EXPERT SERIES
Customer Support WEBINAR

Compile, Optimise and Simulate With Just One Command

Simplifying Questa Usage and Deployment with Qrun

SIEMENS

Agenda

- What is Qrun and why do we need it?
- How to use Qrun? Simple Use Cases
- Qrun Option Sets
- How to use Qrun? Real world use cases
- Demo
- How to integrate Qrun into your environment :Qrun Flows

EXPERT SERIES
Customer Support WEBINAR

Table of Contents

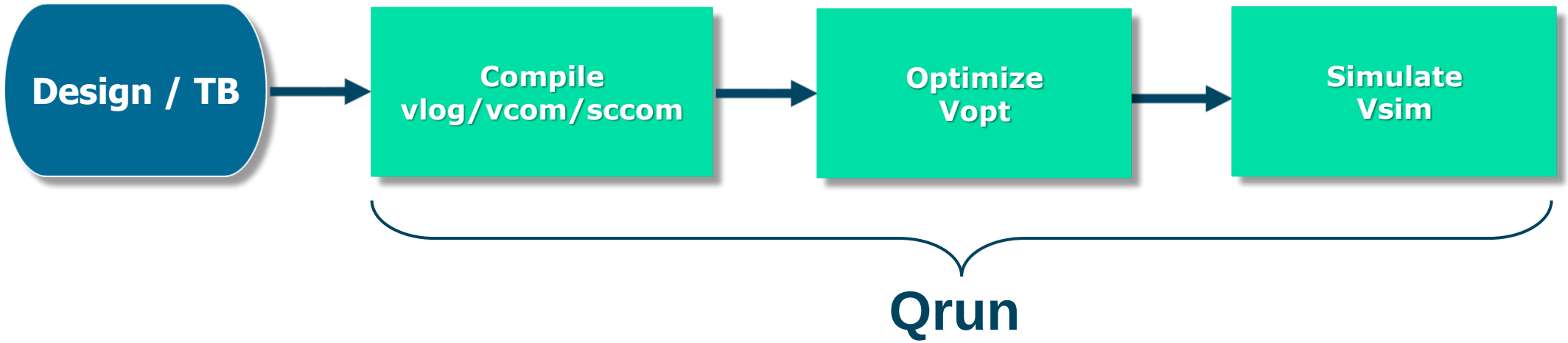
SIEMENS

EXPERT SERIES
Customer Support WEBINAR

WHAT IS QRUN

SIEMENS

Current Questa Run Flow



The Qrun utility simplifies the use of Questa tools to compile, optimize and simulate a design by taking all input files and options, and then automatically executing the correct tool and command-line arguments for each file.

Motivation

1. Source files must be separate by language for each compiler
   - vcom for VHDL
   - vlog for Verilog / SV
   - Sccom for SystemC

   ➡ Dispatch all source files to the appropriate compiler

2. VHDL source files must be ordered manually

   ➡ Determine compilation order for VHDL source files

3. The compile, optimize, and simulate “stages” must be ran Individually

   ➡ Dispatch all command line arguments to the appropriate tool

4. Questa supports various Methodologies
   - PowerAware, Coverage, UVM, etc

   ➡ Build in usage flows so the user doesn’t have learn them (at first)

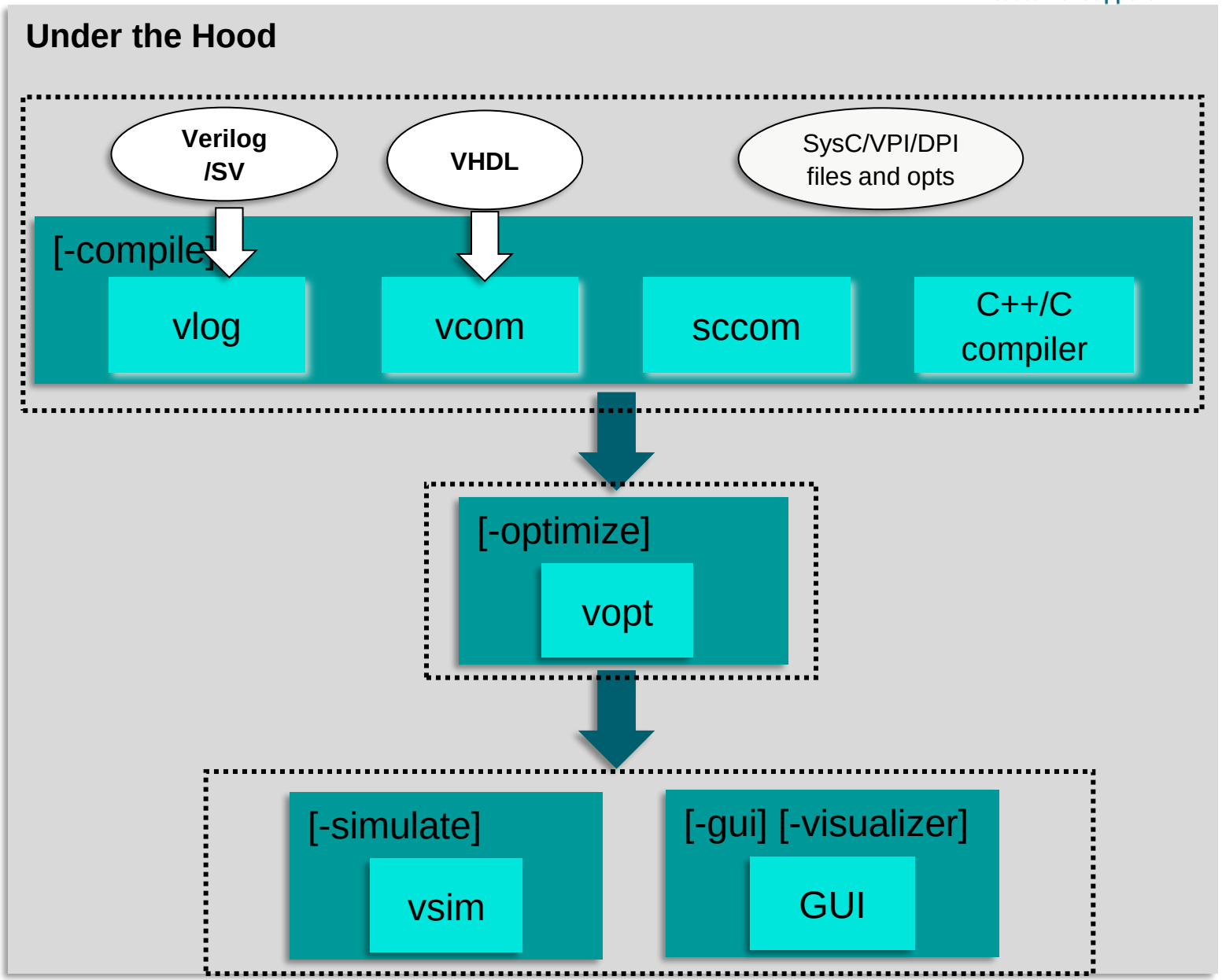
5. Modify vopt and vsim commands to bring up GUI for debug

   ➡ Make it easy to switch between batch run and GUI

Qrun Flow

Qrun

```
qrun file1.v file2.v
file3.vhdl -work work
-o qrun_opt -top top
-do "run -all" -gui
-visualizer
```



How does Qrun tool Simplifies the flow

EXPERT SERIES
Customer Support WEBINAR

- All in one line
  - Additional arguments to vlog/vcom, vopt, vsim are combined to a single command line all together.
- Manages Libraries
  - No need to call vlib/vmap directly
  - Libraries are created inside the qrun environment
- Easy access to other flows
  - Debug, Coverage, UVM, etc
- Abstracts Details
  - Abstracts the details of compiling and simulating

SIEMENS

Qrun Key Features

EXPERT SERIES
Customer Support WEBINAR

- **Single Step Flow -** for compilation ,optimization and simulation
- **Automatic Compiler Selection -** SV, VHDL, SystemC, DPI, VPI, C/C++
- **VHDL File Ordering**
- **UVM -** Handles all aspects of UVM compilation, include C/C++ compilation
- **GUI -** easy switch from batch mode to interactive mode
- **Incremental Compilation and Optimization -** skips if there is no change
- **Parallel Compilation -** Automatically determines compile order and parallelization groupings

SIEMENS

EXPERT SERIES
Customer Support WEBINAR

HOW TO USE QRUN

SIMPLE USE CASES

SIEMENS

Example 1 : Hello world - QuestaSim

test.sv

```
module test;
    initial
      $display ("hello world!");
endmodule
```

To test this in **batch mode**:

- `vlog test.sv -work work`
- `vopt -work work -o out_opt`
- `vsim -lib work -c  -do "run -all"`

Example 1 : Hello world - QuestaSim

test.sv

```
module test;
    initial
        $display ("hello world!");
endmodule
```

To test this in **GUI mode**:

- `vlog test.sv -work work`
- `vopt -work work -o out_opt -debug,livesim -designfile=design.bin`
- `vsim -lib work ~~-c~~ -do "run -all" –visualizer`

Example 1 : Hello world - Qrun

test.sv

```
module test;
    initial
        $display ("hello world!");
endmodule
```

To test this using Qrun:

```
qrun test.sv  -gui -visualizer
```

```
vlog test.sv -work qrun.out/work
vopt test -work qrun.out/work -o qrun_opt
vsim -lib qrun.out/work -c -do "run -all;
quit -f" qrun_opt
```

Example 1 : Hello world - Results

EXPERT SERIES
Customer Support WEBINAR

```
QuestaSim-64 qrun QA Baseline: 2023_weekly_231020 - 5695866 Utility 2023.10 Oct 21 2023
Start time: 05:45:58 on Nov 13,2023
qrun test.sv
Creating library 'qrun.out/work'.
QuestaSim-64 vlog QA Baseline: 2023_weekly_231020 - 5695866 Compiler 2023.10 Oct 21 2023
Start time: 05:45:59 on Nov 13,2023
vlog test.sv -work qrun.out/work -statslog qrun.out/stats_log -writesessionid "+qrun.out/top_dus" -csession=incr -csessionid=0
-- Compiling module test

Top level modules:
            test
End time: 05:45:59 on Nov 13,2023, Elapsed time: 0:00:00
Errors: 0, Warnings: 0
QuestaSim-64 vopt QA Baseline: 2023_weekly_231020 - 5695866 Compiler 2023.10 Oct 21 2023
Start time: 05:46:00 on Nov 13,2023
vopt -findtoplevels ../test1/hello_world/qrun.out/work+0+ -work qrun.out/work -statslog qrun.out/stats_log -o qrun_opt -csession=incr
-csessionid=1

Top level modules:
            test

Analyzing design...
-- Loading module test
Optimizing 1 design-unit (inlining 0/1 module instances):
-- Optimizing module test(fast)
Optimized design name is qrun_opt
End time: 05:46:01 on Nov 13,2023, Elapsed time: 0:00:01
Errors: 0, Warnings: 0
```


SIEMENS

Example 1 : Hello world - Results

EXPERT SERIES
Customer Support WEBINAR

```
# vsim -lib qrun.out/work -c -do "run -all; quit -f" -statslog qrun.out/stats_log qrun_opt -appendlog -l qrun.log -csession=incr -csessionid=2
# Start time: 05:46:03 on Nov 13,2023
# //  Questa Sim-64
# //  Version QA Baseline: 2023_weekly_231020 - 5695866 linux_x86_64 Oct 21 2023
# //
# //  Copyright 1991-2023 Mentor Graphics Corporation
# //  All Rights Reserved.
# //
# //  QuestaSim and its associated documentation contain trade
# //  secrets and commercial or financial information that are the property of
# //  Mentor Graphics Corporation and are privileged, confidential,
# //  and exempt from disclosure under the Freedom of Information Act,
# //  5 U.S.C. Section 552. Furthermore, this information
# //  is prohibited from disclosure under the Trade Secrets Act,
# //  18 U.S.C. Section 1905.
# //
# Loading sv_std.std
# Loading work.test(fast)
# run -all
# hello world!
#  quit -f
# End time: 05:46:04 on Nov 13,2023, Elapsed time: 0:00:01
# Errors: 0, Warnings: 0
# *** Summary ********************************************
#     qrun: Errors:   0, Warnings:   0
#     vlog: Errors:   0, Warnings:   0
#     vopt: Errors:   0, Warnings:   0
#     vsim: Errors:   0, Warnings:   0
#   Totals: Errors:   0, Warnings:   0
```


SIEMENS

Example 2 : Randomization - Visualizer

test.sv

```
module top;

   bit clk;
   reg [3:0] a, b, c;

   always
        #10 clk = ~clk;

   initial
     begin
        $monitor("a:%02d, b:%02d, c:%02d", a, b, c);
        `ifdef A
        $display("`A == %0d", `A);
        `endif

        repeat(10)
          begin
             assert(
                    std::randomize(a, b, c) with { a  < 8;
                                                   b  > 6;
                                                   c == 5;
                                                   }

                    );
             @(posedge clk);
          end
        $finish();
     end

endmodule // top
```

Run CMD:

```
qrun test.sv
```

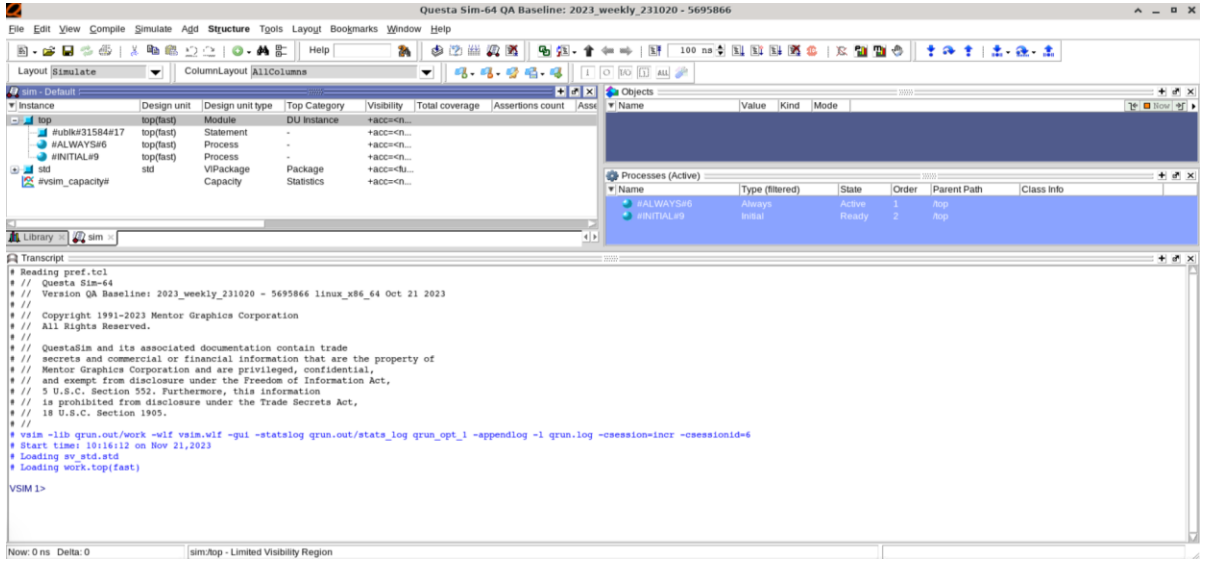
Example 2 : Randomization - Visualizer

test.sv

```
module top;

   bit clk;
   reg [3:0] a, b, c;

   always
        #10 clk = ~clk;

   initial
     begin
        $monitor("a:%02d, b:%02d, c:%02d", a, b, c);
        `ifdef A
        $display("`A == %0d", `A);
        `endif

        repeat(10)
          begin
             assert(
                    std::randomize(a, b, c) with { a  < 8;
                                                   b  > 6;
                                                   c == 5;
                                                   }

                    );
             @(posedge clk);
          end
        $finish();
     end

endmodule // top
```

Run CMD:

```
qrun test.sv -gui
```



SIEMENS

Example 2 : Randomization - Visualizer

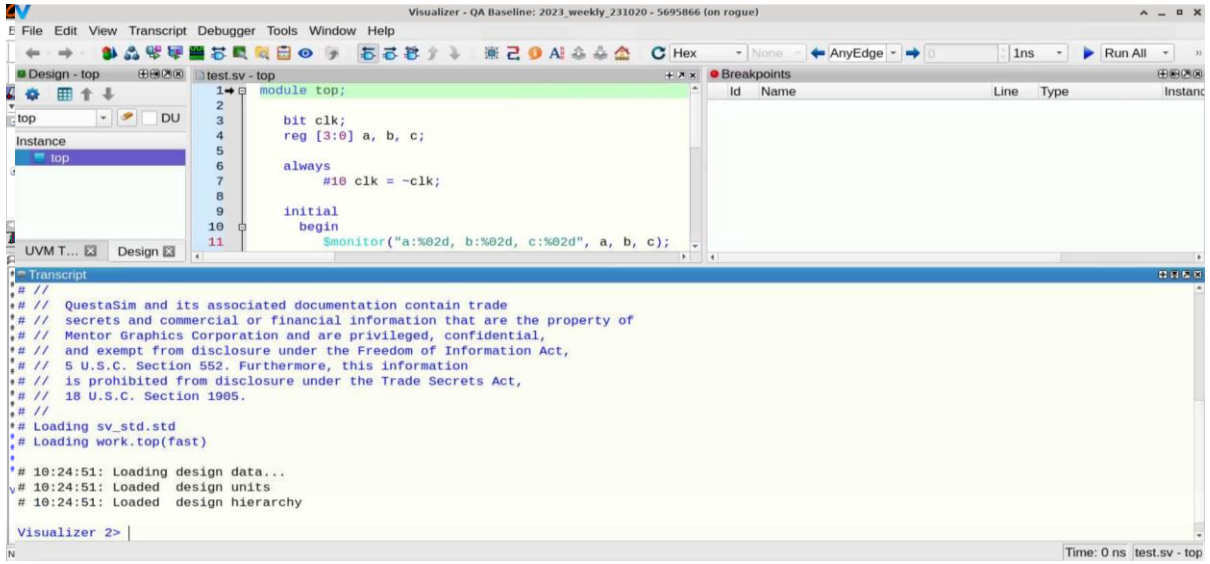
test.sv

```
module top;

   bit clk;
   reg [3:0] a, b, c;

   always
        #10 clk = ~clk;

   initial
     begin
        $monitor("a:%02d, b:%02d, c:%02d", a, b, c);
        `ifdef A
        $display("`A == %0d", `A);
        `endif

        repeat(10)
          begin
             assert(
                    std::randomize(a, b, c) with { a  < 8;
                                                   b  > 6;
                                                   c == 5;
                                                   }

                    );
             @(posedge clk);
          end
        $finish();
     end

endmodule // top
```

