154B Discussion 9

March 8th, 2023

Outline

- Intro to Assignment 5
- Quiz 9

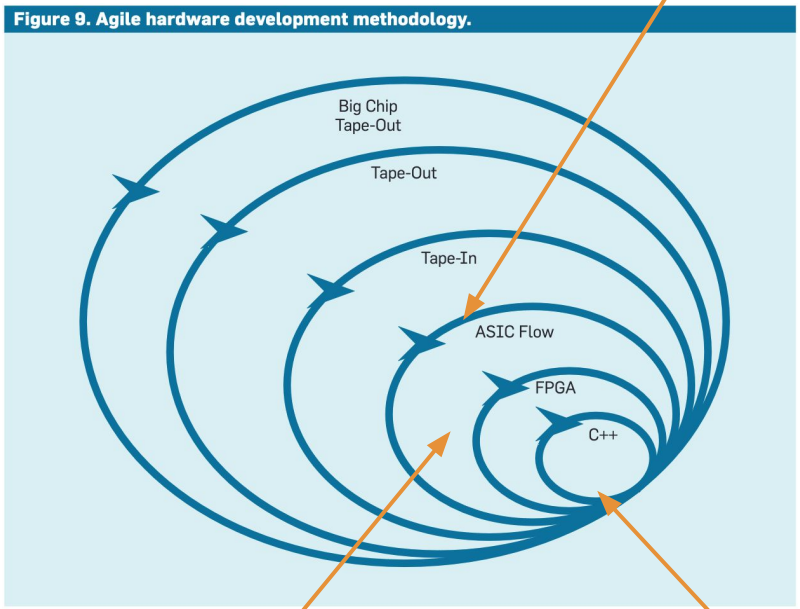
Assignment 5

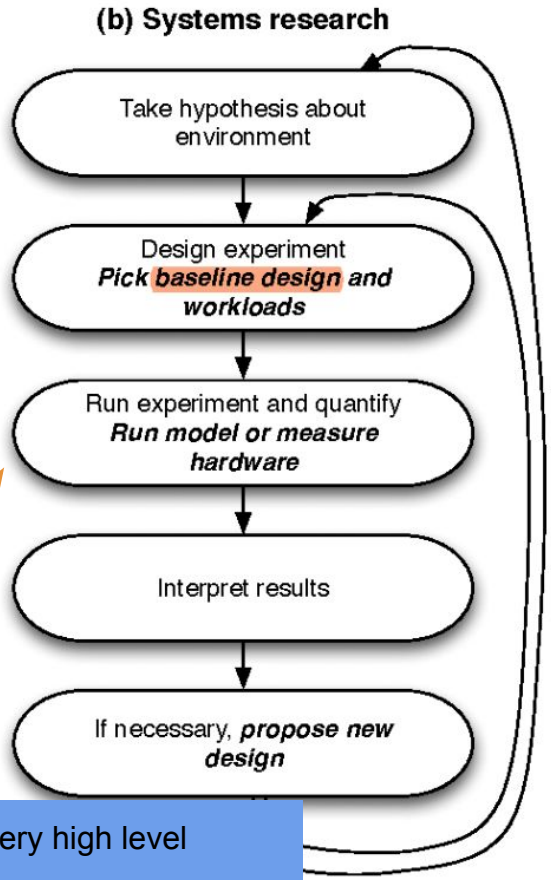
- Multiple-core system using gem5 simulator.
- Benchmarks: different ways of doing summation of an array using multi-threading.

A bigger picture

Physical placement of a chip

(a) Industry





Chisel: called RTL, wire-by-wire details, determine design complexity

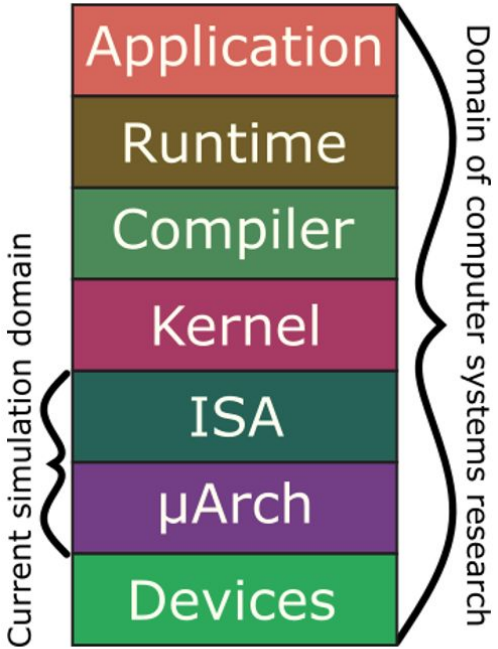
gem5: for very early evaluation, for making very high level design decisions

