6.2.6全局变量  
在TestPlan中，开发人员可以定义两种全局变量。

* 系统全局变量，可以定义如下：

Store= XXX> SYS\_info.YYY

Store是关键字，无法更改。  
XXX是测试步骤结果的关键字，开发人员可以自定义。  
SYS\_info是关键字，无法更改。  
YYY是全局变量的名称，开发人员可以自定义。  
例如，如果测试步骤的结果是“Ch1 = Low”，则开发人员想要存储此信息，其定义如下：  
 Store= Ch1 > SYS\_info.LidStatus  
开发人员可以将多个变量存储到系统全局变量中，格式如下：  
 Store = XXX> SYS\_info.YYY; ZZZ> SYS\_info.NNN; TTT> SYS\_info.QQQ

* Slot全局变量，每个槽的存储空间是分开的，可以定义如下：

Store = XXX> Slots\_info.YYY

Store是关键字，无法更改。

XXX是测试步骤结果的关键字，开发人员可以自定义。

Slots\_info是关键字，无法更改。

YYY是全局变量的名称，开发人员可以自定义。  
例如，如果测试步骤的结果是“serialnumber = 123456”，则开发人员想要存储此信息，其定义如下：

Store = serialnumber> Slots\_info.SN

开发人员可以将多个变量存储到系统全局变量中，格式如下：

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

* 如何获取全局变量值？

1. 开发人员需要定义如下，它意味着告诉平台你想要的变量  
    SYS\_INFO = LidStatus％
2. 在vi中，开发人员需要在输入参数中搜索“SYS\_info.LidStatus”关键字

开发人员还可以获取多个变量值，格式如下：  
 Slots\_info = SN％CylinderStatus％| SYS\_INFO = PanelSN％DUTCom％

* 如何直接在测试计划中使用全局变量？

“测试名称”，“输入参数”，“LSL”，“USL”和“ErrorCode”可以包含全局

TestPlan文件中的变量，并使用<<和>>符号标记如下：

<< Slots\_info.SN >>或<< SYS\_info.PanelSN >>

例如，如果“输入参数”是CMD = idme serial << Slots\_info.SN >>，则为该值

Slots\_info.SN是P012345678，因此“输入参数”将被CMD = idme替换

串行P012345678

* 保留变量名称  
  PanelSN是系统全局变量的保留名称，开发人员可以从此全局变量中获取面板序列号

SFC是系统全局变量的保留名称，开发人员可以从此全局变量获取SFC的状态，SFC的值为ON和OFF

SN是插槽全局变量的保留名称，开发人员可以从此全局变量中获取插槽序列号，并且在测试期间可由开发人员修改。  
Operator是系统全局变量的保留名称，开发人员可以从此全局变量中获取运算符ID，并且在测试期间可由开发人员修改。  
Diag\_Ver是slot全局变量的保留名称，开发人员可以从此全局变量中获取slot diag版本，并且在测试期间可由开发人员修改。  
Build\_Config是插槽全局变量的保留名称，开发人员可以从此全局变量中获取插槽构建配置信息，并且开发人员可以在测试期间对其进行修改。可能需要从SFC服务器读取然后写入此全局变量  
RunMode是slot全局变量的保留名称，开发人员可以从此全局变量中获取slot运行模式信息。只有两种模式，操作员模式和循环模式，例如：RunMode = LOOP或RunMode = OPERATOR

Image\_Name\_Ver是slot全局变量的保留名称，可以通过修改开发人员在测试期间。 它是设备软件映像包名称和版本。

Amazon\_HW\_Ver是slot全局变量的保留名称，它可以通过修改开发人员在测试期间。 它是硬件版本，例如HVT2。  
格式如下  
 SYS\_INFO = PanelSN％SFC％Operator％| Slots\_info = SN％Diag\_Ver％Build\_Config％RunMode％

6.3开发步骤  
1.使用DRV\_template.vi开发驱动程序vi  
2.使用TST\_template.vi开发测试vi  
3.使用log\_Template.vi开发日志vi（如果开发人员需要特殊日志）  
4.编辑硬件配置文件  
5.编辑TestPlan  
6.编辑shell配置文件  
7.编辑工作台配置文件  
8.构建Labview项目，如图33所示  
9.选择源文件夹，它是供应商项目的编译路径，如图34所示  
10.选择目标文件夹，它是Kazam.exe文件夹的路径，如图35所示  
注意：编译前需要关闭所有CSV文件，例如TestPlan，shell配置文件等

