POBC自动化测试系统

|  |  |  |  |
| --- | --- | --- | --- |
| **修改记录** | | | |
| **版本** | **修改人** | **描述** | **修改时间** |
| **V1.0** | **旷凯** | **初次修改** | **20190530** |

# 目录

[POBC自动化测试系统 1](#_Toc10029396)

[目录 1](#_Toc10029397)

[需求分析 2](#_Toc10029398)

[整体设计 3](#_Toc10029399)

[硬件连接设计图 3](#_Toc10029400)

[软件设计 3](#_Toc10029401)

[图形化管理 5](#_Toc10029402)

[访问方式 5](#_Toc10029403)

[访问步骤 5](#_Toc10029404)

[信息展示 5](#_Toc10029405)

[功能测试 5](#_Toc10029406)

[通知功能 6](#_Toc10029407)

[记录查询 6](#_Toc10029408)

[命令行管理 7](#_Toc10029409)

[一键部署 7](#_Toc10029410)

[功能展示 7](#_Toc10029411)

[编程接口 8](#_Toc10029412)

[文档和代码管理 8](#_Toc10029413)

[远程调试 8](#_Toc10029414)

# 需求分析

POBC板组件和接口较多，测试不方便，需要一个功能完善，方便易用的自动化测试方案（以下简称方案）。

根据POBC测试的实际需求，自动化测试方案主要分为硬件对接和软件设计两部分。

1. 硬件对接：由于树莓派以Python为基础编程语言，接口操作方便，树莓派最新版本3B+提供了大部分的测试所需接口，对于POBC提供的两组接口，可以通过多路2选1的选择器进行切换测试。
2. 软件设计：为简化测试流程，降低测试复杂度，在硬件对接完成的基础上，还需要提供一个自动化的测试程序，该程序使用python开发，运行在树莓派官方系统上，提供命令行测试管理工具和图形化测试管理平台，此外，为提供后续的功能开发和修改，提供一套测试程序的编程接口。

# 整体设计

## 硬件连接设计图



## 软件设计

根据硬件设计的要求，控制端需要能通过WIFI连接进行测试。远程控制可以通过SSH协议或者HTTP协议实现，在树莓派上启动一个web服务来实现从浏览器访问进行测试和管理，同时，将功能实现转化为Python可执行脚本，提供命令行测试支持，方便在无法使用图形界面时进行测试。

由于树莓派可用系统资源有限，所有方案设计尽量以简洁和复用功能为主，测试框架分为三个部分，可以拆分独立使用，也可合并工作：

* API（树莓派接口控制部分）
* CMD（命令行测试部分）
* GUI（Graphic User Interface图形化测试部分）

CMD在API的基础上提供功能封装，GUI在CMD的基础上实现图形化控制和结果返回。

图片包含 屏幕截图

描述已自动生成

# 图形化管理

方案提供一个基于Django开发的图形化管理界面，Django是目前功能最强大的前后端统一的Python开发框架可以完全满足自动化测试的需求。本项目基于树莓派开发，可用资源有限，适合使用集成度较高的工具，以减少依赖工具的安装和系统资源的占用。

图形化管理界面主要包含：

* 信息展示
* 功能测试
* 参数配置
* 记录查询

## 访问方式

图形化管理界面采用B/S架构，通过浏览器即可登录管理界面完成相应的测试管理工作。方案将在管理界面设计中使用HTML的新特性，要求浏览器支持HTML5，并打开cookie。我们推荐使用最新版本的Chrome或者FirFox浏览器。