(a) https://ieeexplore.ieee.org/document/8416813

(b) Computer Architecture Performance Evaluation Methods by Lieven Eeckhout

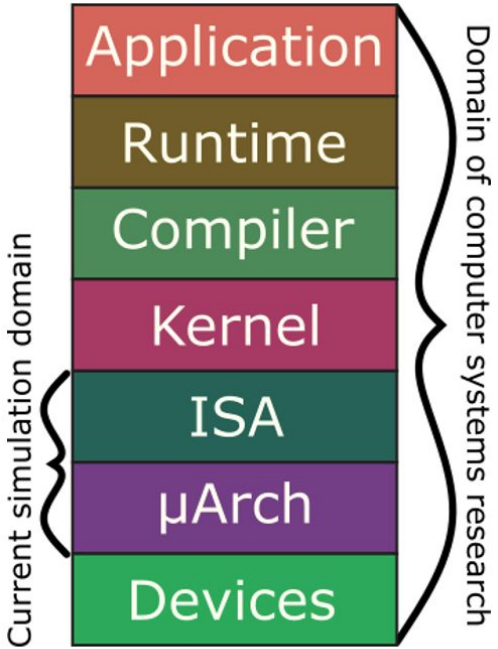
Computer System Development / Research

- Hardware development doesn’t stop at designing an architecture, or designing a micro-architecture.



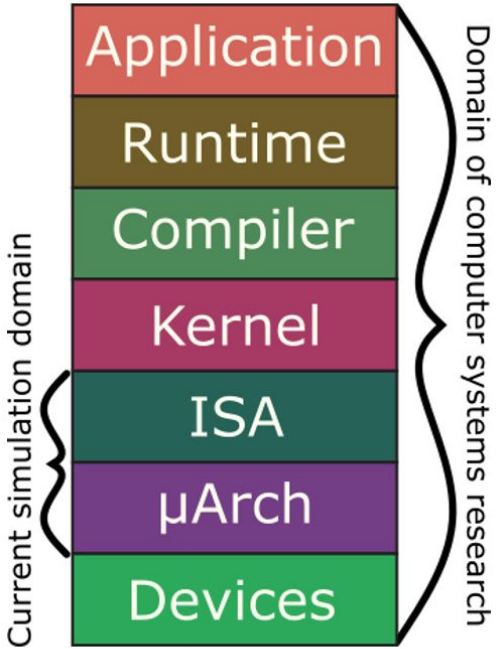
Computer System Development / Research

- Hardware development doesn’t stop at designing an architecture, or designing a micro-architecture.
- New hardware requires support many other levels for compatibility and performance. E.g., a transition from x86 to arm is not straightforward.



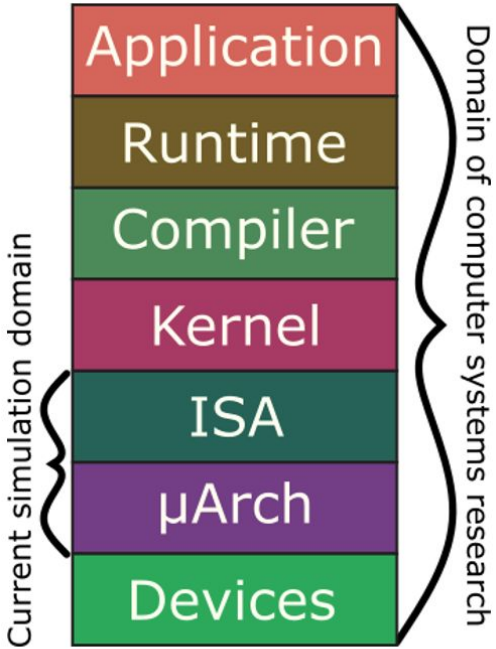
Computer System Development / Research

- Hardware development doesn’t stop at designing an architecture, or designing a micro-architecture.
- New hardware requires support many other levels for compatibility and performance. E.g., a transition from x86 to arm is not straightforward.
  - arm develops architecture/micro-architecture IPs
  - They also have to provide kernel and compiler supports.
  - They also have to provide domain-specific libraries, e.g. math library, to get more performance for important applications.