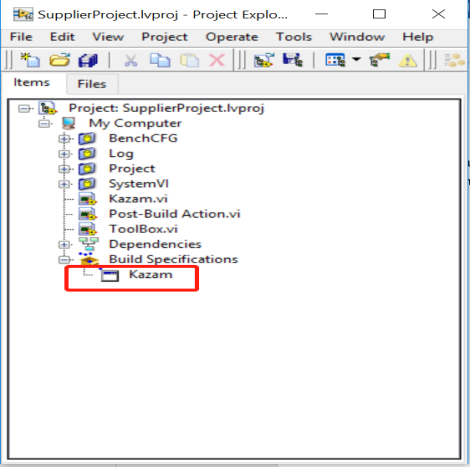


图33

建立

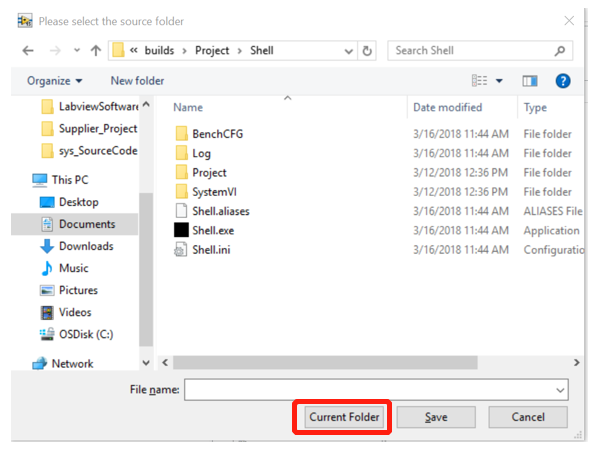


图34

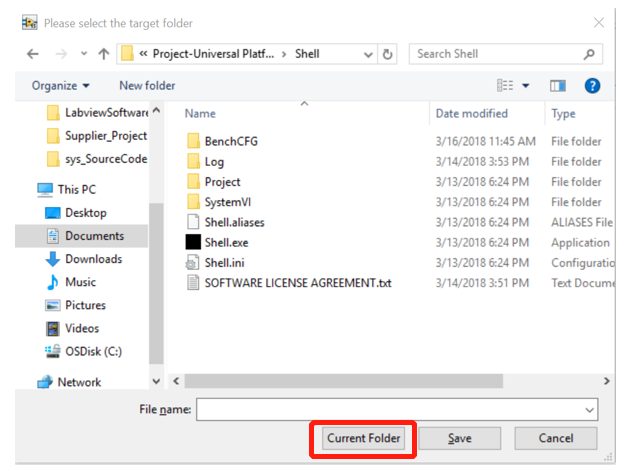


图35

6.4 Labview运行时版本  
开发人员需要从以下网站下载“LabVIEW Run-Time Engine 2016 - （64位） - Windows”：

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

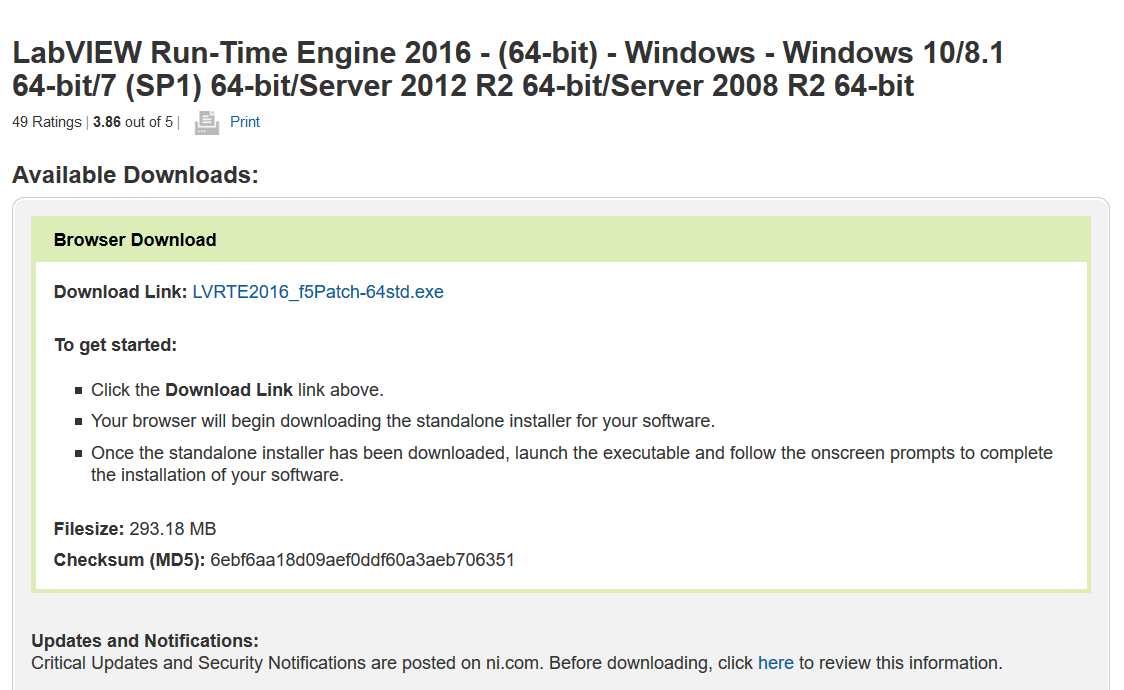


图36

6.5其他语言或EXE支持

6.5.1 EXE支持  
该平台使用Windows命令行来调用执行文件。 输入参数的定义如下：

XXX= 123| YYY= ABC| ZZZ= DEF

XXX是参数名称，开发人员可以自定义  
123是参数值，开发人员可以自定义  
“=”和“|”是分隔符，不能更改

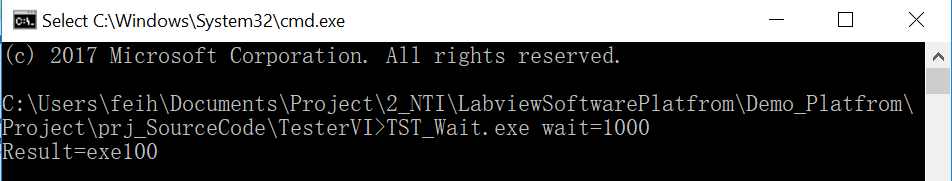


图35

如图35所示，Result = exe100是返回值  
6.5.2 C / C ++ DLL支持  
该平台可以直接调用C或C++ DLL。 DLL函数定义如下：

CStr AME\_Function（const CStr ParametersIn）;

AME\_Function是函数名，它不能改变，函数返回值的类型是字符串，格式与执行文件返回值相同。  
const CStr ParametersIn是函数的输入，参数名不能改变，输入参数的类型是字符串，格式与执行文件返回值相同

7.外部API  
