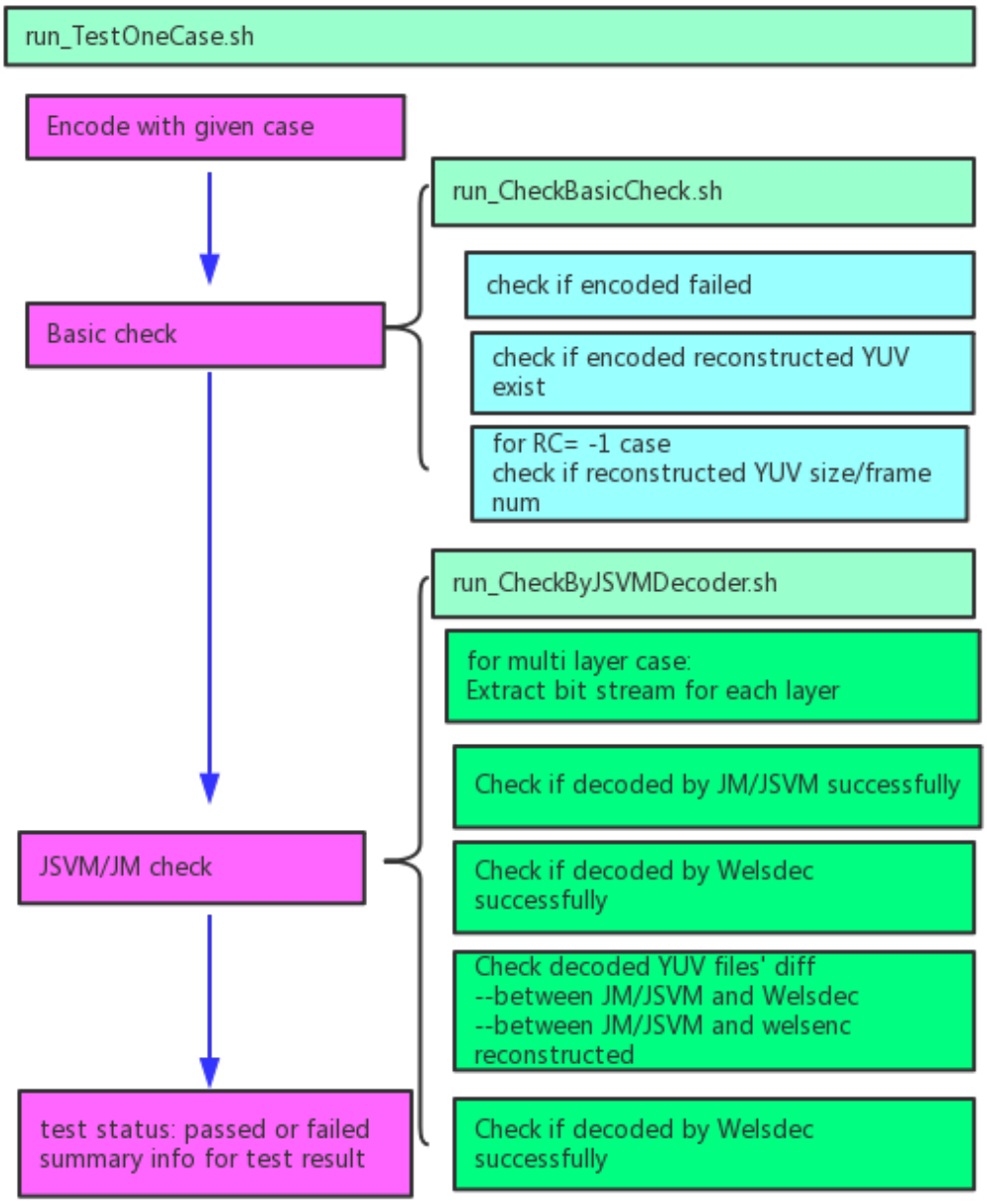
OpenH264 Encoder Conformance Test

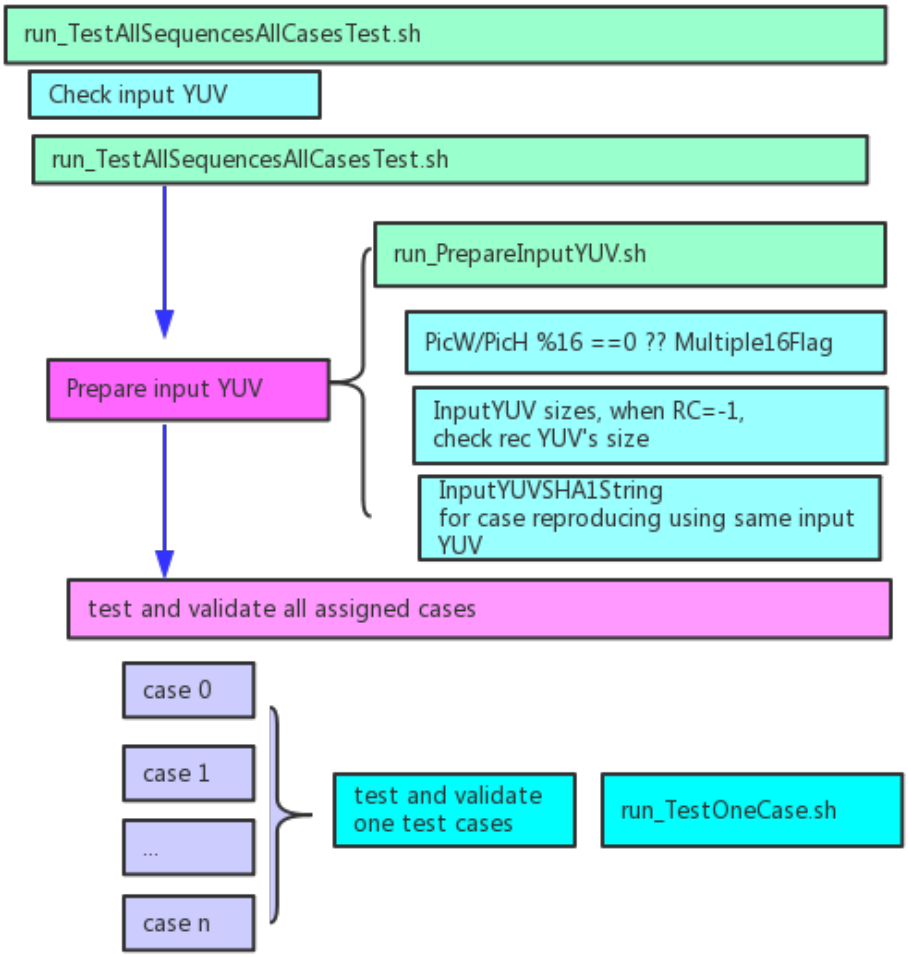
Table of Contents

1. Conformance Test brief introduction

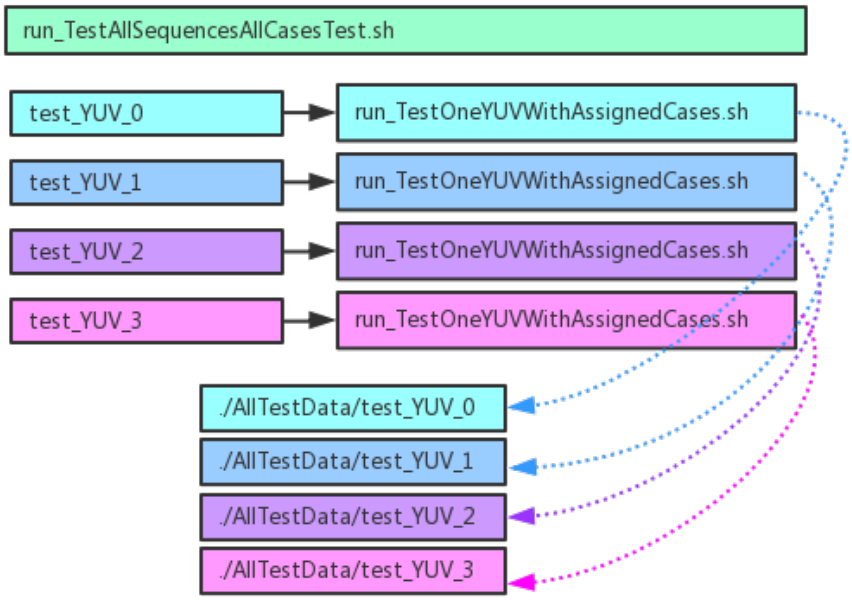
1.1 Conformance test for one case



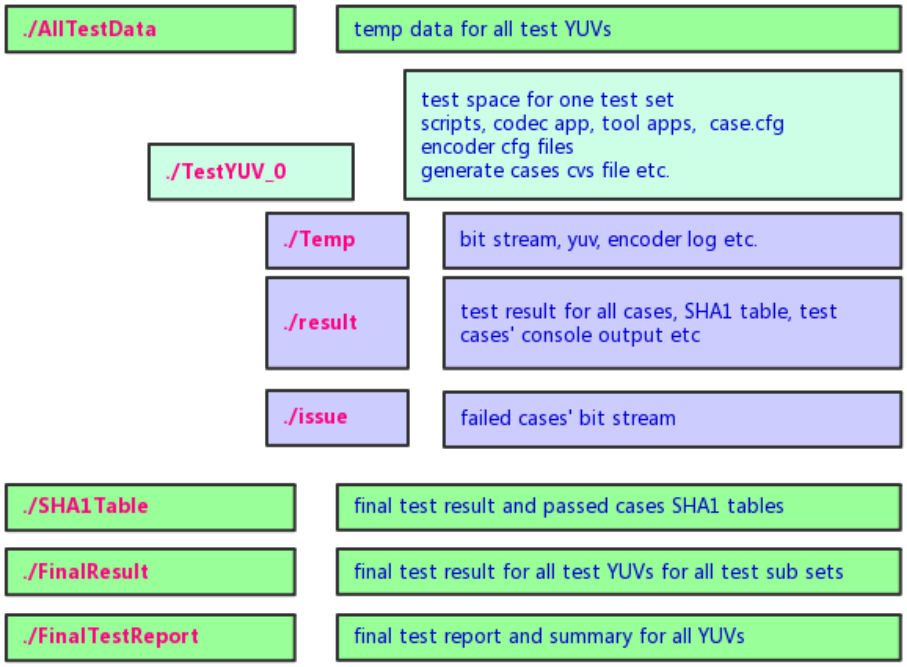
1.2 Conformance for one YUV



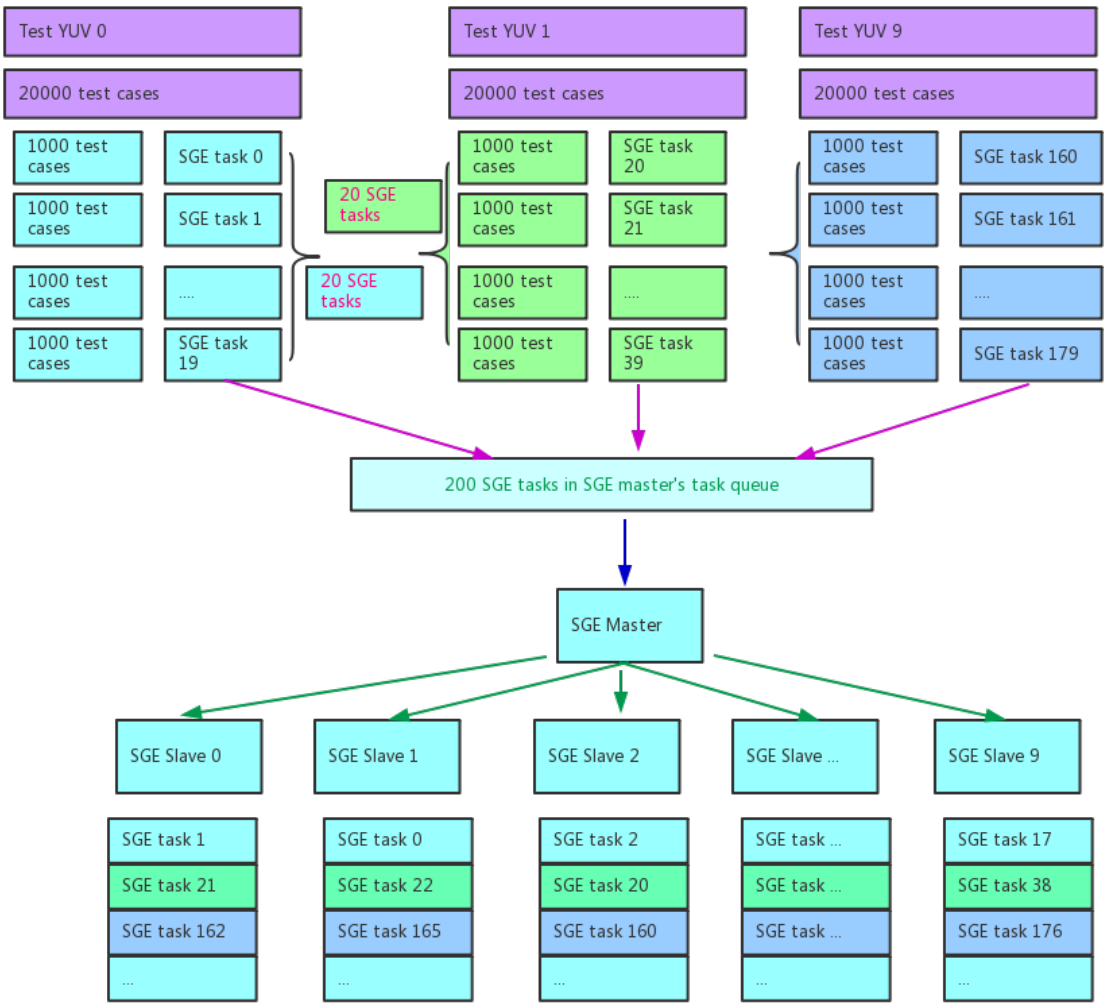
1.3 Conformance test for all YUVs



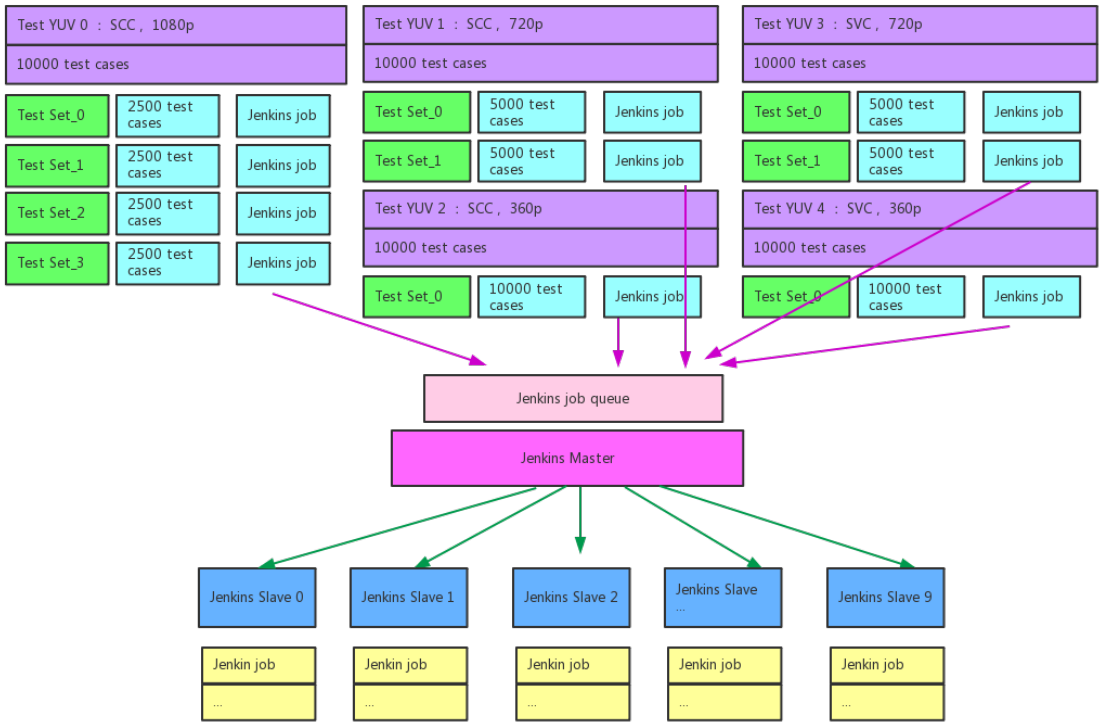
1.4 Test data structure



1.5 SGE test


Test YUV 0
20000 test cases
1000 test cases
1000 test cases
1000 test cases
1000 test cases
SGE task 0
SGE task 1
....
SGE task 19
20 SGE tasks
20 SGE tasks
Test YUV 1
20000 test cases
1000 test cases
1000 test cases
1000 test cases
1000 test cases
SGE task 20
SGE task 21
....
SGE task 39
Test YUV 9
20000 test cases
1000 test cases
1000 test cases
1000 test cases
1000 test cases
SGE task 160
SGE task 161
....
SGE task 179
200 SGE tasks in SGE master's task queue
SGE Master
SGE Slave 0
SGE Slave 1
SGE Slave 2
SGE Slave ...
SGE Slave 9
SGE task 1
SGE task 21
SGE task 162
...
SGE task 0
SGE task 22
SGE task 165
...
SGE task 2
SGE task 20
SGE task 160
...
SGE task ...
SGE task ...
SGE task ...
...
SGE task 17
SGE task 38
SGE task 176
...

