Constructive Computer Architecture

Tutorial 4: SMIPS on FPGA

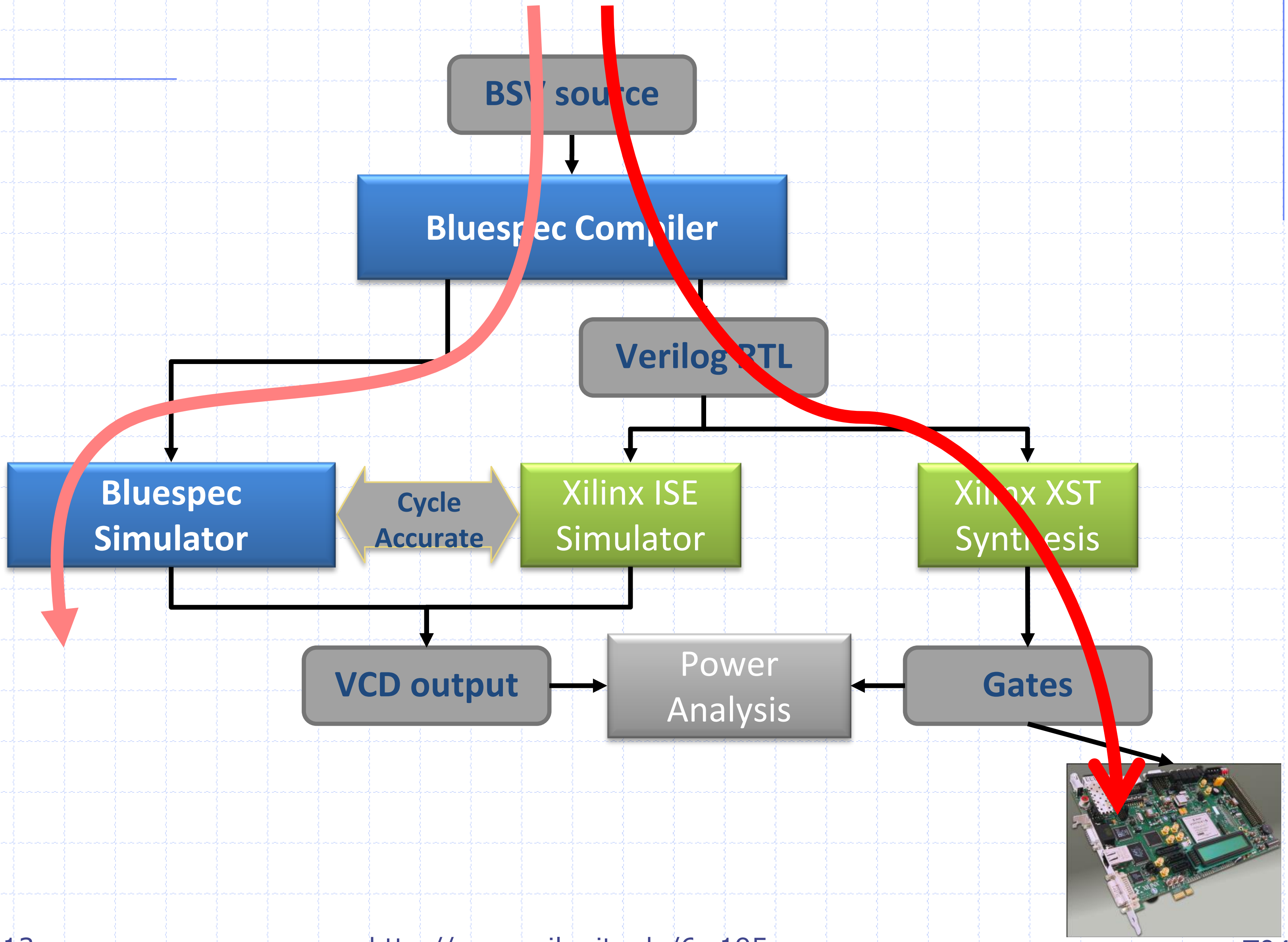
Andy Wright
6.S195 TA

Introduction

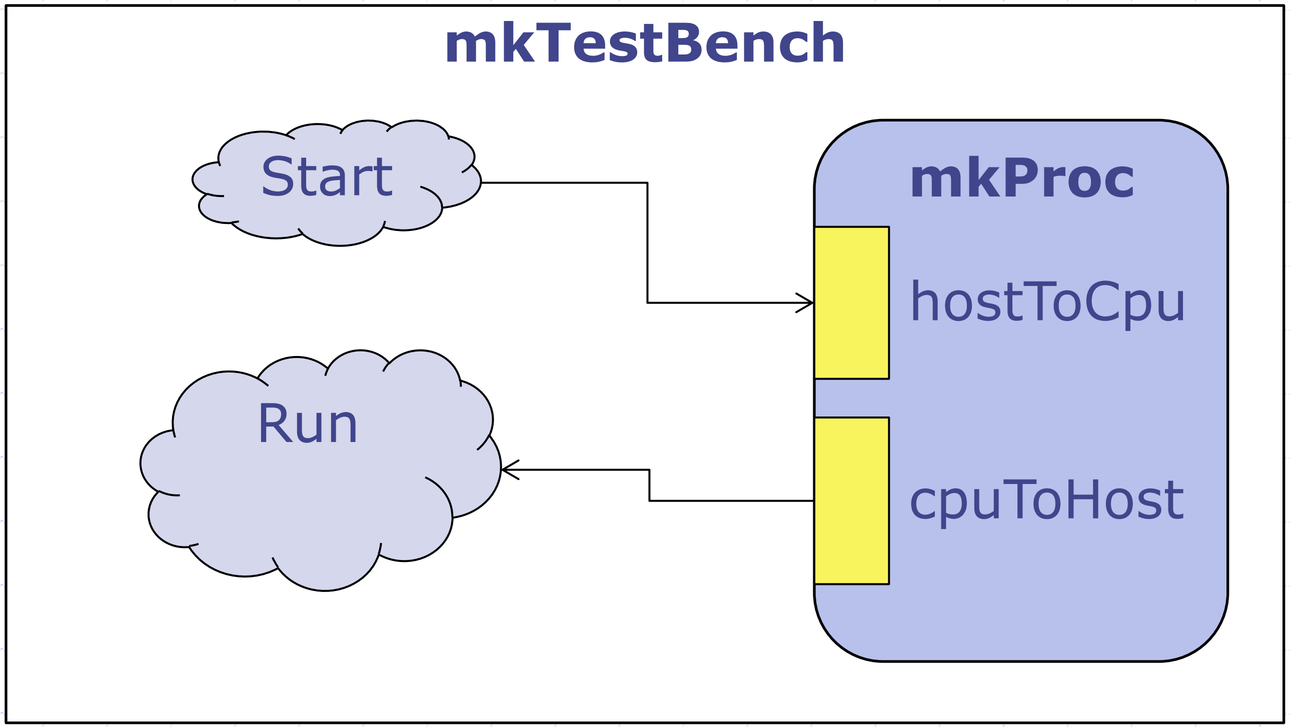
- Field programmable gate arrays (FPGAs) are chips filled with fine grained configurable hardware
  - Can think of it as logic gates that you can connect together
- They are a prototyping tool in ASIC design flow
- They are the end-target for many designs
  - Low volume custom hardware
- They are also very useful for simulation

