Cygnus SDK平台自动化框架概要设计

© 2022 Mooresilicon All rights reserved.

本文档版权归TCL摩星半导体所有，受相关法律法规的保护。未经书面许可不得复制传播。

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 作者 | 说明 |
| draft | 2022-02-25 | 孙茂杰 | 起草 |
| 0.1 | 2022-03-31 | 薛鹏宇 | 初稿 |

目录

[1 引言 4](#_Toc99560559)

[1.1背景-测试对象与测试方法 4](#_Toc99560560)

[1.2 约束 4](#_Toc99560561)

[1.3术语 4](#_Toc99560562)

[1.4参考资料 5](#_Toc99560563)

[2总体设计 6](#_Toc99560564)

[2.1需求规定 6](#_Toc99560565)

[2.2架构设计 6](#_Toc99560566)

[3运行设计 10](#_Toc99560567)

[3.1 测试启动流程 10](#_Toc99560568)

[3.2测试过程中硬件板配置流程 10](#_Toc99560569)

[4.接口设计 11](#_Toc99560570)

[4.1用户接口 11](#_Toc99560571)

[4.2外部接口 11](#_Toc99560572)

[5. 模块设计 12](#_Toc99560573)

[5.1 测试流程管理模块 12](#_Toc99560574)

[5.1.1模块逻辑框图 12](#_Toc99560575)

[5.1.2实现逻辑说明 12](#_Toc99560576)

[5.2 测试bin下载模块 （略） 13](#_Toc99560577)

[5.3 功能测试用例脚本解析(python脚本) 13](#_Toc99560578)

[5.3.1模块逻辑框图 13](#_Toc99560579)

[5.4 Arduino / STM32程序（命令和配置控制模块） 13](#_Toc99560580)

[5.4.1模块逻辑框图 (Arduino为例） 13](#_Toc99560581)

[5.4.2通信命令定义(Arduino为例） 14](#_Toc99560582)

[5.5 硬件板通信模块 14](#_Toc99560583)

[5.6 测试结果和报表模块 (略) 14](#_Toc99560584)

[5.7 自动化测试后台服务模块 15](#_Toc99560585)

[5.7.1模块逻辑框图 15](#_Toc99560586)

[5.7.2 后台服务和PC端通信定义 15](#_Toc99560587)

[6.数据结构设计 16](#_Toc99560588)

[6.1数据结构一（从模块之间的接口中抽象而来） 16](#_Toc99560589)

[数据流一(*数据流是根据系统的运行控制中抽象而来的*) 16](#_Toc99560590)

[逻辑结构设计 16](#_Toc99560591)

[7. 异常处理设计 17](#_Toc99560592)

[8. 系统扩展 18](#_Toc99560593)

# 1 引言

## 1.1背景-测试对象与测试方法

Cygnus SDK平台的自动化测试有三个被测对象

1. 平台提供的组件(component)。例如platform驱动，OS，BT，PM等。每个组件都写了单元测试用例（准确描述为接口测试），因此主要测试方法为自动化单元测试。
2. 基于这些组件开发的example apps。例如SDK平台提供了蓝牙遥控器app。这个app用到SDK平台的驱动，BT，PM等组件，可以作为客户开发定制app的原型。事实上 对这些app进行测试的同时，也对组件进行了测试。对于这些app本身，主要测试方法是自动化黑盒功能测试。
3. 芯片。芯片测试代码是基于 SDK平台 platform组件的HAL和LL层编写。芯片测试代码和组件的单元测试一样，通过单元测试框架进行测试。主要测试方法为自动化单元测试。

综上，SDK平台的自动化测试框架，要求能够对芯片测试，平台的组件单元测试以及 example apps进行自动化测试，支撑整个测试流程包括 配置测试内容，下载并执行可执行测试文件，汇总测试结果等。