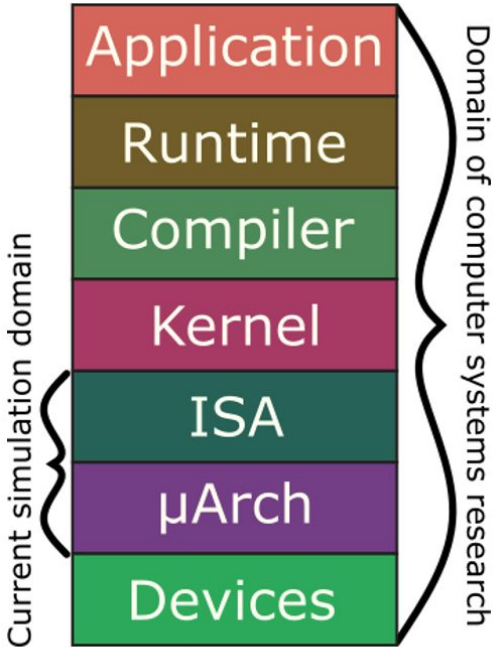
Computer System Development / Research

- Hardware development doesn’t stop at designing an architecture, or designing a micro-architecture.
- New hardware requires support many other levels for compatibility and performance. E.g., a transition from x86 to arm is not straightforward.
  - arm develops architecture/micro-architecture IPs
  - They also have to provide kernel and compiler supports.
  - They also have to provide domain-specific libraries, e.g. math library, to get more performance for important applications.
- RISC-V world lacks for high-performance hardware and tooling, e.g., valgrind.



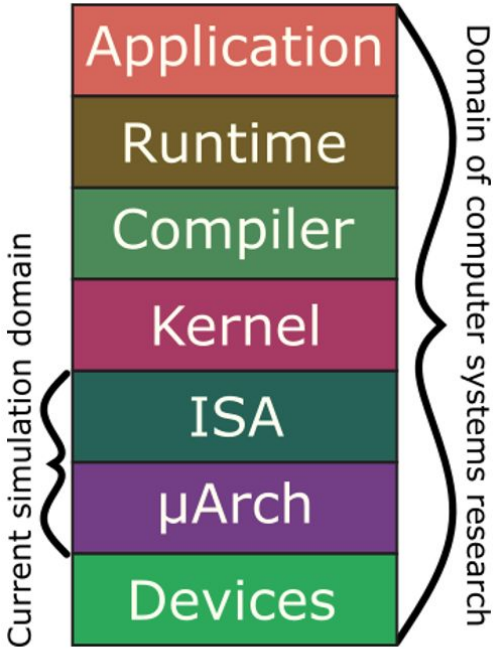
Computer System Development / Research

- Hardware development doesn’t stop at designing an architecture, or designing a micro-architecture.
- New hardware requires support many other levels for compatibility and performance. E.g., a transition from x86 to arm is not straightforward.
  - arm develops architecture/micro-architecture IPs
  - They also have to provide kernel and compiler supports.
  - They also have to provide domain-specific libraries, e.g. math library, to get more performance for important applications.
- RISC-V world lacks for high-performance hardware and tooling, e.g., valgrind.
- Software development is also expensive.



Computer System Development / Research

- Later parts of the development process are increasingly costly.
- First things first:
  - Must be correct!
  - New architectures must enable microarchitectures to achieve performance and efficiency.
- Needs something that does not take years to develop a new design.



Simulators

- Simulator is an economy of verifying something that does not physically exist.
- Every simulator has its own purposes.

Every simulator has its own purposes

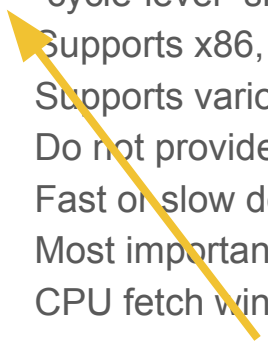
- QEMU, Spike: called functional simulator, focus on emulating *architectures*, do not focus on providing microarchitecture details.
  - Do not provide microarchitecture details, e.g., cache hits, cache accesses.
  - Very fast.

Every simulator has its own purposes

- QEMU, Spike: called functional simulator, focus on emulating *architectures*, do not focus on providing microarchitecture details.
- gem5: architecture + micro-architecuture simulator, supports fewer architectures, provides a lot more high-level microarchitecture details.
  - Supports x86, ARM, RISC-V.
  - Supports various CPU designs: high-performance in-order, and out-of-order CPUs.
  - Do not provide RTL details, i.e., no physical design complexity details.
  - Fast or slow depending on micro-architecture details.

Every simulator has its own purposes

- QEMU, Spike: called functional simulator, focus on emulating *architectures*, do not focus on providing microarchitecture details.
- gem5: architecture + micro-architecuture simulator, supports fewer architectures, provides a lot more high-level microarchitecture details.
  - “cycle-level” simulator.
  - Supports x86, ARM, RISC-V.
  - Supports various CPU designs: high-performance in-order, and out-of-order CPUs.
  - Do not provide RTL details, i.e., no physical design complexity details.
  - Fast or slow depending on the level micro-architecture details.
  - Most importantly: can easily change design parameters easily! (e.g., cache size, associativity, CPU fetch window, CPU execute width, etc.)