Run CMD:

```
qrun test.sv -gui -visualizer
```

```
vlog test.sv -work qrun.out/work
vopt top -work qrun.out/work -o qrun_opt -debug -
designfile design.bin
vsim -lib qrun.out/work -qwavedb=+signal+class
qrun_opt -visualizer
```



SIEMENS

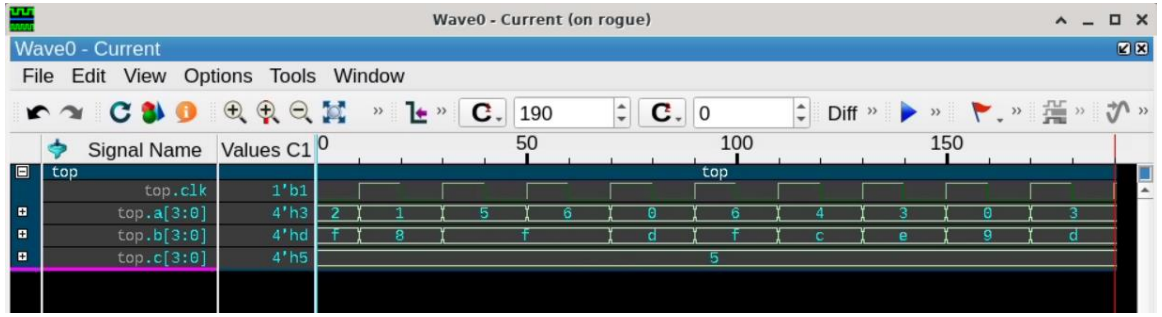
Example 2 : Randomization - Visualizer

test.sv

```
module top;

   bit clk;
   reg [3:0] a, b, c;

   always
        #10 clk = ~clk;

   initial
     begin
        $monitor("a:%02d, b:%02d, c:%02d", a, b, c);
        `ifdef A
        $display("`A == %0d", `A);
        `endif

        repeat(10)
          begin
             assert(
                    std::randomize(a, b, c) with { a  < 8;
                                                   b  > 6;
                                                   c == 5;
                                                   }

                    );
             @(posedge clk);
          end
        $finish();
     end

endmodule // top
```

transcript

```
# qwavedb_dumpvars: Done initiating qwavedb waveform capturing
#
# a: 2, b:15, c: 5
# a: 1, b: 8, c: 5
# a: 5, b:15, c: 5
# a: 6, b:15, c: 5
# a: 0, b:13, c: 5
# a: 6, b:15, c: 5
# a: 4, b:12, c: 5
# a: 3, b:14, c: 5
# a: 0, b: 9, c: 5
# a: 3, b:13, c: 5
# ** Note: $finish    : test.sv(26)
#    Time: 190 ns  Iteration: 1  Instance: .top
# Stopped at test.sv line 26
# 1
# Break in Module top at test.sv line 26
# 10:26:00: Loading waveform header...
# 10:26:00: Loaded  waveform header
```



SIEMENS

EXPERT SERIES
Customer Support WEBINAR

QRUN OPTION SETS

SIEMENS

Library Management

EXPERT SERIES
Customer Support WEBINAR

Qrun creates and maps libraries automatically.

- How to separate files and options into different libraries?

```
-makelib <source files> <options> -end
```

  - Chooses a default name for the library
  - Creates library under ./qrun.out

- How can we control the name and location of libraries?

```
-makelib <libpath[:logical name]> <source files> <options> -end
```

  - libpath: Physical location of the library
  - logical name: Maps the logical name to the physical path (vmap)

- Any number of -makelib's are allowed on the qrun command line
- Only the source files in the options set will be compiled into the library

SIEMENS

Library Management

EXPERT SERIES
Customer Support WEBINAR

- Until now:

```
vlib /u/myFiles/project/lib1
vlib /u/myFiles/project/lib2

vmap myLib1 /u/myFiles/project/lib1
vmap myLib2 /u/myFiles/project/lib2

vlog file1.sv files2.sv -warning 1902 -work myLib1
vlog file3.sv -permissive -work myLib2
```

- On Qrun:

```
-makelib /u/myFiles/project/lib1 myDir1 file1.sv files2.sv -warning 1902 -end
-makelib /u/myFiles/project/lib2 myDir2 file3.sv -permissive -end
```


SIEMENS

Library Management - Grouping

EXPERT SERIES
Customer Support WEBINAR

- Groups options together and applies them to specific files, libraries, or tools

```
-<group type> <files> or <options> -end
```

  - Unspecified number of arguments
  - The files and options are only valid within the option set
  - You can have multiple option sets on the Qrun command line

SIEMENS

Library Management - Grouping

EXPERT SERIES
Customer Support WEBINAR

- Groups options together and applies them to specific files, libraries, or tools

```
-<group type> <files> or <options> -end
```

  - Unspecified number of arguments
  - The files and options are only valid within the option set
  - You can have multiple option sets on the Qrun command line

- Option set types
  - `-makelib    <library> <files> <options> -end|-endlib`
  - `-precompile <library> <files> <options> -end`
  - `-filemap    <options> <files>           -end|-endfile`
  - `-<tool>.options -opt1 -opt2 +op3        -end`

  Ex: `-vopt.options +nospecify –l vopt.log -end`

SIEMENS

Library Management

EXPERT SERIES
Customer Support WEBINAR

Useful Options:

- Pre-compiled libraries are reference with either the '-L' or '-reflib' switches

```
qrun    -L   /path/to/vendor/libs/my_gate_lib … <other arguments>
qrun -reflib /path/to/vendor/libs/my_gate_lib … <other arguments>
```

- The default qrun.out directory can be re-name or relocated with –outdir switch:

```
qrun -outdir /path/to/output/dir … <other arguments>
```


SIEMENS

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

command to launch qrun

```
qrun -f qrun_commands.f -verbose -gui -visualizer
```

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

- `-f <filename>` - concatenate all arguments listed in the file to qrun command
- `-verbose` - prints additional output
- `-visualizer` - run simulation in visualizer

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

command to launch qrun

```
qrun -f qrun_commands.f -verbose -gui -visualizer
```

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008 -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

Design comprises 4 libraries:

- Logical name, without physical path
- Each library has its own file list
- File lists contain both VHDL + Verilog files
- Can specify specific compile switches for individual libs e.g. -2008
- Libraries can be specified in any order

```
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/baudGen.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uart2BusTop.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uart2BusTop_pkg.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uartParser.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uartRx.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uartTop.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/uart/uartTx.vhd
```

```
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/clock_manager.v
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/dual_port_ram.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/reset_manager.v
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/spi_slave_dut.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/RTL_Code/spi_uart_top.vhd
```

```
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/blk_mem_gen_v8_2/blk_mem_gen_v8_2.vhd
```

```
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/testbenches/bidir_txrx_testbench_rtl.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/testbenches/spi_uart_testbench_rtl.vhd
$PROJECT_ROOT/SPI2UART_VHDL_CODE/Code/testbenches/uart_2_spi_testbench_rtl.vhd
```

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

command to launch qrun

```
qrun -f qrun_commands.f -verbose -gui -visualizer
```

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

Design comprises 4 libraries:

- Logical name, without physical path
- Each library has its own file list
- File lists contain both VHDL + Verilog files
- Can specify specific compile switches for individual libs e.g. -2008
- Libraries can be specified in any order

Top Design Unit:

- Define the top-level design unit with '-top'

Vopt options:

- Here we specify -debug access, any required libs, and code coverage options

Vsim Options:

- Timescale and coverage enabled
- Define TCL commands to be ran

Example 3 : usage of –makelib

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

1. Create the libraries

qrun.log transcript

```
QuestaSim-64 qrun QA Baseline: 2023_weekly_231020 - 5695866 Utility 2023.10 Oct 21 2023
Start time: 05:18:31 on Nov 27,2023
qrun -f qrun_commands2.f -verbose -gui
***Vlib command: vlib -quiet qrun.out/work
Creating library 'qrun.out/work'.
***Vlib command: vlib -quiet qrun.out/uart_lib
Creating library 'qrun.out/uart_lib'.
***Vlib command: vlib -quiet qrun.out/spi_uart_lib
Creating library 'qrun.out/spi_uart_lib'.
***Vlib command: vlib -quiet qrun.out/tb_lib
Creating library 'qrun.out/tb_lib'.
***Vlib command: vlib -quiet qrun.out/blk_mem_gen_v8_2
Creating library 'qrun.out/blk_mem_gen_v8_2'.
```

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

qrun.log transcript

```
***Vlog '-makelib' command: vlog -adaptive -L unisim -L ../SPI2UART_VHDL_CODE/libraries/unisim
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/RTL_Code/clock_manager.v
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/RTL_Code/reset_manager.v -work qrun.out/spi_uart_lib
-statslog qrun.out/stats_log -writesessionid +qrun.out/top_dus -csession=incr -csessionid=0
QuestaSim-64 vlog QA Baseline: 2023_weekly_231020 - 5695866 Compiler 2023.10 Oct 21 2023
Start time: 05:18:31 on Nov 27,2023
vlog -adaptive -L unisim -L ../SPI2UART_VHDL_CODE/libraries/unisim
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/RTL_Code/clock_manager.v
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/RTL_Code/reset_manager.v -work qrun.out/spi_uart_lib
-statslog qrun.out/stats_log -writesessionid "+qrun.out/top_dus" -csession=incr -csessionid=0
-- Compiling module clock_manager
-- Compiling module reset_manager

Top level modules:
            clock_manager
            reset_manager
End time: 05:18:31 on Nov 27,2023, Elapsed time: 0:00:00
Errors: 0, Warnings: 0
```

1. Create the libraries
2. Compiles any Verilog/SV files

```
qrun.out/
|-- blk_mem_gen_v8_2
|   |-- _info
|   |-- _lib1_1.qdb
|   |-- _lib1_1.qpg
|   |-- _lib1_1.qtl
|   |-- _lib.qdb
|   `-- _vmake
|-- history
|-- history.cnt
|-- spi_uart_lib
|   |-- _info
|   |-- _lib1_1.qdb
|   |-- _lib1_1.qpg
|   |-- _lib1_1.qtl
|   |-- _lib.qdb
|   `-- _vmake
|-- stats_log
|-- tb_lib
|   |-- _info
|   |-- _lib1_1.qdb
|   |-- _lib1_1.qpg
|   |-- _lib1_1.qtl
|   |-- _lib.qdb
|   `-- _vmake
|-- top_dus
|-- uart_lib
|   |-- _info
|   |-- _lib1_1.qdb
|   |-- _lib1_1.qpg
|   |-- _lib1_1.qtl
|   |-- _lib.qdb
|   `-- _vmake
|-- version
`-- work
    `-- _info
```


SIEMENS

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

1. Create the libraries
2. Compiles any Verilog/SV files
3. Scan the VHDL libraries to work out compile dependencies
4. Compiles any VHDL files in correct dependency order

qrun.log transcript

```
vcom ../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/testbenches/bidir_txrx_testbench_rtl.vhd
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/testbenches/spi_uart_testbench_rtl.vhd
../VRM_VHDL_Qrun_Version2/SPI2UART_VHDL_CODE/Code/testbenches/uart_2_spi_testbench_rtl.vhd -work
qrun.out/tb_lib -noautoorderrefresh -autoorder -statslog qrun.out/stats_log -writesessionid
"+qrun.out/top_dus" -csession=incr -csessionid=0
-- Loading package STANDARD
-- Scanning entity bidir_txrx_testbench
-- Scanning architecture rtl of bidir_txrx_testbench
-- Loading entity spi_uart_top
-- Scanning entity spi_uart_testbench
-- Scanning architecture rtl of spi_uart_testbench
-- Scanning entity uart_2_spi_testbench
-- Scanning architecture rtl of uart_2_spi_testbench
End time: 06:16:26 on Nov 27,2023, Elapsed time: 0:00:01
Errors: 0, Warnings: 0
```


SIEMENS

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

qrun.log transcript

```
vopt -debug -adaptive -L unisim "+cover=sbfec+spi_uart_top." -L ../SPI2UART_VHDL_CODE/libraries/unisim
tb_lib.spi_uart_testbench -work qrun.out/work -L uart_lib -L spi_uart_lib -L tb_lib -L blk_mem_gen_v8_2
-designfile design.bin -statslog qrun.out/stats_log -o qrun_opt -csession=incr -csessionid=0

Top level modules:
             spi_uart_testbench

Analyzing design...
-- Loading package STANDARD
-- Loading package TEXTIO
-- Loading architecture behaviour of spi_slave_dut
.
.
.
-- Optimizing package uart2BusTop_pkg
-- Optimizing architecture BUFG_V of BUFG

Optimized design name is qrun_opt
End time: 01:36:29 on Nov 28,2023, Elapsed time: 0:00:01
Errors: 0, Warnings: 0
```

1. Create the libraries
2. Compiles any Verilog/SV files
3. Scan the VHDL libraries to work out compile dependencies
4. Compiles any VHDL files in correct dependency order
5. Runs vopt on the testbench

- Optimizes the design with -debug
- Applies –adaptive switch
- Applies code coverage only to the DUT
- Automatically adds the –L switches

SIEMENS

Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

qrun.log transcript

```
# vsim -t ps -coverage -L ../SPI2UART_VHDL_CODE/libraries/unisim -lib qrun.out/work -
qwavedb=+livesim+signal+class -statslog qrun.out/stats_log qrun_opt -appendlog -csession=incr -
csessionid=1 "+no_qwavedb_threading" -classdebug
# Start time: 06:16:36 on Nov 27,2023
# Loading std.standard
# Loading std.standard
# Loading std.textio(body)
# Loading ieee.std_logic_1164(body)
# Loading ieee.std_logic_arith(body)
# Loading ieee.numeric_std(body)
# Loading tb_lib.spi_uart_testbench(rtl)#1
# Loading verilog.vl_types(body)
# Loading ieee.std_logic_unsigned(body)
# Loading uart_lib.uart2bustop_pkg
# Loading spi_uart_lib.spi_uart_top(behaviour)#1
# Loading spi_uart_lib.clock_manager(fast)
.
.
.
```

1. Create the libraries
2. Compiles any Verilog/SV files
3. Scan the VHDL libraries to work out compile dependencies
4. Compiles any VHDL files in correct dependency order
5. Runs vopt on the testbench
6. Run vsim to launch the simulation

- Automatically applies specified vsim switches & do file commands

SIEMENS

Example 3 : usage of –makelib

qrun_commands.f

```
-reflib ../SPI2UART_VHDL_CODE/libraries/unisim

-makelib uart_lib -f uart_lib_files.f -endlib
-makelib spi_uart_lib -f spi_uart_lib_files.f -2008  -endlib
-makelib tb_lib -f work_files.f -endlib
-makelib blk_mem_gen_v8_2 -f blk_mem_gen_v8_2_files.f -endlib

-top tb_lib.spi_uart_testbench

-vopt.options -debug -adaptive -L unisim +cover=sbfec+spi_uart_top. -end
-vsim.options -t ps -coverage -end
-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

qrun.log transcript