Later labs will use FPGA tools to compile Bluespec code for an FPGA

BSV Design Flow

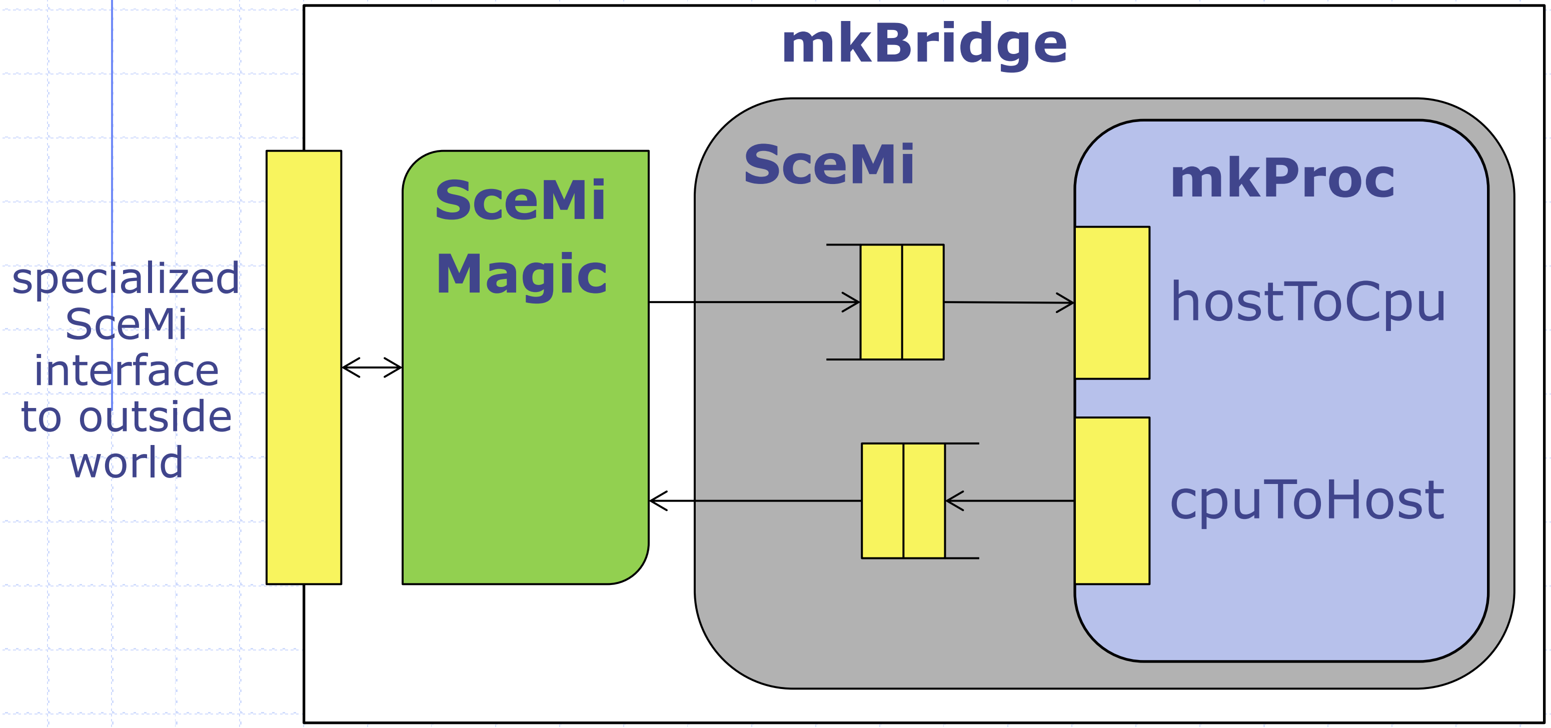


Original Test Bench Setup



- This setup isn’t suited to test a design on an FPGA.
- None of the output from the test can be seen by the user since all $… commands are ignored during FPGA synthesis

SceMi Infrastructure



- The SceMi infrastructure sets up a way to access interface methods of a DUT through a common connection to the outside world

