Git-Openh264-Conformance Test

Contents

huashi@cisco.com 2014.1.6

1. 总概

一致性测试

一致性测试就是比较两个版本的 encoder 在相同的 case 下输出码流是否一致。在脚本中的一致性比较，是用修改更新的 codec(下面用 **codec_target** 表示)，同之前经过测试通过的 codec(下面用 **codec_benchmark** 表示)，比较两者的码流在配置文件中的所有 case 下的码流是否一致。

功能说明

1. 过配置文件，来修改测试所使用的 case
2. 过配置文件，来修改测试所使用的测试序列
3. 过配置文件，修改用于比较的 codec 的 github 地址，或者 svn 地址以及版本（该脚本采用 git 管理，可以切换不同的 branch 来选用 git 或者 svn）

4.自动下载指定的 codec,并自动 build

5. 自动生成配置文件对应的 case 文件（如 XX._case.csv;

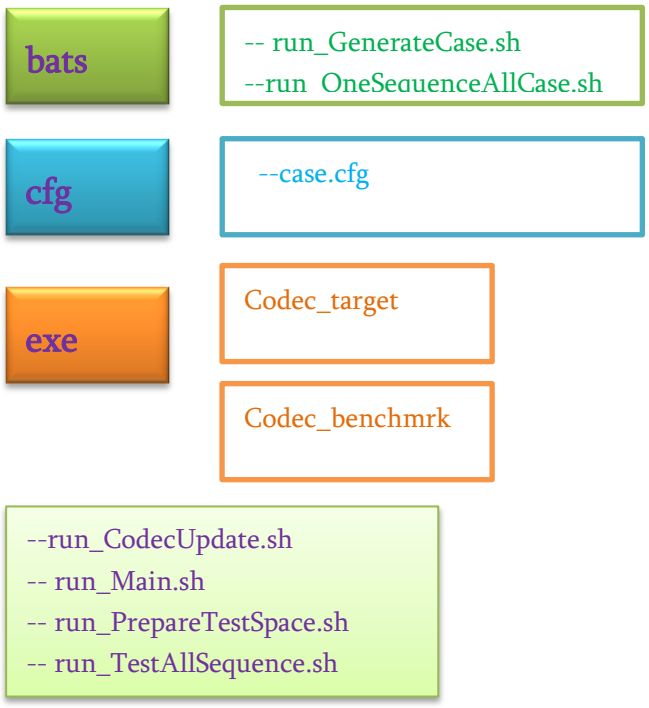
6.自动测试所有序列的所有 case,并自动生成测试报告，

7. 可结合 CI 系统，集成为自动化的测试环境

2. 测试目录说明

测试脚本目录结构

--TestScript

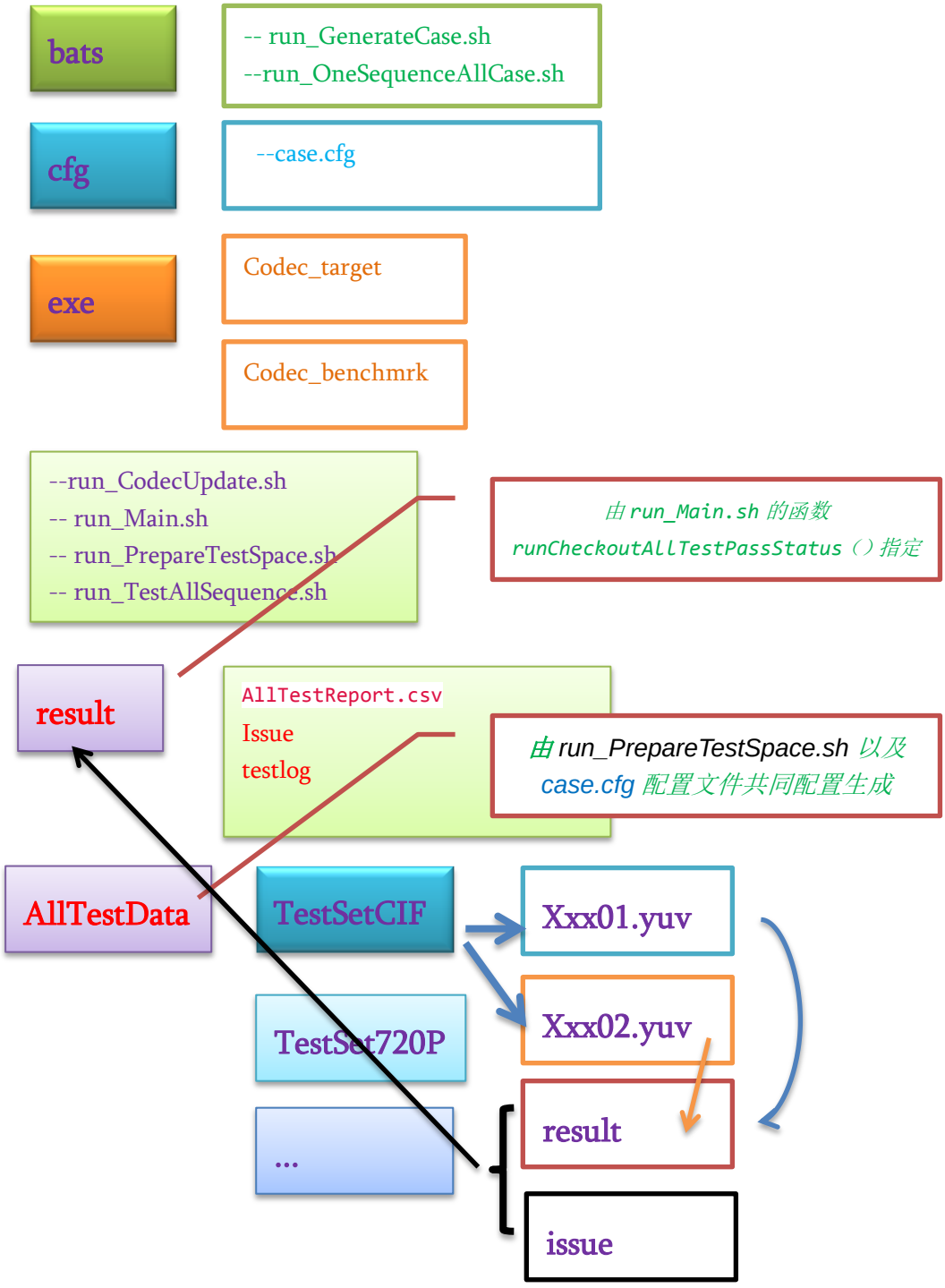


脚本+测试数据目录结构

-- Source-openh264-git

run_CodecUpdate.sh 语句 SourDir="../Source-openh264-git"指定，可修，为 codec_target 的源文件）

--TestScript



3. 测试脚本文件使用说明

Codec 的自动下载和 build

执行 ./run_CodecUpdate.sh $GitAddr

如 ./run_CodecUpdate.sh https://github.com/cisco/openh264

脚本文件工作流程：

1） 从$GitAddr 下载 codec 代码，自动 build，

2） 删除./exe/codec_target 所有文件

3） 将 build 成功的 .exe .a 等文件拷贝到 ./exe/codec_target 目录下

4） 将 codec 中的配置文件 ../testbin/*.cfg 拷贝到./exe/codec_target 目录下

5） 为统一两个 codec 的配置文件，用./exe/codec_target 下最新的*.cfg 文件覆盖./exe/codec_benchmark 下的配置文件

测试数据空间的创建

执行 ./ **run_PrepareTestSpace.sh** $ConfigureFile

如 . /**run_PrepareTestSpace.sh** ./cfg/case.cfg