## 1.2 约束

1. SDK平台的自动化测试框架 需要硬件测试板的支持。自动化测试框架本身包含了 板子端和PC端的部分。PC端部分以后我们都放到测试服务器运行。在测试服务器搭建之前，PC端部分就是一个发起测试执行的应用程序，主体是下文描述的测试理层管理。

1. 测试用例管理需要满足以下条件（在测试服务器搭建之前，可以在测试目录里面管理）：

-单元测试用例： 测试服务器上的每个测试用例只包含测试描述（前置条件 / 测试步骤 / 期望结果）

-自动化功能测试用例： 测试服务器上的每个测试用例包含测试描述以及对应的自动化测试python脚本

-手动功能测试用例: 测试服务器上的每个测试用例只包含测试描述（不在本文档描述范围内）

## 1.3术语

* 硬件测试板：为支持自动化测试定制的硬件板子。该板子包含 FGPA小板(后续可替换成SOC芯片板)，支撑自动化测试的外围器件板(Arduino板，STM32板)
* 测试服务器：为了支持CICD搭建的测试服务器，主要功能包括测试用例管理，测试配置，自动化测试服务，测试结果展现等功能
* 组合test case：特殊情况下两个测试用例是需要在组合在一起测试的。例如进入睡眠的测试用例和推出睡眠之后的系统检查用例（或者说这两个case放在一起才是完整的测试）。

## 1.4参考资料

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 资料名称 | 版本 | 作者 |
|  |  |  |  |
|  |  |  |  |

# 2总体设计

## 2.1需求规定

1. 三个测试对象的 自动化测试前置条件以及自动化测试要达成的目的说明如下：

|  |  |
| --- | --- |
| 测试对象 | 目的说明 |
| SDK平台各组件 | 前置条件：   1. 各个组件有覆盖充分的单元测试用例 2. 单元测试用例通过单元测试框架(unity test)串起来，可以编译生成可执行程序下载运行   平台自动化测试要达成的目的：   1. 能够根据测试配置选择需要做单元测试的组件和测试用例 2. 部分单元测试用例需要外界硬件支持（例如SPI传输，串口传输测试），自动化测试框架要能响应单元测试的要求，对硬件测试板做相应的配置。 |
| SDK平台example apps | 前置条件：   1. 平台的每个example app 可以编译成独立的可执行文件，可以单独下载运行 2. 对每个example app，有相应的自动化功能测试用例 (用python脚本写的黑盒功能测试用例)   平台自动化测试要达成的目的：   1. 能够根据测试配置选择需要测试的app 和测试用例。 2. 部分app运行过程中需要用户数据输入（例如按键输入，语音输入），自动化测试框架能够解析测试用例的脚本，通过和Arduino/STM32的命令交互，在合适的时间点，通过硬件测试板做响应的输入（达到类似机械手的作用）。 |
| 芯片 | 前置条件：   1. 芯片测试程序基于SDK 平台 platform组件的 hal层和ll层来写   2． 芯片测试程序 通过单元测试框架(unity test)串起来，生成芯片测试可执行程序  平台自动化测试要达成的目的：同SDK平台各组件测试。 |

1. 自动化测试框架需要稳定运行。

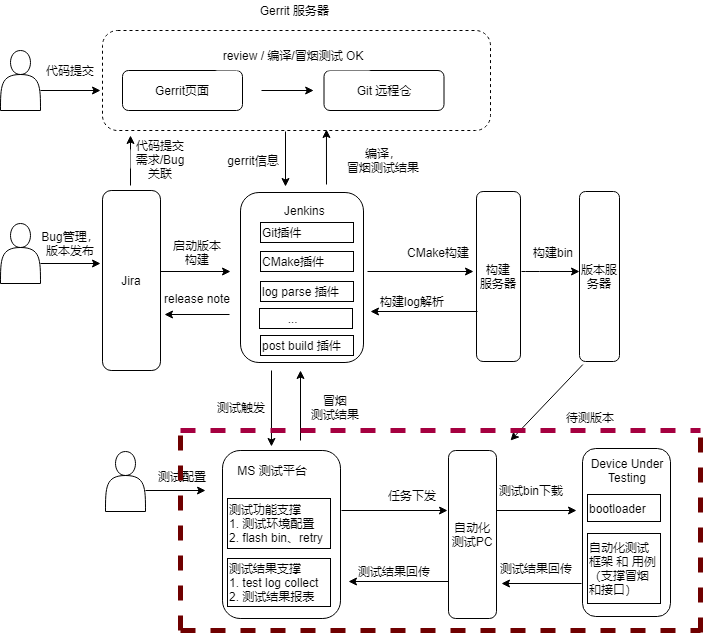
指标（TBD）：在自动化测试的问题点中，自动化框架本身引入的问题不能高于 1%。