Sample Scemi Interfaces

- TCP – for bsim simulations of scemi connection
- XUPV5

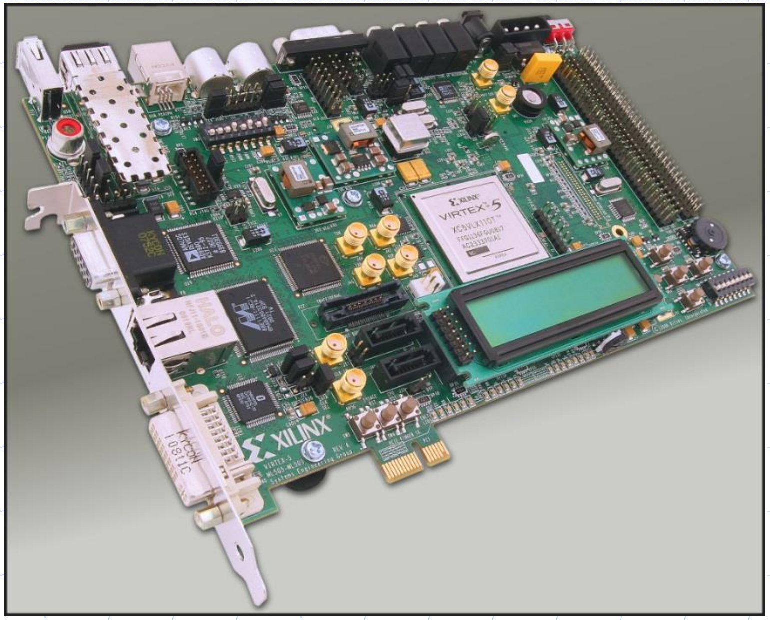
