Integrating Verilog Designs in Chipyard

Abraham Gonzalez

UC Berkeley

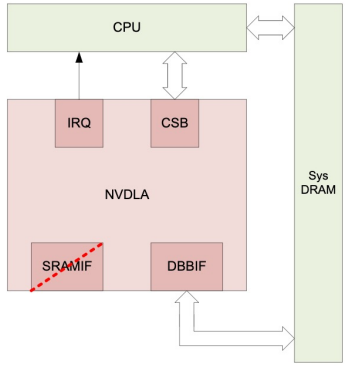
abe.gonzalez@berkeley.edu

Berkeley Architecture Research

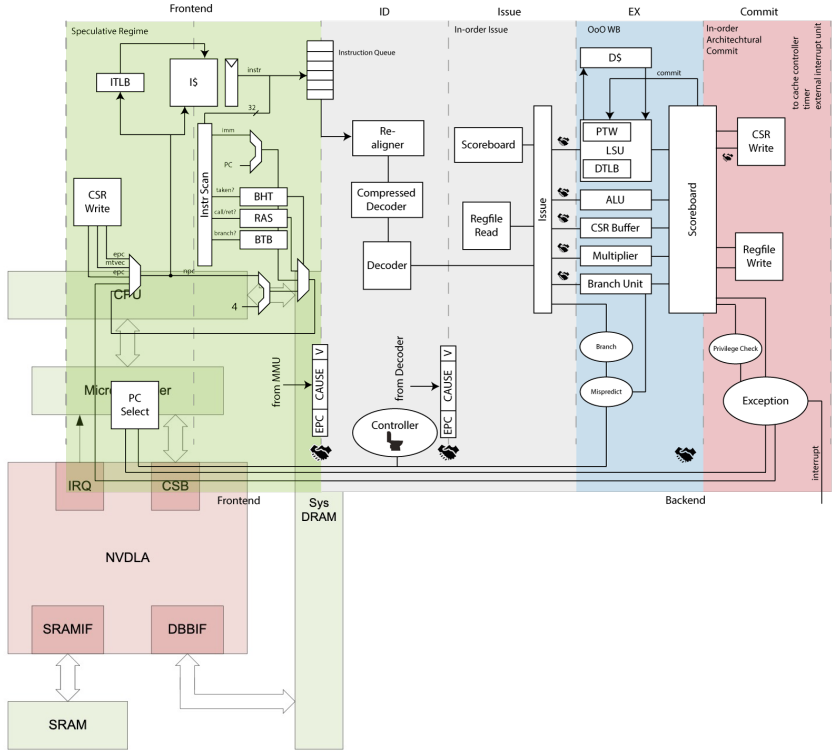
CHIPYARD

But what about Verilog designs!

- There are still plenty of exciting Verilog-based projects out there
- Many pre-made Verilog IP blocks
- Chipyard + Chisel allows for mixed languages
- Chipyard Verilog integrated projects
  - CVA6 (Ariane) Core
  - NVDLA
  - Example GCD MMIO accelerator
  - Much more!



"Small" NVDLA system

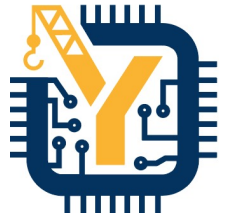


"Large" NVDLA system

Integrating a Verilog GCD accelerator

Background

- Let us look at the pre-integrated GCD accelerator
  - Simple GCD implementation
- Integrating involves a few steps
  1. Add the Verilog file(s) to a Chisel project
  2. Create a Chisel blackbox that matches Verilog module
  3. Instantiating Chisel blackbox in your Chisel design
- At this point you can connect the new Chisel module to
  - SoC buses
  - MMIO register maps
  - And other system signals