1.6 Jenkins based test set test mode



2. Conformance Test script basic info

2.1 Brief introduction

Repos info

- GitHub URL： https://github.com/shihuade/Conformance-Test-Openh264.git
- Branch: master
- Basic introduction

Support test mode

- Mac/Linux local test mode for both SCC and SVC
- Linux Jenkins based test set mode for both SCC and SVC
- Linux SGE test mode for both SVC and SCC
- Mac/Linux SHA1 tables generation test mode for both SCC and SVC

Support platform

- Linux:
- Mac OS/Unix

For more detail, please refer related chapters in this document

2.2 How to use

2.2.1 Linux single machine local test

- Step 1:
  Change **Testplatform** to **Linux**,in one cfg file under CaseConfigure/*.cfg which you want to test

```
#============================== Test platform ======================
TestPlatform:  Linux            #test platform, Mac or Linux
#============================== Test platform ======================
```

- Step 2:
  For SVC, run below command like
  `./run_Main.sh LocalTest ./CaseConfigure/case_SVC.cfg`

  For SCC, run below command like
  `./run_Main.sh LocalTest ./CaseConfigure/case_SCC.cfg`

- Step 3:
  Wait final test result and test result will be under
  SHA1Table/*
  FinalResult/*
  FinalTestReport/*

2.2.2 Linux Jenkins based test

Jenkins based test, will assign test cases to Jenkins slaves.
Jenkins job, please refer to:

http://10.140.198.27:8080/view/ConformanceTest/job/OpenH264-Encoder-Conformance-Test/

For example,

in CaseConfigure/, there are 20 cfg files for SVC and SCC test case

```
case_Jenkins_SCC_TestSet0.cfg
case_Jenkins_SCC_TestSet1.cfg
case_Jenkins_SCC_TestSet2.cfg
case_Jenkins_SCC_TestSet3.cfg
case_Jenkins_SCC_TestSet4.cfg
case_Jenkins_SCC_TestSet5.cfg
case_Jenkins_SCC_TestSet6.cfg
case_Jenkins_SCC_TestSet7.cfg
case_Jenkins_SCC_TestSet8.cfg
case_Jenkins_SCC_TestSet9.cfg

case_Jenkins_SVC_TestSet0.cfg
case_Jenkins_SVC_TestSet1.cfg
case_Jenkins_SVC_TestSet2.cfg
case_Jenkins_SVC_TestSet3.cfg
case_Jenkins_SVC_TestSet4.cfg
case_Jenkins_SVC_TestSet5.cfg
case_Jenkins_SVC_TestSet6.cfg
case_Jenkins_SVC_TestSet7.cfg
case_Jenkins_SVC_TestSet8.cfg
case_Jenkins_SVC_TestSet9.cfg
```

For more detail about test set, please refer to:

AboutJenkinsTestSet.txt

For each Jenkins slaves, run below command, take SCC test set 0 for example:

```
./run_Main.sh   LocalTest  case_Jenkins_SCC_TestSet0.cfg
```

note: please make sure that there in test YUV desktop_dialog_1920x1080_i420.yuv under /home/Video/YUV on test slave or you can change the YUV location

For Jenkins job run for one test set, take SCC test set 0 for example,
you can refer to below Jenkins job URL for more detail.

http://10.140.198.27:8080/job/OpenH264_ConformanceTest_SCC_TestSet0/configure

To Optimize overall test runtime for all test cases,

- Test set cfg file will need to optimize based on actual historical run time in future.
- And the script level optimization is also need to decrease run time for one test cases

2.2.3 Linux SGE based test

SGE test is based on SGE distribution system and assign test tasks to SGE slaves.
For more detail about SGE test, please refer to

- Chapter SGE based conformance test
- Chapter SGE install and configuration

SGE based test will be transform to 2.2.2 test mode,
and will be removed in future

For SGE test on SGE master:

Run below command:

- `SCC: ./run_Main.sh SGETest ./CaseConfigure/case_SCC.cfg`
- `SVC: ./run_Main.sh SGETest ./CaseConfigure/case_SVC.cfg`

For SGE test on SGE master based on Jenkins:

Please refer to Jenkins job(will be removed)

1) Openh264-SGE-Job-Submit

http://10.140.198.27:8080/view/SGE-Test/job/Openh264-SGE-Job-Submit/configure

after running this job, SGE test tasks will be assigned to SGE slaves
job artifacts will give more detail about SGE jobs submit, as show below:

And need to wait for final test result, sometimes may need 6~12 hours to compete all test cases on SGE which run in background
If you want to check the test status and result, please go to 2).

2) penh264-SGE-Job-status-And-TestResult

http://10.140.198.27:8080/view/SGE-Test/job/Openh264-SGE-Job-status-And-TestResult/configure

if there are failed cases, this job status will be marked as failed, and the artifacts files will give more detail about which YUV in which case on which SGE slaves failed

for more detail about test status and result please refer to artifacts as show below:

take SVC_JobReport.txt for example:
below shows that test YUV candyHF2_640x480.yuv with 1000 test cases which running on ZhangFei SGE Slaves

```
report file: /opt/sge62u2_1/SGE_room2/OpenH264ConformanceTest/NewSGE-SVC-Test/FinalResu
[34m ********************************************************************************
[34m  Test report for YUV candyHF2_640x480.yuv                [0m
[34m ********************************************************************************
[35m   test host          is: ZhangFei              [0m
[35m   test type          is: SGETest              [0m
[35m   test directory     is: /home/ZhangFei/SGEJobID_10522      [0m
[35m   AssignedCasesFile  is: /opt/sge62u2_1/SGE_room2/OpenH264ConformanceTest/NewSGE-S
[32m   test YUV full path is: /home/Video/YUV/candyHF2_640x480.yuv      [0m
[34m ********************************************************************************
[36m   Sub-Case Index is: 0             [0m
[36m   Host name      is: ZhangFei               [0m
[36m   SGE job ID     is: 10522                 [0m
[36m   SGE job name   is: ----candyHF2_640x480.yuv_SubCaseIndex_0----                  [
[34m ********************************************************************************
[34m ********************************************************************************
[34m              Test summary for candyHF2_640x480.yuv      [0m
[34m ********************************************************************************
[33m    TestStartTime    is: Wed Apr 19 00:08:30 CST 2017               [0m
[33m    TestEndTime      is: Wed Apr 19 01:49:05 CST 2017                 [0m
[34m ********************************************************************************
[32m    total case  Num     is : 1000        [0m
[32m    EncoderPassedNum    is : 1000    [0m
[31m    EncoderUnPassedNum  is : 0  [0m
[32m    DecoderPassedNum    is : 1000    [0m
[31m    DecoderUpPassedNum  is : 0  [0m
[31m    DecoderUnCheckNum   is : 0   [0m
[34m ********************************************************************************
[32m     --issue bitstream can be found in /home/ZhangFei/SGEJobID_10522/issue
[32m     --detail result can be found in   /home/ZhangFei/SGEJobID_10522/result [0m
[34m ********************************************************************************

[32m Succeed!                          [0m
[32m All Cases passed the test!
```

2.2.4 Mac/Unix single machine local test

note:

```
#*****************************************************************
For downsample and extractor app tools,
 which using for multiple layer cases,
 currently, there are no mac version,
 so, on mac, only support AVC cases.
#*****************************************************************
```

Below tools app only for Linux under

Codec_Linux/Tools32Bits/
Codec_Linux/Tools64Bits/

- **DownConvertStatic**
- **JSVMDecoder**
- **extractor.app**

```
#*****************************************************************
```

multi layer setting should set to 0

```
#============================== Multiple Layer option==================
MultiLayer:     0    # 0 single layer  1 multi layer
                     #  2 mult layer and single layer
Multiple16Flag: 1    # all sub layers' resolution is multiple of 16
                     # 0:disable ;  1:enable
#============================== Multiple Layer option==================
```

```
#*****************************************************************
```

Usage:

- Step 1:
  Change **Testplatform** to **Mac**,in one cfg file under CaseConfigure/*.cfg which you want to test

```
#============================== Test platform ======================
TestPlatform:  Mac            #test platform, Mac or Linux
#============================== Test platform ======================
```

- Step 2:
  For SVC(AVC only), run below command like
  ./run_Main.sh LocalTest ./CaseConfigure/case_SVC.cfg

  For SCC, run below command like
  ./run_Main.sh LocalTest ./CaseConfigure/case_SCC.cfg

- Step 3:
  Wait final test result and test result will be under
  SHA1Table/*
  FinalResult/*
  FinalTestReport/*

2.2.5 SHA1 table generation

SHA1 tables are used for encoder output bit stream validation.
For more detail, please refer to chapter:

- SHA1 tables generation

1) Generate SHA1 tables on Mac/Linux local machine:
   - SCC
     ```
     ./run_Main.sh LocalTest case_SHA1Table_SCC.cfg
     ```
   - SVC:
     ```
     ./run_Main.sh LocalTest case_SHA1Table_SVC.cfg
     ```

   Final SHA1 tables for test YUV can be found under folder:
   - ./SHA1Table/

2) Jenkins Jobs:
   http://10.140.198.27:8080/job/Openh264_Travis_SHA1Table_Generation_weekly/configure

   Final SHA1 tables for 3 YUVs, you can download from artifacts files

2.3 Codec and test tools

2.3.1 Codec for test

1) switch to your codec branch and run conformance test

if you change algorithm or you refactor code, you need to run conformance test for your code change. For example, you push your code change to your repos under branch Algorithm_V1.0,

- Branch="Algorithm_V1.0"
- Repos=" https://github.com/shihuade/openh264.git"

and run conformance test on Linux,

you can:

1) modify below parameters in case configuration file

```
#====================== Git Repository setting ==============
GitAddress   https://github.com/shihuade/openh264.git
GitBranch    Algorithm_V1.0
#============================================================
```

or

2) you can overwrite parameters by passing to run_Main.sh

```
./run_Main.sh  LocalTest  ./CaseConfigure/case_SVC.cfg  ${ Branch }  ${ Repos }
```

2) Update codec by scripts

- checkout openh264 repos and switch to test branch
  more detail, please refer to script files
  ./run_CheckoutRepos.sh

- build codec with enable YUV dump macro
  more detail, please refer to script file
  ./run_UpdateCodec.sh

- Update flow list as below:

```
./run_Main.sh
      ➔run_PrepareAllTestData.sh
        ➔runUpdateCodec()
             ./run_CheckoutRepos.sh
             ./run_UpdateCodec.sh
```

2.3.2 App tools for test

for more detail about test tools, please refer to below files:

- Codec_Linux/Tools32Bits/AboutTools.txt
- Codec_Linux/Tools64Bits/AboutTools.txt

For linux 32 bits test tools, list as below:

```
#*************************************************************
Tools app build on centos 6.5 32 bits version
i686
#*************************************************************
    ----JMDecoder/DownConvertStatic
        git clone from: https://github.com/shihuade/JSVM.git
        bin file from JSVM/bin/
        with build from JSVM/JSVM/H264Extension/build/linux/
        with default make setting
#*************************************************************
    ----JMDecoder
        git clone from: https://github.com/shihuade/JM.git
        JM19.0
        bin file from JM/bin/
        with build from JM/ with default make setting
#*************************************************************
    ----extractor.app
        from train pangu project
        bin file: under pangu/extractor/build/linux/bin/
        build under pangu/extractor/build/linux/bld/
#*************************************************************
```

Tools introduction:

- DownConvertStatic

  Downsample YUV for input test YUV

  ```
  In Scripts/run_PrepareInputYUV.sh
  ./DownConvertStatic  ${OriginW} ${OriginH} ${OriginYUV}
                       ${OutputW} ${OutputH} ${OutPutYUVFile}"
  ```

  Truncate frame num

  ```
  In Scripts/run_TruncateYUV.sh
  ./DownConvertStatic  ${PicW} ${PicH} ${InputYUV}
                       ${PicW} ${PicH} ${OutputYUV}
                       0 0 0 ${OutputFrmNum}
  ```

- JMDecoder

  It is for validating encoded bit stream which encoded by test cases, and, check if pass below iterms:

  I. Bit stream decoder by JMDecoder successfully
  II. Decoded YUV is the same with reconstruction YUV by test encoder

  For more detail, please refer to Scripts/run_CheckByJSVMDecoder.sh

- JSVMDecoder

  The same with JMDecoder, and support SVC multiple layer cases
  For more detail, please refer to Scripts/run_CheckByJSVMDecoder.sh

- extractor.app

  extract single layer bit stream from multiple layer cases' bit stream.

  For more detail, please refer to Scripts/run_ExtractLayerBitStream.sh

2.4 Test cases

2.4.1 Test case configuration files

For Jenkins based test sets assignment configure:

in CaseConfigure/, there are 20 cfg files for SVC and SCC test case

```
case_Jenkins_SCC_TestSet0.cfg
case_Jenkins_SCC_TestSet1.cfg
case_Jenkins_SCC_TestSet2.cfg
case_Jenkins_SCC_TestSet3.cfg
case_Jenkins_SCC_TestSet4.cfg
case_Jenkins_SCC_TestSet5.cfg
case_Jenkins_SCC_TestSet6.cfg
case_Jenkins_SCC_TestSet7.cfg
case_Jenkins_SCC_TestSet8.cfg
case_Jenkins_SCC_TestSet9.cfg

case_Jenkins_SVC_TestSet0.cfg
case_Jenkins_SVC_TestSet1.cfg
case_Jenkins_SVC_TestSet2.cfg
case_Jenkins_SVC_TestSet3.cfg
case_Jenkins_SVC_TestSet4.cfg
case_Jenkins_SVC_TestSet5.cfg
case_Jenkins_SVC_TestSet6.cfg
case_Jenkins_SVC_TestSet7.cfg
case_Jenkins_SVC_TestSet8.cfg
case_Jenkins_SVC_TestSet9.cfg
```

For local /SGE test mode, configure files:

```
SVC: case_SCC.cfg
SCC: case_SCC.cfg
```

For SHA1 table generation, configure files:

```
SCC: case_SHA1Table_SCC.cfg
SVC: case_SHA1Table_SVC.cfg
```

For default encode parameters, encoder configure files:

```
welsenc.cfg
layer0.cfg
layer1.cfg
layer2.cfg
layer3.cfg
which copied from ${Openh264SrcDir}/testbin

you can refer to script in
scripts/run_UpdateCodec.sh
➔runCopyFile

 all those encode parameters in welenc.cfg and layerx.cfg
will be overwrited by parameters in case_xxx.cfg

you can refer to script in
      Scripts/run_TestOneCase.sh
      ➔runEncodeOneCase()
```

2.4.2 Test case generation

more detail, please refer to script file

./Scripts/run_GenerateCase.sh

Brief:

Combine all test cases in configure file like case_xxx.cfg

And output cvs file which contain all cases

```
./run_GenerateCase.sh $Case.cfg $TestSequence $OutputCaseFile
```

for example;

```
./run_GenerateCase.sh case_SVC.cfg ABC_1920X1080.yuv AllCase.csv
```

will output all test cases for ABC_1920X1080.yuv based on case_SVC.cfg

output cases csv file may look like:

UsageType	EncodedNur	NumSpaceLa	NumTempLa	PicWidth	PicHeiht	PicWLayer0	PicHLayer0	PicWLayer1
0	32	1	1	320	192	320	192	0
0	32	1	1	320	192	320	192	0
0	32	1	1	320	192	320	192	0
0	32	1	1	320	192	320	192	0
0	32	1	4	320	192	320	192	0
0	32	1	4	320	192	320	192	0
0	32	1	4	320	192	320	192	0

RCMode	FrameSkip	BROverAll	BRLayer0	BRLayer1	BRLayer2	BRLayer3	MaxBRLayer	MaxBRLayer	MaxBRLayer
-1	1	240.00	240	0	0	0	240	0	0
-1	1	240.00	240	0	0	0	240	0	0
-1	1	72.00	72	0	0	0	72	0	0
-1	1	72.00	72	0	0	0	72	0	0
-1	1	240.00	240	0	0	0	240	0	0
-1	1	240.00	240	0	0	0	240	0	0
-1	1	72.00	72	0	0	0	72	0	0
-1	1	72.00	72	0	0	0	72	0	0

IntraPeriod	MultipleThre	LoadBalanci	EnableLongT	LoopFilterDi	DenoiseFlag	SceneChang	Backgroundl	AQFlag
0	1	0	1	0	1	1	0	1
0	1	0	1	0	1	1	1	1
0	1	0	1	0	1	1	0	1
0	1	0	1	0	1	1	1	1
0	1	0	1	0	1	1	0	1
0	1	0	1	0	1	1	1	1
0	1	0	1	0	1	1	0	1
0	1	0	1	0	1	1	1	1

2.5 Test data

2.5.1 Test data for one case during test

When start to run test for one test set, script will go to test space,

- TestSpace=./AllTestData/${TestYUV}
- And all temp files will be output to:
  ${TestSpace}/${TempDataPath},
  which empDataPath=" TempData"
- And all temp files list below

```
CheckLogFile="${TempDataPath}/CaseCheck.log";
EncoderLog="${TempDataPath}/encoder.log"
RecYUVFile0="${TempDataPath}/${TestYUVName}_rec_0.yuv";
RecYUVFile1="${TempDataPath}/${TestYUVName}_rec_1.yuv"
RecYUVFile2="${TempDataPath}/${TestYUVName}_rec_2.yuv";
RecYUVFile3="${TempDataPath}/${TestYUVName}_rec_3.yuv"
RecCropYUV0="${TempDataPath}/${TestYUVName}_rec_0_cropped.yuv";
RecCropYUV1="${TempDataPath}/${TestYUVName}_rec_1_cropped.yuv"
RecCropYUV2="${TempDataPath}/${TestYUVName}_rec_2_cropped.yuv";
RecCropYUV3="${TempDataPath}/${TestYUVName}_rec_3_cropped.yuv"
```

For more detail, please refer to script

run_TestAssignedCases.sh
run_TestOneCase.sh

2.5.2 Test data for one test cases set

1. One test cases set is sub-set of all test cases for one YUV.

For example, for desktop_dialog_1920x1080_i420.yuv

- In case_SCC.cfg, oveall test cases is num 10368

- In case_Jenkins_SCC_TestSet0.cfg, test case num is 2592
  When run below command in local test machine:
  ./run_Main.sh LocalTest case_Jenkins_SCC_TestSet0.cfg
  only sub-set of 2592 cases of all 10368 cases.

