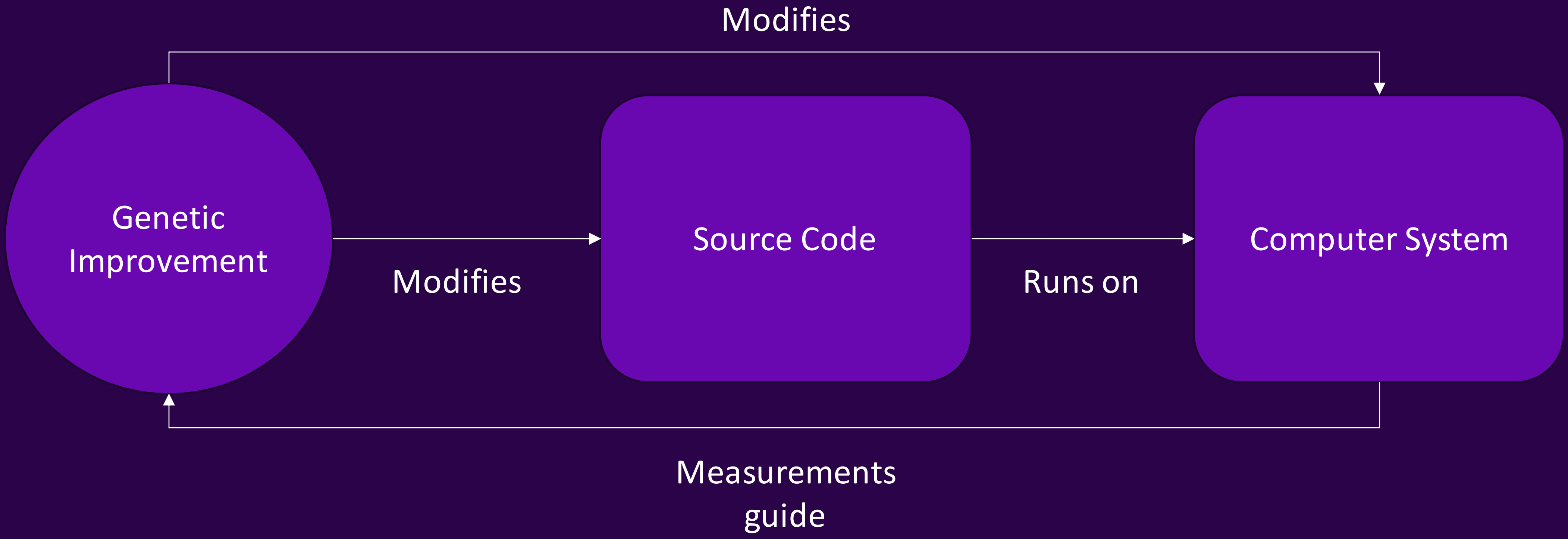
AUTOMATICALLY EXPLORING COMPUTER SYSTEM DESIGN SPACES

Bobby R. Bruce

THE GENERAL IDEA



THE GENERAL IDEA

Genetic Improvement

Typically, in GI we optimize software with respect to a static target computer system.