Caveats: not as fast as qemu, not as accurate as RTL, but we can get useful information on high-level micro-architecture details.

Every simulator has its own purposes

- QEMU, Spike: called functional simulator, focus on emulating *architectures*, do not focus on providing microarchitecture details.
- gem5: architecture + high-level micro-architecuture simulator, supports fewer architectures, provides a lot more high-level microarchitecture details.
- Chisel, Verilog: called HDL, provides plenty of micro-architecture information on physical design complexity.
  - “cycle-accurate” simulator.
  - Very slow.

gem5 Architecture

- Backend: C++ Simulation Objects.
- Frontend: Python script, specifying how simulation objects are connected, object parameters.

gem5 Architecture

- Backend: C++ Simulation Objects.
- Frontend: Python script, specifying how simulation objects are connected, object parameters.
- For assignment 5, you'll only need to write python script(s).

gem5 Architecture

- Input to gem5: a python script.
- Output of gem5: inside the m5out folder.
  - config.ini: how everything are connected.
  - stats.txt: micro-architecture statistics.
- Running simulation

```
gem5 my_system.py
```

Hello world script

```
if __name__ == "__m5_main__":
    print("hello")
```

Importing Objects from the Backend

```
from components.boards import HW5X86Board
from components.cache_hierarchies import HW5MESITwoLevelCacheHierarchy
...
```

Specifying Objects

```
from components.boards import HW5X86Board
from components.cache_hierarchies import HW5MESITwoLevelCacheHierarchy
...

my_cache = HW5MESITwoLevelCacheHierarchy()
my_board = HW5X86Board(clk_freq = ...,
                       processor = ...,
                       memory = ...,
                       cache_hierarchy = my_cache)
```

xbar_latency = 1

workload = my_board.set_workload(workload)

Running the Simulation

```
simulator = Simulator(board=my_board,
                      full_system=False,
                      on_exit_event=exit_event_handler)
simulator.run()
```

Hints

- Optionally, you can use the Python's argparse library to parameterize the workload, e.g.

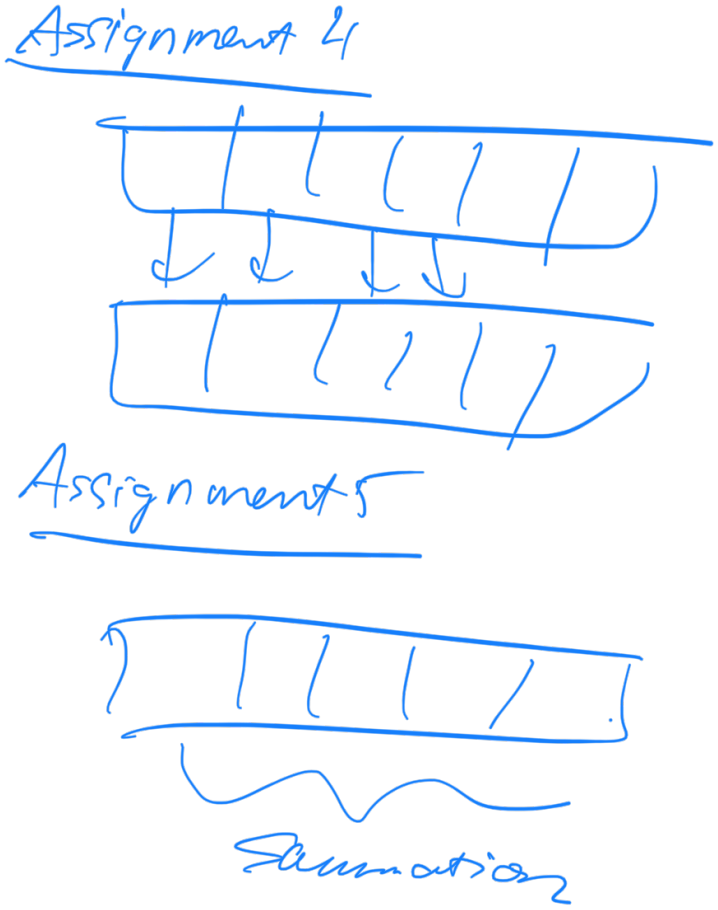
```
gem5 my_system.py --workload=1
```

Assignment 5: Approach

- You can start from 201A's assignment 0.
- What do we learn from assignment 4?