创建空间即为第二部分提及的文件目录 AllTestData

脚本文件工作流程：

1） 根据配置文件，解析测试序列集包含的测试序列信息

2） 为每一个序列集(TestSet)创建一个目录

3） 对每一个 YUV 序列，在其对应的序列集目录下建一个 YUV 目录

4） 将./exe ./cfg 以及./bats 目录下的每一个文件拷到每一个 YUV 目录下

5） 在每一个 YUV 目录，调用 run_GenerateCase.sh，为其生成对应的 case 文件

Case 文件的生成

执行 ./run_GenerateCase.sh $ConfigureFile $OutputFile $TestSetName

如./run_GenerateCase.sh ./cfg/case.cfg \
TestSet_CIF_Cisco_392X192.yuv_AllCase.csv \
CIF

将配置文件的配置信息转化为一个参数排列组合而成的二维 case 矩阵，以.csv 数据文件作为输出。

脚本文件工作流程：

1） 根据 YUV 所在目录，解析当前 YUVName，TestSetName 等信息
如 ./AllTestData/TestSetCIF/Cisco_392X192.yuv 目录信息，可解析出上述信息

2） 根据配置文件，解析编码参数的对应参数集

3） 排列组合，生成输出文件

- 单个序列所有 case 的测试

执行 ./ run_OneSequenceAllCase.sh （无参数）
对当前 YUV 目录下的 case 文件调用，对每一个 case 进行一次编码比较。
caseFile=${TestSetIndex}_${TestSequenceName}.csv
脚本文件工作流程：

1） 根据 YUV 所在目录，解析当前 YUVName，TestSetName，PicW,PicH 等信息,同时对 YUV 的文件位置进行确认，是否存在测试序列
2） 解析每一行 case,将其与编码命令行参数意义匹配；
3） 对 codec_target 和 codec_benchmak 分别编码比较同一 case 下的输出码流是否一致
4） 对所有 case 的比较结果进行统计记录并将结果放到../result 目录下

- ./run_Main.sh

执行 ./run_Main.sh $ConfigureFile
如 ./run_Main.sh ./case.sh
执行出 run_CodecUpdate.sh 外的所有功能

1） 创建 AllTestData 测试目录；
2） 对所有的 YUV 进行所有 case 的测试
3） 对所有的测试结果进行总结，并将结果放到./result 目录下，生成 AllTestReport.csv 文件

4. 脚本语法

1） If [[ ]] 正则表达式

2） Echo $string | awk ' ' 文本信息提取

3） 一维数组及伪二维数组使用技巧

4） 变量的变量

5. 其他