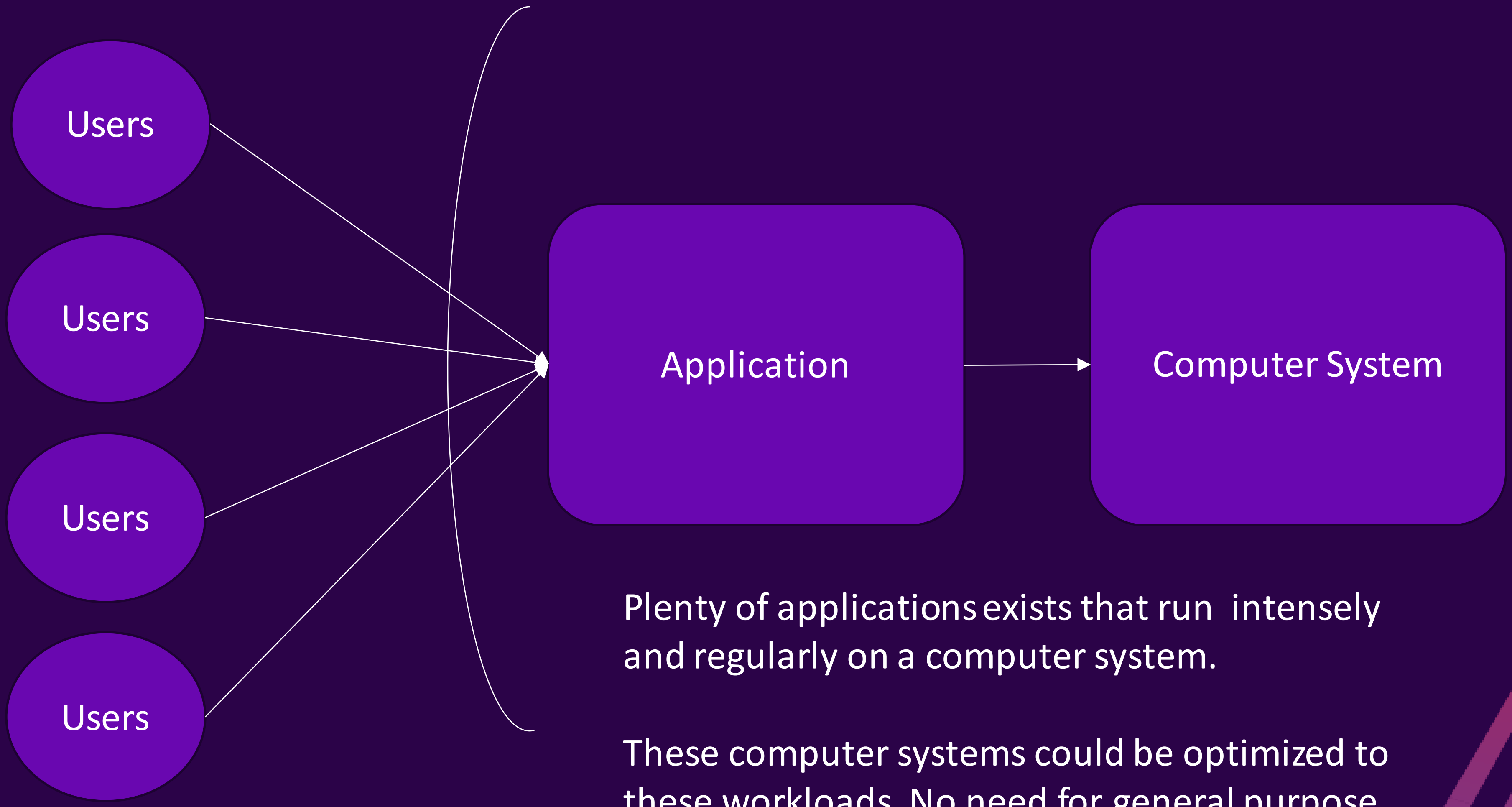
THE GENERAL IDEA

Genetic Improvement

We *could* also use GI to optimize a computer system with respect to a static software target

Isn't source-code the definition of computer system behavior?

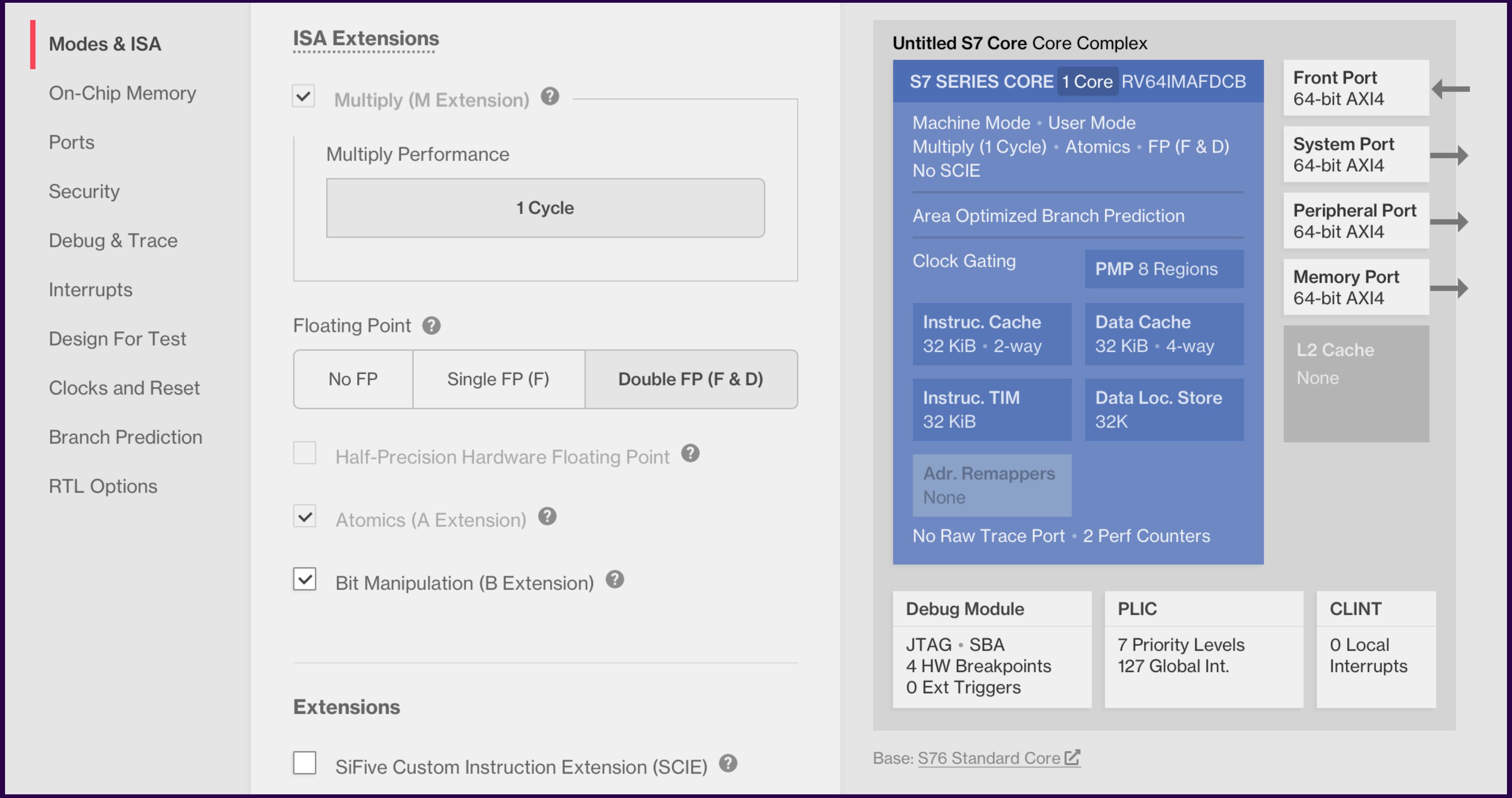
A SIMPLE MOTIVATING EXAMPLE



Plenty of applications exists that run intensely and regularly on a computer system.

These computer systems could be optimized to these workloads. No need for general purpose systems. They could be specialized.

OTHER MOTIVATIONS



Modes & ISA
On-Chip Memory
Ports
Security
Debug & Trace
Interrupts
Design For Test
Clocks and Reset
Branch Prediction
RTL Options

ISA Extensions

☑ Multiply (M Extension)

Multiply Performance: 1 Cycle

Floating Point: No FP | Single FP (F) | [✓ Double FP (F & D)]

☐ Half-Precision Hardware Floating Point

☑ Atomics (A Extension)

☑ Bit Manipulation (B Extension)

Extensions

☐ SiFive Custom Instruction Extension (SCIE)

Untitled S7 Core Core Complex

S7 SERIES CORE 1 Core RV64IMAFDCB

Machine Mode • User Mode
Multiply (1 Cycle) • Atomics • FP (F & D)
No SCIE

Area Optimized Branch Prediction

Clock Gating

PMP 8 Regions

Instruc. Cache 32 KiB • 2-way

Data Cache 32 KiB • 4-way

Instruc. TIM 32 KiB

Data Loc. Store 32K

Adr. Remappers None

No Raw Trace Port • 2 Perf Counters

Front Port 64-bit AXI4
System Port 64-bit AXI4
Peripheral Port 64-bit AXI4
Memory Port 64-bit AXI4
L2 Cache None

Debug Module: JTAG • SBA, 4 HW Breakpoints, 0 Ext Triggers

PLIC: 7 Priority Levels, 127 Global Int.