```
# run 10 us
# ** Note: ################################################
#    Time: 0 ps  Iteration: 0  Instance: /spi_uart_testbench
# ** Note: ##     param1 value = 8
#    Time: 0 ps  Iteration: 0  Instance: /spi_uart_testbench
# ** Note: ################################################
#    Time: 0 ps  Iteration: 0  Instance: /spi_uart_testbench
# coverage save sim.ucdb
# coverage report -summary
# Coverage Report Totals BY INSTANCES: Number of Instances 18
#
#     Enabled Coverage              Bins      Hits    Misses    Weight  Coverage
#     ----------------              ----      ----    ------    ------  --------
#     Branches                       583       187       396         1    32.07%
#     Conditions                     201         7       194         1     3.48%
#     Expressions                     31         4        27         1    12.90%
#     FSM States                      36         3        33         1     8.33%
#     FSM Transitions                 73         0        73         1     0.00%
#     Statements                     527       236       291         1    44.78%
# Total coverage (filtered view): 16.92%
```

1. Create the libraries
2. Compiles any Verilog/SV files
3. Scan the VHDL libraries to work out compile dependencies
4. Compiles any VHDL files in correct dependency order
5. Runs vopt on the testbench
6. Run vsim to launch the simulation
7. Exec do command

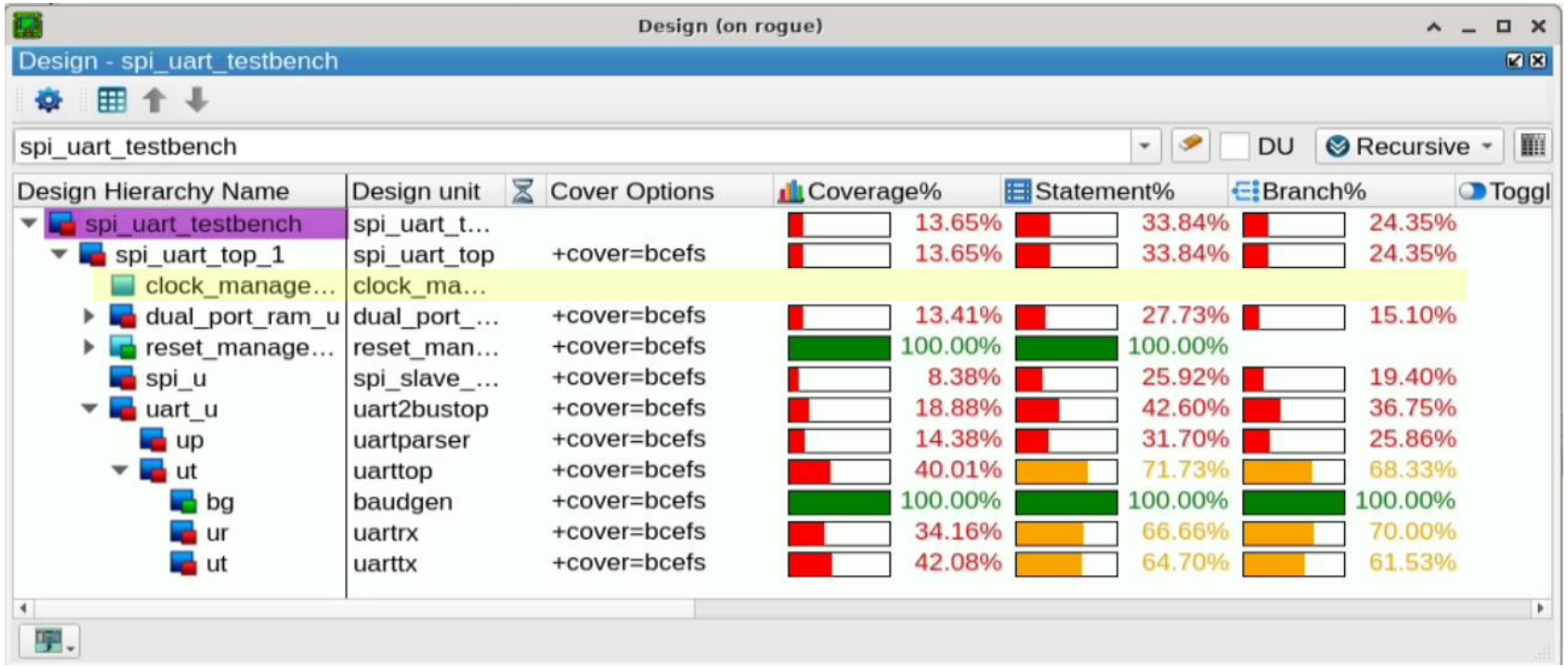
Example 3 : usage of –makelib

EXPERT SERIES
Customer Support WEBINAR

command to view coverage

```
visualizer -viewcov sim.ucdb
```

- View coverage Visualizer:
  View -> Coverage -> Testplan Tracker
- Code coverage applied to only the DUT



SIEMENS

Help Menu

EXPERT SERIES
Customer Support WEBINAR

Help menu is available on the manual page

`qrun -help`

```
 The qrun utility simplifies the use of Questa tools to compile, optimize
    and simulate a design by taking all input files and options, and then
    automatically executing the correct tool and command-line arguments for
    each file.

...

    Try the following to get help on specific options or categories:

    qrun -help all            : List all categories and options
    qrun -help category       : List all categories
    qrun -help <option>       : Help on an option
    qrun -help <command-line> : Help on all options in a command-line
    qrun -help <category>     : List all options in a category

    Category:       The list of categories in qrun tool.

    General         List most common general options
    Compiler        List options for compiler, and optimizer controls
    Design          List options for design units, and libraries
    File            List options for file and option specifications
    Flow            List options for Flow (Coverage, PA, Debug, ...)
    Messages        List all warn, error, note, fatal messages options
    Script          List options for scripts, environment, history, and log
    Subgroup        List options for handle grouping files and options
    SystemC         List options supporting for SystemC
```

SIEMENS

Help Menu

EXPERT SERIES
Customer Support WEBINAR

Help menu is available on the manual page

`qrun -help Subgroup`

```
-----------------------------------Subgroup-------------------------------------
Grouping files and options

--------------------------------------------------------------------------------
-filemap <vcom/vlog files, options> -end|-endfilemap
                                Compile the Verilog and VHDL files with the
                                given options added.

-makelib <libpath[:logical]> <vcom/vlog files, options> -end|-endlib
                                Compile the Verilog and VHDL files into library
                                'libname'. The optional 'logical' name will be
                                mapped to the path.

-precompile [libpath[:logical]] <vcom/vlog files, options> -end
                                Compile the Verilog and VHDL files with the
                                given options before other files.

-reflib <library>               Add to library search path for all tools.

-refreshlibs                    Refresh all libraries referenced by
                                -makelib/-reflib switches.
```

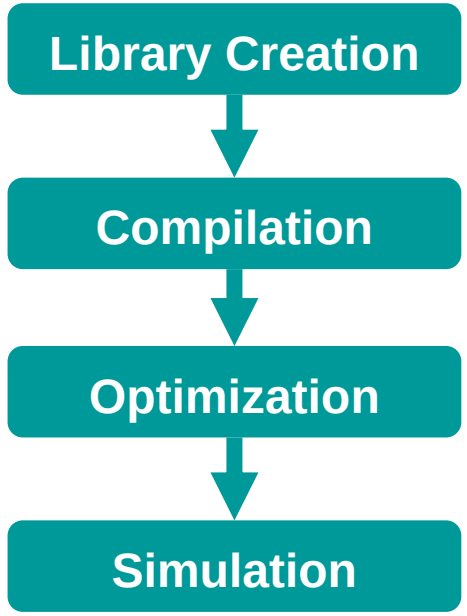
SIEMENS

Incremental Compilation

Assuming running Qrun again, after it has compiled and optimized a design.

- If there is no change in source files or options:
  - All compilation and optimization will be skipped
  - Qrun will print messages like the following:

```
-- Skipping 'vlog pkg.sv t.sv -work qrun.out/work...'
-- Skipping vopt
```



- A change in vsim options will **not** trigger a re-compile:
  - For instance: +UVM_TESTNAME, sv_seed, -do

- A change in "-makelib" commands will trigger compilation (and optimization)
- A change in "-makepdu" commands will trigger optimization

For instance:

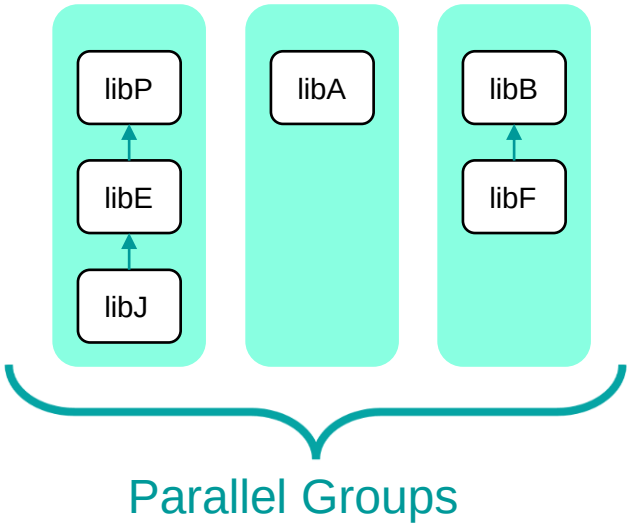
```
-makelib lib1 t1.sv -endlib
-makelib lib2 t2.sv -endlib
```

if only t1.sv is changed then only –makelib lib1 will be re-run.

Parallel Compilation

- **Multiple vlog and vcom occur at the same time**
  - Use `-parallel` to enable this feature
  - Use `-jcomp <n>` to indicate how many cores to use. The default is 4

- **Packages and some design units must be compiled before others**
  - Qrun determines these dependencies and orders the compiles accordingly.
  - This results in “parallel groups”



- **Best parallel performance**
  - -makelibs must be present to benefit from this feature
  - -makelibs should use different libraries (-makelibs to the same library will be serialized)

EXPERT SERIES
Customer Support WEBINAR

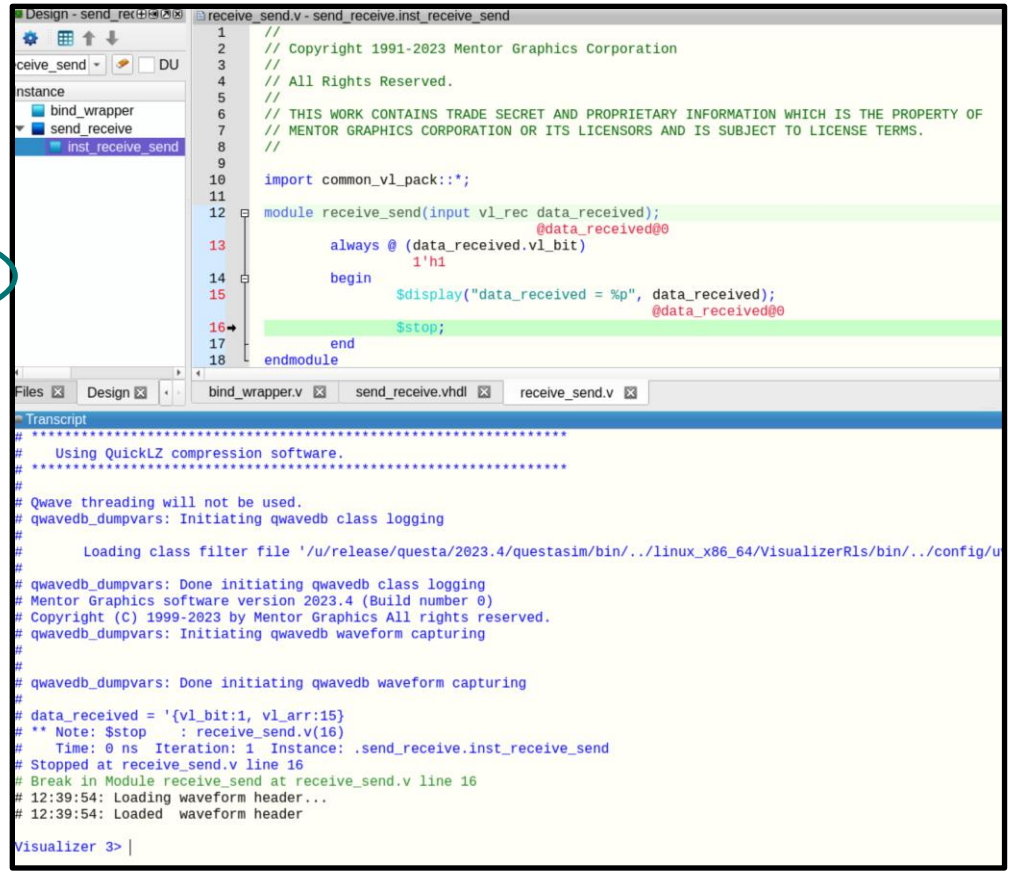
HOW TO USE QRUN

REAL WORLD USE CASES

SIEMENS

Invoking Visualizer

- `qrun -f qrun_commands.f -gui -visualizer`
  - Adding `-visualizer` with `-gui` invokes the Visualizer GUI
- qrun adds the following switches to vopt and vsim
  - vopt switches: `-debug,livesim -designfile design.bin`
  - vsim switches: `-qwavedb=+signal+class`
- Brings up Visualizer in live simulation mode
  - Allows you to set breakpoints and add signals to the wave



DPI/VPI and SystemC

EXPERT SERIES
Customer Support WEBINAR

- C/C++ Source code
  - Put C, VPI and DPI source files directly on qrun command line
- Precompiled .so (shared objects)
  - User `-pli` <library.so> or `-sv_lib` <library.so>
- SystemC
  - All source files and arguments follow the "`-sysc`" switch

SIEMENS

SystemC/DPI Example

hello.cpp

```
#include "systemc.h"
#include "dpiheader.h"

SC_MODULE(hello)
{
    void call_verilog_task();
    SC_CTOR(hello)
    {
        SC_THREAD(call_verilog_task);
    }
    ~hello() {};
};


void hello::call_verilog_task()
{
    svSetScope(svGetScopeFromName("top"));
    for(int i = 0; i < 3; ++i)
    {
      verilog_task();
    }
}
SC_MODULE_EXPORT(hello);
```

hello.sv

```
module top;

export "DPI-C" task verilog_task;

task verilog_task();
  $display("[%t] hello from verilog_task.", $time);
  #200;
  $display("[%t] exiting verilog_task.", $time);
endtask

initial
begin
   $display("starting initial block");
  #200000 $finish;
end
endmodule
```

SystemC/DPI Example

qrun.f

```
# Compile the HDL source(s)
-precompile
-sv -dpiheader dpiheader.h hello.sv –ccflags -g
-endlib

-dpioutoftheblue 2
# Simulate the design
-sysc
-g
hello.cpp
-top hello
-top top
```

command to launch qrun

```
qrun –f qrun.f
```

```
vlog -sv -dpiheader dpiheader.h -ccflags -g hello.sv -work qrun.out/work
sccom -g -incr hello.cpp -work qrun.out/work
sccom -link -statslog qrun.out/stats_log -work qrun.out/work
vopt hello top -work qrun.out/work -o qrun_opt
vsim -dpioutoftheblue 2 -lib qrun.out/work -c -do "run -all; quit -f" qrun_opt
```

```
# run -all
# starting initial block
# [                  0] hello from verilog_task.
# [                200] exiting verilog_task.
# [                200] hello from verilog_task.
# [                400] exiting verilog_task.
# [                400] hello from verilog_task.
# [                600] exiting verilog_task.
# ** Note: $finish    : hello.sv(14)
```

UVM with Qrun

- Qrun handles UVM compilation for the user
  - You don't have to add any UVM information to the qrun command line
- Add the "`-uvm`" switch to the qrun command line
  - Qrun adds the required `+incdirs` and compiles uvm_dpi.cc
  - The Questa UVM libraries are used by default
  - The default uvm version is uvm-1.1d
- To change the default UVM version (uvm-1.1d)

```
qrun -uvm -uvmhome uvm-1.1c
```

- To use a custom version of UVM

```
qrun -uvm -uvmhome <path to UVM src>
```

UVM - Simple “Hello World” Example

my_pkg.sv

```
`include "uvm_macros.svh"
package my_pkg;
  import uvm_pkg::*;
  class my_env extends uvm_env;
    `uvm_component_utils(my_env)
    function new(string name, uvm_component parent);
      super.new(name, parent);
    endfunction
  endclass: my_env
  class my_test extends uvm_test;
    `uvm_component_utils(my_test)
    my_env m_env;
    function new(string name, uvm_component parent);
      super.new(name, parent);
    endfunction
    function void build_phase(uvm_phase phase);
      m_env = my_env::type_id::create("m_env", this);
    endfunction
    task run_phase(uvm_phase phase);
      phase.raise_objection(this);
      #10;
      `uvm_info("", "Hello World", UVM_MEDIUM)
      phase.drop_objection(this);
    endtask
  endclass: my_test
endpackage: my_pkg
```

design.sv

```
interface dut_if;
endinterface
module dut(dut_if dif);
endmodule
```

testbench.sv

```
module top;
  import uvm_pkg::*;
  import my_pkg::*;
  dut_if dut_if1 ();
  dut dut1 (.dif(dut_if1) );
  initial
  begin
    run_test("my_test");
  end
endmodule: top
```

UVM - Simple “Hello World” Example

qrun.f

```
-sv my_pkg.sv testbench.sv design.sv
-uvm -uvmhome uvm-1.1c
```

command to launch qrun

```
qrun –f qrun.f
```

```
vlog -work qrun.out/work +incdir+/u/release/questa/2023.4/questasim/verilog_src/uvm-1.1c/src -
ccflags -Wno-missing-declarations -ccflags -Wno-maybe-uninitialized -ccflags -Wno-return-type -
ccflags -DQUESTA /u/release/questa/2023.4/questasim/verilog_src/uvm-1.1c/src/dpi/uvm_dpi.cc
/u/release/questa/2023.4/questasim/verilog_src/uvm-1.1c/src/uvm_pkg.sv
/u/release/questa/2023.4/questasim/verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv
vlog -sv testbench.sv design.sv -work qrun.out/work
+incdir+/u/release/questa/2023.4/questasim/verilog_src/uvm-1.1c/src
vopt <TOP-DESIGN-UNIT> -work qrun.out/work -o qrun_opt
vsim -lib qrun.out/work -c -do "run -all; quit -f" qrun_opt
```

```
# run -all
# ----------------------------------------------------------------
# UVM-1.1c
# (C) 2007-2012 Mentor Graphics Corporation
# (C) 2007-2012 Cadence Design Systems, Inc.
# (C) 2006-2012 Synopsys, Inc.
# (C) 2011-2012 Cypress Semiconductor Corp.
.....
# UVM_INFO my_pkg.sv(30) @ 10: uvm_test_top [] Hello World
```

ALU UVM Example

- Here we will see how to use qrun in complete ALU UVM Testbench

qrun_commands.f

```
-vlog.options -sv -timescale 1ps/1ps -suppress 2223 -suppress 2286
+define+UVM_REPORT_DISABLE_FILE_LINE -end

-makelib work -f work_files.f -endlib

-top hdl_top
-top hvl_top

-vopt.options –debug +cover=sbfec+alu. -end
-o opt_debug

 -do " set NoQuitOnFinish 1; run 0; do wave.do "

-vsim.options -sv_seed random  +UVM_TESTNAME=test_top -permit_unmatched_virtual_intf  +notimingchecks -suppress 8887 -suppress 8887 -suppress 8887 -suppress
8887 -solvefaildebug -solvefailtestcase -assertdebug -uvmcontrol=all +uvm_set_config_int=*,enable_transaction_viewing,1 -classdebug -msgmode tran –coverage
-qwavedb=+signal+transaction+class+uvm_schematic -end


-do "run 10 us; coverage save sim.ucdb; coverage report -summary"
```

ALU UVM Example

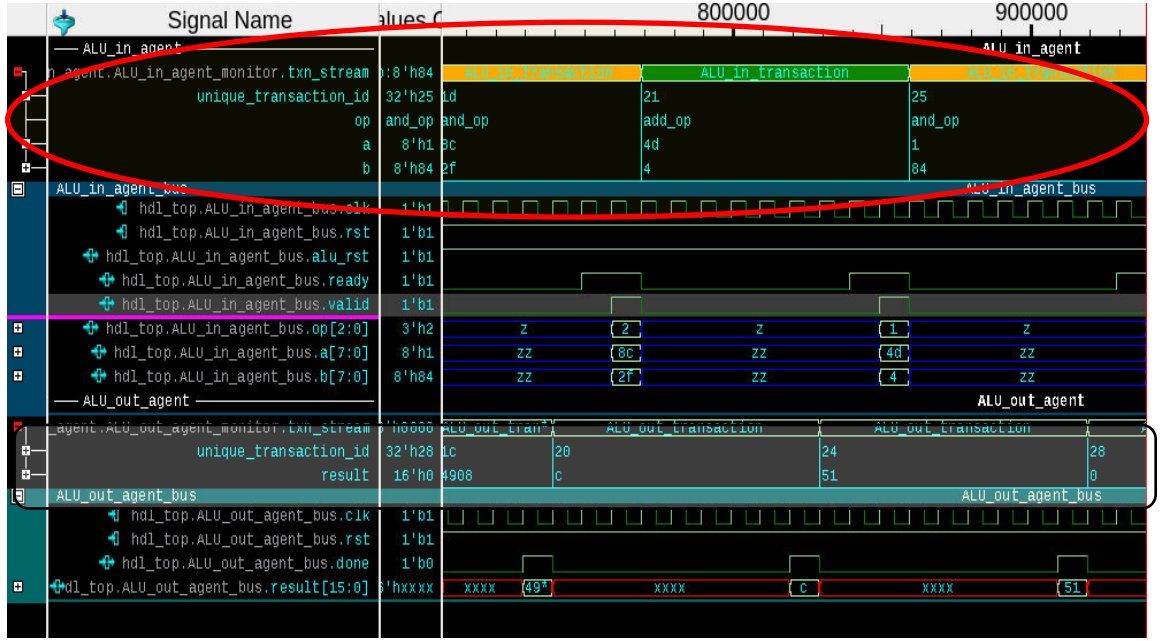
EXPERT SERIES
Customer Support WEBINAR

```
## Compile the DUT
$PROJ_TOP/Source_Code/project_benches/ALU/rtl/verilog/alu.v
$PROJ_TOP/Source_Code/project_benches/ALU/rtl/vhdl/vhdl_dut.vhd
## Compile UVMF Base Package
+incdir+$UVMF_HOME/common/fli_pkg
+incdir+$UVMF_HOME/uvmf_base_pkg
$UVMF_HOME/common/fli_pkg/fli_pkg.sv
-F $UVMF_HOME/uvmf_base_pkg/uvmf_base_pkg_filelist_hdl.f
-F $UVMF_HOME/uvmf_base_pkg/uvmf_base_pkg_filelist_hvl.f
## Compile UVMF Agent Code
+incdir+$PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_in_pkg
+incdir+$PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_out_pkg
-F $PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_in_pkg/ALU_in_filelist_hvl.f
-F $PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_in_pkg/ALU_in_filelist_hdl.f
-F $PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_out_pkg/ALU_out_filelist_hvl.f
-F $PROJ_TOP/Source_Code/verification_ip/interface_packages/ALU_out_pkg/ALU_out_filelist_hdl.f
## Compile UVMF Environment Code
+incdir+$PROJ_TOP/Source_Code/verification_ip/environment_packages/ALU_env_pkg
$PROJ_TOP/Source_Code/verification_ip/environment_packages/ALU_env_pkg/ALU_env_pkg.sv
## Compile UVMF testbench
+incdir+$PROJ_TOP/Source_Code/project_benches/ALU/tb/parameters
$PROJ_TOP/Source_Code/project_benches/ALU/tb/parameters/ALU_parameters_pkg.sv
+incdir+$PROJ_TOP/Source_Code/project_benches/ALU/tb/sequences
$PROJ_TOP/Source_Code/project_benches/ALU/tb/sequences/ALU_sequences_pkg.sv
+incdir+$PROJ_TOP/Source_Code/project_benches/ALU/tb/tests
$PROJ_TOP/Source_Code/project_benches/ALU/tb/tests/ALU_tests_pkg.sv
+incdir+$PROJ_TOP/Source_Code/project_benches/ALU/tb/testbench
-F $PROJ_TOP/Source_Code/project_benches/ALU/tb/testbench/top_filelist_hvl.f
-F $PROJ_TOP/Source_Code/project_benches/ALU/tb/testbench/top_filelist_hdl.f
```

work_files.f

command to launch qrun

```
qrun -f qrun_commands.f –gui -visualizer
```



UVMF_HOME* refers to UVM Framework Home whose base classes are extended from UVM base classes

SIEMENS

Preoptimised Design Unit – PDU Flow - Visualizer

EXPERT SERIES
Customer Support WEBINAR

- PDU flow is often used to avoid optimizing your design over and over again specially when it takes a long time
- When your design freezes, it is possible to create a pre-optimized image and use it for simulating with various tests for regression
  - reuse the PDU and exclude it from being optimized
  - We create pdu's with qrun using `-makepdu/-end`
  - `vopt -access` inside the `-makepdu/-end` if you have hierarchical references that crosses the PDU boundary.
  - `-access=rw+/.` (global read/write access) negatively impact performance.
  - `-access=rw+m1/u2/u2/m1_bot2_reg` point specifically to the objects that need visibility

SIEMENS

PDU Flow - Visualizer

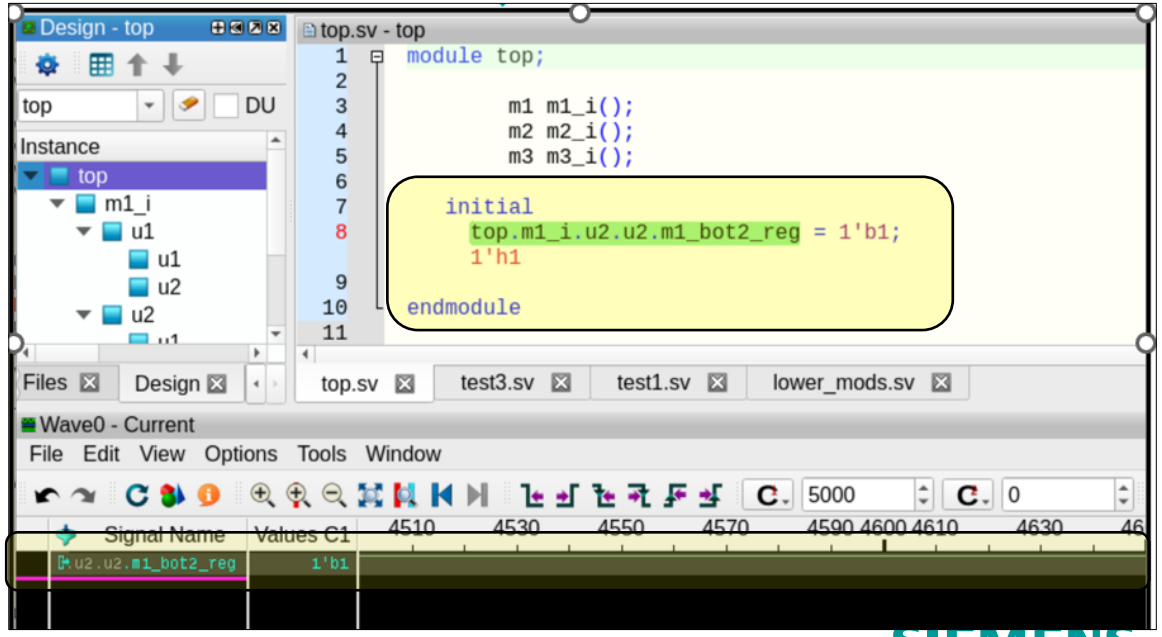
qrun.f