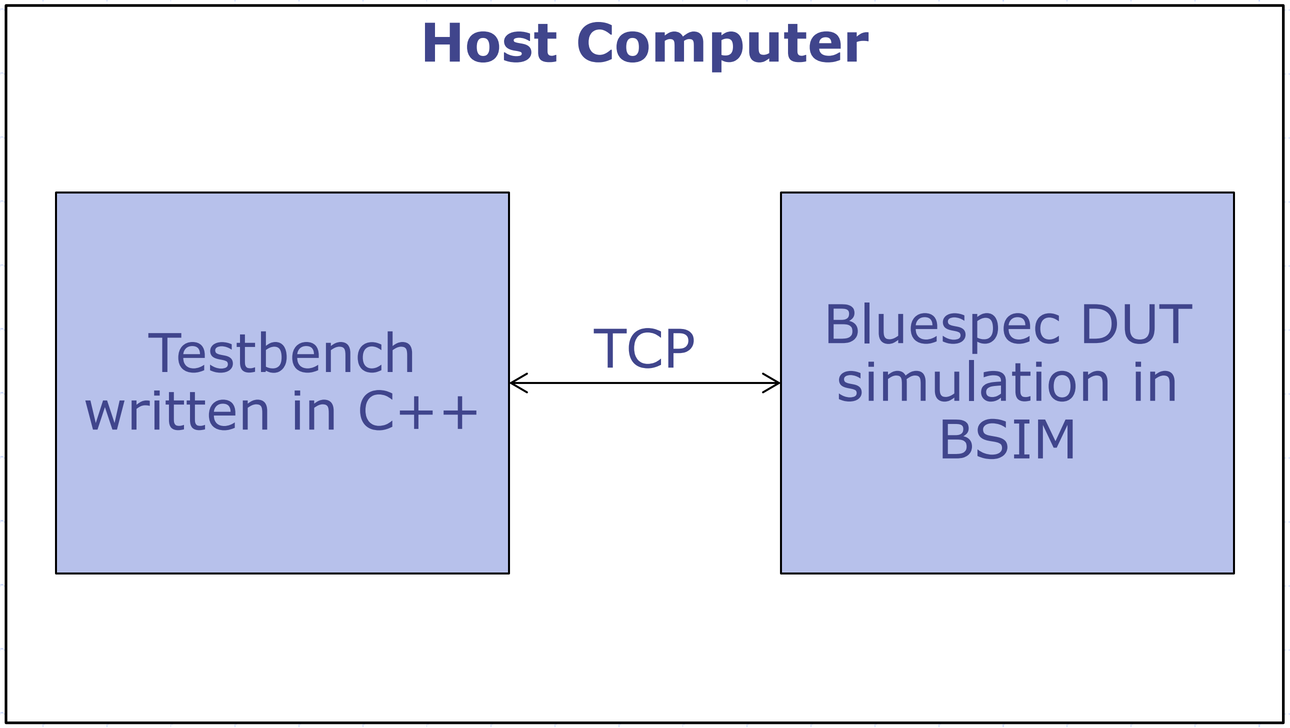
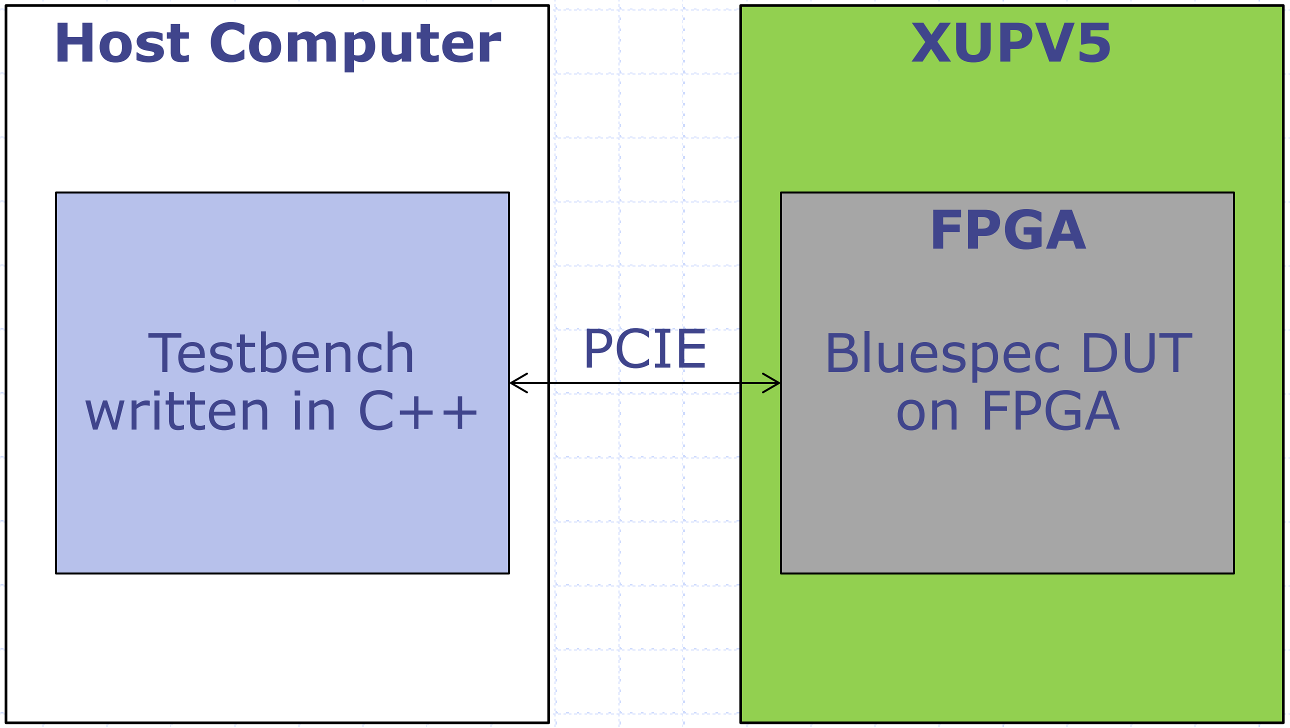


