Verification of Digital Designs: Week 4

Martin Schoeberl

Technical University of Denmark
Embedded Systems Engineering

September 8, 2021

Overview

- Discuss your designs and tests
- ChiselTest
- Class hierarchy and testing
- Mixed language testing
- Verilog and Verilator
- S4NOC
- Concurrent testing
- Lab: some concurrent testing of a FIFO

Lab, Project, and Grading

- There is no exam in this course
- Grade is based on lab work and final project
- Reminder: 5 ECT = average 140 hours work
- Let us look into your design/verification examples

Literature

- We searched for some literature
- I browsed your suggestions and a few more
- No really good textbook, some are simply bad
- Following was the best, but a bit dated (no UVM)
- *Writing Testbenches - Functional Verification of HDL Models, J. Bergeron*
- Get the 2003 edition

ChiselTest

- New testing framework
- Long called “tester2”
- Developed by Richard Lin at UCB
- Still in beta
- Will be the fully supported testing framework for Chisel
- see: `https://github.com/ucb-bar/chisel-testers2`

ChiselTest

- Core operations: `peek`, `poke`, `expect`
- Similar to iotester
- Inverted syntax
  - Instead of `poke(port, value)`
  - use `port.poke(value)`
- Values are Chisel literals (not `BigInt`)
- Based on ScalaTest

Example DUT

```
class Add extends Module {
  val io = IO(new Bundle {
    val a = Input(UInt(width = 8))
    val b = Input(UInt(width = 8))
    val c = Output(UInt(width = 8))
  })

  val reg = RegInit(UInt(0, width = 8))
  reg := io.a + io.b

  io.c := reg
}
```

Old PeekPokeTester

```
class AddTester(dut: Add) extends
    PeekPokeTester(dut) {

  for (a <- 0 to 2) {
    for (b <- 0 to 3) {
      val result = a + b
      poke(dut.io.a, a)
      poke(dut.io.b, b)
      step(1)
      expect(dut.io.c, result)
    }
  }
}

object AddTester extends App {
  iotesters.Driver.execute(Array[String](), () =>
     new Add()) { c => new AddTester(c) }
}
```

ChiselTest the Adder

```
class AddNewTester extends FlatSpec with
    ChiselScalatestTester with Matchers {

  behavior of "Adder with Testers2"

  it should "test addition" in {
    test(new Add()) { c =>
      for (a <- 0 to 2) {
        for (b <- 0 to 3) {
          val result = a + b
          c.io.a.poke(a.U)
          c.io.b.poke(b.U)
          c.clock.step(1)
          c.io.c.expect(result.U)
        }
      }
    }
  }
}
```

Using ChiselTest

- Define in `build.sbt` (both testers)

```
libraryDependencies += "edu.berkeley.cs" %%
   "chisel-iotesters" % "1.4.2"
libraryDependencies += "edu.berkeley.cs" %%
   "chiseltest" % "0.2.2"
```

- Chisel and ScalaTest come as a dependency of chiseltest
- No need to specify, easier with version numbers
- Import additional packages

```
import chiseltest._
import org.scalatest._
```

More Examples

- Show code examples: NITester, NetworkCompare, NocTester
- Code is in `https://github.com/schoeberl/soc-comm`

Test Different Implementations

- Modules need to extend a base class
- Test code expects the base class
- Need to use some generic magic
- Show code in `chisel_uvm` project

Mixed Language Testing

- Black box for Verilog code
- Test backend using Verilator
- There are two (three) tester backends: Treadle, Verilator, and Synopsis VCS
- Show code in `chisel_uvm` project
- Code is in https://github.com/chisel-uvm/chisel-uvm

Concurrent Testing

- Threaded concurrency with `fork` and `join`
- Needed for more complex testing
- E.g., Model several masters on a shared bus
- Concurrency is implicit in VHDL or Verilog
- Was added with ChiselTest to Chisel
- Show example: NetworkTester
- Code is in `https://github.com/schoeberl/soc-comm`

Lab Time

- Write a concurrent tester for a bubble FIFO
- First define some test criteria (in ScalaTest strings)
- To avoid name collisions, use your name in the test class

Home Work

- Read up on a topic
- Anything related to testing and verification
- Give a 15’ presentation on it next week
- Following list is just possible examples
  - Test coverage (Hans)
  - Testing in SW
  - Agile development and TDD (Victor)
  - Test categories (in SW, in HW)
  - Testing in open-source projects
  - Available test infrastructure (e.g., AXI transactions)
  - Testing a processor, e.g., what is Rocket doing?
  - Testing of Chisel itself
  - Your idea/interest