该平台使用TCP / IP与外部软件通信，默认端口为7200，开发人员可以在Bench\_Default\_Config.csv文件中对其进行修改。 Kazam在服务器模式下工作，外部软件需要使用TCP客户端进行访问。

7.1命令摘要  
命令格式如下：

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

CMD是外部命令的关键字，不能更改。  
XXX是一个命令，可以在下表的命令列中找到它们。  
RequiredParameter是一个必须的关键字，可以在下表所需的参数列中找到它们，有些命令不需要这个参数。  
YYY是所需参数的值。  
OptionalParameter是可选参数的关键字，开发人员可以决定是否需要它们。  
ZZZ是可选参数的值。

\ r \ n表示CR（回车）后跟LF（换行），它是结束字符。

注意：所有命令和参数都不区分大小写

|  |  |  |  |
| --- | --- | --- | --- |
| 命令 | 必需参数 | 可选参数 | 描述 |
| StartAll |  | PanelSN | 开始所有测试 |
| StartOne | 插槽 | SN | 开始测试一个插槽 |
| SetMode | 模式=操作员 模式=循环 |  | 设置测试模式 |
| SetSFC | SFC = ON SFC = OFF |  | 设置SFC |
| SetLoopCounter | 循环计数器= 100 |  | 设置循环次数 |
| ForceExit |  |  | 强行退出Kazam.exe |

例如：  
外部软件发送以下命令：

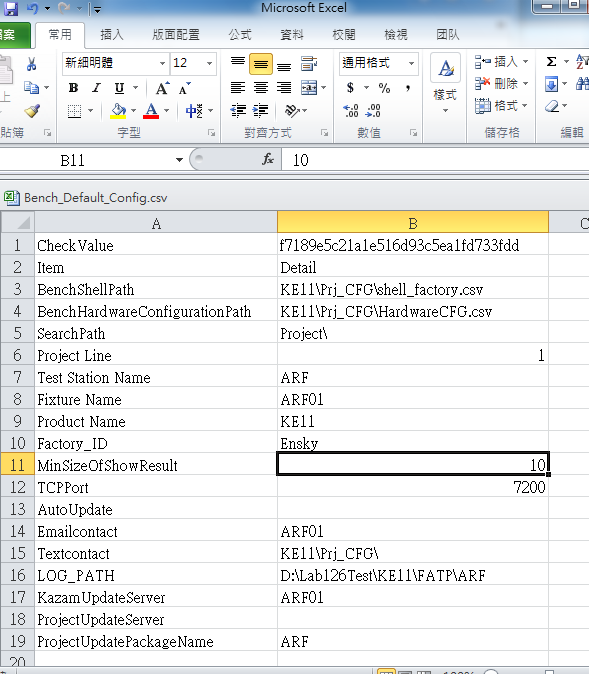
CMD = StartAll | PanelSN = 30975  
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息

