嵌入式软件自动化

嵌入式软件自动化测试即利用脚本来自动化驱动嵌入式软件运行，并自动收集相关数据进行分析，最终生成相应的最终生成相应的测试报告。虽然，嵌入式软件的自动化测试流程与一般 PC 机应用软件的自动化测试流程相同。但是，由于嵌入式软件软件对嵌入式设备的高度依赖性，以及嵌入式设备受周围环境影响较重，从而导致嵌入式软件的自动化测试平台存在诸多问题。

Sanity Check

为了简化、加速测试过程，Zephyr 创建了 Sanity Check,以实现独立于嵌入式设备的自动化测试。Sanity Check 可自动完成测试样例选择、配置、归档，测试执行，测试结果收集，并生成测试报告。

Sanity Check 可在主机和嵌入式设备之间进行全双工串口连接，并在目标设备上执行测试，过程由 Sanity Check 提供的 python 脚本进行监管，不与任何用户进行交互。脚本生成一系列输出内容和测试报告，测试人员只需查看测试报告即可。

Sanity Check 框架

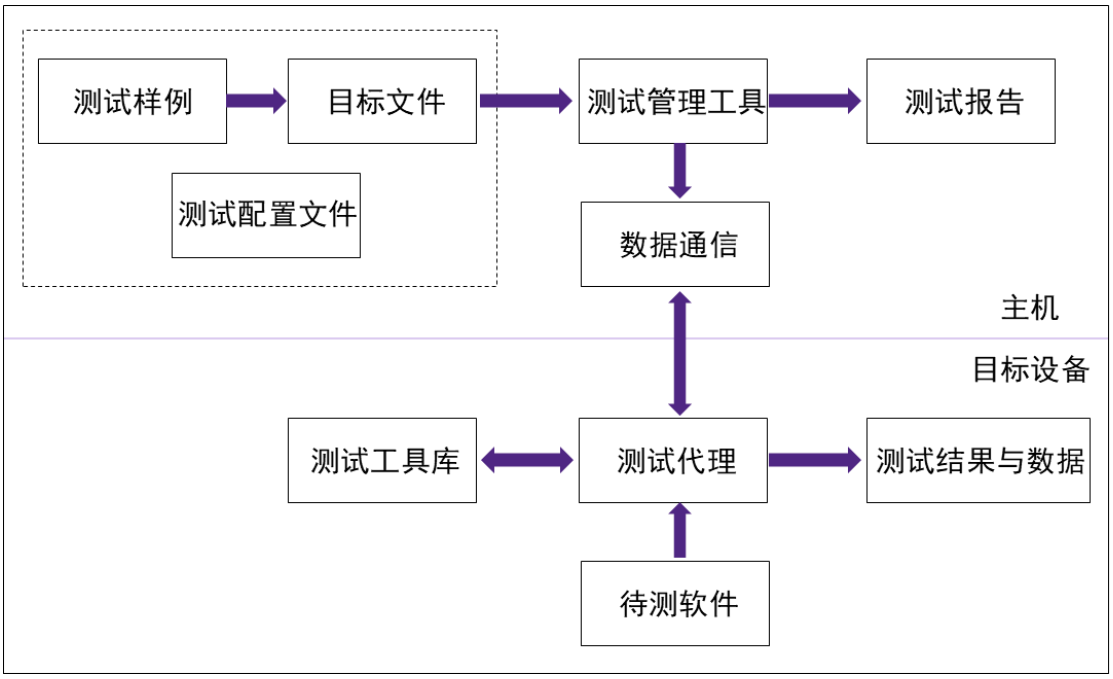
测试配置文件。每个测试样例对应一个配置文件“testcase.yml”，Sanity Check 通过该文件判断是否存在测试样例。在“testcase.yaml”中测试元数据以 key/value 的方式定义。

测试管理工具。为了管理、控制测试流程，同时减少嵌入式设备端资源占用，Sanity Check 实现了多个功能模块。

（1）日志模块。日志模块用于跟踪测试状态，记录测试信息，分析测试结果并生成测试报告等。
（2）测试样例管理模块。根据输入参数和测试配置文件，查找、选择并归档需要一起执行的测试样例。
（3）测试执行模块。解析生成 CMake 脚本，根据目标设备选择执行器，并采用异步方式编译、执行测试样例。

测试代理。测试代理基于 openocd,pyocd 等调试工具实现，运行在目标设备上，通过远程串行协议与主机通信。

测试工具库。测试代理通过调用测试工具库中的通用函数完成待测软件的测试。



Sanity Check 采用嵌入式软件通用的 Host/Target 测试策略。

当建立了管理工具和测试代理之间的通信之后，向测试代理发送测试请求，并将目标文件传送到测试代理上。测试代理将目标文件存放到特定的区域执行，并且将测试结果返回给主机端。当主机端接收到测试代理所发送执行完毕信号后，发送新的目标文件给测试代理，直到所有的目标文件测试完成之后，对测试代理所返回的测试结果进行分析，得出最终的测试报告。

Sanity Check 主要特点

- 易用性

Sanity Check 能够自动识别测试样例，并根据用户输入的参数进行过滤，生成测试套件，使用者无需关注实现细节，就能完成相关测试。

- **健壮性**

Sanity Check 在执行自动化测试时，存在一定的概率会失败。通过设置 timeout 和出错重试逻辑，提高测试稳定性。

- **扩展性**

Zephyr 在其存储库中提供了多项 Benchmark，以满足性能测试的基本需求。如果用户想要对其进行扩展，仅需根据 Ztest 编写测试样例，添加配置文件`testcase.yaml`。 Sanity Check 具备自动检测变更并加载配置的能力，大大降低了维护成本。

- **支持嵌入式设备测试**

Sanity Check 能够直接使用交叉编译生成的目标机代码，在模拟器或目标机上进行测试。既能实现程序逻辑的单元验证，又能够对嵌入式设备组装为产品后可能发生的问题等进行具有高信赖度的白盒测试。