1. 自动化测试框架需要高度扩展性，可以支持后续不断增加的单元测试用例，example app，芯片测试。
2. 自动化测试框架需要能够融合到后续要开展的CI流程，作为CI流程的一部分。

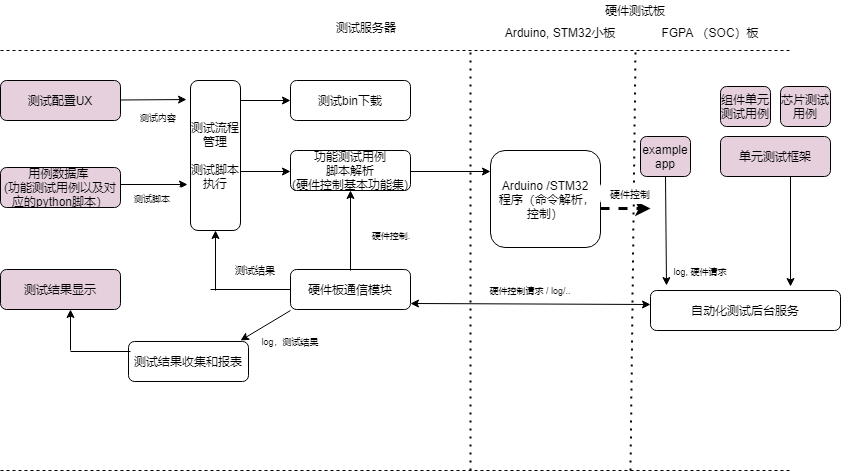
## 2.2架构设计

1) 自动化测试 软件整体架构

在整个CI流程框图中，自动化测试框架需要支撑的是 深红色虚线 部分。



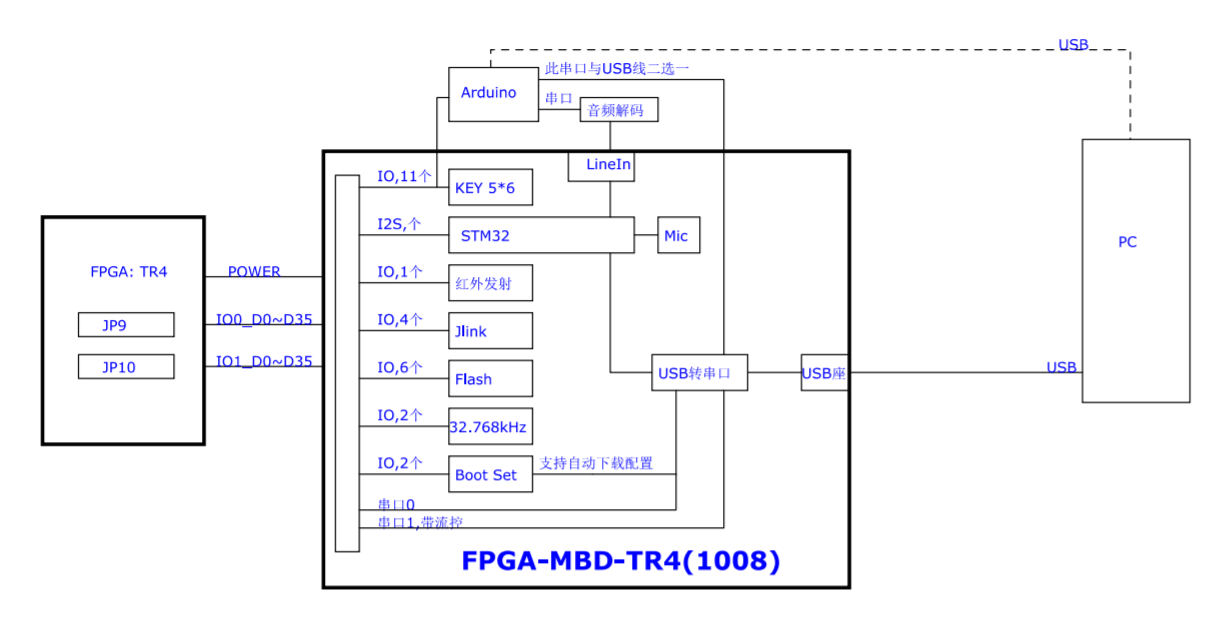
自动化测试框架的软件构架图（红色的模块 不属于自动化测试框架包含的内容）：



自动化测试框架的各模块主要功能描述：

|  |  |  |
| --- | --- | --- |
| 序号 | 模块名 | 主要功能 |
| 1 | 测试流程管理模块 | 1. 根据用户（或者构建服务器）的配置 获取自动化测试项（要测试的组件和用例） 2. 根据测试项，从测试用例数据库获取功能测试用例脚本以及下载正确的测试bin文件 3. 调用功能测试用例脚本进行自动化测试 4. 得到每个测试用例执行完成的通知并开始下一个测试用例。（注意：组件单元测试和芯片测试用例 没有测试脚本，因为测试代码已经包含在测试程序里面。但是单元测试用例的用例 需要在包含在测试用例管理的用例库里面。测试流程管理模块需要知道当前要测试的所有用例集 以便显示当前的测试进度） |
| 2 | 测试bin下载模块 | 自动烧入程序。将来和烧入工具使用一样的底层库来完成。 |
| 3 | 功能测试用例脚本解析模块 (硬件控制基本功能集) | 功能自动化测试用例是用 脚本 (python)写的。每个测试用例对应一个自动化测试脚本。一般地，我们先写好黑盒测试用例的流程（前置条件 / 测试步骤 / 期望结果），再把这个流程用python脚本写出来。功能测试用例脚本解析模块实际上是提供了测试用例自动化脚本里面用到的所有的硬件控制基本功能集合。例如 短按键，长按键，播放声音，停止播放，配置SPI/I2C/UART参数等硬件控制能力。通过命令把这些硬件控制基本功能交给Arduino或者STM32实现（如果是串口的配置则直接在本地PC配置）。这样测试用例着重关注自己的测试逻辑，不需要关注基础控制能力。 |