7.1.1命令详细说明  
StartAll：此命令将使所有插槽开始测试。  
外部软件发送以下命令：  
CMD = StartAll | PanelSN = 30975  
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息  
PanelSN用于告诉平台面板序列号。  
-------------------------------------------------- ----------------------------------------------  
StartOne：此命令将使一个插槽开始测试。  
外部软件发送以下命令：

CMD = StartOne | Slot = 1 | SN = 12345678  
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息  
插槽用于告诉平台选择了哪个插槽。它的范围从1到32。  
SN用于告诉平台该插槽的序列号。  
------------------------------------------------------------------------------------------------  
SetMode：此命令设置平台的运行模式，循环模式或操作员模式。  
外部软件发送以下命令：  
CMD = SetMode | Mode =Operator   
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息

模式用于告诉平台选择哪种模式。它的值是循环或运算符。  
----------------------------------------------------------------------------------------------------  
SetSFC：此命令将设置SFC的状态，在线或离线。  
外部软件发送以下命令：  
CMD = SetSFC | SFC = ON  
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息  
SFC用于告诉平台是在线还是离线测试。其值为ON或OFF。  
----------------------------------------------------------------------------------------------  
SetLoopCounter：此命令将设置循环次数。  
外部软件发送以下命令：  
CMD = SetLoopCounter | LoopCounter = 100  
如果没有错误，平台将返回“成功”  
如果有错误，平台将返回错误信息  
LoopCounter用于告诉平台需要循环多少次。它的值是-1或1~2147483648，“ - 1”表示永远。  
-------------------------------------------------- --------------------------------------------------  
ForceExit：此命令用于强制退出平台。  
外部软件发送以下命令：  
CMD = ForceExit  
此命令不返回值。

8.重要参数  
8.1 Bench\_Default\_Config.csv  
Bench\_Default\_Config.csv文件中预定义了一些重要参数



关键字的值

关键字

图36

Project Line：用于定义项目行的名称。它将用于定义文件名登录。

AutoUpdate：用于设置是否自动更新项目文件夹。它的价值在于启用或禁用。

注意：启用项目文件夹时，Kazam会自动检查每个文件的“CheckValue”更新

ProjectUpdateServer：当Kazam在exe中运行时，用于定义项目服务器路径模式，它会自动检查更新服务器中是否有更新。

ProjectUpdatePackageName：当Kazam为时，用于定义更新包的名称在exe模式下运行时，它会自动检查更新包的名称，看是否有在更新服务器中更新。

