

CI FOR FPGA D&V

Continuous Integration for FPGA Design and Verification

Verification Futures 2015-02-05

AGENDA



- › Ericsson TV background
- › Ericsson TV Firmware Group design flow
- › Issues
- › Motivation for continuous integration
- › Continuous Integration to the rescue!
- › Application to FPGA Build
- › Application to FPGA Simulation
- › Conclusions

ERICSSON TV



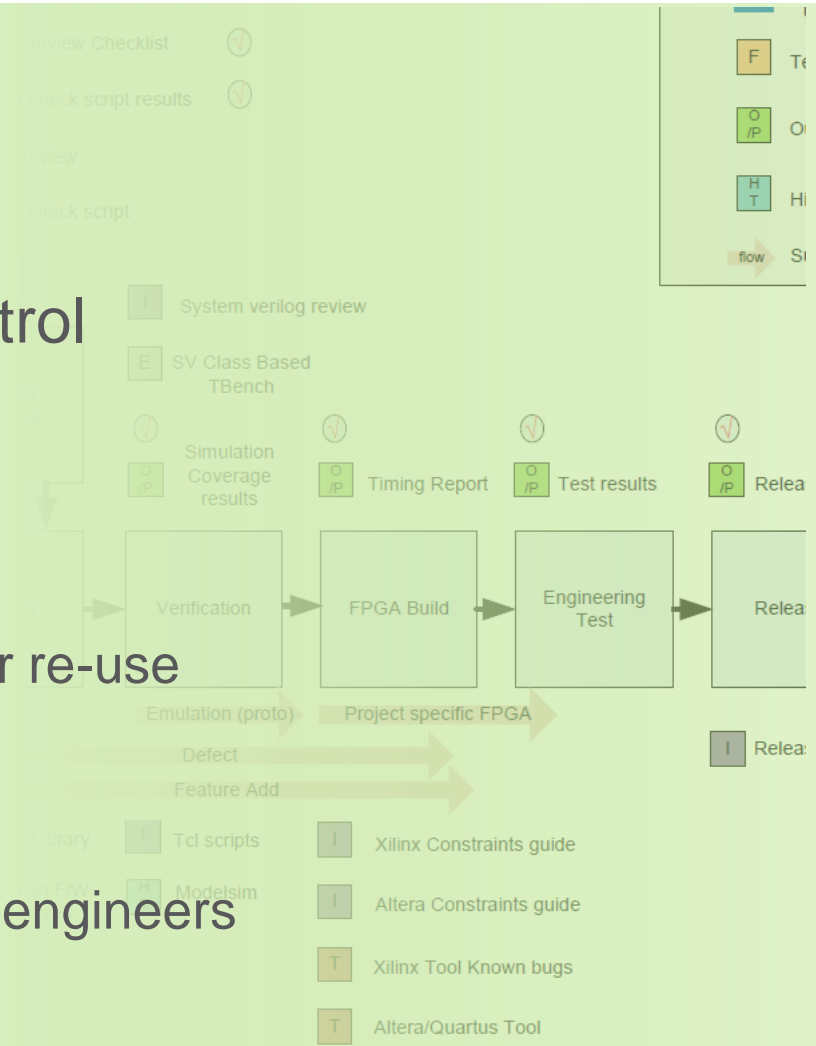
- › Market leader in professional video processing solutions for compression
- › Primary site Hedge End, Southampton UK
- › About 200 engineering staff
- › FPGA (“Firmware”)group 30 staff
- › Developing FPGAs
 - Largest project – 5 x 1M logic elements
 - Many existing designs in maintenance



DESIGN FLOW



- › VHDL Designs
- › Tcl compile/build scripts
- › VHDL testbenches
- › Subversion for revision control
- › Largest projects
 - UVM test environments
 - Standardized base classes for re-use
- › Mixture of roles
 - Some specialized verification engineers



ISSUES



› FPGA Build

- Make “push-button”
- Schedule builds over the weekend

› FPGA Simulation

- Use regression with UVM/SV (non-proprietary)
- Extend regression to VHDL
- Make sure existing sims still compile and run (“smoke test”)
 - › Avoid “software rot”
- Efficient use of licenses (e.g. overnight/weekends)

CI TO THE RESCUE!



- › We could have extended existing Tcl flow
 - But luckily local software groups had already implemented Jenkins
- › Jenkins was applied to FPGA Build and Simulation
 - Work carried during “innovation days”, time allocated to process improvement at beginning of Agile Sprints
- › Used for UVM regressions
- › Extended to VHDL testbenches

APPLICATION TO BUILD



- › Tcl flow already carries out automated build
 - But still required user checkout of large Subversion database
 - Knowing what directory to use and which script options to supply
- › The Jenkins CI server allows arbitrary commands to be run on a Jenkins slave node
- › The firmware slave node is a Grid Engine Master, which then queues the jobs on to a Linux server client

FPGA BUILD MADE EASY



Welcome to Jenkins - the Continuous Integration server for Engineering.

Scheduled builds

S	W	Name ↓	Last Success	Last Failure	Last Duration	Built On	Cron Trigger	Number of builds
		EN8100_VIPER_asi_io_card	5 mo 6 days - #15	N/A	58 min	typhon	Poll SCM: # 19:00 on Sunday 0 19 * * 7	1 0 0

Icon: [S](#) [M](#) [L](#)

[Legend](#) [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Press button

Someone else has done the hard work...

Execute shell

```
Command #!/bin/sh -xe
pwd
if [[ "$RELEASE_BUILD" == "true" ]]; then
  OPTION="-rel"
fi
cd asi_io_block/synthesis/scripts
JobRun ise_toplevel.tcl $OPTION
echo "Compile Done"
tclsh < ../scripts/xilinx_check_timing.tcl
```

See [the list of available environment variables](#)

Add build step

Post-build Actions

Archive the artifacts

Files to archive

CI WITH SIMULATION



› First applied with UVM

- Jenkins interprets results in XML – JUNIT report format

```
function void scoreboard::report_phase(uvm_phase phase);  
    // . . .  
    // instance the junit class  
    junit_results = new(junit_res_file_name);  
    junit_results.start_tests();  
  
    if (((svr.get_severity_count(UVM_FATAL) + svr.get_severity_count(UVM_ERROR)) == 0))  
    begin  
        junit_results.add_test("uvm_tb", ctrl_knobs.get_str_value("test_name"), `PASS, , , );  
    end  
  
    // . . .  
endfunction: report_phase
```

Scoreboard base class

XML writer class

```
<?xml version="1.0" encoding="UTF-8"?>  
<testsuite>  
    <testcase name="test1"  
        classname="uvm_tb"  
        time="35">  
    </testcase>  
</testsuite>
```

JENKINS SIMULATION



S	W	Name ↓	Last Success	Last Failure	Last Duration	Built On	Cron Trigger
		COMMON_CSM_Interfaces_regression	2 days 2 hr - #302	2 hr 36 min - #304	8 min 10 sec	vishnu	
		overnight_regression_COMMON	2 hr 37 min - #287	5 days 18 hr - #280	1 min 1 sec	vishnu	Build periodic <i>midnight ev</i>
		project_crawler ▼	28 min - #7769	1 hr 28 min - #7768	19 sec	vishnu	Build periodic
		SVN_checkout	2 hr 37 min - #1697	N/A	47 sec	typhon	

Python script builds project coverage web-page

Define build flow using flow DSL

```
1 build( "      SVN_checkout" )
2 ignore(FAILURE) {
3     parallel (
4         { build("      _COMMON_CSM_Interfaces_regression") }
5     )
6 }
```

DSL Plugin

Post-build Actions

E-mail Notification

Recipients

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, I

Send e-mail for every unstable build

ISSUES



- › Jenkins connects to slaves through ssh
 - Loss of ssh connection for any reason stops periodic builds
 - Investigating a “slave monitor” Jenkins plugin

- › It’s better to write XML status from wrapper scripts / log scanners (which we now do)
 - Applies to all languages (VHDL/SV) equally

CONCLUSIONS



- › Jenkins is a powerful way to do regression and build
 - Unexpected benefit – helps engineers keep on top of code changes during rapid development
- › Push-button or time-triggered builds/sims
- › Common CI across software/firmware
- › Plans
 - Extend into department to run “smoke tests” on VHDL sims
 - TV display of Coverage/Build status “build radiator”

<http://jenkins-ci.org>



ERICSSON