→ think about spatial/ locality temporal/ ...

→ assignment 4:
cache block size is 8 bytes
element size is 2 bytes
assignment 5:
cache block size is 64 bytes
element size is 8 bytes

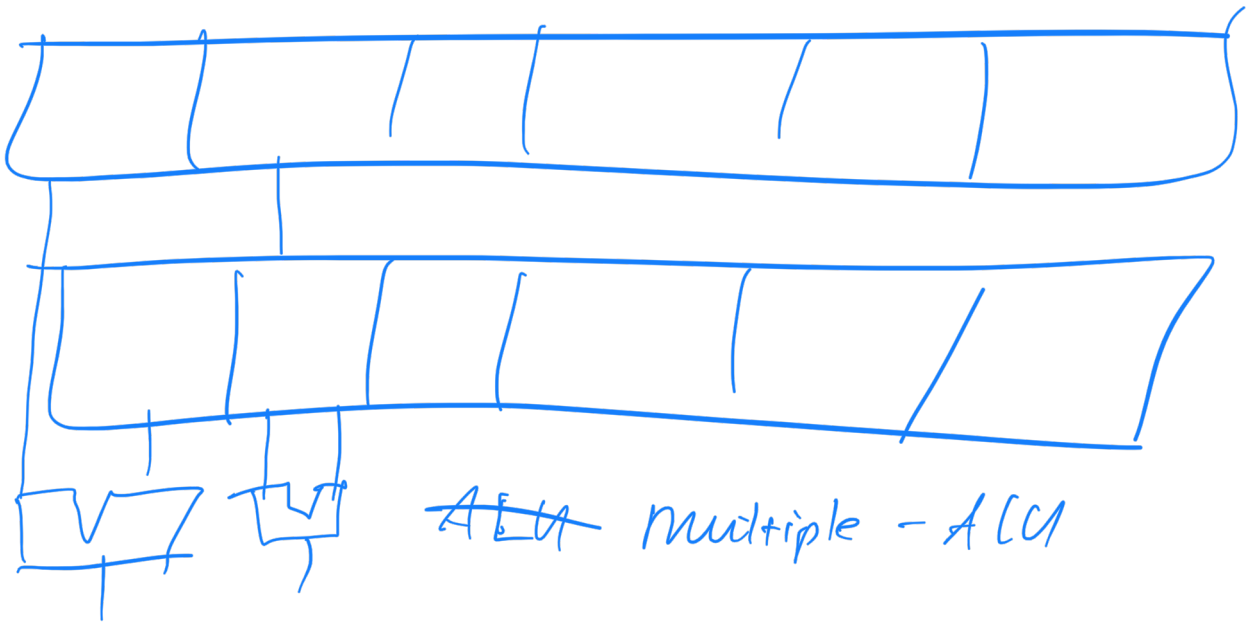


Assignment 5: Approach

Quiz 9

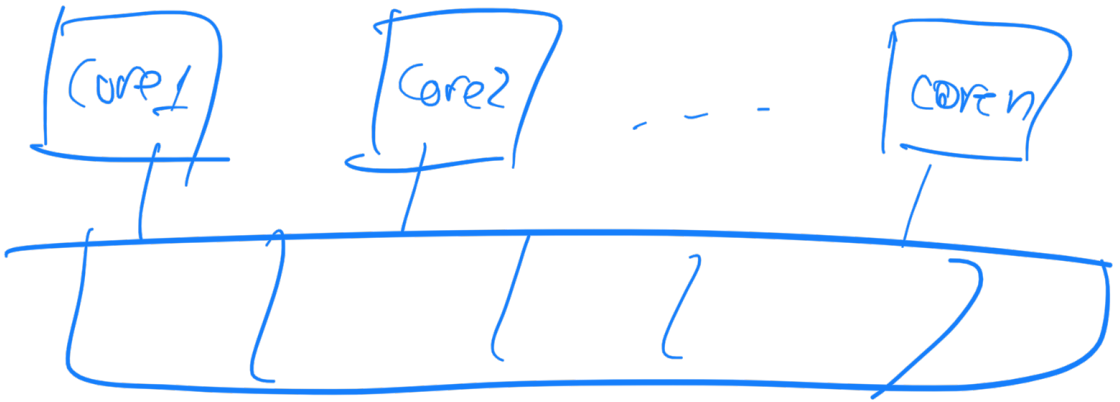
SIMD

1 instruction process multiple data



Quiz 9

MIMD:
has to handle multiple instruction streams / execution contexts



Quiz 9

time = Amdahl's law $= 0.25 + \frac{0.75}{\#\text{cores}}$

power = CV^2f

energy = power x time