KazamUpdateServer：用于定义Kazam在exe中运行时的kazam服务器路径

模式，它会自动检查更新服务器中是否有更新。

Emailcontact：用于设置联系人的邮箱。它将被写入日常日志。

Textcontact：用于设置联系人的电话号码。它将被写入日常日志。

LOG\_PATH：用于设置日志的存储路径。它可以设置多个路径，并隔离每个路径

用分号的路径。

BenchShellPath：用于定义shell配置文件路径。

Ben chHardwareConfigurationPath：用于定义硬件配置文件路径。

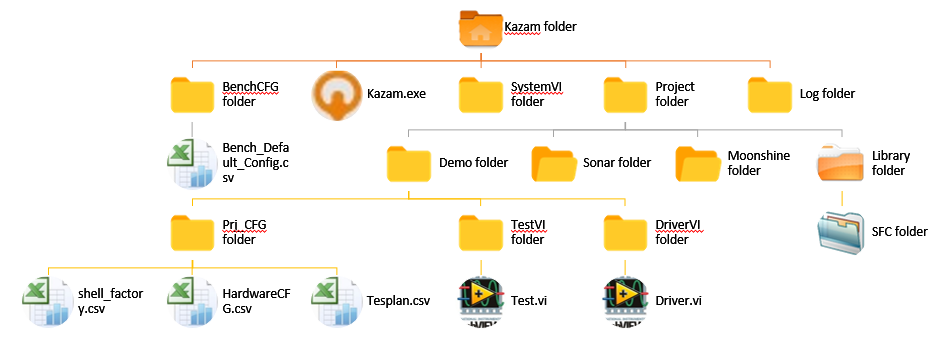


图37

SearchPath：它用于定义搜索路径，它可以是多个路径，它允许开发人员进入

绝对路径和相对路径，并用分号隔离每个路径，例如：

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

在这种情况下，Kazma将首先搜索kazam.exe路径下的“Project \ Demo”路径，如果kazam

找到具体的测试计划，kazam将停止搜索，否则将继续搜索下一条路径。该

第二步，kazam将从第一个搜索路径开始搜索驱动程序并测试vi。

测试站名称：用于定义测试站的类型，如AFT，VnI，MT，PT等

上。它将被写入日常日志

夹具名称：用于定义夹具名称，例如BFT01，BFT02，AFT01等。它会

被写入日常日志，将用于定义日志的文件名。

产品名称：用于定义产品名称，如Moonshine，Muffin，Sonar等

上。它将被写入日常日志，并将用于定义日志的文件名。

Factory\_ID：用于定义工厂名称。也许将来会有新的规则。这将是

翻到日常日志中。

MinSizeOfShowResult：用于定义显示结果的长度。如图38所示，有11个结果需要在GUI上显示。所以节目结果的长度是11。

如果开发人员想要控制每个节目结果的“下一步”，那么MinSizeOfShowResult

需要设置为大于11，例如12。

如果开发人员不想控制每个节目结果的“下一步”，那么MinSizeOfShowResult需要设置为小于11，例如10。

MinSizeOfShowResult是一个非常重要的参数，请注意设置它，否则测试时间可能会变得很长

TCPPort：用于定义TCP端口。 它的默认端口是7200。

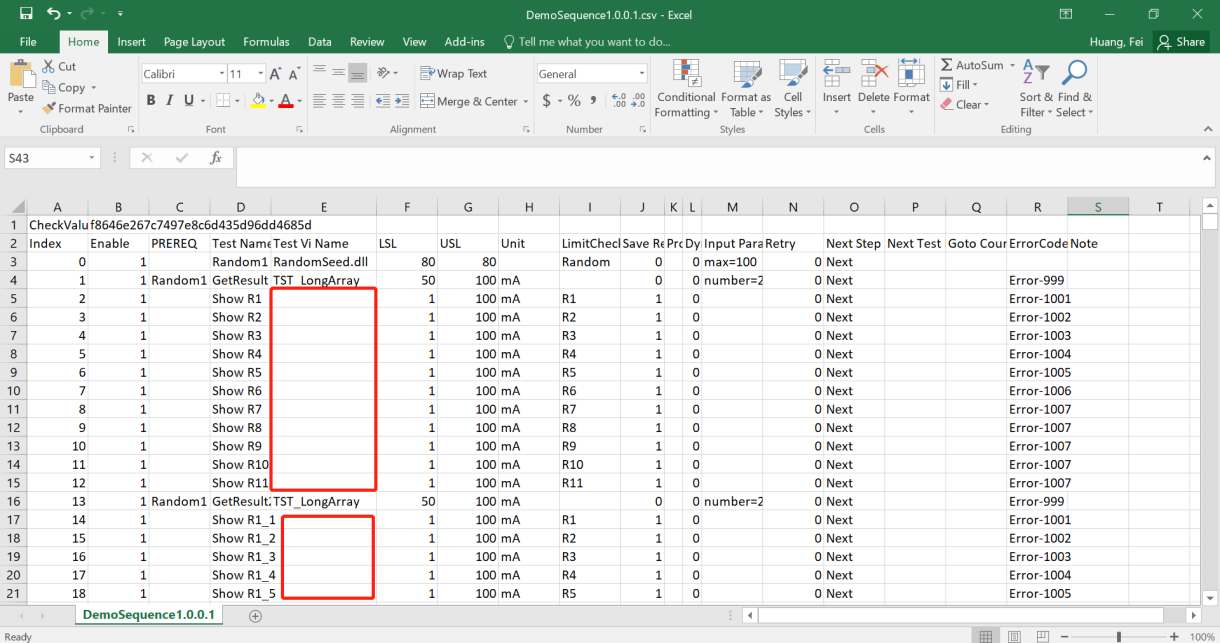


图38

**8.2 Kazam配置文件**

Kazam配置文件用于定义Kazam的功能，例如使用什么日志vi，什么GUI使用，等等。

如图42所示，它是一个Kazam配置文件，参见下面的参数定义：



图39

**8.2.1通用参数**  
可以在Kazam配置文件中的任何位置使用公共参数。

|  |  |  |
| --- | --- | --- |
| 常用参数名称 | 描述 | 示例 |
| WaitUntilDone | 它用于告诉平台等待此vi执行的完成 WaitUntilDone = True，表示平台需要等待此vi执行完成 空，表示无需等待执行结束 | WaitUntilDone=True |