2. Test data files for one test set is under ResultPath and IssueDataPath,

- TestSpace=./AllTestData/${TestYUV}
- ResultPath=${ TestSpace }/result
- IssueDataPath=${ TestSpace }/issue

For more detail, please refer to script
run_TestAssignedCases.sh
Below is part of script for test data path setting in *run_TestAssignedCases.sh*

```
AssignedCasesPassStatusFile="${ResultPath}/${TestYUVName}_AllCasesOutput_SubCasesIndex_${SubCaseIndex}.csv"
UnPassedCasesFile="${ResultPath}/${TestYUVName}_UnpassedCasesOutput_SubCasesIndex_${SubCaseIndex}.csv"
AssignedCasesSHATableFile="${ResultPath}/${TestYUVName}_AllCases_SHA1_Table_SubCasesIndex_${SubCaseIndex}.csv"
CaseSummaryFile="${ResultPath}/${TestYUVName}_SubCasesIndex_${SubCaseIndex}.Summary.log"
AssignedCasesConsoleLogFile="${ResultPath}/${TestYUVName}_AssignedCases_SubCaseIndex_${SubCaseIndex}_0.TestLog"
```

Below is test result files for CiscoVT2people_320x192_12fps.yuv

```
ls -l AllTestData/CiscoVT2people_320x192_12fps.yuv/result/

- 29 13:47 CiscoVT2people_320x192_12fps.yuv.passFlag
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_AllCases.csv
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_AllCases.csv
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_AssignedCases_SubCaseIndex_AllCases_0.TestLog
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_AllCases.Summary.log
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_AllCases.csv
```

For subcase index,
- SGE test mode: index = 0, 1, 2, …
- Local test mode: index = AllCases

For more detail, please refer to run_CasesPartition.sh

3. For all sub test cases console output, you can refer to
AssignedCasesConsoleLogFile
Below is test console output for CiscoVT2people_320x192_12fps.yuv
For each cases, there is detail console output log include:

- Test cases detail parameters
- Encoder log, include encoder command line, encoder output
- Encoder reconstruction YUV's check
- JSVM/JM check log
- Wlsdecoder check log
- Final check summary info

```
***************case index is 0***********

*************************************************************
     call bash file is ./run_TestOneCase.sh
     input parameters are:
        ./run_TestOneCase.sh 0, 32, 1, 1, 320, 192, 320,192,0,0,0,0,0,0, 12, 12,12
, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1
*************************************************************


 Encoded command line is:
./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer
 -dw 0 320 -dh 0 192 -dw 1 0 -dh 1 0 -dw 2 0 -dh 2 0 -dw 3 0 -dh 3 0 -frout 0 12 -
 -1 -fs 1 -tarb 240.00 -ltarb 0 240.00 -ltarb 1 0 -ltarb 2 0 -ltarb 3 0 -lmaxb 0 2
um 1 0 -slcmd 2 0 -slcnum 2 0 -slcmd 3 0 -slcnum 3 0 -nalsize 0 -iper 0 -thread 1
coVT2people_320x192_12fps.yuv_SubCaseIndex_AllCases_CaseIndex_0_openh264.264 -org
192_12fps.yuv/CiscoVT2people_320x192_12fps.yuv -drec 0 TempData/CiscoVT2people_326
_1.yuv -drec 2 TempData/CiscoVT2people_320x192_12fps.yuv_rec_2.yuv -drec 3 TempDat
Width:          320
Height:         192
Frames:         9
encode time:    0.060529 sec
FPS:            148.689058 fps


*************************************************************
     call bash file is ./run_CheckBasicCheck.sh
     input parameters are:
        ./run_CheckBasicCheck.sh 0 1 -1
*************************************************************


---------------Basic Check--------------------------------------------
-------------------1. Basic Check--Encoded Failed Check
-------------------2. Basic Check--RecYUV Check
-------------------3. Basic Check--Crop RecYUV for JSVM comparison

*************************************************************
     call bash file is ./run_CropYUV.sh
     input parameters are:
        ./run_CropYUV.sh TempData/CiscoVT2people_320x192_12fps.yuv_rec_0.yuv TempD
*************************************************************


  YUV resolution is multiple of 16, no need to crop

-------------------4. Basic Check--Encoded Number Check
RecYUV   size: 829440
InputYUV size: 829440

 basic check passed!
    1.encoded failed check passed!
    2.cropped YUV check passed!
    3.encoded number check  passed!


*************************************************************
     call bash file is ./run_CheckByJSVMDecoder.sh
     input parameters are:
        ./run_CheckByJSVMDecoder.sh TempData/CiscoVT2people_320x192_12fps.yuv_SubC
Sat Apr 29 13:47:04 CST 2017

*************************************************************

---------------JSVM Check---------------------------------------------
-------------------1. JSVM Check--extract bit stream

*************************************************************
     call bash file is ./run_ExtractLayerBitStream.sh
     input parameters are:
        ./run_ExtractLayerBitStream.sh 1 TempData/CiscoVT2people_320x192_12fps.yuv
2_12fps.yuv_SubCaseIndex_AllCases_CaseIndex_0_openh264.264 TempData/BitStream_Laye
*************************************************************


  bit stream extraction succeed

-------------------2. JSVM Check--JSVM Decode Check
-------------------3. JSVM Check--WelsDecoder Decode Check
H264 source file name: TempData/CiscoVT2people_320x192_12fps.yuv_SubCaseIndex_AllC
Sequence output file name: TempData/Dec_WelsDec_0.yuv..
-------------------------------------------------------
iWidth:         320
height:         192
Frames:         9
decode time:    0.003610 sec
FPS:            2493.074792 fps
-------------------------------------------------------
-------------------4. Generate SHA1
-------------------5. JSVM Check--RecYUV-JSVMDecYUV-WelsDecYUV Comparison
-------------------6. JSVM Check--Check Result

WelsDecodedFailedFlag  0
FinalCheckFlag         0
RecJSVMFlag            0
WelsDecJSVMFlag        0
SpatialLayerNum        1

  Passed!:RecYUV--JSVMDecYUV WelsDecYUV--JSVMDecYUV

Sat Apr 29 13:47:04 CST 2017

---------------parse and Cat Check Log file--------------------
EncoderPassedNum:   1
EncoderUnPassedNum: 0
DecoderPassedNum:   1
DecoderUpPassedNum: 0
DecoderUnCheckNum:  0
EncoderCheckResult: 0-Encoder passed!
DecoderCheckResult: 0-Decoder passed!
```

2.5.3 How to reproduce failed case

1. **Reproduce based on console output log**
   As mentioned in #3, there are console output log for each case,
   you can find out the root cause for failed cases, and run that failed case step by step based on log

2. **Reproduce based on csv file**
   Below is test result files for CiscoVT2people_320x192_12fps.yuv
   Scv file: CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_AllCases.csv
   Include all test cases status and encoder command line.

   You can run that cases with:
   - encoder command
   - using JSVM/JM to decode bit stream
   - check decoded YUV which decoded by JSVM/JM/Welsdecoder
   - validate bit stream and YUVs

```
ls -l AllTestData/CiscoVT2people_320x192_12fps.yuv/result/

 29 13:47 CiscoVT2people_320x192_12fps.yuv.passFlag
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_AllCases.csv
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_AllCases.csv
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AssignedCases_SubCaseIndex_AllCases_0.TestLog
 29 13:47 CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_AllCases.Summary.log
 29 13:47 CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_AllCases.csv
```

BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL
-denois	-scene	-bgd	-aq	Command						
1	1	0	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	1	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	0	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	1	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	0	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	1	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	0	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						
1	1	1	1	./h264enc welsenc.cfg -lconfig 0 layer0.cfg -lconfig 1 layer1.cfg -lconfig 2 layer2.cfg -lconfig 3 laye						

2.5.4 Test data for all YUVs

During running test cases and after completed all test cases for one YUVs, test result files are in ./FinalResult directory.

1. **Below is example for only one YUV in test**
   - test YUV is CiscoVT2people_320x192_12fps.yuv
   - test set num is 1

```
ls -l FinalResult/

- 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_AllCases.csv
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_AllCases.csv
- 29 13:47 CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_AllCases.csv
- 29 13:47 TestReport_CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_AllCases.report.log
```

if there are 4 set for CiscoVT2people_320x192_12fps.yuv,
the result file may looks like:

```
CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_0.csv
CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_1.csv
CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_2.csv
CiscoVT2people_320x192_12fps.yuv_AllCasesOutput_SubCasesIndex_3.csv

CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_0.csv
CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_1.csv
CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_2.csv
CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table_SubCasesIndex_3.csv

CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_0.csv
CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_1.csv
CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_2.csv
CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput_SubCasesIndex_3.csv

TestReport_CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_0.report.log
TestReport_CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_1.report.log
TestReport_CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_2.report.log
TestReport_CiscoVT2people_320x192_12fps.yuv_SubCasesIndex_3.report.log
```

2. Below is the final test result setting in scripts

run_Main.sh
➔runMain()

```
runMain()
 {
    runCheck

    #dir translation
    AllTestDataFolder="AllTestData"
    CodecFolder="Codec"
    ScriptFolder="Scripts"
    SHA1TableFolder="SHA1Table"
    ConfigureFolder="CaseConfigure"
    FinalResultDir="FinalResult"
    AllJobsCompletedFlagFile="AllSGEJobsCompleted.flag"
    AllTestResultPassFlag="AllCasesPass.flag"
```

run_Main.sh
➔runMain()

```
echo -e "\033[32m ********************************************************************************\
echo -e "\033[32m   testing all cases for all test sequences......                               \
echo -e "\033[32m ********************************************************************************\
./run_TestAllSequencesAllCasesTest.sh  ${TestType}  ${AllTestDataFolder}  ${FinalResultDir} $
AllTestFlag=$?
```

run_AllTestSequencesAllCasesTest.sh
➔runLocalTest

```
        cd  ${SubFolder}
        CasesFile=${TestYUV}_AllCase.csv
        echo ""
        echo "test YUV is ${TestYUV}"
        echo ""
        ./run_TestOneYUVWithAssignedCases.sh  ${TestType}      ${TestYUV}  ${FinalResultDir} \
                                              ${ConfigureFile}  AllCases   ${CasesFile}

        let "AllSequencesAllCassesPassedFlag = $?"
```

run_TestOneYUVWithAssignedCases.sh
➔runGlobalVariableInitial

```
runGlobalVariableInitial()
{
    HostName=`hostname`
    TestReport="${FinalResultDir}/TestReport_${TestYUVName}_SubCasesIndex_${SubCaseIndex}.report.log"
    TestSummaryFileName="${TestYUVName}_SubCasesIndex_${SubCaseIndex}.Summary.log"
    InputYUVCheck="InputYUVCheck.log"
    YUVDeleteLog="DeletedYUVList.log"
    ConfigureFile=`echo ${ConfigureFile} | awk 'BEGIN {FS="/"} {print $NF}'`
    LocalWorkingDir=""
    TestYUVFullPath=""
    CurrentDir=`pwd`
    TestResult=""
```

2.6 Test report

After all test cases for all YUVs have been completed,
final test result and report for all YUVs will be generated
and create under directory
./FinalTestReport

1. **Below is example for only one YUV in test,**

   test YUV is CiscoVT2people_320x192_12fps.yuv

```
ls -l FinalTestReport/

 29 13:47 AllTestResultCombineConsole.txt
 29 13:47 AllTestYUVsAllSlavesDetailTestReport.txt
 29 13:47 AllTestYUVsSummary.txt
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCasesAllSlaves.Summary.log
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCasesOutput.csv
 29 13:47 CiscoVT2people_320x192_12fps.yuv_AllCases_SHA1_Table.csv
 29 13:47 CiscoVT2people_320x192_12fps.yuv_UnpassedCasesOutput.csv
 29 13:47 TestReport_CiscoVT2people_320x192_12fps.yuv.log
```

   output of TestReport_CiscoVT2people_320x192_12fps.yuv.log

```
*************************************************************************
*        Test report of all cases for YUV CiscoVT2people_320x192_12fps.yuv
*************************************************************************
  All Cases passed the test!
    Succeed!
total case  Num      is : 8
EncoderPassedNum     is : 8
EncoderUnPassedNum   is : 0
DecoderPassedNum     is : 8
DecoderUpPassedNum   is : 0
DecoderUnCheckNum    is : 0
*************************************************************************
*************************************************************************
```

   output of AllTestYUVsAllSlavesDetailTestReport.txt

```
HUASHI-M-400W:ConformanceTest huashi$ cat FinalTestReport/AllTestYUVsAllSlavesDetailTestReport.txt

*****************************************************************************************************
 Test report for YUV CiscoVT2people_320x192_12fps.yuv
*****************************************************************************************************
  test host          is: HUASHI-M-400W
  test type          is: LocalTest
  test directory     is: /Users/huashi/project/Openh264/ConformanceTest/AllTestData/CiscoVT2people_320x192_12fps.yuv
  AssignedCasesFile  is: CiscoVT2people_320x192_12fps.yuv_AllCase.csv
  test YUV full path is: /Users/huashi/project/Openh264/YUV/CiscoVT2people_320x192_12fps.yuv
*****************************************************************************************************
  Sub-Case Index is: AllCases
  Host name      is: HUASHI-M-400W
  SGE job ID     is:
  SGE job name   is:
*****************************************************************************************************
@runCheckInputYUV,TestYUVFullPath is /Users/huashi/project/Openh264/YUV/CiscoVT2people_320x192_12fps.yuv
*****************************************************************************************************
            Test summary for CiscoVT2people_320x192_12fps.yuv
*****************************************************************************************************
   TestStartTime    is: Sat Apr 29 13:47:04 CST 2017
   TestEndTime      is: Sat Apr 29 13:47:07 CST 2017
*****************************************************************************************************
   total case  Num     is : 8
   EncoderPassedNum    is : 8
   EncoderUnPassedNum  is : 0
   DecoderPassedNum    is : 8
   DecoderUpPassedNum  is : 0
   DecoderUnCheckNum   is : 0
*****************************************************************************************************
    --issue bitstream can be found in /Users/huashi/project/Openh264/ConformanceTest/AllTestData/CiscoVT2people_320x192_12fps.yuv/issue
    --detail result can be found in   /Users/huashi/project/Openh264/ConformanceTest/AllTestData/CiscoVT2people_320x192_12fps.yuv/result
*****************************************************************************************************

Succeed!
All Cases passed the test!
```

2. Below is the final test report directoty setting in scripts

run_GetAllTestResult.sh
➔runMain()

```
runMain()
{
    CurrentDir=`pwd`
    #get full path info
    FinalResultDir=${CurrentDir}/FinalResult
    SHA1TableDir=${CurrentDir}/SHA1Table
    FinalTestReportDir=${CurrentDir}/FinalTestReport
    #check input parameters
    runCheck

    #Summary report for all test sequences
    AllTestSummary="${FinalTestReportDir}/AllTestYUVsSummary.txt"
    AllTestResultCombineConsole="${FinalTestReportDir}/AllTestResultCombineConsole.txt"
    #Detail report file for all test sequences under all slaves test report
    AllYUVAllSlavesTestReort="${FinalTestReportDir}/AllTestYUVsAllSlavesDetailTestReport.txt"
    let "AllTestFlag=0"
```

