BQS script requirement

**Jason Wang**

**Abstract:**

BQS script is used for all diamond flow and simulation flow support.

REVISION HISTORY

|  |  |  |
| --- | --- | --- |
| REVISION | RELEASE DATE | COMMENTS |
| V1.0 | 1/21/2014 | Initial draft – Jason Wang |
| V1.1 | 1/26/2014 | Rename implement flow as: TCL, CMD flow |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 1 Objective

BQS script will be creating to support all cases in LSH with the same case format. This script will cover normal diamond flow, individual diamond flow, simulation work flow and finally the check & report work flow.

# 2 Key Structure

BQS script should cover the following structure:

1. Design entry
2. Implement flow
3. Simulation flow
4. Check flow
5. Report flow

## 2.1 Design entry

Design entry used to collect all design and flow information.

Every design case will use the following format:

./<case name>

./models

./others

./par

./sim

./source

Bqs.info(or omit) --project information

Bqs.config(or omit) --project configure

BQS script will scan this folder, If find bqs.info file use this file to get the design entry. Normally there are two conditions in this file: with/without ldf file address.

When script get ldf file address, use this ldf file as design entry

When script not get ldf file, it means this is a standalone flow, script should try to collect all input files to run the standalone flow specified by option (e.g. map/par)

While BQS script cannot find the bqs.info file, script should try to find the ldf file in ./par folder, and use this file as design entry.

## 2.2 Implement flow

Implement flow used to run the whole design project. Here implement flow should support:

### 2.2.1 Normal design flow

A normal diamond flow means a diamond project run in TCL flow (pnmainc.exe), normal diamond flow should support all milestones we can find in Diamond process view.

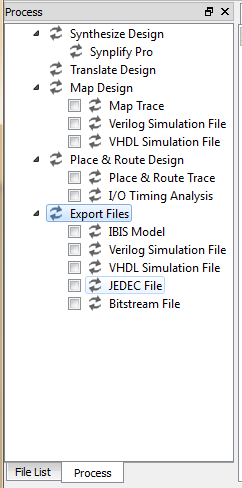


Figure1. All milestones in process view

It means we can run every process like Diamond GUI flow through options, such as: map, map trce, par, i/o timing analysis, ibis…

Some extra requirements:

1. Sweeping --run design with different target fmax
2. Multi-seed --run design with different seed
3. No-scuba -- not update module
4. Device-kit --run design with different device
5. Strategy update support --strategy overwrite

### 2.2.2 Individual design flow

Individual design flow is required for stand-alone flow support, such as when a user just have map ncd and lpf files, BQS script can run par flow with these files.

## 2.3 Simulation flow

BQS script should support simulation for all of the following styles with both of ModelSim & Active-HDL.

1. RTL simulation flow
2. POST map simulation flow
3. POST par simulation flow

For every of these three flows, BQS script should support:

1. Source file simulation flow
2. Macro(.do) file simulation flow

Here source file simulation flow means run simulation with RTL/map back annotation/par back annotation and test bench files. BQS script will generate macro file first, then run simulation flow with this generate macro file.

Simulation with Macro file means a user provide the Macro files already, BQS script should use this file to run simulation with little change(such as file address).

Priority: Macro file > source file, If find Macro file, ignore source file.

## 2.4 Check flow

Check flow will run according with the bqs.config file.

### 2.4.1 With bqs.config file

If case has conf file, scripts will do behavior check with the following methods:

#### 2.4.1.1 check\_lines

1) format

[check\_lines]

title = check\_test1

file = <path>\<file>

check\_1 = <string\_1>

times = <number> --optional

check\_<num> = <string\_2>

2) description

This method will try to find the <string\_1> assigned by check\_1 in <file> and treat it as a start point (line 1),

(If times option is used, scripts will find <string\_1> <number> times and then treat the last one as start point),

Then try to check whether line <num> (a shift value) has <string\_2>. if yes, the result of this method is true.

#### 2.4.1.2 check\_data

1) Format

[check\_data]

file=<path>\<file>

start\_line = <string\_1>

times = <number> --optional

result = <num> / <line>,<shift>

line<num1> = <num>,<operation\_symbol> / <line>,<shift>,<operation\_symbol>

line<num2> = <num>,<operation\_symbol> / <line>,<shift>,<operation\_symbol>

...

line<numn> = <num> / <line>,<shift>

2) Description

this method will try to find the <string\_1> assigned by start\_line in <file> and treat it as a start point (line 1),

(if times option is used, scripts will find <string\_1> <number> times and then treat the last one as start point),

Then try to calculate the result.

User can either use the absolute number or to use the line + shift to indicate the number.

#### 2.4.1.3 check\_block

1) Format

[check\_block]

Title = check\_blocks

compare\_file = <path>\<file>

golden\_file = <Gold\_file>

2) Description

This method will try to compare the <Gold\_file> and <file>, if <Gold\_file> is included in <file>, return true.