**8.2.2其他参数**  
对于每个函数，开发人员可以定义其参数。 以下是平台定义的参数

|  |  |  |
| --- | --- | --- |
| sys\_SequenceLauncher.vi  参数名称 | 描述 | 示例 |
| 插槽 | 它用于告诉平台将使用多少个插槽 | SLOTS= 4 |
| SeqEngineName | 它无法修改 | SeqEngineName=TestSequece.vi |
| 测试计划 | 它用于告诉平台您要使用哪个测试计划。“TestPlan”是默认值所有插槽的关键字 | TestPlan = DemoSequence1.0.0.1.csv |
| TestPlan1 | 它用于告诉平台slot1将使用此测试计划。 | TestPlan1 = Seq\_Slot1.csv |
| TestPlan2 | 它用于告诉平台slot2将使用此测试计划。 | TestPlan2 = Seq\_Slot2.csv |
| ...... | ...... | ...... |
| TestPlan32 | 它用于告诉平台slot32将使用此测试计划。 | TestPlan32 = Seq\_Slot32.csv |

|  |  |  |
| --- | --- | --- |
| GUI\_Operator.vi  GUI\_Single.vi  参数名称 | 描述 | 示例 |
| FP\_Show | 它用于告诉平台需要显示此GUI | FP\_Show=true |
| RefreshTime | 它用于设置GUI的刷新时间，  刷新时间必须设置为10批量生产的秒数 | RefreshTime= 10000 |
| 即时的 | 它用于设置是否实时显示测试结果，其值为0或1 对于批量生产，实时必须设置为0 | Real-time=0 |
| 可最小化 | 它用于禁用GUI最小化，其值为Disable或Enable | Minimizable=Enable |
| SFC | 它用于告诉平台默认情况下SFC是ON还是OFF | SFC = OFF |
| 延迟 | 在循环模式下，它用于设置此运行和下一个运行之间的延迟 | Delay=3000 |
| SlotPosition | 它用于告诉平台此GUI的插槽号，其值为1,2,3，... 32 | SlotPosition= 1 |
| AutoScrolling | 它用于启用自动滚动，其值为禁用或 | AutoScrolling= Enable |
| ToolPath | 它用于指定路径开发人员自己的工具vi | ToolPath =demo\ TesterVI\ TST\_tool.vi  ToolPath= C：\Tools \ TST\_tool.vi |
| log\_Daily.vi  log\_single.vi  log\_timelog.vi  参数名称 | 描述 | 示例 |
| LOG\_PATH | 用于设置日志存储路径它可以是多条路径，用分号分隔 | LOG\_PATH=[\\10.117.119.235\Moo](file:///\\\\10.117.119.235\\Moo)shine\ PVT\ SMT\ BFT\ BFT03;.. \.. \ ..\ LOG\ |
| 模块 | 用于定义方法区分不同序列号中的模块。其格式如下：  Module=ModuleName[Offset tosearch string/Length of specialstring]: Special string  用户可以使用';'来分隔不同的模块，并使用  ' – Else = ModuleName' | Module=WAN[3/3]:1AF/15S/1EA/1EQ/1F4/1F1/;WTN[3/3]:4RU/9I9-Else=WIFI |

log\_SFC.vi需要由CM或供应商开发，而这个vi需要保存到开发人员的SFC文件夹中需要为他们的工厂创建一个新文件夹。

|  |  |  |
| --- | --- | --- |
| log\_SFC.vi  参数名称 | 描述 | 示例 |
| LOG\_PATH | 它用于设置日志存储路径  它可以是多条路径，用分号分隔 | LOG\_PATH=[\\10.117.119.235\Moo](file:///\\\\10.117.119.235\\Moo)shine\ PVT\ SMT\ BFT\ BFT03; .. \.. \ ..\ LOG\ |
| VI\_PATH | 它用于设置日志vi路径 | VI\_PATH=..\Project\SFC\Compal\BFT\ |