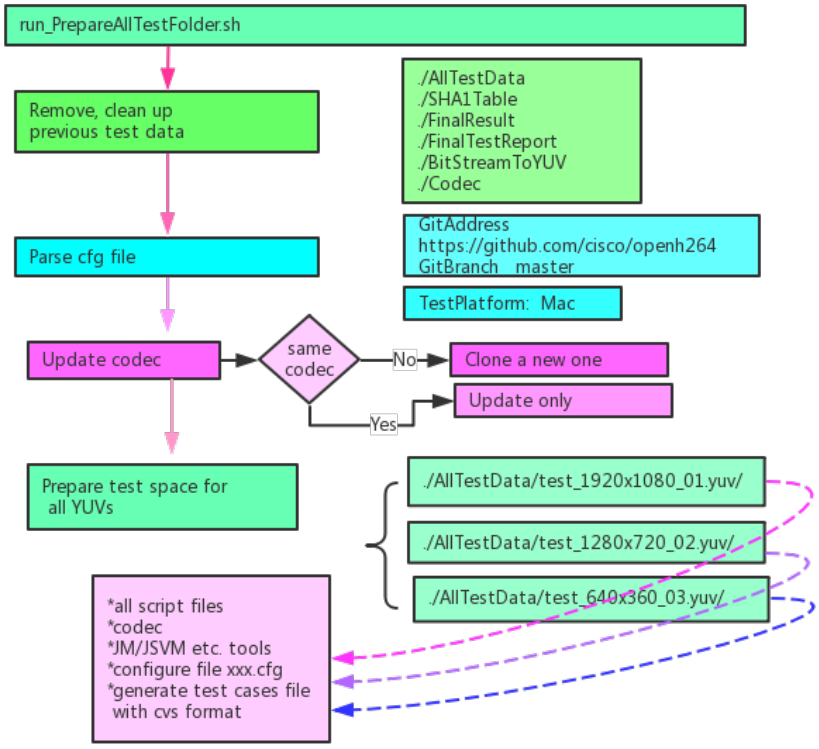
run_GetAllTestResult.sh
➔runGetAllYUVTestResult

```
runGetAllYUVTestResult()
{
    echo """>${AllTestSummary}
    echo """>${AllYUVAllSlavesTestReort}
    echo "YUV list based cfg file ${ConfigureFile} is:"
    echo "  ${aTestYUVList[@]}"
    for TestYUV in ${aTestYUVList[@]}
    do
        # combine sub-cases files into single all cases file
        echo ""
        echo "combining sub-set cases files into single all cases file..."
        echo ""
        DetailSummaryFile="${FinalTestReportDir}/${TestYUV}_AllCasesAllSlaves.Summary.log"
        SummaryFile="${FinalTestReportDir}/TestReport_${TestYUV}.log"
        SHA1TableFile="${FinalTestReportDir}/${TestYUV}_AllCases_SHA1_Table.csv"
        ./Scripts/run_SubCasesToAllCasesCombination.sh  ${FinalResultDir} ${TestYUV} 0
        ./Scripts/run_SubCasesToAllCasesCombination.sh  ${FinalResultDir} ${TestYUV} 1
        ./Scripts/run_SubCasesToAllCasesCombination.sh  ${FinalResultDir} ${TestYUV} 2
        ./Scripts/run_SubCasesToAllCasesCombination.sh  ${FinalResultDir} ${TestYUV} 3
        ./Scripts/run_SubCasesToAllCasesSummary.sh ${TestYUV} ${DetailSummaryFile} ${SummaryFile}

        if [ ! $? -eq 0 ]
        then
            let "AllTestFlag=1"
```

2.7 Basic flow chart for test preparation

2.7.1 basic flow chart

More detail, please refer to script file
run_PrepareAllTestFolder.sh



2.7.2 Example

1. **Below is example after updating codec**
   all codec and test tools will be copied to
   ./Codec folder

```
huashi$ ls -l Codec

   936 May  3 13:39 AboutTools.txt
 55805 May  3 13:39 DownConvertStatic
827539 May  3 13:39 JMDecoder
1026399 May  3 13:39 JSVMDecoder
 106044 May  3 13:39 extractor.a
  43199 May  3 13:39 extractor.app
  66647 May  3 13:39 extractor.so
 439676 May  3 13:42 h264dec
 853448 May  3 13:42 h264enc
   2185 May  3 13:42 layer0.cfg
   2185 May  3 13:42 layer1.cfg
   2185 May  3 13:42 layer2.cfg
   2185 May  3 13:42 layer3.cfg
   4444 May  3 13:42 welsenc.cfg
```

2. Below is example for removing previous test data and update codec

```
     call bash file is ./run_PrepareAllTestData.sh
     input parameters is:
        ./run_PrepareAllTestData.sh LocalTest CaseConfigure/case_SVC.cfg
*************************************************************

 ********************************************************************
    Removing previous test data
 ********************************************************************
deleted folder is:  AllTestData
deleted folder is:  SHA1Table
deleted folder is:  FinalResult
deleted folder is:  FinalTestReport
deleted folder is:  BitStreamToYUV
deleted folder is:  Codec
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/BitStreamToYUV.log
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/CiscoVT2people_320x1
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/CodecBuildInfo.log
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/CodecReposInfo.log
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/ReposCheckout.log
deleted file is :  /Users/huashi/project/Openh264/ConformanceTest/AllCasesPass.flag
 ************************************************************
 Repository         is https://github.com/cisco/openh264
 Branch             is master
 SGEJobSubCasesNum  is 1000
 SGEJobSubCasesNum  is 1000
 ************************************************************
 ********************************************************************
    updating test codec
 ********************************************************************
Test platform is Linux; copy JM/JSVM etc. tools to codec
uname: illegal option -- -
usage: uname [-amnprsv]

*************************************************************
     call bash file is ./run_CheckoutRepos.sh
     input parameters is:
        ./run_CheckoutRepos.sh https://github.com/cisco/openh264 master Source fast
*************************************************************

Source folder Source does not exist!
now change ReposUpdateOption to clone, which will clone a new repos to Source
***********************************************
   ReposUpdateOption is: clone
   repos addr        is: https://github.com/cisco/openh264
   branch            is: master
   update src dir    is: Source
***********************************************
***********************************************
  clone a new repos
```

3. Below is example after preparing all test space for all YUVs

```
 ********************************************************************************
   Test space preparation succeed
   All test data, please refer to: AllTestData
 ********************************************************************************
HUASHI-M-400W:ConformanceTest huashi$ ls -l AllTestData/
total 0
drwxr-xr-x  63 huashi  staff  2142 May  3 13:45 CiscoVT_2people_160x96_6.25fps.yuv
drwxr-xr-x  63 huashi  staff  2142 May  3 14:06 Jiessie_James_talking_1280x720_30.yuv
drwxr-xr-x  63 huashi  staff  2142 May  3 13:52 MSHD_320x192_12fps.yuv
drwxr-xr-x  63 huashi  staff  2142 May  3 13:59 candyHF2_640x480.yuv
```

4. **Below is example after preparing all test space for all YUVs**

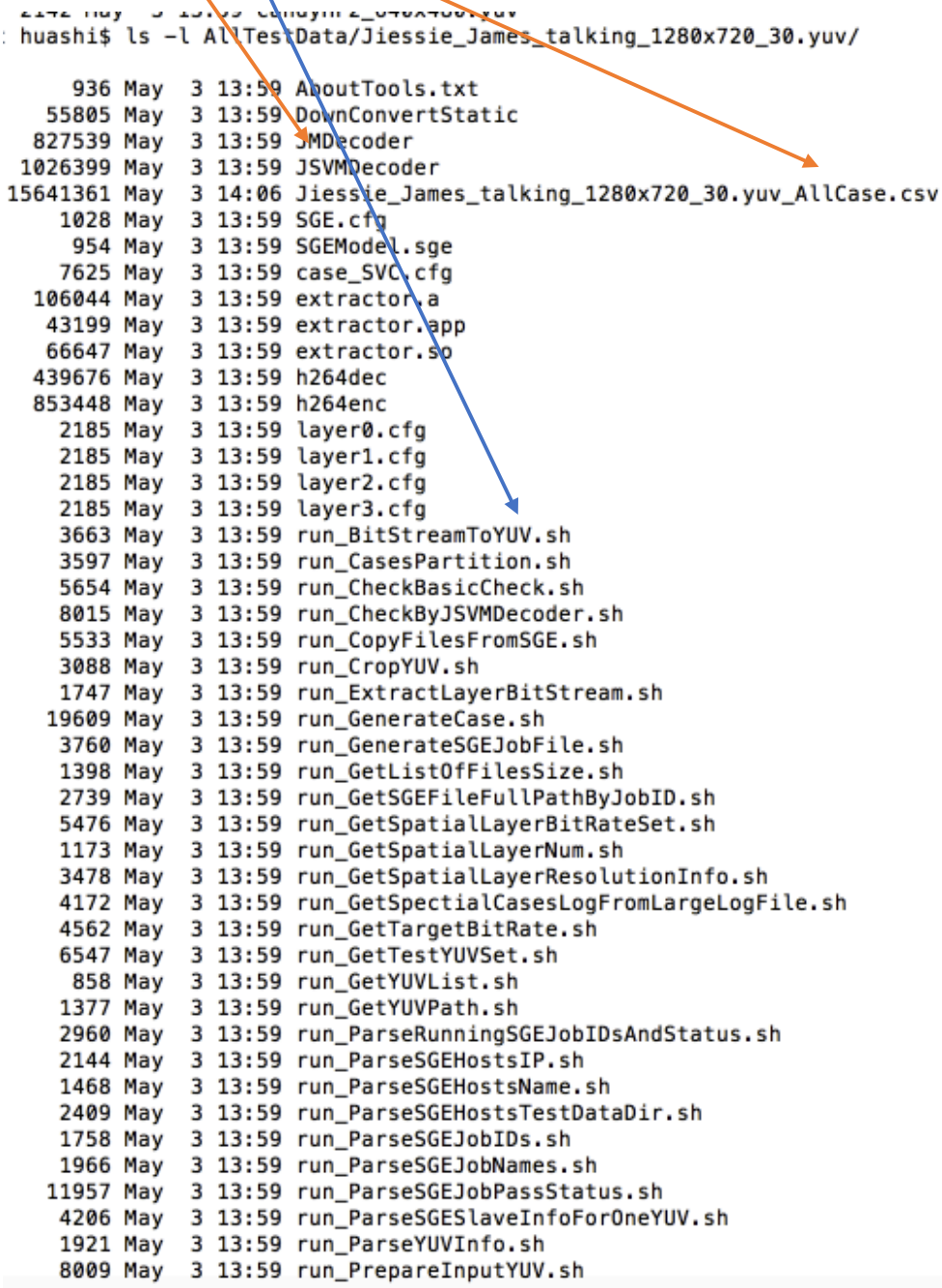
Take `Jiessie_James_talking_1280x720_30.yuv` **for example,**

Test space is:

`AllTestData/Jiessie_James_talking_1280x720_30.yuv/`

And files before run all test cases include:

- Encoder configure file welsenc.cfg etc.
- Codec app, JM app etc
- All test script files
- Csv format test cases file



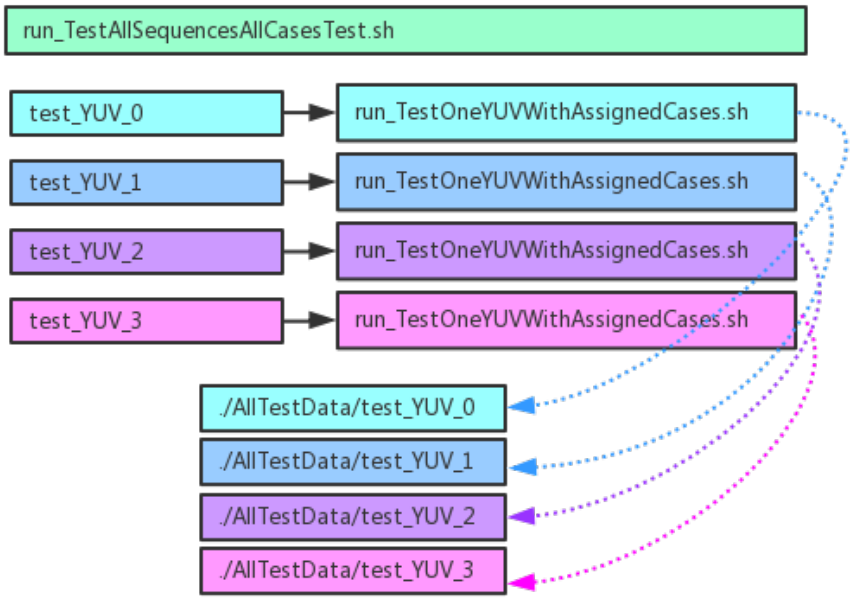
```
huashi$ ls -l AllTestData/Jiessie_James_talking_1280x720_30.yuv/

     936 May  3 13:59 AboutTools.txt
   55805 May  3 13:59 DownConvertStatic
  827539 May  3 13:59 JMDecoder
 1026399 May  3 13:59 JSVMDecoder
15641361 May  3 14:06 Jiessie_James_talking_1280x720_30.yuv_AllCase.csv
    1028 May  3 13:59 SGE.cfg
     954 May  3 13:59 SGEModel.sge
    7625 May  3 13:59 case_SVC.cfg
  106044 May  3 13:59 extractor.a
   43199 May  3 13:59 extractor.app
   66647 May  3 13:59 extractor.so
  439676 May  3 13:59 h264dec
  853448 May  3 13:59 h264enc
    2185 May  3 13:59 layer0.cfg
    2185 May  3 13:59 layer1.cfg
    2185 May  3 13:59 layer2.cfg
    2185 May  3 13:59 layer3.cfg
    3663 May  3 13:59 run_BitStreamToYUV.sh
    3597 May  3 13:59 run_CasesPartition.sh
    5654 May  3 13:59 run_CheckBasicCheck.sh
    8015 May  3 13:59 run_CheckByJSVMDecoder.sh
    5533 May  3 13:59 run_CopyFilesFromSGE.sh
    3088 May  3 13:59 run_CropYUV.sh
    1747 May  3 13:59 run_ExtractLayerBitStream.sh
   19609 May  3 13:59 run_GenerateCase.sh
    3760 May  3 13:59 run_GenerateSGEJobFile.sh
    1398 May  3 13:59 run_GetListOfFilesSize.sh
    2739 May  3 13:59 run_GetSGEFileFullPathByJobID.sh
    5476 May  3 13:59 run_GetSpatialLayerBitRateSet.sh
    1173 May  3 13:59 run_GetSpatialLayerNum.sh
    3478 May  3 13:59 run_GetSpatialLayerResolutionInfo.sh
    4172 May  3 13:59 run_GetSpectialCasesLogFromLargeLogFile.sh
    4562 May  3 13:59 run_GetTargetBitRate.sh
    6547 May  3 13:59 run_GetTestYUVSet.sh
     858 May  3 13:59 run_GetYUVList.sh
    1377 May  3 13:59 run_GetYUVPath.sh
    2960 May  3 13:59 run_ParseRunningSGEJobIDsAndStatus.sh
    2144 May  3 13:59 run_ParseSGEHostsIP.sh
    1468 May  3 13:59 run_ParseSGEHostsName.sh
    2409 May  3 13:59 run_ParseSGEHostsTestDataDir.sh
    1758 May  3 13:59 run_ParseSGEJobIDs.sh
    1966 May  3 13:59 run_ParseSGEJobNames.sh
   11957 May  3 13:59 run_ParseSGEJobPassStatus.sh
    4206 May  3 13:59 run_ParseSGESlaveInfoForOneYUV.sh
    1921 May  3 13:59 run_ParseYUVInfo.sh
    8009 May  3 13:59 run_PrepareInputYUV.sh
```

2.8 Basic flow chart for testing all cases

More detail, please refer to script file

run_TestAllSequencesAllCasesTest.sh

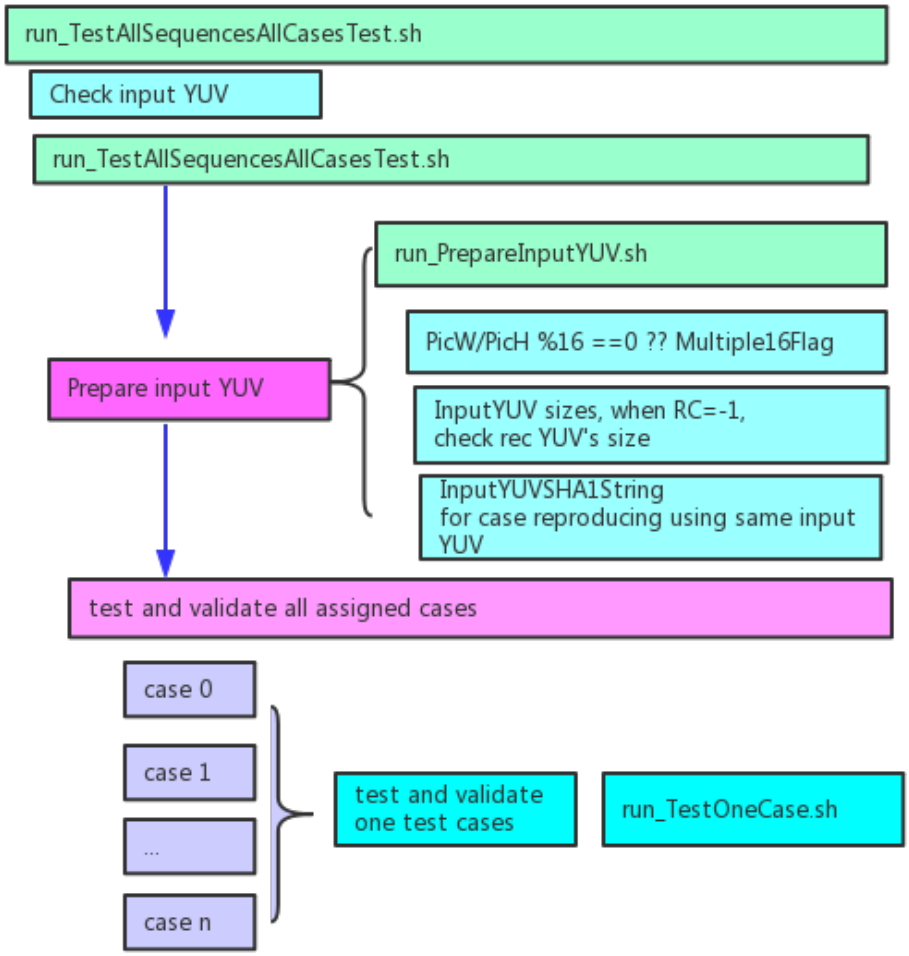
2.8.1 Test all cases for all YUVs



2.8.2 Test assigned cases within one test cases set for one YUV

More detail, please refer to script file

run_TestOneYUVWithAssignedCases.sh
run_TestAssignedCases.sh
run_PrepareInputYUV.sh