#### 2.4.1.4 check\_flow

1) Format

[check\_flow]

file=<path>\<file>

2) Description

this method will try to find the string "All signals are completely routed." in the par report.

<file> need to be a par report.

#### 2.4.1.5 check\_multiline

1) Format

[check\_multiline]

file = <path>\<file>

check\_line = <total\_string>

2) Description

This method will try to find the <total\_string> assigned by check\_line in <file> regardless the "space" and "line feed".

This method will benefit the situation that the <total\_string> won't be at the same line and changing for each regression.

### 2.4.2 without bqs.config file

If case doesn’t have configured file, scripts will do nothing or run default check when user required in options.

## 2.5 Report flow

Report flow is used to generate final report for all test cases.

Scripts will generate a report (csv) based on configure file:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Area | Type | Case | Device\_syn | Result | Comments |
| STA | Features | 02\_ClosestClockEdge | LCMXO2-2000HC-5TG100C | TRUE |  |
| STA | Features | 01\_UGROUP\_anchored | LFE3-95EA-6FN484C | TRUE |  |

If there is no configure file, just list the test flow result (specified by run option).

# 3 Demo Case

## 3.1 Case Demo:

--models                    folder to place your model files.

--others                      folder to place ngo ngd and some other files which will be include in your diamond project

--par                            folder to place project files. Before you check in your case, please clean up this file

--sim                            folder to place files for simulation

--source                      folder to place source files here

--xx.info                   total info record here for script use.

--xx.config total configure information for check and report flow

## 3.2 info file demo

Info file will be dividing into two sections: project section, simulation section.

Project section is for design information such as: ldf file, ngd file, ncd file, lpf file… BQS script scans these files to get the right design entry.

Simulation section is for designs require simulation flow, user should list simulation files: tb file, input file, source file, simulation info…

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* bqs.info \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Project section used to record project information with the following options:

Ldf\_file= used to address the ldf file

Map\_ncd= used to address the map ncd file

Lpf\_file= used to address lpf file

….

#Style1:

tb\_file= used to address the test bench file

tb\_vector= used to address the simulation input vector

tb\_name= used to address test bench top model name

sim\_lib= used to address which device lib should be used

pri\_lib= used to address which lib should be used first

sim\_time= how long is the simulation will be run

do\_file= simulation macro file

#Style2:

do\_file= simulation macro file

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NOTE:

priority: style2 > style1

## 3.2 config file demo

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* bqs.conf \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[configuration information]

area = STA

type = Arc

[method]

check\_lines\_1 = 1

check\_lines\_2 = 1

check\_block = 1

check\_data = 1

check\_multiline = 1

check\_flow = 1

[check\_lines\_1]

title = check\_test1

file = <path>\NoPortName\_one.twr

check\_1 = Preference: FREQUENCY PORT "clka" 10.000000 MHz ;

check\_2 = 2 items scored

[check\_lines\_2]

title = check\_test2

file = <path>\NoPortName\_one.twr

check\_1 = Preference: FREQUENCY 20.000000 MHz ;

check\_2 = 4 items scored

[check\_block]

title = check\_blocks

compare\_file = <path>\NoPortName\_one.jed

golden\_file = Gold\_NoPortName\_one.jed

[check\_data]

file=<path>\NoPortName\_one.twr

start\_line = Preference: INPUT\_SETUP GROUP "Data" 5.000000 ns CLKPORT "Clock" ;

result=6,8

line1=20,1,-

line2=19,1,+

line3=21,1,-

line4=22,1,-

line5=23,1

[check\_multiline]

file = <path>/GSR\_NET\_impl.mrp

check\_line = Syntax error in "GSR\_NET NETAAAAA "Reset1\_c";": error on token "NETAAAAA"

[check\_flow]

file=<path>\NoPortName\_one.par

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NOTE:

(1) for auto check, one (or more than one) method failure will lead to the case's failure in report.

(2) if the item value is set to 1, then script will enable this method

(3) if one method is used more than once, it can be named as check\_lines\_1, check\_lines\_2, ...

(4) <path> means relative path compared with conf file.

## 3. 3 macro file demo(.do demo)

1. You need collect all the variables(will be change version by version and will be include by script) in the top side.
2. List source file first           --source file will be replaced with map.vo/par.vo files in the post map and par simulation.
3. List simulation test bench file

For example:

-----------------------------------------------sim.do---------------------------------------------------------

Set diamond=XXXXX

set device\_lib=XXXXXX

set PMI\_lib=XXXXX

vlib work

#add some more lib here

Vmap device\_lib

Vmap pmi\_lib

##compile <source start>

-----all source files here will be replaced in the post map and post par simulation

Vlog/vcom rtl\_source.v/vhd

Vlog/vcom rtl\_source.v/vhd

##compile <source end>

#sim tb here <testbench start>

Vlog/vcom sim\_source.v/vhd

Vlog/vcom sim\_source.v/vhd

#start simulation

            vsim -novopt -lib work tb\_name –L pmi\_lib –l device\_lib

           run 10us

-----------------------------------------------END---------------------------------------------------------