| 4 | Arduino/STM32程序（命令解析和控制配置等） | 可以在Arduino硬件小板 以及可以在 STM32硬件小板上运行的程序。功能测试用例脚本解析模块 需要发送命令给这两个程序，分别完成对Arduino 和 STM32的硬件输出配置。Arduino主要负责输出GPIO。 STM32的SPI， I2C, I2S和 FPGA对应的模块连接，支持语音的输入以及这些外设的测试。 |
| 5 | 硬件板通信模块 | PC端运行的模块。主要功能如下：   1. 在进行单元测试的时候，根据用户的测试配置，发送测试配置给自动化后台服务程序，告知本次进行的是哪个测试。 2. 收集FPGA板测试程序发来的数据。包括log，单元测试结果，组合test case处理请求以及对 Arduino 和 STM32的硬件输出配置的请求。 在根据请求完成Arduino, STM32的配置后，把配置结果反馈给FPGA板测试程序。 3. 测试开始之后如果长时间没有收到FPGA板测试程序发来的数据，判断为出现异常，则对FPGA板子进行复位，并把本次测试结果判为fail，开始下一个测试case（见异常处理）。 |
| 6 | 测试结果和报表 | 收集测试结果。统计汇总结果，生成当此测试的报表。包括测试对象，测试版本，测试内容，测试结果。 |
| 7 | 自动化测试服务模块 | 工作在板子端的后台任务。   1. 收集测试log发给PC端。 2. 在有需要外设硬件配合做测试的时候，example app或者单元测试程序发送固定格式的请求到该模块。该模块转发请求到PC，并等待请求结果，反馈给对应的测试程序。 3. 组合test case，测试用例需要通过该后台任务发送必要的信息给到PC（信息内容包括需要PC端测试流程管理做什么操作，以及完成该操作之后要执行的下一个用例号） |

