Testing
Chisel provides a evolving family of testers with different capabilities. A tester typically is invoked from scalatest. For example:

```scala
class CounterSpec extends ChiselPropSpec {
  property("Counter should wrap") {
    assertTesterPasses {
      new WrapTester(42)
    }
  }
}
```

**BasicTester:**
BasicTester: supports creation of a circuit and provides simple chisel operations and a family of assert statements.

```scala
class WrapTester(max: Int)
  extends BasicTester {
    val (cnt, wrap) = Counter[Boolean(true), max)
    when(wrap) {
      assert(cnt === UInt(max - 1))
      stop()
    }
  }
```

**Chisel-Testers:**
Additional testers can be found in the ucb-arg/chisel-testers repository. Current Testers are:

**StandardTester:**
Standard Tester: is a class with functions for testing Modules, connecting and communicating with a simulator:

```scala
class MuxTester(c: Mux) extends Tester(c) {
  for (sel <- 0 until 2) {
    poke(c.io.sel, sel)
    poke(c.io.in0, 0); poke(c.io.in1, 1)
    step(1)
    expect(c.io.out, sel)
  }
}
```

**Hardware IO Testers:**
The Hardware IO Testers run all tests by implementing small FSM's and vectors of input and output values. In general hardware testers are faster than the standard tester. There currently two forms

**Stepped Tester:**
SteppedHWIOTester: works like the Standard Tester but does not support the peek command

**Decoupled Tester:**
OrderedDecoupledHWIOTester: tests Modules with IO implementing DeqIO, EnqIO, Valid or Decoupled interfaces, automatically handling the ready/valid protocol.

**Defining:**
Subclass Tester with testing code:

```scala
class MuxTester(c: Mux) extends Tester(c) {
  for (sel <- 0 until 2) {
    poke(c.io.sel, sel)
    poke(c.io.in0, 0); poke(c.io.in1, 1)
    step(1)
    expect(c.io.out, sel)
  }
}
```

**Defining:**
Subclass OrderedDecoupledHWIOTester with testing code:

```scala
class MuxTester(c: Mux) extends Tester(c) {
  for (sel <- 0 until 2) {
    poke(c.io.sel, sel)
    poke(c.io.in0, 0); poke(c.io.in1, 1)
    step(1)
    expect(c.io.out, sel)
  }
}
```

**Defining:**
Subclass OrderedDecoupledHWIOTester with testing code: