Accelerating Simulation via AI-derived Reduced-Order Models

Serge Leef, DARPA/MTO

MODSIM'21 Workshop

October 6, 2021

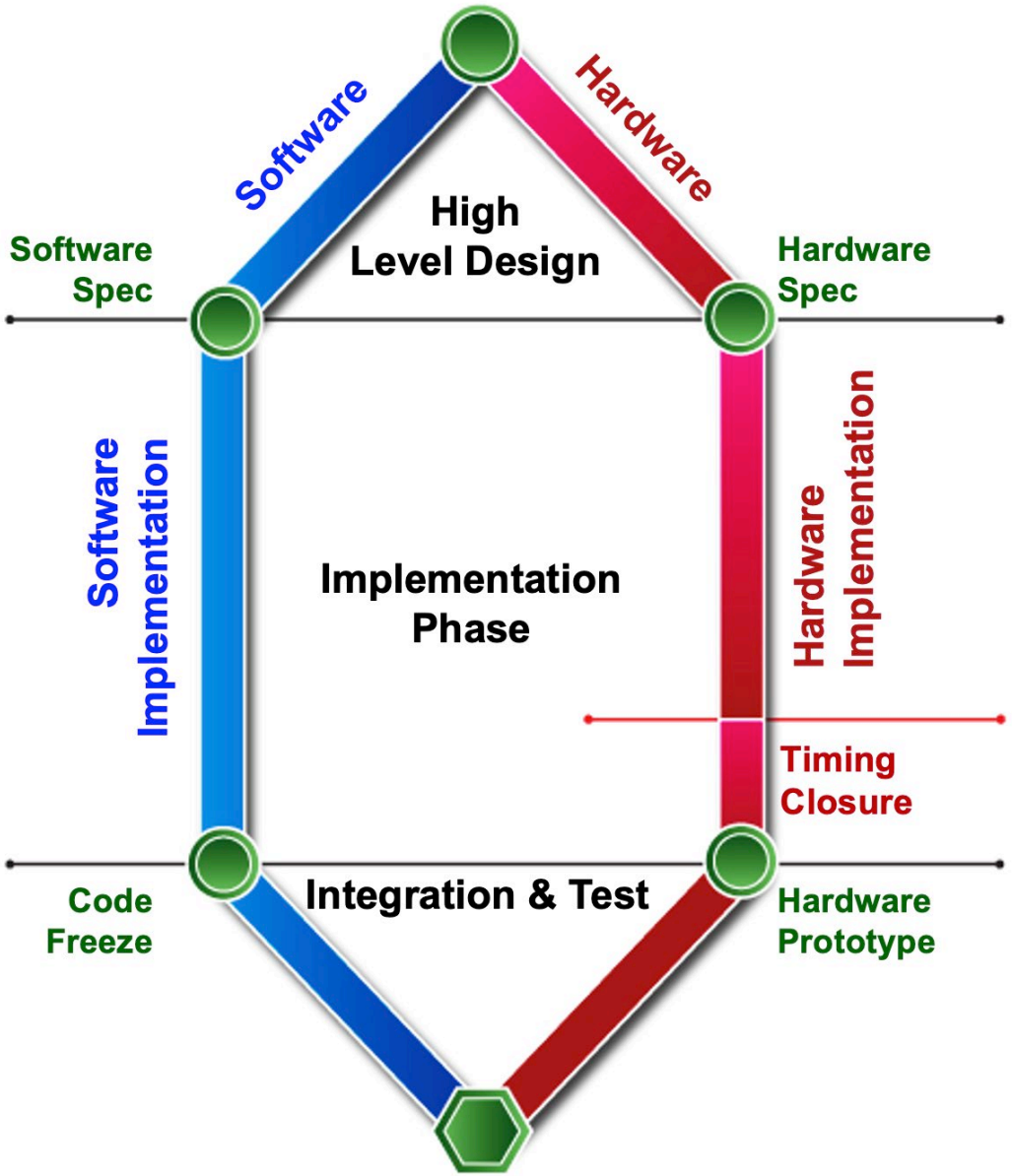
Mr. Serge Leef

Microsystems Technology Office (MTO)

Program Manager

Mr. Serge Leef joined DARPA in August 2018 as a program manager in the Microsystems Technology Office (MTO). His research interests include computer architecture, chip design tools, simulation, synthesis, semiconductor intellectual property (IP), cyber-physical modeling, distributed systems, secure design flows, and supply chain management. He is also interested in the facilitation of startup ecosystems and business aspects of technology.