2.8.3 Test one case

More detail, please refer to script file

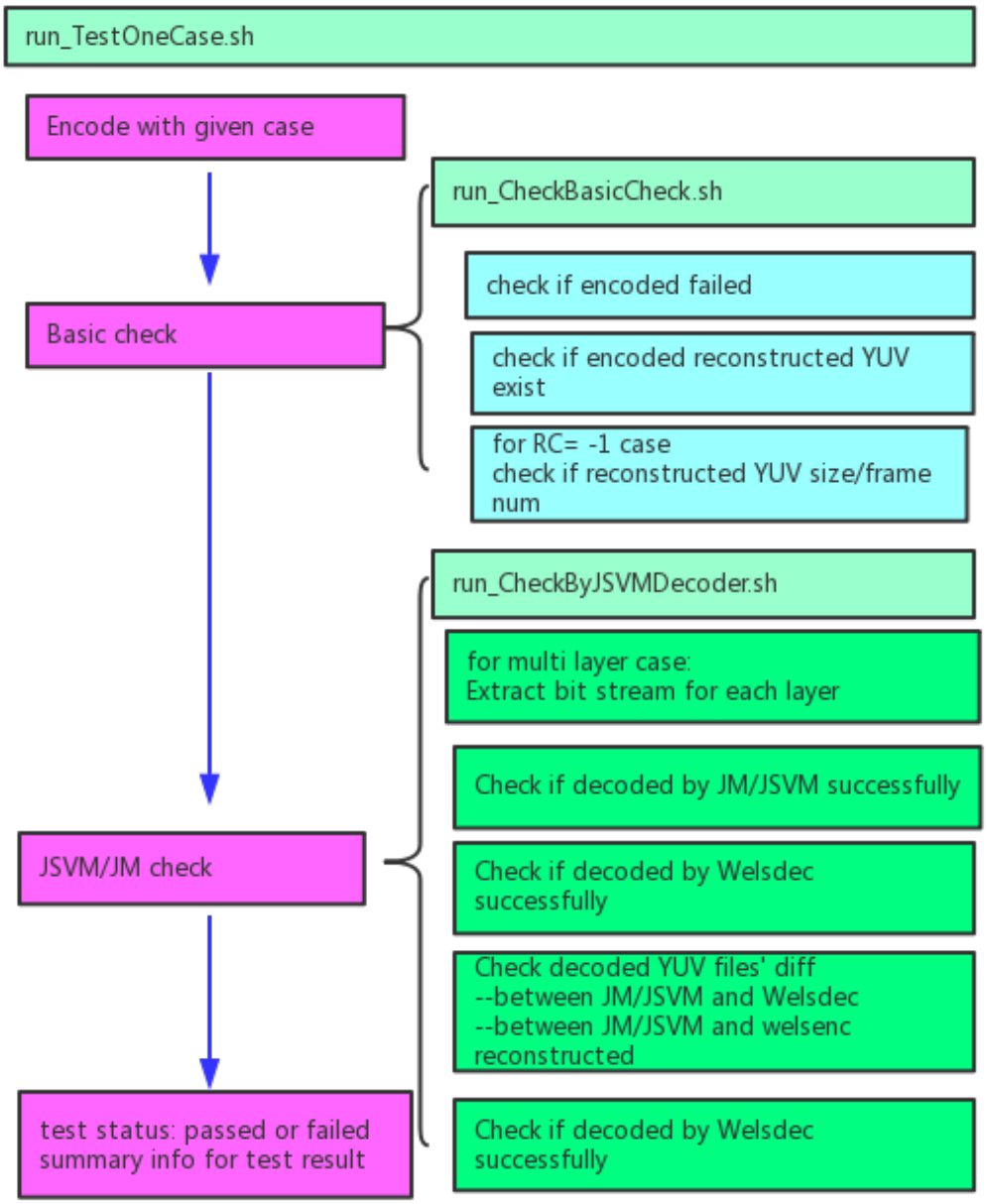
run_TestOneCase.sh

run_CheckBasicCheck.sh

➔ *run_CropYUV.sh*

run_CheckByJSVMDecoder.sh

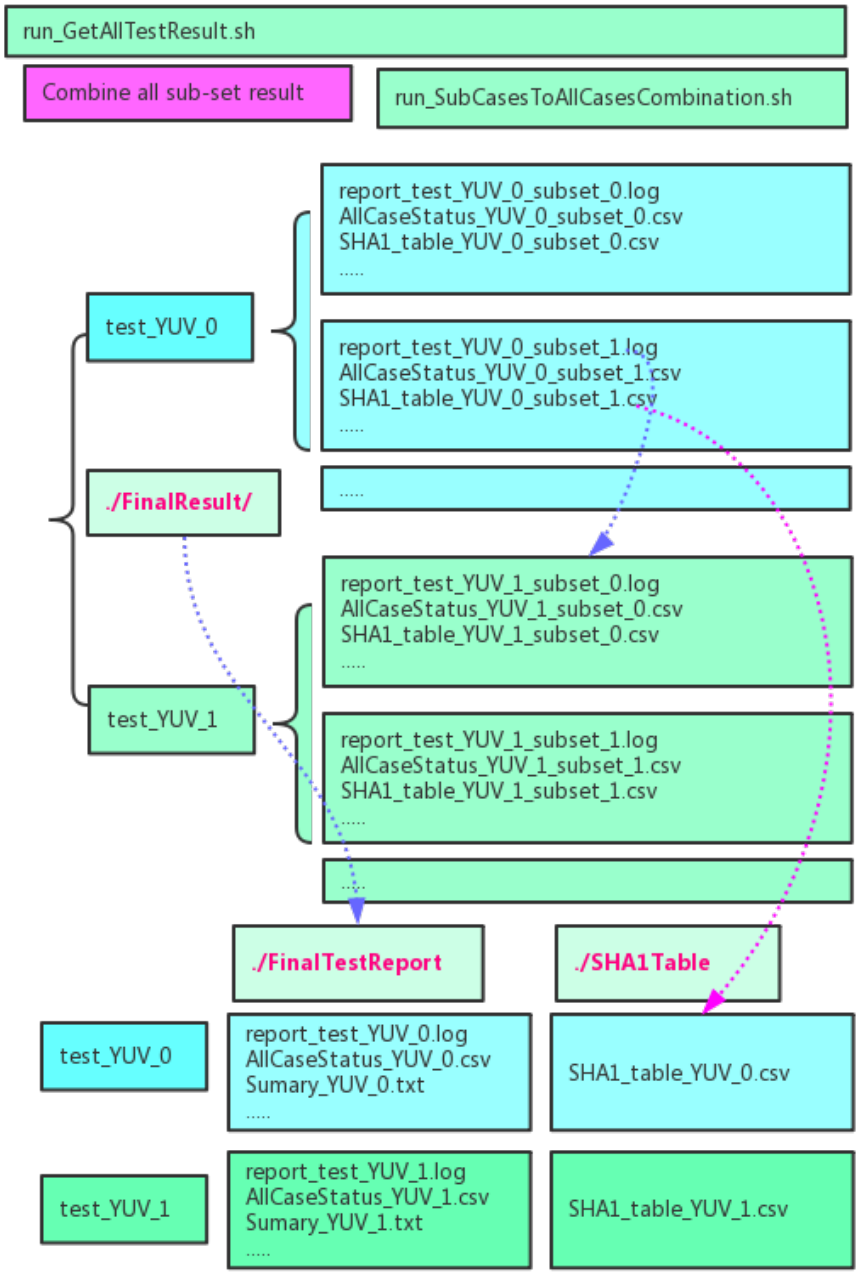
➔ *run_ExtractLayerBitStream.sh*



2.9 Basic flow chart for check final test result

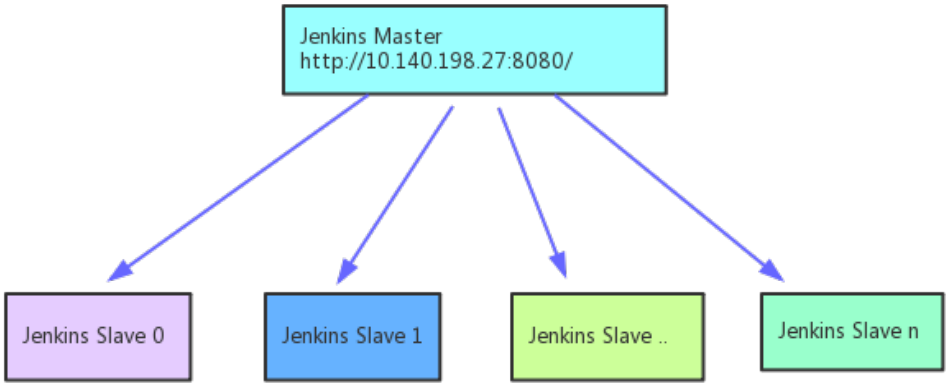
More detail, please refer to script file

run_GetAllTestResult.sh
run_SubCasesToAllCasesSummary.sh

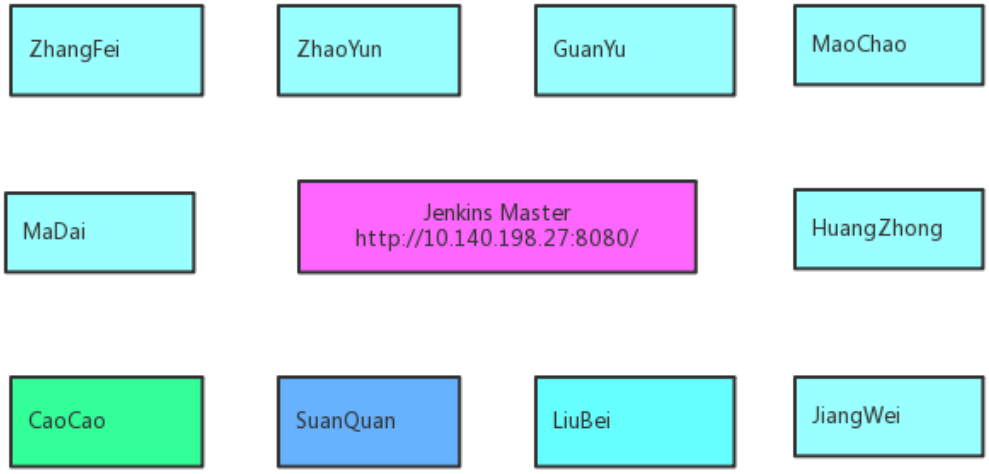


3 Jenkins based conformance test

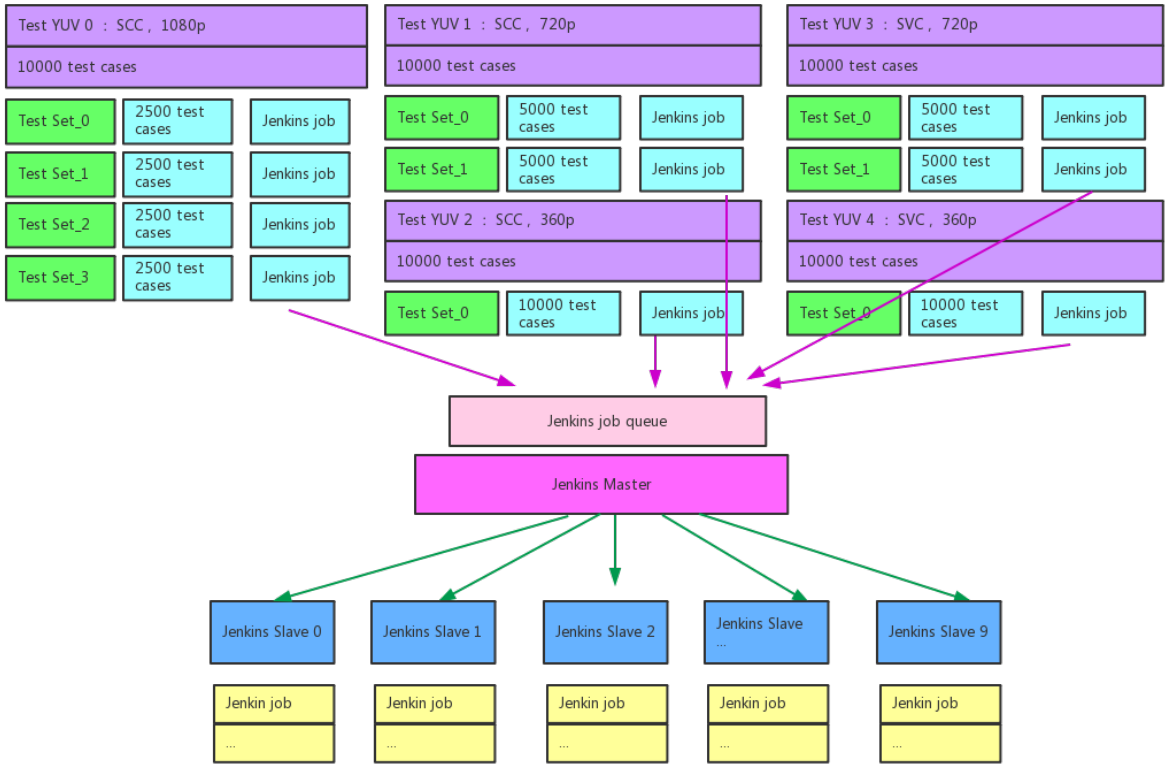
1. Basic test architecture



Current Jenkins slaves for conformance test in HZ site



2. Test set and Jenkins slaves task assignment



Test set for SCC/SVC:

in CaseConfigure/, there are 20 cfg files for SVC and SCC test case

```
case_Jenkins_SCC_TestSet0.cfg
case_Jenkins_SCC_TestSet1.cfg
case_Jenkins_SCC_TestSet2.cfg
case_Jenkins_SCC_TestSet3.cfg
case_Jenkins_SCC_TestSet4.cfg
case_Jenkins_SCC_TestSet5.cfg
case_Jenkins_SCC_TestSet6.cfg
case_Jenkins_SCC_TestSet7.cfg
case_Jenkins_SCC_TestSet8.cfg
case_Jenkins_SCC_TestSet9.cfg

case_Jenkins_SVC_TestSet0.cfg
case_Jenkins_SVC_TestSet1.cfg
case_Jenkins_SVC_TestSet2.cfg
case_Jenkins_SVC_TestSet3.cfg
case_Jenkins_SVC_TestSet4.cfg
case_Jenkins_SVC_TestSet5.cfg
case_Jenkins_SVC_TestSet6.cfg
case_Jenkins_SVC_TestSet7.cfg
case_Jenkins_SVC_TestSet8.cfg
case_Jenkins_SVC_TestSet9.cfg
```

For more detail about test set, please refer to:

```
AboutJenkinsTestSet.txt
```

3. Jenkins jobs

Jenkins job, please refer to:
http://10.140.198.27:8080/view/ConformanceTest/job/OpenH264-Encoder-Conformance-Test/

http://10.140.198.27:8080/job/OpenH264_ConformanceTest_SCC_TestSet0/configure

4. Test data and test report

For test result of each test set, you can refer to job artifact, which show as below:

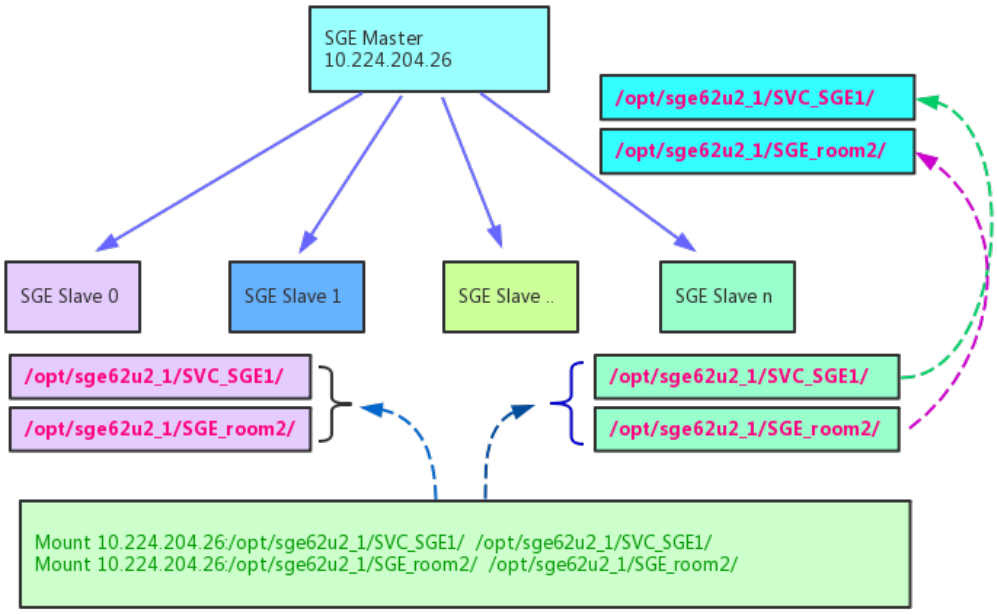
5. Failed cases reproduce and analysis

Please refer to 2.5.3 How to reproduce failed case

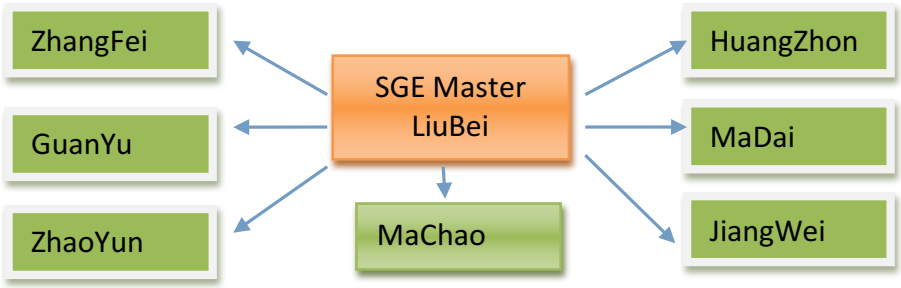
4. SGE based conformance test

1. Basic test architecture

For how to install and configure SGE system, please refer to Charpter SGE install and configuration