1st: Add Verilog Files to Chipyard Sub-Project

Directory Structure

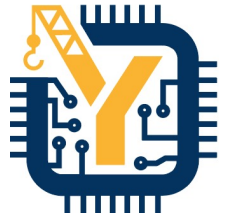
```
chipyard/
  generators/
    rocket-chip/
    sha3/
    chipyard/
      src/main/
        scala/example/
          GCD.scala
        resources/vsrc/
          GCDMMIOBlackBox.v
build.sbt
```

Our library of Chisel generators

Top-level Chipyard integration project

Verilog file holding IP

GCD Verilog IP



```
module GCDMMIOBlackBox
  #(parameter WIDTH)
   (
    input                   clock,
    input                   reset,
    output                  input_ready,
    input                   input_valid,
    input [WIDTH-1:0]       x,
    input [WIDTH-1:0]       y,
    input                   output_ready,
    output                  output_valid,
    output reg [WIDTH-1:0] gcd,
    output                  busy
   );
```

```
chipyard/
  generators/
    chipyard/
      src/main/
        resources/vsrc
          GCDMMIOBlackBox.v
    rocket-chip/
    boom/
    sha3/
  sims/
    verilator/
  tools/
    chisel/
    firrtl/
  tests/
  build.sbt
```

2nd: Create matching Chisel blackbox

Chisel Wrapper around Verilog BBox

```
class GCDMMIOBlackBox(val w: Int)
  extends BlackBox(Map("WIDTH" -> IntParam(w)))
  with HasBlackBoxResource
{
  val io = IO(new Bundle {
    val clock = Input(Clock())
    val reset = Input(Bool())
    val input_ready = Output(Bool())
    val input_valid = Input(Bool())
    val x = Input(UInt(w.W))
    val y = Input(UInt(w.W))
    val output_ready = Input(Bool())
    val output_valid = Output(Bool())
    val gcd = Output(UInt(w.W))
    val busy = Output(Bool())
  })

  addResource("/vsrc/GCDMMIOBlackBox.v")
}
```

- Must have the same IOs as the Verilog module
- Uses `addResource` to "link" the two modules (given by adding `HasBlackBoxResource`)
- The blackbox `Map` is used to allow Chisel to update Verilog parameters

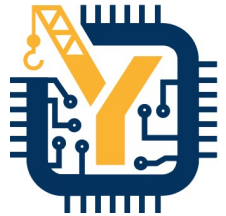
```
chipyard/
  generators/
    chipyard/
      src/main/
        scala/example
          GCD.scala
    rocket-chip/
    boom/
    sha3/
  sims/
    verilator/
  tools/
    chisel/
    firrtl/
  tests/
  build.sbt
```

3rd: Instantiate Chisel blackbox in your design

+

4th: Connect your blackbox to Chipyard components

Using the blackbox module



```
val impl = if (params.useBlackBox) {
  Module(new GCDMMIOBlackBox(params.width))
} else {
  Module(new GCDMMIOChiselModule(params.width))
}
```

- Now it is in Chisel!
- From here you can use the Verilog module to anything else in Chipyard
  - Accelerators
  - Cores
  - Memory modules
  - And more
- The example GCD Verilog IP is integrated into an MMIO accelerator

```
chipyard/
  generators/
    chipyard/
      src/main/
        scala/example
          GCD.scala
    rocket-chip/
    boom/
    sha3/
  sims/
    verilator/
  tools/
    chisel/
    firrtl/
  tests/
  build.sbt
```

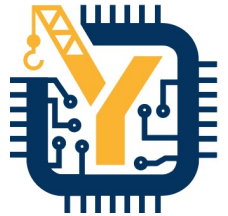
Quick Summary

- Steps
  - Add Verilog file
  - Wrap in Chisel module
  - Connect to Chipyard+Chisel IP
- Automatically works with VLSI and FireSim simulation! Coming up soon!

https://chipyard.readthedocs.io/en/1.5.0/Customization/Incorporating-Verilog-Blocks.html

Covers what we did in this section and more!

That’s it!



Coming up after the coffee break…

Using the Hammer VLSI Flow