```
// Create library lib1 in qrun.out and compile all source into it
-makelib lowerLibs
  lower_mods.sv
-end
-makelib lib1
      top.sv test1.sv  test2.sv  test3.sv
-end
// Create a PDU from module m1 in lib1
-makepdu m1_opt_pdu lib1.m1
  -L lowerLibs
  // Add design access for hierarchical ref
 //    -access=rw+/.
  -access=rw+m1/u2/u2/m1_bot2_reg
  // Create a visualizer design.bin file for the
PDU
  -designfile m1_opt_design.bin -debug
-end
// Create a PDU from module m2 in lib1
// This PDU will be a "blackbox" in Visualizer since we don't
generate
// a design.bin for this PDU
-makepdu m2_opt_pdu lib1.m2
  -L lowerLibs
-end
-top top
// Invoke visualizer in live simulation mode
-gui -visualizer
-do "run -all;"
```

```
vlog lower_mods.sv -work qrun.out/lowerLibs
vlog top.sv test1.sv test2.sv test3.sv -work qrun.out/lib1
vopt -pdu -o m1_opt_pdu lib1.m1 -L lowerLibs -access=rw+m1/u2/u2/m1_bot2_reg
-designfile m1_opt_design.bin -debug -work lib1
vopt -pdu -o m2_opt_pdu lib1.m2 -L lowerLibs -work lib1
vopt top -work qrun.out/work -L lowerLibs -L lib1 -o qrun_opt -debug,livesim
-designfile design.bin
vsim -lib qrun.out/work -qwavedb=+signal+class -visualizer -do "run -all;"
qrun_opt
```

command to launch qrun

```
qrun -f qrun.f
```

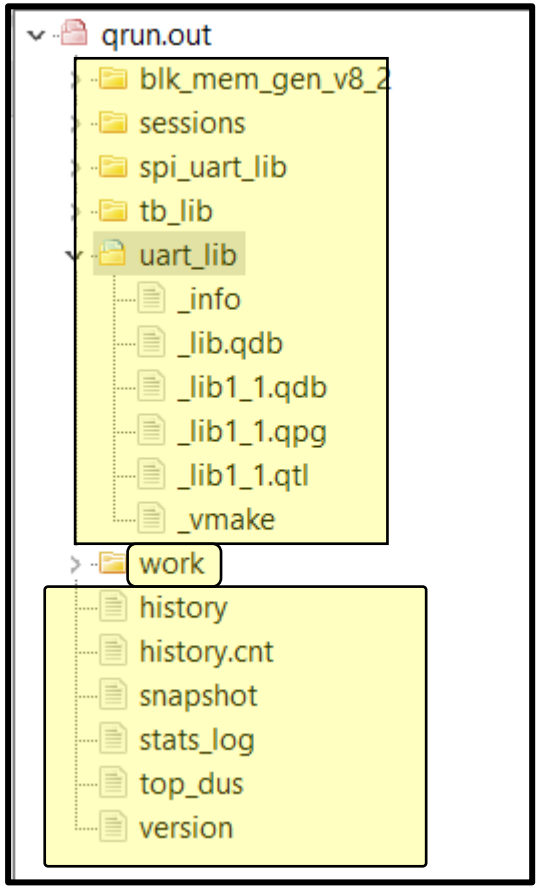


SIEMENS

Qrun directory structure

- qrun puts all output in the directory ./qrun.out
  - All `-makelib` libraries are created in ./qrun.out by default
- qrun.out contains
  - The work library
  - Any library that was created with `-makelib` without a libpath
  - A command history file
  - Other qrun housekeeping information
- Renaming qrun.out
  - The qrun output directory can be renamed
  - `-outdir` <path and directory name>

qrun.out directory



EXPERT SERIES
Customer Support WEBINAR

DEMO

SIEMENS

EXPERT SERIES
Customer Support WEBINAR

HOW TO INTEGRATE QRUN INTO YOUR ENVIRONMENT

QRUN FLOWS

SIEMENS

Qun 1, 2, and 3 step flows

EXPERT SERIES
Customer Support WEBINAR

- Qrun flows
  - Qrun supports 1, 2, and 3 step flow
  - You can also create an “elaboration file”
- Flow descriptions
  - `qrun -f qrun.f <no switch>`: compile, optimize, and simulate are all run together
  - `qrun -f qrun.f -optimize`: only compile and optimize are run
  - `qrun -f qrun.f -simulate`: only runs simulate
- Makes qrun more flexible
  - Fits into existing environments easier

SIEMENS

Running Regressions with qrun

EXPERT SERIES
Customer Support WEBINAR

- Single step flow
  - Qrun compiles, optimizations, and simulates the first time
  - Only runs simulation on subsequent runs if nothing changes
    - Skips compilation and optimization
  - Changing test dependent switches does not cause a recompile
    - i.e. +UVM_TESTNAME, -sv_seed, and others
  - Changes to src files or compile/opt switches will trigger a recompile
    - i.e. adding +defines, +incdir, etc will trigger a recompile and optimization

SIEMENS

Running Regressions with qrun

EXPERT SERIES
Customer Support WEBINAR

- Two step flow
  - `qrun -f qrun.f -optimize`
  - Compiles and optimizes for the first test
    - Skips both on subsequent tests if no src files or switches change
  - `qrun -f qrun.f -simulate`
  - prevents re-compilation and/or re-optimization

- Three step flow
  - `qrun -f qrun.f -compile`
  - `qrun -f qrun.f -optimize`
  - `qrun -f qrun.f -simulate`

- Elaboration flow – Four step flow
  - `qrun -f qrun.f -elab <filename>`
  - `qrun -f qrun.f -load_elab <filename>`

SIEMENS

Snapshots

EXPERT SERIES
Customer Support WEBINAR

- A Snapshot is the named output of the optimizer
  - Allows you to create several “flavors” of optimized output

- Examples
  - Creating the snapshot
    - `qrun -f qrun.f -optimize -snapshot fullOpt_sn`
    - `qrun -f qrun.f -optimize -snapshot coverage_sn`
    - `qrun -f qrun.f -optimize -snapshot sdf_sn`
  - Using the Snapshot
    - `qrun -simulate -snapshot fullOpt_sn`
- Qrun uses the default snapshot name “`qrun_opt`”
  - If you don’t use the `-snapshot` switch, the default named is `qrun_opt`

SIEMENS

Qrun usage tips

EXPERT SERIES
Customer Support WEBINAR

- Users commonly use three step mode
  - Their scripts are generally setup to run three step flow
  - Three step mode makes it easy to integrate qrun into existing scripts
- Separate build and run directories
  - Use two step mode "`qrun –qrun.f -optimize`" to build in one location
  - Use "`qrun -f qrun.f - outdir <path> -simulate`" to simulate from a different directory
    - `- outdir` `<path>` points the build location
- `- top` <top level du>
  - Use `- top` to specify top level module. Multiple -top switches allowed
  - qrun can usually determine the top level module
    - `- top` not always necessary
  - Required with SystemC and sometimes with VHDL and definitely with several top designs

SIEMENS

Qrun troubleshooting

EXPERT SERIES
Customer Support WEBINAR

- `-script <filename>`
  - Creates a script of all the tool command lines
    - Helps to understand qrun behavior
  - Example:
  - `qrun –f qrun.f –script t.csh`
    - File t.csh will contain all the Questa commands that qrun will call
- `-env <filename>(format)`
  - Same as –script plus it dumps all the current environment variables
  - Environment variable output can be formatted
    - Ex: `-env "env.out(setenv %s = %s)“`:
    - The first %s is the environment variable
    - The second %s is the value

SIEMENS

Qrun Script example

- qrun test.sv -top top -script script.out

```
cat script.out
vlog test.sv -work qrun.out/work
vopt top -work qrun.out/work -o qrun_opt
vsim -lib qrun.out/work -c -do "run -all; quit -f" qrun_opt
```

- qrun test.sv -top top -env "env.out(setenv %s %s)"

```
cat env.out

setenv MTI_VCO_MODE 64
setenv HOSTTYPE x86_64-linux

setenv MODEL_TECH /u/release/questa/2023.4/questasim/linux_x86_64
vlog test.sv -work qrun.out/work
vopt top -work qrun.out/work -o qrun_opt
vsim -lib qrun.out/work -c -do "run -all; quit -f" qrun_opt
```

Summary

EXPERT SERIES
Customer Support WEBINAR

- Qrun simplifies the use of Questa tools
  - Eliminates the need to think at the tool level
- It provides shortcuts for complex tool usage
  - UVM, Coverage, etc.
  - Getting the switches right can be challenging
- Makes Questa adoption easier for new users
  - You don’t have to understand all the tools and switches
- Simplifies debug
  - Makes bringing up the GUI and setup it for debug easy
- Can be used in the user’s Regression flow
  - Is flexible enough to fit into many regression flows
- It makes it easier to switch between Questa Classic and Visualizer flow

SIEMENS

EXPERT SERIES
Customer Support WEBINAR

Contact

Tal Nahari

Applications Engineer
Functional Verification Support / EMEA / Siemens EDA

Phone: +972544334709

E-mail: tal.nahari@siemens.com

Nomita Goswami

Applications Engineer
Functional Verification Support / EMEA / Siemens EDA

Phone: +447921741154

E-mail: nomita.goswami@siemens.com