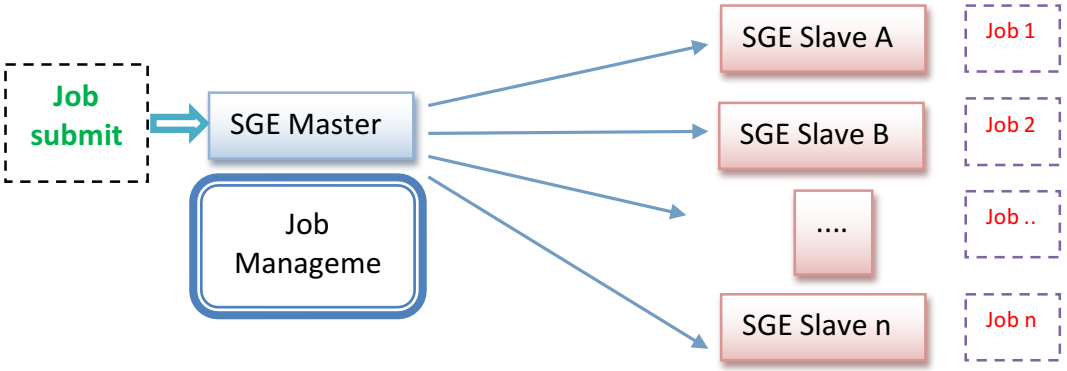


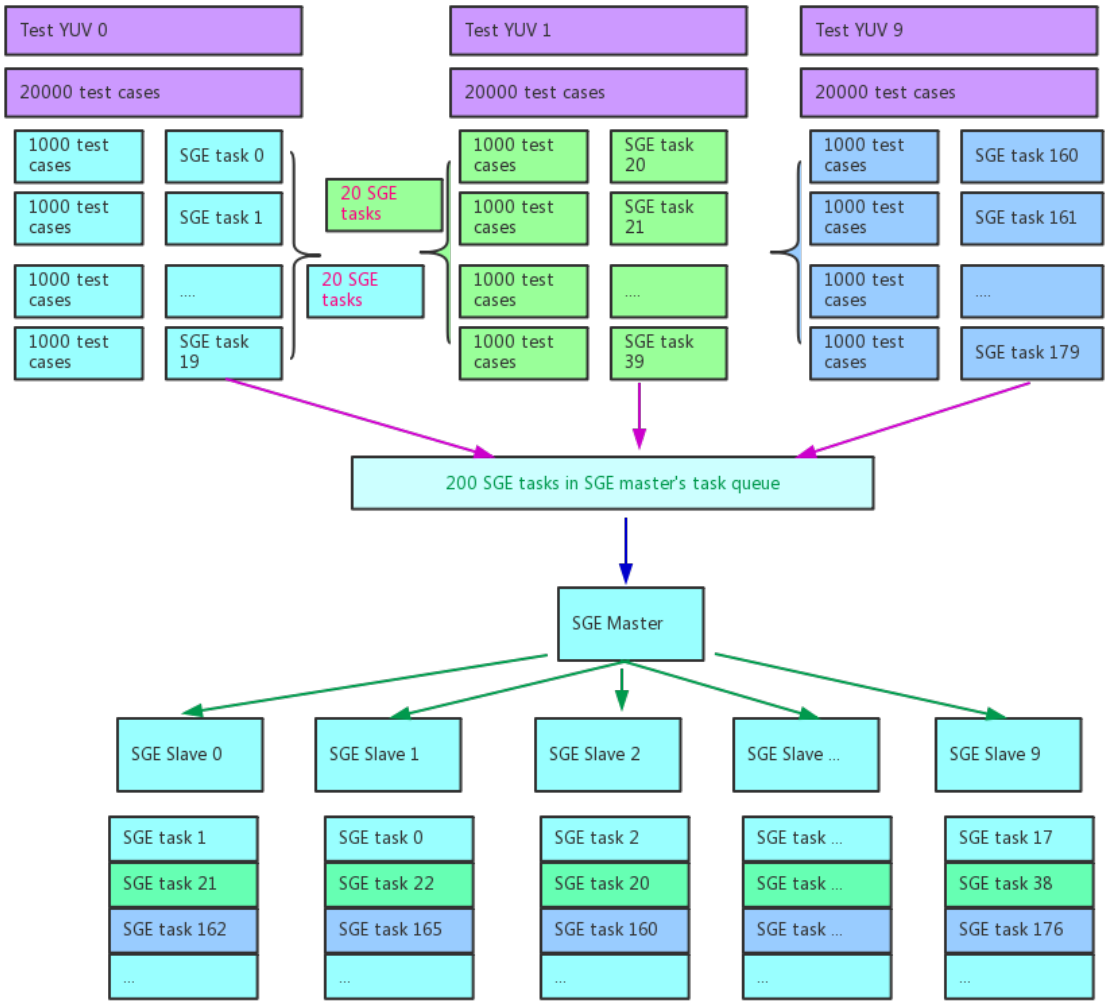
Current SGE in HZ



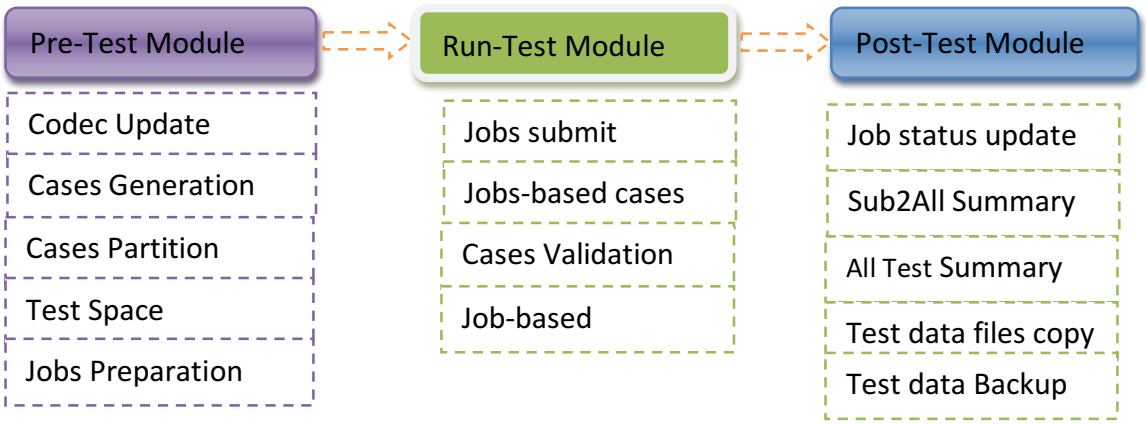
2. Test set and Jenkins slaves task assignment