CLINT: 0 Local Interrupts

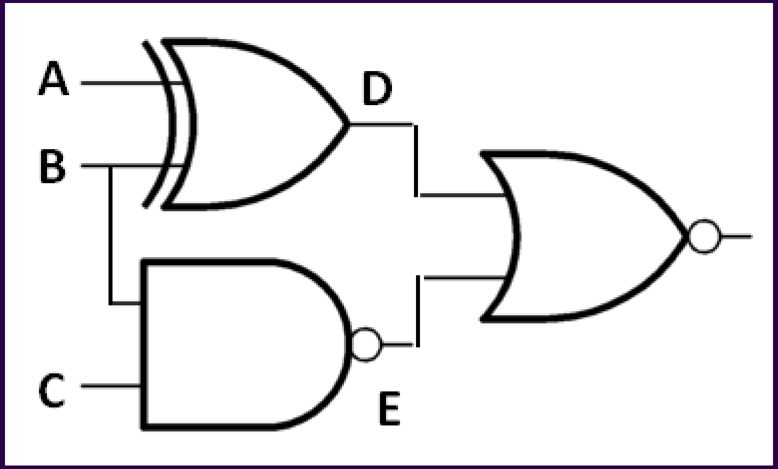
Base: S76 Standard Core

This is the SiFive Core Design Tool

Open Architecture projects such as RISC-V are making this easier.

Costs of silicon customization, “design to tape-out” are dropping dramatically

AT WHAT LEVEL SHOULD WE WORK?

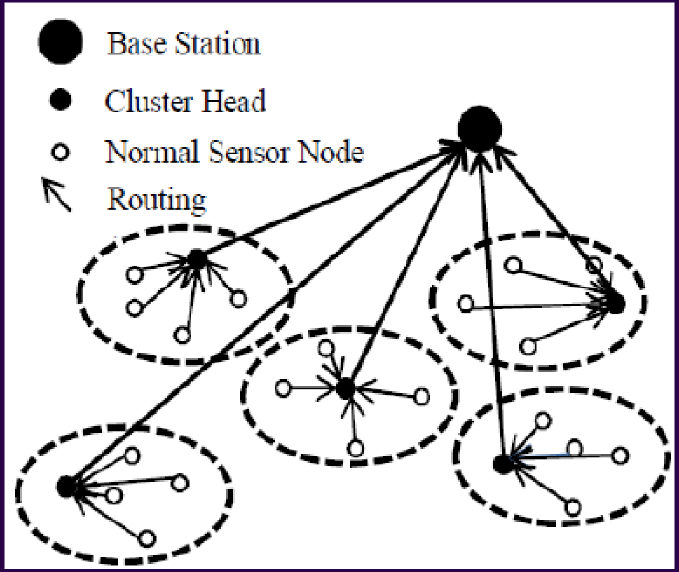


Logic gates, transistors, circuit-level?

A few cons:

- It’s been done!
- Run into scalability problems
- Customizing at this level is very expensive

AT WHAT LEVEL SHOULD WE WORK?



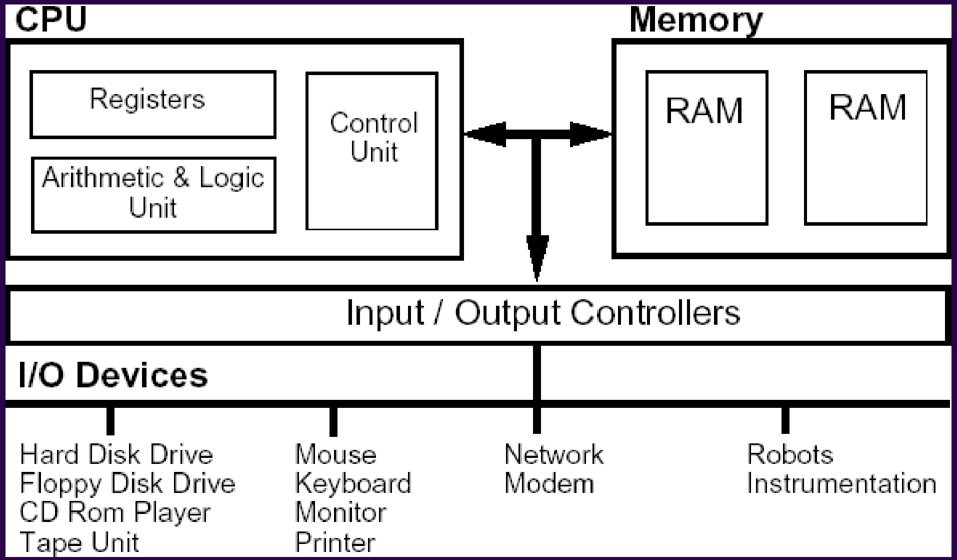
<u>Clusters, computer networks, etc.?</u>

A few cons:

- Difficult to simulate workloads sufficiently for optimization

- (I’m not really all that interested!)

AT WHAT LEVEL SHOULD WE WORK?



Computer architecture?

Some Pros:

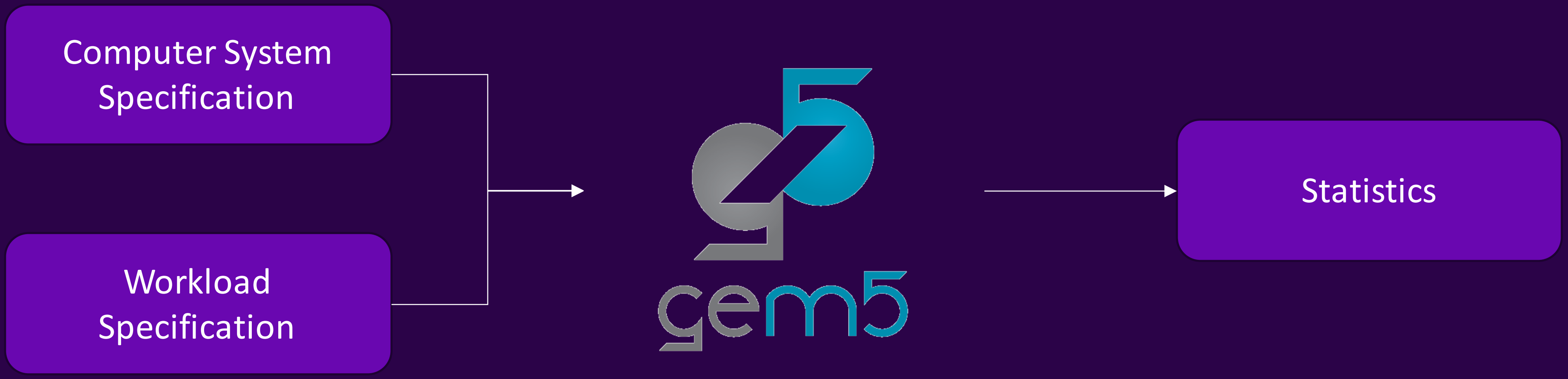
- Common APIs and standards allow for interchanging components (think genes and alleles)
- We can utilize off the shelf components and designs.
- Plenty of research opportunity --- computer architecture designing and optimization is very manual.

HOW DO WE EVALUATE ARCHITECTURE DESIGNS?