2) 自动化测试 硬件环境



说明： Arduino和 FPGA是 GPIO的连接 （即测试FPGA的按键输入，GPIO需要配置Arduino）

STM32和FPGA是 I2S, I2C,SPI模块的连接（即测试FPGA的这些模块需要配置STM32）

PC和FPGA是串口的连接（即测试FPGA的这些模块需要配置PC的串口）

# 3运行设计

典型流程的运行过程：

## 3.1 测试启动流程

开始执行脚本测试

App bin下载

App测试FPGAPGA

测试流程管理模块

测试配置模块

通过硬件通信模块，通知后台服务程序，开始按照按case测试

单元测试bin下载

单元测试FPGAPGA

用例库

测试用例管理模块

注：当前因为测试服务器（涉及配置模块，测试用例库，测试用例管理模块）还没开发，测试流程管理模块作为自动化程序的发起端发起测试。

对example app测试，直接在本地存放测试脚本的目录，顺序读取测试脚本进行测试。

对组件和芯片的单元测试，由硬件板通信模块通知后台测试服务程序，开始每个单元测试用例的测试。

## 3.2测试过程中硬件板配置流程

Arduino/STM32程序

PC端硬件通信模块

测试后台模块

测试case

注：假设某个测试用例，在做SPI master测试中，需要将Arduino SPI设置为slave mode 并进行数据传输。该测试用例需要调用 测试后台模块提供的公共接口，把该请求通过测试后台模块🡪PC端硬件响应请求模块🡪Arduino设置模块 发给Arduino程序，并把设置的结果最终返回到该测试用例。

支撑本流程的硬件示意图：

FPGA 板

Arduino板

PC

GPIO /SPI/I2C..

UART

请求，反馈，log

# 4.接口设计

## 4.1用户接口

## 4.2外部接口

# 5. 模块设计

## 测试流程管理模块

### 5.1.1模块逻辑框图

读取测试配置

YFPGAPGA

最后测试bin?

NFPGAPGA

下载bin，获取测试用例

测试结束

YFPGAPGA

用例测试完?

NFPGAPGA

NFPGAPGA

单元测试?

YFPGAPGA

执行测试用例脚本

通知测试后台执行

获取测试结果

获取测试结果

### 5.1.2实现逻辑说明

1. 当前组件单元测试，芯片单元测试，以及每个example app都是编译成独立的可执行bin。在测试过程中可能需要下载不同的测试bin。（比如我们可能需要下载 单元测试bin做单元测试，下载 蓝牙遥控器app bin做蓝牙遥控器相关的测试）

2. 对于单元测试bin，测试用例本身已经包含在 test bin，测试流程管理模块需要根据配置，通过 自动化测试服务模块 告诉单元测试bin当前需要测试的组件和测试用例。

3. 当前只有蓝牙遥控器example app 及其对应的测试用例，并且没有测试服务器做测试用例管理和配置管理。测试流程管理模块可以先设计为从某个固定目录获取蓝牙遥控器的测试用例 并全部执行。

4. 在没有测试服务器支撑的情况下，测试流程管理模块作为自动化测试的发起者，在测试过程中需要展现当前测试的总体进度条和正在测试的内容。

## 测试bin下载模块 （略）

等后续下载工具一起开发。先手工下载测试bin，运行，再从PC上执行测试管理模块。

## 功能测试用例脚本解析(python脚本)

### 5.3.1模块逻辑框图

Test Case N…. …

测试脚本 2: 蓝牙遥控器语音

Test start

do\_key\_long\_press (up)

do\_voice\_send ()

….

Do\_voice\_play

测试脚本1: 蓝牙遥控器按键

Test start

do\_key\_short\_press (up)

do\_key\_short\_press (down)

repeat

……

Arduino / STM32 命令解析，执行

功能测试用例脚本基本操作实现，将基本操作翻译成 Arduino / STM32 可以识别的命令

Def: do\_key\_short\_press

Def: do\_key\_long\_press

Def： do\_volume\_up

Def： do\_voice\_send

Arduino / STM32 串口驱动

Python 系统函数(串口收发数据，timer等)

如上图，每个功能测试用例对应一个python脚本，这些脚本调用到的硬件基本操作依赖 功能测试用例脚本解析模块 提供。

本模块提供的硬件基本操作 需要涵盖所有 Arduino，STM32能对 FPGA板子控制的能力：

--GPIO (按键控制)

--I2S(语音输入)

--外设设置(uart, spi, i2c)

这些基本操作通过PC端从串口发给 Arduino，STM32的命令来实现。

## Arduino / STM32程序（命令和配置控制模块）

### 5.4.1模块逻辑框图 (Arduino为例）

开始

串口数据监听

Command接收

Command处理

反馈(optional)

### 5.4.2通信命令定义(Arduino为例）

|  |  |  |
| --- | --- | --- |
| Command | 用途 | 参数 |
| AA 80 | 短按键 | 键值 |
| AA 81 | 长按键 | 键值 |
| .. |  |  |