图形化管理界面基于用户密码认证，设置唯一的管理员，默认用户test，默认初始密码为：test123

## 访问步骤

1. 打开浏览器（Chrome或FirFox）
2. 在浏览器URL栏输入GUI访问点，如下：

<http://124.16.113.55:22280/>

其中124.16.113.55为树莓派的IP，默认的访问端口为22280

1. 按照界面提示，输入管理员密码，即可登录管理界面。
2. 点击管理界面右上角“退出”或者关闭浏览器“POBC自动化测试系统”标签页均可退出。

## 信息展示

主要展示POBC的各组件状态信息和详细参数，测试进程，通知信息等。

## 功能测试

前段功能测试主要调用后端模块完成指定的测试，包含组件测试和接口测试，可以单独测试，也可以按根据流程一步步测试。

组件测试包括：

* 电源部分：测试板上各路供电是否正常；
* 时钟：测试板上晶振是否正常工作；
* 程序下载及启动：测试Z7020 JTAG接口下载程序以及SPI Flash自启动是否正常；
* DDR部分：测试DDR读写是否正常；
* EMMC部分：测试EMMC读写是否正常；
* 功耗测试

接口测试包括：

* 两路ETH；
* 2路RS422 UART；
* 2路RS422 SPI；
* 2路CAN；
* 2路 Camera link；
* 6路LVDS或12路GPIO；
* 1路调试串口；（USB）
* 其他GPIO；

## 通知功能

方案提供一个可配置的通知功能，支持邮件，短信，微信通知，方便用户及时了解测试进程。

* 邮件发送使用常用的smtp服务器，指定一个发送邮箱即可。
* 短信发送需要一个短信网关来支持短信发送，可以购买相关服务，约0.03-0.05元/条
* 微信通知需要公众号支持，开通一个企业公众号，后台通过Http请求访问微信后台即可发送通知信息。

## 记录查询

测试历史记录留存和查询，本地使用新建一个MySQL或者简易的Sqlite3数据库保存测试记录，需要参考实际需求和资源占用决定选用哪种数据库。

表结构：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Item | 序号 | 开始时间 | 结束时间 | 负责人 | 测试项目 | 测试结果 |
| Value | 1 | 2019/5/29 | 2019/5/30 | 张三 | POBC接口 | Result-20190530.tgz |

# 命令行管理

命令行管理依托于树莓派接口操作模块，封装具体测试项目，主要包含：

* 接口测试
* 组件测试
* 其他测试项目

其他测试功能都可以在树莓派接口操作模块基础上，添加测试流程，设置测试参数来完成。

## 一键部署

根据网络连接情况分为离线部署和在线部署，以在线部署为例：

1. 获取安装包：

wget https://github.com/kk47/POBC\_automated\_test/archive/POBC-release-v0.1.bin

1. 自解压安装

./POBC-release-v0.1.bin

1. 浏览器访问

http://172.1.1.200:8080

## 功能展示

keven-MBP-3:POBC\_automated\_test dayu$ ./pobc\_test.py -h

Usage:

Name:

pobc\_test.py - POBC automate test tool

Synopsis:

pobc\_test.py [-h] [-c cmd] [-p params]

Description:

Arguments are as following:

-h Print the help message

-c Command mode, ex:

TestAll test all case

InterfaceTest interface test

ComponentTest component test

ApiTest Raspberrypi interface test

keven-MBP-3:POBC\_automated\_test dayu$ ./pobc\_test.py -c InterfaceTest

Interface test

# 编程接口

方案提供API编程接口，用来简化树莓派的接口控制，方便功能的迭代和新功能的开发。编程接口主要使用python模块控制各个接口

* GPIO接口，主要使用python GPIO模块实现接口控制
* SPI接口，主要通过python SPI模块实现控制
* ETH接口，网口使用python ping模块实现控制
* CAN接口，SPI转CAN，封装SPI接口实现。
* UART接口，SPI转UART，封装SPI接口实现。
* LVDS接口，通过GPIO连接，封装GPIO模块实现。

# 文档和代码管理

目前主流的版本控制软件主要有GitHub和svn，GitHub较为建议，而且无需搭建服务器，所以我们选用GitHub来托管代码和文档。目前已经建好GitHub项目，搭建好开发框架，后续使用GitHub进行文档和代码的管理。

GitHub站点：

git clone https://github.com/kk47/POBC\_automated\_test.git

# 远程调试

由于项目属于远程协作开发模式，需要提供一个远程调试方式来进行测试，对于硬件的改动也可以通过远程调试的方式进行代码测试。

目前已经搭建一个临时的远程调试的环境：

ssh <124.16.113.55> -p 22222 -l root