SIMULATION

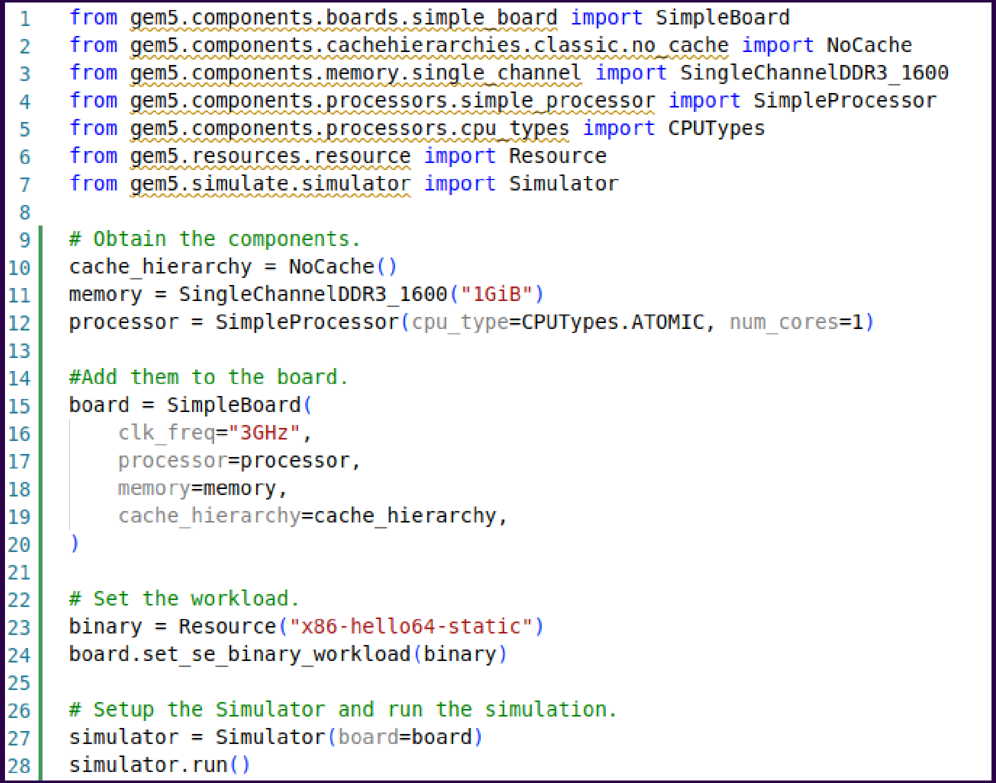
MODERN SIMULATORS

They provide a language for GI to modify



MODERN SIMULATORS

That language follows a grammar like any other



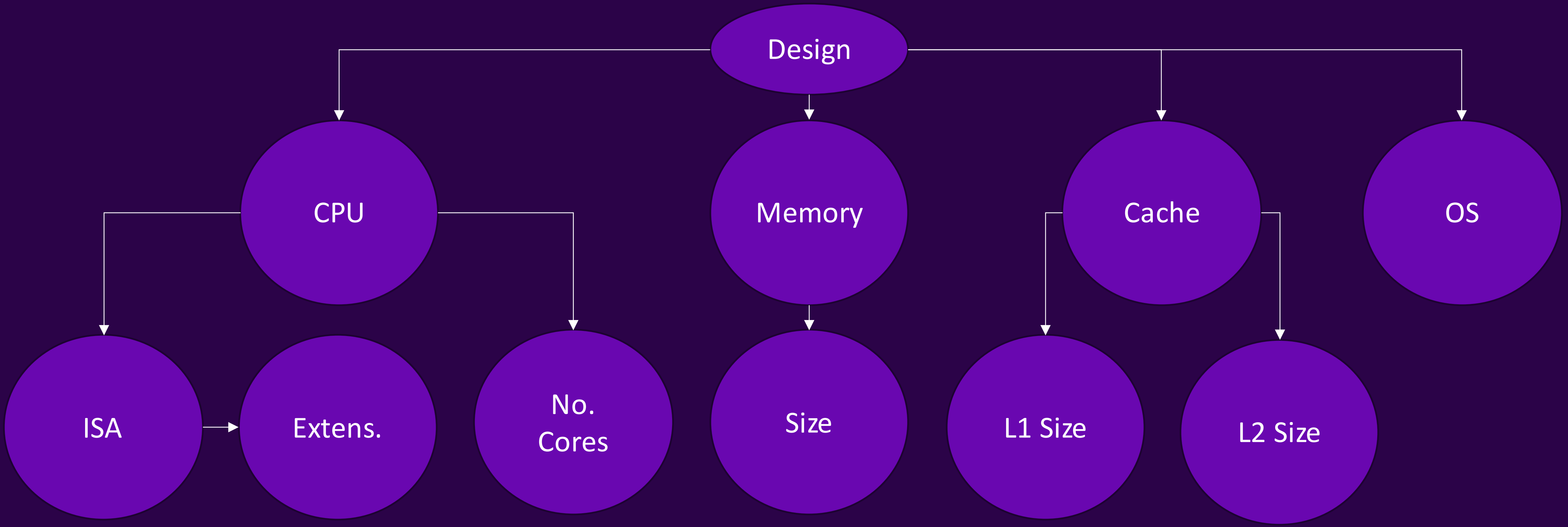
```python
from gem5.components.boards.simple_board import SimpleBoard
from gem5.components.cachehierarchies.classic.no_cache import NoCache
from gem5.components.memory.single_channel import SingleChannelDDR3_1600
from gem5.components.processors.simple_processor import SimpleProcessor
from gem5.components.processors.cpu_types import CPUTypes
from gem5.resources.resource import Resource
from gem5.simulate.simulator import Simulator

# Obtain the components.
cache_hierarchy = NoCache()
memory = SingleChannelDDR3_1600("1GiB")
processor = SimpleProcessor(cpu_type=CPUTypes.ATOMIC, num_cores=1)

#Add them to the board.
board = SimpleBoard(
    clk_freq="3GHz",
    processor=processor,
    memory=memory,
    cache_hierarchy=cache_hierarchy,
)

# Set the workload.
binary = Resource("x86-hello64-static")
board.set_se_binary_workload(binary)

# Setup the Simulator and run the simulation.
simulator = Simulator(board=board)
simulator.run()
```

