ScalaHDL

Yao Li

Contents

- Syntax
  - annotation
- Simulator
- Test Bench

Syntax: Annotation

- Our original design:

```
@init(reset=1)
@sync(clock=1)
def mod_adder2(a,b,res=0){
    res.size = max(a.size,b.size)+1
    res = a + b
}
```

Syntax Annotation

- Our current design:

```
module.adder('a, 'b, 'res) {
  'res := 'a + 'b
  'res := 'res + 'a
}
```

- Problem:
  - Scala annotations must be in front of a class or method definition.

Solutions

- the Chisel approach:

```
class ShiftRegister extends Module {
  val io = new Bundle {
    val in  = UInt(INPUT, 1)
    val out = UInt(OUTPUT, 1)
  }
  val r0 = Reg(next = io.in)
  val r1 = Reg(next = r0)
  val r2 = Reg(next = r1)
  val r3 = Reg(next = r2)
  io.out := r3
}
```

Solutions

- the Chisel approach:
  - the always @ (posedge clk) will be added to an assignment of Reg or a when statement.
  - I haven't found how Chisel handles async block of logic yet

Solutions

- a method similar to annotation:

```
init(rst = 1)
sync(clk = 1)
module.adder('a, 'b, 'res) {
  'res := 'a + 'b
  'res := 'res + 'a
}
```

Implementation

- rst and clk are variables of ScalaHDL class.
- the init and sync function will record information in ScalaHDL class.
- the information recorded will work on the next module definition.
- the information will be cleared before other module definitions.

Simulator

- I'm still studying the simulator implementation of myHDL.

Test Bench

- Chisel test bench:

```
class ByteSelectorTests(c: ByteSelector)
  extends Tester(c, Array(c.io)) {
  defTests {
    ...
  }
}
```

- However, You may prefer:

```
defTest.ByteSelectorTests {
  ...
}
```

Any Question?

Thanks!