4.2.1 Basic module

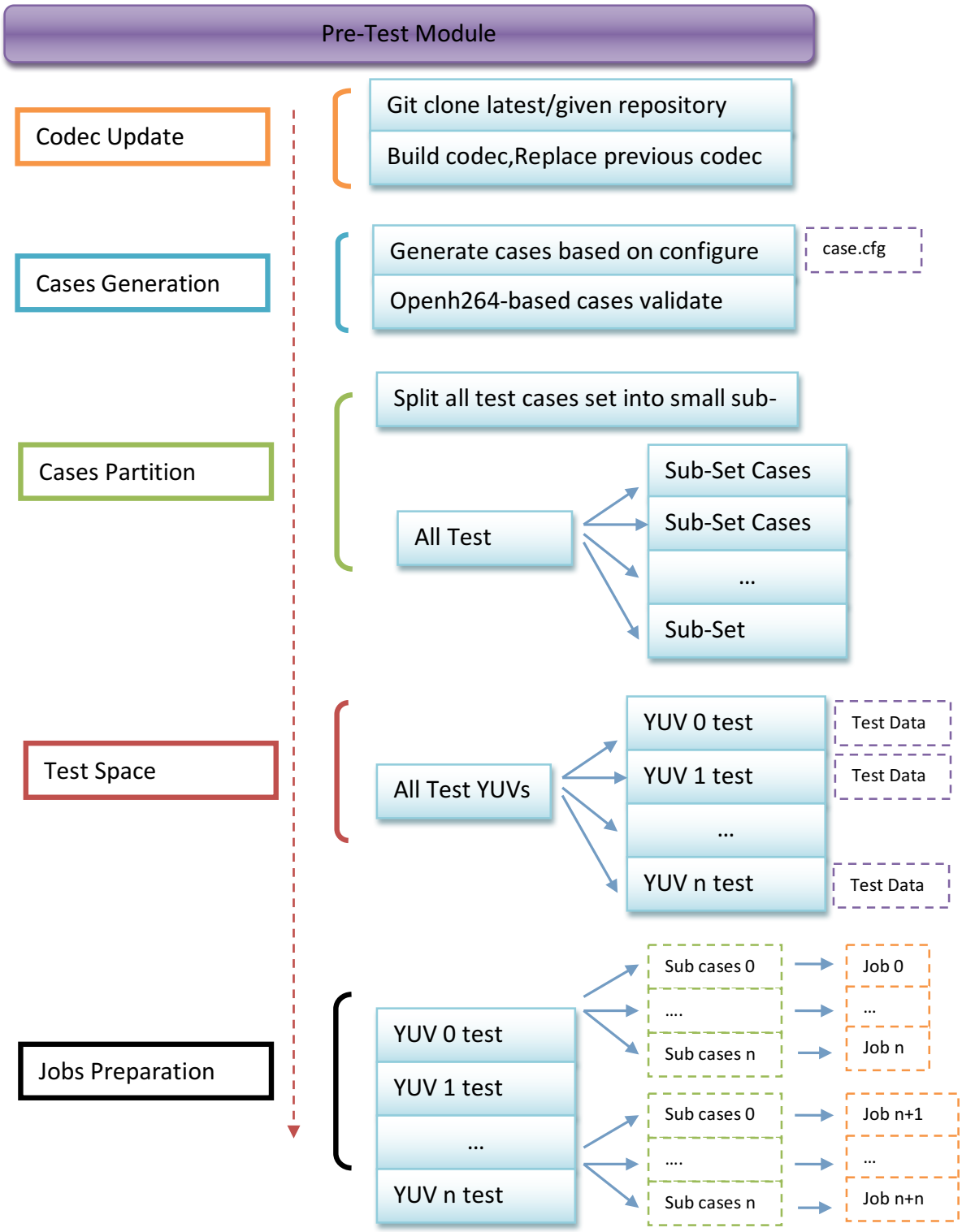




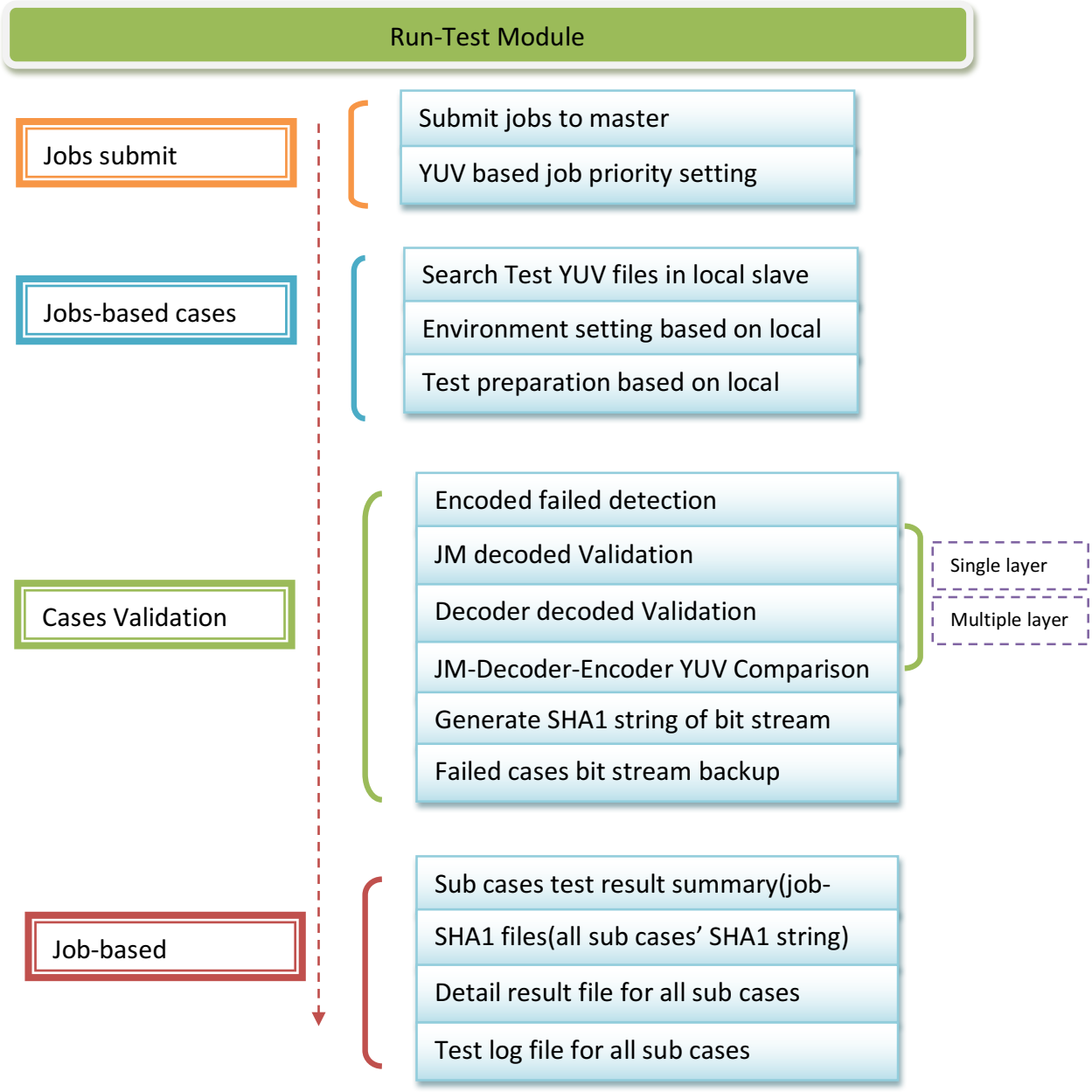
4.2.2 Basic SGE Test flowchart



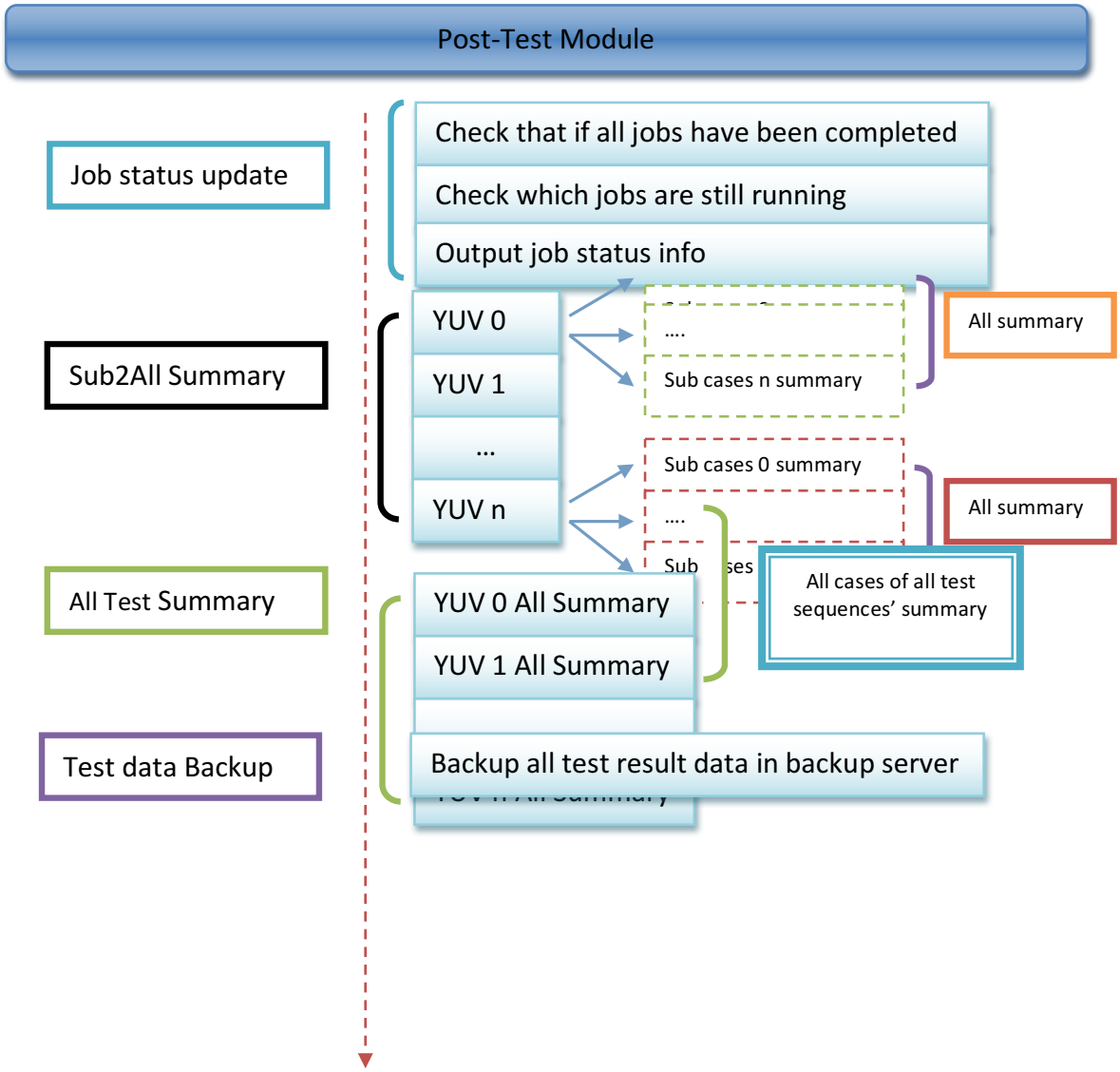
4.3 Pre-Module of SGE Test



4.4 Test Module of SGE Test



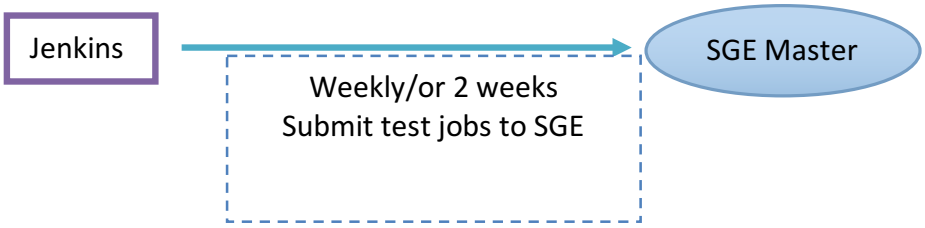
4.5 Post-Module of SGE Test



4.6 Jenkins jobs for SGE test

4.6.1 Period SGE submit

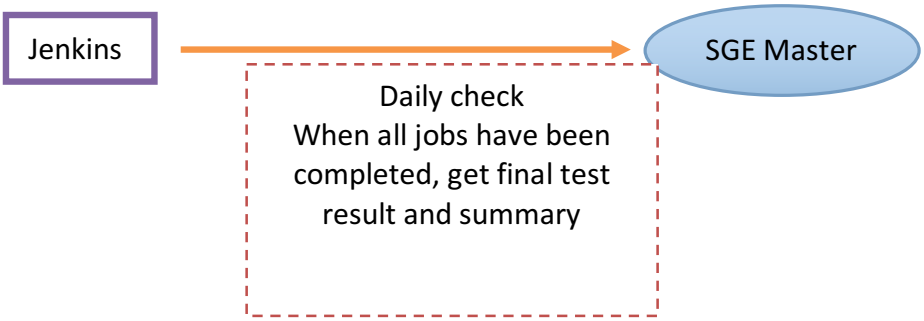
more detail, please refer to Jenkins job
http://10.140.198.27:8080/view/SGE-Test/job/Openh264-SGE-Job-Submit/configure



4.6.2 SGE test status detection jobs

more detail, please refer to Jenkins job

http://10.140.198.27:8080/view/SGE-Test/job/Openh264-SGE-Job-status-And-TestResult/configure



3. Test data and test report

For test report of each SGE submit,
you can refer to job artifact to get all test report and log files

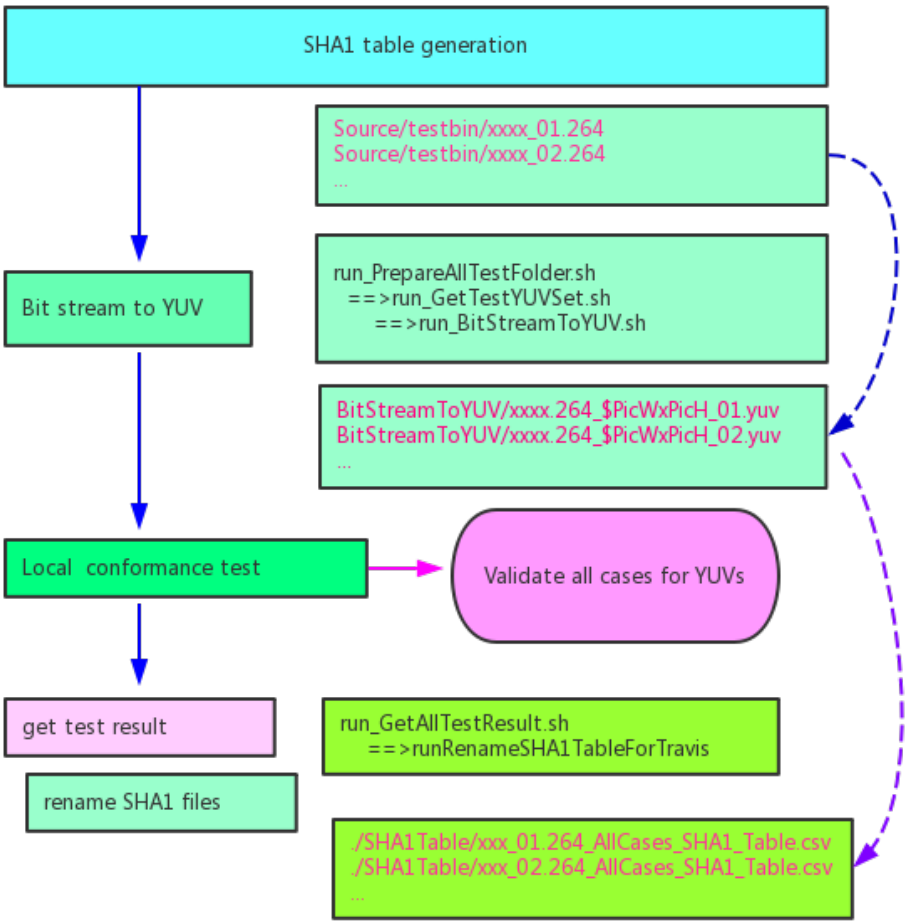
http://10.140.198.27:8080/view/SGE-Test/job/Openh264-SGE-Job-status-And-TestResult/lastSuccessfulBuild/artifact/Openh264-SGETest/Jenkins-Job-Status-Check-Log/

4. Failed cases reproduce and analysis

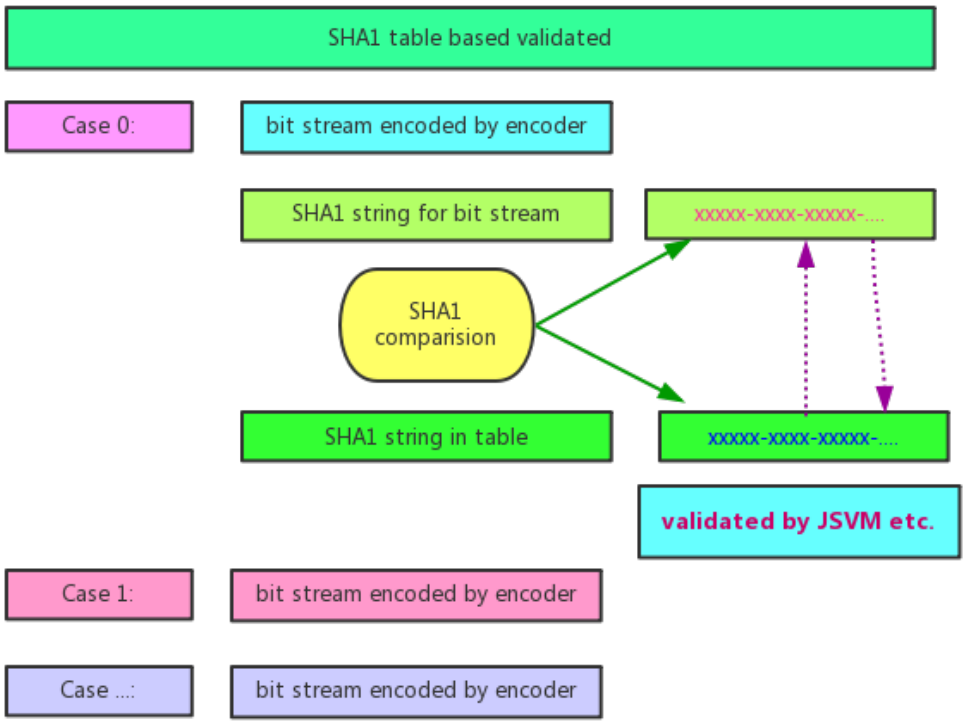
Please refer to **2.5.3 How to reproduce failed case**

5. SHA1 tables generation

5.1 basic arch



basic validate logic



5.2 setting in case configure file

In CaseConfureFile/*.cfg, like:

- ./CaseConfigure/case_SHA1Table_SCC.cfg
- ./CaseConfigure/case_SHA1Table_SVC.cfg

chane InputFormat to 1, and bitstream directory is Source/res

```
#=========== Test Input bit stream/TestSequence setting=================
TestBitStreamDir:   Source/res      #  derectory of input test bit stream files

InputFormat:          1      #Input file format, 0:YUV file; 1 bit stream file,
#======================================================================
```

5.3 SHA1 tables for openh264 travis test

1) SHA1 tables for openh264
   - OpenH264/test/encoder_binary_comparision
   - https://github.com/cisco/openh264/tree/master/test/encoder_binary_comparison/SHA1Table

2) Travis test

   Travis test setting:

   https://github.com/cisco/openh264/blob/master/.travis.yml

```
env:
  - TASK=UnitTest;      TestParameter=""
  - TASK=BinaryCompare; TestParameter=BA_MW_D.264
  - TASK=BinaryCompare; TestParameter=Zhling_1280x720.264
  - TASK=BinaryCompare; TestParameter=Adobe_PDF_sample_a_1024x768_50Frms.264
matrix:
  exclude:
    - compiler: clang
      env: TASK=BinaryCompare; TestParameter=BA_MW_D.264
    - compiler: clang
      env: TASK=BinaryCompare; TestParameter=Zhling_1280x720.264
    - compiler: clang
      env: TASK=BinaryCompare; TestParameter=Adobe_PDF_sample_a_1024x768_50Frms.264
script:
  -  echo "currrent test is for ${TASK}"
  -  echo "test parameter is ${TestParameter}"
  -  ./run_Test.sh ${TASK}  ${TestParameter}
```

Called by OpenH264/run_Test.sh

```
if [  "${TestType}"  = "BinaryCompare" ]
then
  set -e
  runPrepareAndBinaryTest ${TestBitStream} TravisTest
  return $?
fi
```

Travis test for one bit stream

```
  local BinaryTestDir="test/encoder_binary_comparison"
  local TestSpacePrepareLog="AllTestSpacePrepare.log"
  cd ${BinaryTestDir}
  ./run_PrepareAllTestData.sh 64 2>${TestSpacePrepareLog}
  cd ${WorkingDir}
  echo ""
  echo " binary compare test, test bit stream is ${TestBitStream}"
  echo ""
./test/encoder_binary_comparison/run_OneBitStream.sh  ${TestBitStream} ${TestType}
  return $?
```

Travis test script for SHA1 validation

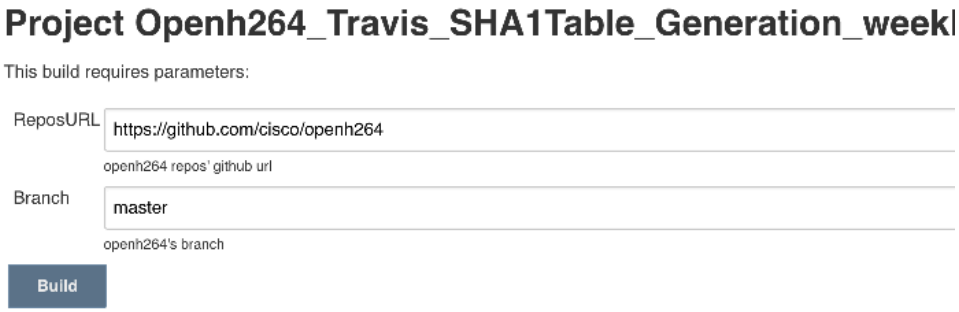
```
ls -l Source/test/encoder_binary_comparison/

 3 13:41 AboutTest
 3 13:41 SHA1Table
 3 13:41 Scripts
 3 13:41 run_Main.sh
 3 13:41 run_OneBitStream.sh
 3 13:41 run_PrepareAllTestData.sh
```

5.4 Jenkins job

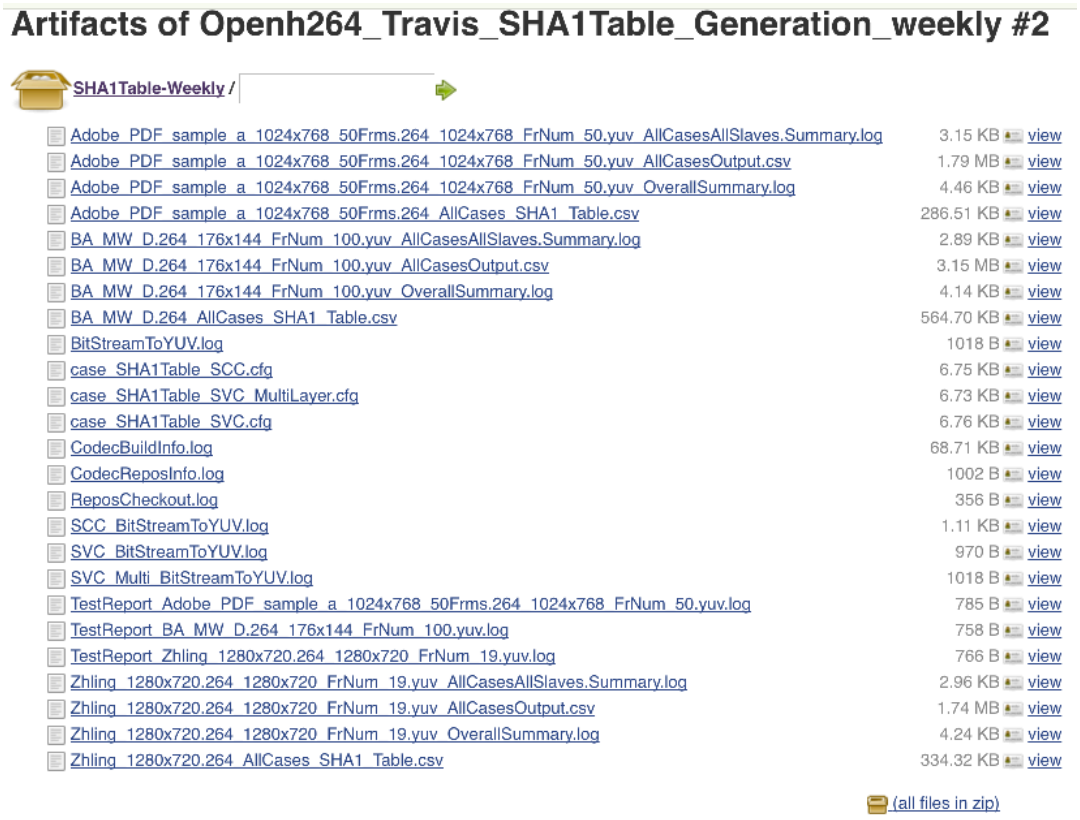
1. Jenkins job will generate SHA1 table for given bit stream setting in `case_SHA1Table_SCC.cfg` file

2. for more detail, please refer to Jenkins job
   http://10.140.198.27:8080/job/Openh264_Travis_SHA1Table_Generation_weekly/configure

3. You can update your branch's SHA1 tables
   when you change code which may lead to bit steam change



4. for SHA1 tables, you can download from job's artifact



6 SGE install and configuration

- *The following is an introduction of the installation SGE*
- *This chapter is copied from video team's SGE installation and configuration documents*

6.1 Install sge-qmaster

1. update these packages to make sure the consistency of system environment with other hosts.
   a) yum install *gcc*
   b) yum install *make*
   c) yum install *gnuplot*
   d) yum install *glibc*
   e) uname –a -------to check the kernel version is same as other hosts, otherwise
      yum install *kernel*
   f) Reboot system
2. download the package:
   sge62u2_1_linux24-i586_targz.zip from sun website

3. decompression this package in xp. Then download the two tar files to your centos machine.

4. Mkdir /opt/sge62u2_1
   then use
   tar zxvf sge-6_2u2_1-bin-linux24-i586.tar.gz
   tar xzvf sge-6_2u2-common.tar.gz

5. Useradd –g root –u 501 sgeadmin

6. Vi /etc/hosts
   # Do not remove the following line, or various programs
   # that require network functionality will fail.
   #127.0.0.1 localhost.localdomain localhost
   127.0.0.1 localhost0.localdomain localhost0
   10.224.204.26 sge-qmaster.webex.com sge-qmaster

7. vi /etc/sysconfig/network

   NETWORKING=yes
   NETWORKING_IPV6=no
   HOSTNAME=sge-qmaster

8. Remove the krb5-workstation package of rsh

rpm -qf `which rlogin` this command is to check rsh version.
rpm -e krb5-workstation remove the krb5-workstation package.

9. Install a latest version of rsh:
   a) yum install rsh
   b) 运行 setup，进入系统服务，将 rsh/rlogin/rexec 选中，退出
   c) 查看/etc/xinetd.d 目录下有没有 rsh、rexec、rlogin 文件，若没有，请添加，内容请参考第一大步骤 3 小步骤
   d) 若/etc/xinetd.d 目录下有 rsh、rexec、rlogin 文件，vi /etc/xinetd.d/rs,将 disable = yes 改为 disable = no（受控端设置）
   e) 若服务器设置了防火墙，请打开 512\513\514 端口（受控端设置）
   f) 修改配置文件（受控端设置）
   g) #vi /etc/securetty 再里面添加 rsh rexec rlogin 。或者
      #echo "rexec" >>/etc/securetty
      #echo "rlogin" >>/etc/securetty
      #echo "rsh" >>/etc/securetty
10. Install a latest version of rsh:
    vi /root/.bash_profile:
    PATH=$PATH:$HOME/bin:/opt/sge62u2_1:/opt/sge62u2_1/bin/lx24-x86:/opt/SDK/bin:/opt/SDK
    SGE_ROOT=/opt/sge62u2_1;export SGE_ROOT
    SGE_CELL=SVC_SGE1;export SGE_CELL
    SGE_QMASTER_PORT="10536";export SGE_QMASTER_PORT
    SGE_EXECD_PORT="10537";export SGE_EXECD_PORT
    JAVA_HOME=/opt/SDK;export JAVA_HOME
    PATH=$PATH:$SGE_ROOT/bin
    export PATH
    unset USERNAME
    note: you need to install Java and change the JAVA_HOME to /opt/SDK
11. Restart .bash_profile:

    . .bash_profile

12. service network restart

13. service xinetd restart

14. cd /opt/sgeu62u2_1/

15. ./install_qmaster
    (follow the instruction and use default setting)

6.2 Install sge-host

1. update these packages to make sure the consistency of system environment with other hosts.
   a) yum install *gcc*
   b) yum install *make*
   c) yum install *gnuplot*
   d) yum install *glibc*
   e) uname –a -------to check the kernel version is same as other hosts, otherwise yum install *kernel*
   f) Reboot system

2. Change the computer name or host name like Carl, Jeny….

   g) Vi /etc/sysconfig/network HOSTNAME=Yula

3. Remove the krb5-workstation package of rsh
   h) rpm -qf `which rlogin` this command is to check rsh version.
   i) rpm -e krb5-workstation remove the krb5-workstation package.

4. Install a latest version of rsh:
   j) yum install rsh (sometime you need to yum install xinetd firstly)
   k) 命令行输入 setup,进入系统服务，将 rsh/rlogin/rexec 选中，save,退出；
   l) 查看/etc/xinetd.d 目录下有没有 rsh、rexec、rlogin 文件，如果有，表明 rsh 安装成功；
   m) 若/etc/xinetd.d 目录下有 rsh、rexec、rlogin 文件，vi /etc/xinetd.d/rsh,将 disable = yes 改为 disable = no（受控端设置）；
   n) 若服务器设置了防火墙，请打开 512\513\514 端口（受控端设置）；
   o) 命令行输入 setup,进入防火墙配置，关闭防火墙；
   p) 修改配置文件/etc/securetty，在文件最后添加 rsh, rexec ,rlogin
      Echo “rexec">>/etc/securetty
      Echo “rlogin">>/etc/securetty
      Echo "rsh" >>/etc/securetty

5. Add ports permission in /etc/services
   q) Echo “###SGE service”>> /etc/services
   r) Echo “sge_qmaster 10536/tcp”>> /etc/services
   s) Echo “sge_execd 10537/tcp”>>/etc/services

6. SGE install package preparation
   t) Download the package: sge62u2_1_linux24-i586_targz.zip from sun website
   u) Decompression this package in xp. Then download the two tar files to your centos machine.
   v) Mkdir /opt/sge62u2_1 the use tar zxvf sge-6_2u2_1-bin-linux24-i586.tar.gz tar xzvf sge-6_2u2-common.tar.gz
   w) Useradd –g root –u 501 sgeadmin

x) Mkdir SVC_SGE1 (sell name) all machines should have this directory.

y) Chmod 777 SVC_SGE1.

z) Mkdir SGE_room2 (sell name) all machines should have this directory.

aa) Chmod 777 SGE_room2.

bb) Mount the directory.

Mount 10.224.204.26:/opt/sge62u2_1/SVC_SGE1/ /opt/sge62u2_1/SVC_SGE1/

Mount 10.224.204.26:/opt/sge62u2_1/SGE_room2/ /opt/sge62u2_1/SGE_room2/

Note: you need to start you NFS service.

--sge-master side: /etc/init.d/rpcbind restart

--sge-host side: /etc/init.d/nfs restart

If above command does not work, try to change the NFS setting in **sge-master** side as below:

Vi /etc/exports and add host info

```
#for ZhangFei
/opt/sge62u2_1/SVC_SGE1/   10.224.200.95(rw,no_root_squash)
/opt/sge62u2_1/SGE_room2/   10.224.200.95(rw,no_root_squash)

#for ZhaoYun
/opt/sge62u2_1/SVC_SGE1/   10.224.200.124(rw,no_root_squash)
/opt/sge62u2_1/SGE_room2/   10.224.200.124(rw,no_root_squash)
```

7. Modify /root/.bash_profile:

cc) Add these lines at the end of /root/.bash_profile

```
PATH=$PATH:$HOME/bin:/opt/sge62u2_1:/opt/sge62u2_1/bin/lx24-x86:/opt/SDK/bin:/opt/SDK
SGE_ROOT=/opt/sge62u2_1;export SGE_ROOT
SGE_CELL=SVC_SGE1;export SGE_CELL
SGE_QMASTER_PORT="10536";export SGE_QMASTER_PORT
SGE_EXECD_PORT="10537";export SGE_EXECD_PORT
JAVA_HOME=/opt/SDK;export JAVA_HOME
PATH=$PATH:$SGE_ROOT/bin
export PATH
unset USERNAME
```

dd) Restart .bash_profile:

```
. .bash_profile
```

8. update /etc/host related files:

ee) 转到 **sge-qmaster(10.224.204.26),** 在/etc/hosts 文件中加入新机器的 ip 地址与 host name 列表：

```
127.0.0.1 localhost0.localdomain localhost0
10.224.204.26 sge-qmaster.webex.com sge-qmaster
10.224.174.14 Dena
10.224.174.18 High
10.224.174.19 Jeny
10.224.174.20 Yula
10.224.174.21 Gaea
10.224.174.22 Eros
10.224.174.23 Rhea
10.224.174.24 Hera
```

::1 localhost6.localdomain6 localhost6

ff) **在本机上，**同样在/etc/hosts 文件中加入新机器的 ip 地址与 host name 列表

127.0.0.1 localhost.localdomain localhost
10.224.204.26 sge-qmaster.webex.com sge-qmaster
10.224.174.14 Dena
10.224.174.18 High
10.224.174.19 Jeny
10.224.174.20 Yula
10.224.174.21 Gaea
10.224.174.22 Eros
10.224.174.23 Rhea
10.224.174.24 Hera
::1 localhost6.localdomain6 localhost6

gg) **在本机上**，在/etc/hosts.allow 文件中加入如下信息：

in.rshd:10.224.204.26 root pass789
in.rlogind:10.224.204.26 root pass789
portmap:10.224.204.26 root pass789
sshd:10.224.204.26 root pass789

9. 转到 **sge-qmaster(10.224.204.26)**， 使用 qconf –ah 命令，添加新的 SGE host PC:

   hh) qconf –ah Yula.

10. service network restart
    service xinetd restart
    note: you need to remount the folder again:
    Mount 10.224.204.26:/opt/sge62u2_1/SVC_SGE1/ /opt/sge62u2_1/SVC_SGE1/
    Mount 10.224.204.26:/opt/sge62u2_1/SGE_room2/ /opt/sge62u2_1/SGE_room2/
    如果 mount 有问题，请参考 步骤 6
    如果 xinetd 没有安装，yum install xinetd

11. cd /opt/sgeu62u2_1/
12. ./ install_execd

13. Make sure sgeadmin in all machine should have same ID, same group. Check environment variable of sgeadmin. Should be same with root. (this is very important. And this block us too much time to solve it.)

6.3 Sge 断电重启方法

Sge-master:10.224.204.26:

1. cd /opt/sge62u2_1/SVC_SGE1/common
2. ./sgemaster start

Sge-host: 10.224.174.1x

1.mount 10.224.204.26:/opt/sge62u2_1/SVC_SGE1/ /opt/sge62u2_1/SVC_SGE1/
2.mount 10.224.204.26:/opt/sge62u2_1/ SGE_room2/ /opt/sge62u2_1/ SGE_room2/
3.cd /opt/sge62u2_1/SVC_SGE1/common
4. ./sgeexecd start
5.检查 sge 是否正在运行使用命令: ps -ef | grep sge

自动化重启脚本

当前在杭州的机器，/root/目录下已经有写好的自动重启 SGE 的脚本。
--run_SGERestart_For_Host_side.sh（放在各 host 的/root/目录下）
--run_SGERestart_Master_side.sh(放在 master 的/root 目录下)
1. 在 SGE-master,可以按照提示，可以在 master 端重启 master 和 host.
（各 host 的 IP 在脚本 run_SGERestart_Master_side.sh 里，可以根据实际 IP 来增减；
同时要确保脚本 run_SGERestart_For_Host_side.sh 放在各 host 的/root/目录下）
2. 在 host 的机器，也可以单独运行脚本，单个重启 host.
3. 如果在 mount 上遇到问题，请参考 host 装的第 6 步的解决方法。

6.4 Sge 常用命令

1.submit host

- Qconf –as hostname add new host to submit list
- Qconf –ds hostname delete host from the submit list
- As defaut, we used sge-master as submot host,
  If you submit a job by qsub Xxjob, there is error info 'unable to run job: host sge-master is no submit host' or something like that:
  Just use qconf –as sge-master to add sge-master to the submit host list. You can also add another host to the submit host list, and thus this host can also use command qsub XXXjob to submit job

2.queue management

- Qconf –sql list all of the queue
- Qconf –sq queuename list the detail info of queue
- Qconf –aq queuename add new queue
- Qconf –dq queuename delete queue
- Qconf –mq quenename modify the queue parameters,e.g.priority etc.
- Example of qconf –mq Openh264 for adding host in this test queue

```
qname           Openh264
hostlist        ZhangFei GuanYu ZhaoYun MaChao HuangZhong MaDai JiangWei
seq_no          0
load_thresholds     np_load_avg=1.75
suspend_thresholds  NONE
```

3.jop managment

- Qstat –f list all avaliable host and host status
- Qstat list all job (running/queue/waiting)
- Qdel JobId delete job
- Qsub job job should be as a excutable bianry file or script file
- Qsub –q qname job run job under destianted queue
- For more info about submit a job, you can refer to helo info by qsub –help

4. Other command :

- Qmod –s
- ...

6.5 Test-bed 一致性测试说明

一致性测试 testbed 路径:

10.224.204.26: /opt/sge62u2_1/SGE_room2/Wels_Codec_Test

测试序列:

为了避免在 SGE 运行中对于 sge-master 硬盘中视频序列数据的海量读取，在每一台 sge-host 的/opt 路径下都存放了一份测试序列的拷贝，如 Hera 这台 sge-host 的序列路径为/opt/Hera/Test_sequence，所以如果要增加测试序列，需要在 sge-master 的测试序列路径/opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Test_sequence 以及每一台 sge-host 的测试序列路径下都存放一份拷贝。

1. Wels_SVC_decoder_test: decoder 一致性测试(3D4Q)

操作说明:

   a) 将需要测试的 decoder 重新命名为 h264dec.exe 并拷贝到该路径下；
   b) 运行命令 sh run.sh

2. Wels_SVC_encoder_compare: 在同样的编码条件下，在固定 QP 编码条件下，检查两个不同的 encoder 编码结果是否一致(3D2Q)

操作说明:

   a) 将需要测试的两个 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_compare/Wels_SVC_encoder_test 下;
   b) 重新命名两个 encoder 文件，一个命名为 h264enc_opt_b.exe，另外一个命名为 h264enc_per_t.exe;
   c) 运行命令 sh run_no_mgs_slice_compare.sh

3. Wels_SVC_encoder_compare_rc_on: 在同样的编码条件下，Rate control 打开的情况下，检查两个不同的 encoder 编码结果是否一致(3D2Q)
   a) 将需要测试的两个 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_compare_rc_on/Wels_SVC_encoder_test 下；
   b) 重新命名两个 encoder 文件，一个命名为 h264enc_opt_b.exe，另外一个命名为 h264enc_per_t.exe;
   c) 运行命令 sh run_no_mgs_slice_compare.sh

4. Wels_SVC_encoder_HD_test: HD encoder 一致性测试(1D1Q)
   a) 将需要测试的 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_HD_test/Wels_SVC_en_AVC_case 下
   b) 重新命名 encoder 文件，命名为 h264enc.exe
   c) 运行命令 sh Wels_SVC_en_AVC_case.sh(增加 svc 基本测试,已添加)

5. Wels_SVC_encoder_MT_test: 在 Multi-thread 的编码条件下，测试 encoder 的一致性(3D2Q)
   a) 将需要测试的两个 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_MT_test 下，其中一个 encoder 的 multi-thread 编码选项关闭，命名为 h264enc_noMT.exe，另外一个 encoder 的 multi-thread 编码选项打开，命名为 h264enc_MT.exe
   b) 运行命令 sh run_no_mgs_slice_MT.sh

6. Wels_SVC_encoder_test: encoder 一致性测试，其中针对 mgs 的测试包含 1 个 base quality layer 和 3 个 enhancement quanlity layers(fix qp)(3D4Q)
   a) 将需要测试的 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_test 下
   b) 重新命名 encoder 文件，命名为 h264enc.exe
   c) 运行命令 sh run_no_mgs_slice.sh
7. Wels_SVC_encoder_test_cvs_algo: 针对 cvs algo_tune_b branch 的 encoder 一致性测试，其中针对 mgs 的测试包含 1 个 base quality layer 和 1 个 enhancement quanlity layers(3D2Q)
   a) 将需要测试的 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_test_cvs_algo 下
   b) 重新命名 encoder 文件，命名为 h264enc.exe
   c) 运行命令 sh run_no_mgs_slice.sh
8. Wels_SVC_encoder_test_svn_opt_tune: 针对 SVN opt_tune_b branch 的 encoder 一致性测试，其中针对 mgs 的测试包含 1 个 base quality layer 和 1 个 enhancement quanlity layers(3D2Q)
   a) 将需要测试的 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_test_svn_opt_tune 下
   b) 重新命名 encoder 文件，命名为 h264enc.exe
   c) 运行命令 sh run_no_mgs_slice.sh
   d) 测试结果在相应的 result_check 路径下查找。
9. Wels_SVC_encoder_test_rc_on: 在 rate control 模块打开的情况下，针对 encoder 一致性测试(3D2Q)
   a) 将需要测试的 encoder 拷贝到路径 /opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_encoder_test_rc_on 下
   b) 重新命名 encoder 文件，命名为 h264enc.exe
   c) 运行命令 sh run_rc_on.sh

6.6. Test-bed performance 测试说明：

工作路径为：
/opt/sge62u2_1/SGE_room2/Wels_Codec_Test/Wels_SVC_opt_encoder_performance_test/testbed
运行命令为:
sh testbed.sh *.para
qp_multi_spatial_vs_singlelayer_vs_simlcast.para: RD performance comparison among SVC multi-spatial layer coding and single layer coding for the highest resolution picture and simulcast multi-spatial layer coding.
Wels_vs_VP8_rc_30HZ_360p.para: RD performance comparison among Wels encoder and x264 encoder and VP8 encoder.
HD_720p_test.para RD: performance analyzing for HD encoder at the condition of fixed QP coding.
HD_720p_test_rc.para: RD performance analyzing for HD encoder at the condition of rate control module turned on
qp_diff_qp_cascading_gop148_rc_24HZ_360p.para: temporal scalable coding performance testing

ffmpeg

de-mux 使用命令：
ffmpeg -i h264.mp4 -vcodec copy -vbsf h264_mp4toannexb -an out.h264
.→yuv 使用的命令：
`ffmpeg -i input.avi output.yuv`
re-sample 使用命令：
ffmpeg -s 1920x1080 -i Absolute_Power.yuv -s 1280x720 Absolute_Power_720p.yuv”
Should be noted out that for raw YUV format input, the file name must be ended by “.yuv” otherwise the ffmpeg down-scale tool can not recognize the input format.

update version
ffmpeg 更新或者下载某个版本的 code 使用命令：
svn update -r 3729 ffmpeg
或者 svn up svn://svn.ffmpeg.org/ffmpeg/trunk ffmpeg -r 3729
其中-r 3729 是规定了需要更新的版本号

Ffmpeg 编译方法：进入到 ffmpeg 文件路径下：
1. 使用命令./configure
2. 使用命令 make install