Image of XUPV5 from Xilinx Inc.
http://www.xilinx.com/univ/xupv5-lx110t.htm

Scemi using TCP



Scemi using XUPV5



Same testbench from the TCP example

TCP vs XUPV5

- TCP
  - + Can see output from $… commands
  - + Don’t have to wait for FPGA compilation
  - - Not testing hardware
- XUPV5
  - + Actually running Bluespec code on FPGA
  - - More things can go wrong
  - - Harder to Debug
  - + Faster simulation!

Filter:

SMIPS: 867329 cycles

CPU sim: ~10 seconds

FPGA: ~25 ms

Debugging on FPGA

- You can't see the effects of:
  - $display
    - Used to give basic debug information from processor in simulation
  - $fwrite
    - Used to give essential output from the simulation
  - $finish
    - Used to stop the processor early on error
- All output seen by the user must pass through the SceMi Interface
  - The current setup only allows for print statements from SMIPS programs and PASSED/FAILED output
  - An improved interface could give more feedback

Old Processor Interface

```
interface Proc;
    method ActionValue#(...) cpuToHost;
    method Action hostToCpu(Addr startpc);
endinterface
```

Old C++ Test Bench

- Look at Run.cpp
- Look at function that prints to stderr from cpuToHost

Interface Improvements

- GDB Inspired
  - More program flow control:
    - start, step, stop
    - read PC, write PC
  - Read processor state:
    - Read registers, read memory
  - Read profiling stats
    - cycle count, instruction count

Debug Processor Interface

```
interface ProcDebug;
    method ActionValue#(...) cpuToHost;
    method Action start(Bool ignore);
    method Action step(Data steps);
    method Action stop(Bool ignore);
    method Addr read_pc;
    method Action write_pc(Addr d);
    method Action req_read_rfile(Bit#(5) r);
    method ActionValue#(Data) resp_read_rfile();
    method ActionValue#(Data) read_mem32(Addr addr);
    method Addr read_cycle;
    method Addr read_inst;
endinterface
```

New 1cyc.bsv

Look at 1cyc.bsv

Also includes modified coprocessor

New SceMiLayer.bsv

- Look at SceMiLayer.bsv

New C++ Test Bench

- Look at Run-debug.cpp
- Look at the new functions included for debugging

Output from C++ Test Bench

Look at output.txt

SMIPS Debugger Program

- Give user interactive control over processor on FPGA
  - User can type `start`, `step <n>`, `stop`, and other similar commands
  - They can also choose to get the entire processor state when the processor is stopped
  - This is still a work in progress

Adding breakpoints to SMIPS

- The user needs to be able to specify a PC to stop at
  - New interface method to write breakpoint to processor
- The processor needs to be able to store breakpoints
  - The processor needs a breakpoint register
- The processor needs to stop when that PC has been reached
  - Coprocessor needs to monitor PC and change processor state accordingly

SceMi simulation bugs?

Look at bugs.txt

Cycle count and instruction count don't match for print and filter when running 1cyc.bsv

This is not a bug, the coprocessor fifo is getting filled because the C++ test bench is slower at dequeuing from the fifo than the BSV test bench. The filled coprocessor fifo is forcing the processor to stall.

Conclusion

- FPGA simulation is harder, but SceMi's interfaces make it easier.
- The TCP SceMi interface allows for testing software written for FPGA simulation without using the FPGA.
- More advanced debugging techniques can be used by adding to the processor interface.