The “variables” in a design are the components and their proporties

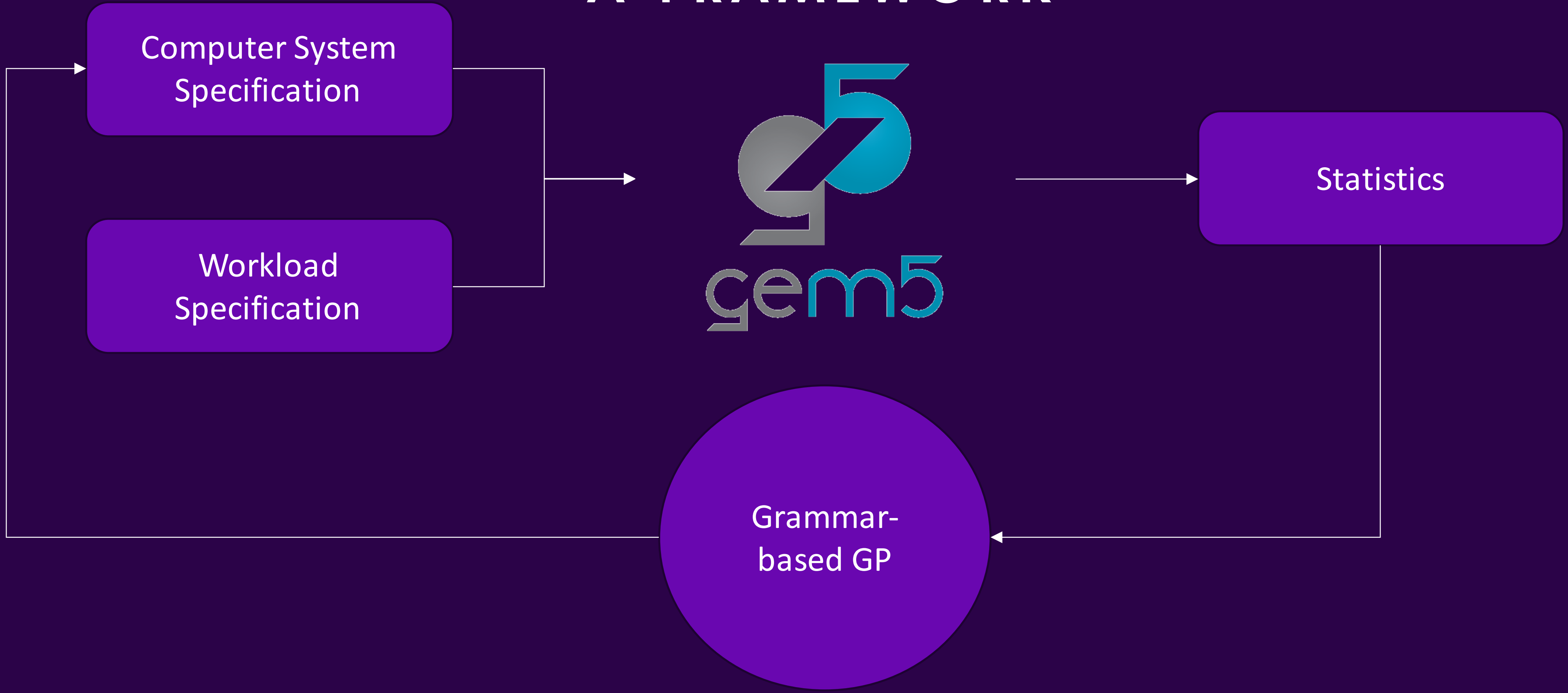
For example, we can swap out the memory system with a different type (single channel or dual) or configure its size

MODERN SIMULATORS

A good target for Grammar-based GP



A FRAMEWORK


Computer System Specification
Workload Specification
gem5
Statistics
Grammar-based GP

WHAT'S NEEDING DONE?

STATS Can we create simulations with good enough fidelity?

ACCURACY What stats do we need from our simulator, what are we optimizing for??

COST MODEL How do we estimate the cost of a design so we can determine the trade off?

SPEED How do we do 1000s of evaluations when 1 can take hours?

BENCHMARKS What workloads should we optimize and are they meaningful?

ANY (NICE) QUESTIONS?

Bobby R. Bruce

bbruce@ucdavis.edu

https://www.bobbybruce.net