## 硬件板通信模块

开始

等待数据处理

FPGA数据包FPGAPGA

测试管理模块 单元测试

硬件配置命令PGAPGA

Arduino/STM32数据包

数据包类型

发送反馈到FPGA

发送测试用例号FPGA

超时

发送fail结果给测试管理模块

发送log和结果到测试结果收集模块以及 测试管理模块

发送命令到Arduino/STM32

## 测试结果和报表模块 (略)

先简单汇总测试结果，展现两个内容：

1. 单元测试bin 文件的每个单元测试用例执行的结果
2. 蓝牙example app的每个测试用例的执行结果

后续实现测试服务器的时候，一起做详细设计和实现：

1. 测试结果和测试bin文件，git版本等相关联
2. 问题点趋势分析
3. 测试问题点解决情况

## 自动化测试后台服务模块

### 5.7.1模块逻辑框图

Task Start

信号量触发

Get response from PC

Get log/request from test app

Send response to test app

Send log/request to PC

注：这里PC是PC端的硬件板通信模块

### 5.7.2 后台服务和PC端通信定义

通信以不加密的数据包进行传输，报文格式：

Log： 数据包开始标志-数据长度-数据包类型标志-数据报文

其他（请求等）：数据包开始标志-数据长度-数据包类型标志-数据报文-checksum

|  |  |  |
| --- | --- | --- |
| 标识符类型 | 标识符子类 | 含义 |
| 0x7E7E | - | 一个完整数据包的开始 |
| 0x7F | 0x5A | 该数据包是板子发给PC端的log |
| 0x7F | 0x5E | 该数据包是板子发给PC端的 硬件配置请求，配置Arduino/GPIO |
| 0x7F | 0x7A | 该数据包是板子发给PC端的 硬件配置请求，配置STM32/SPI |
| 0x7F | 0x7E | 该数据包是板子发给PC端的 硬件配置请求，配置STM32/I2C |
| 0x7F | 0x9A | 该数据包是板子发给PC端的 组合测试请求，报文包含请求类型和下一个测试用例号。 |
| 0xF7 | 0x5A | 该数据包是PC端发给板子的response |
| … |  |  |

# 6.数据结构设计

## 6.1数据结构一（从模块之间的接口中抽象而来）

### 数据流一(*数据流是根据系统的运行控制中抽象而来的*)

（*描述如下内容：*

*1、数据处理：说明关键数据在传送时引起状态变化的数据处理情况，说明对数据进行处理的模块；*

*2、数据状态：说明对每种数据状态的数据存储形式、表现方式*）



### 逻辑结构设计

（*说明系统全局、模块接口所采用的数据结构*）

（*说明对系统数据物理存储的模式和方法*）

# 7. 异常处理设计

|  |  |
| --- | --- |
| 异常类型 | 处理方案 |
| 测试用例执行过程中超时。单元测试用例没有返回测试结果，example app没有log打印。 | 由硬件板通信模块 做超时判断，超时之后判断为程序跑飞，hang。   1. 进行FPGA板子复位。 2. 通知 测试结果收集模块 该测试用例fail 3. 通知 测试流程管理模块 进行下一个测试用例。 |
| 测试用例执行过程中发生exception。 | 由硬件板通信模块 做log关键字判断。  处理方案同上。 |
| 测试用例执行过程中发生重启。 | 由硬件板通信模块 做log关键字判断。  处理方案同上。 |
| 组合test case | 组合test case作为异常处理的一种特例。部分测试需要做一些特殊的操作，比如系统复位，深睡眠后唤醒等，并且在这些特殊操作之后可能再执行另外其他的测试用例（比如深睡眠之后，需要按键唤醒，唤醒之后再检查系统是否正常）。这种情况下两个test case组合成一个完整的测试（组合test case）。前一个test case进入深睡眠之前，通知 测试流程管理模块后续需要做唤醒处理以及唤醒之后首先要执行的另外一个test case （深睡眠之后的检查）。 |
| STM32/Arduino硬件请求超时 | 判断为STM32/Arduino程序问题。当前测试用例的结果为 NA （没有测试）。 |

# 8. 系统扩展

1. 支持增加新的测试用例

需要建测试服务器管理 单元测试用例 和 功能测试用例。自动化测试的框架设计需要能够满足不断增长的测试用例的需求。

1. 自动化测试框架作为CI流程的一部分。当前的框架设计 需要和后续的 构建服务器，代码服务器，版本发布服务器联动，做成完整的CI流程。
2. 自动化测试框架需要能够支撑多个板子同时进行测试