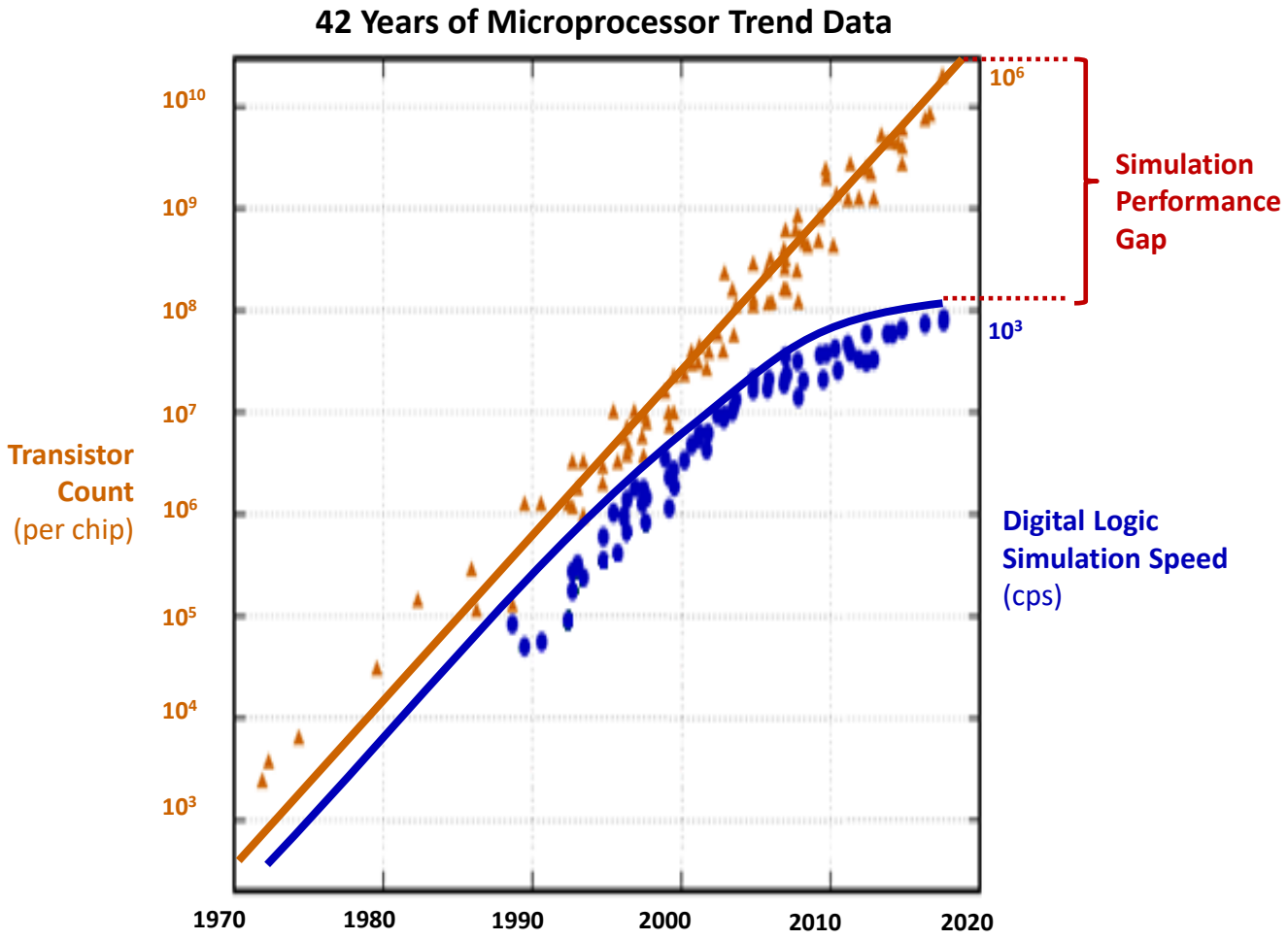
Verification Methodologies Are Severely Strained



- Unpredictable, iterative loop during timing closure and system integration/test phase
- Poor partitioning decisions at the front end of the process are impossible to overcome during the design
- Functional verification is strained by even today's designs; how to verify multi-discipline systems with many billions of gates?
- Even though software is a key, growing system component it is only partially included in hardware verification phases due to insufficient simulation speed
- Interaction with outside world through sensors and actuators is rarely an integral part of the flow

Source: Mentor Graphics, 2009

Enabling Linear Scaling / Hyperscaling of Digital Simulation

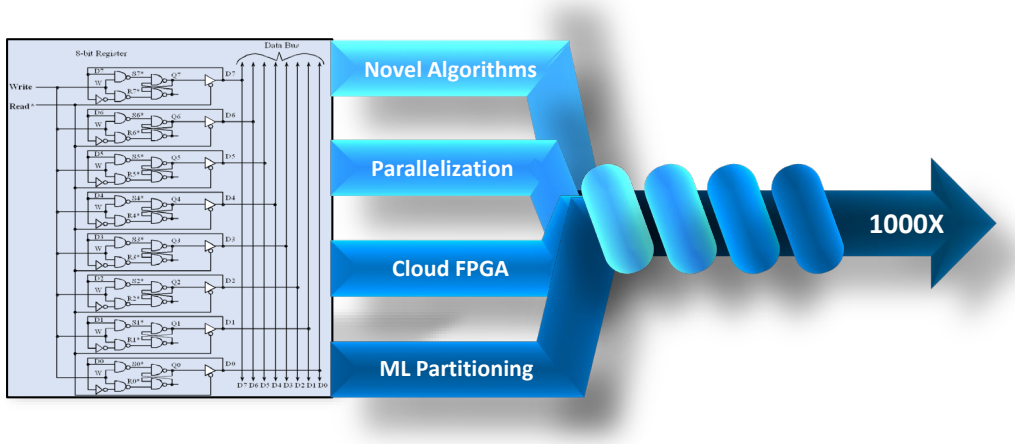


- Chip development timelines are bottlenecked by functional verification, where simulation speed is center-stage
- Simulation speed grew with Moore driven platform performance to ~ 1K cps
- Post-Moore, the architectural direction is multicore/cloud, but modern simulation algorithms are not designed take advantage of distributed computational fabric

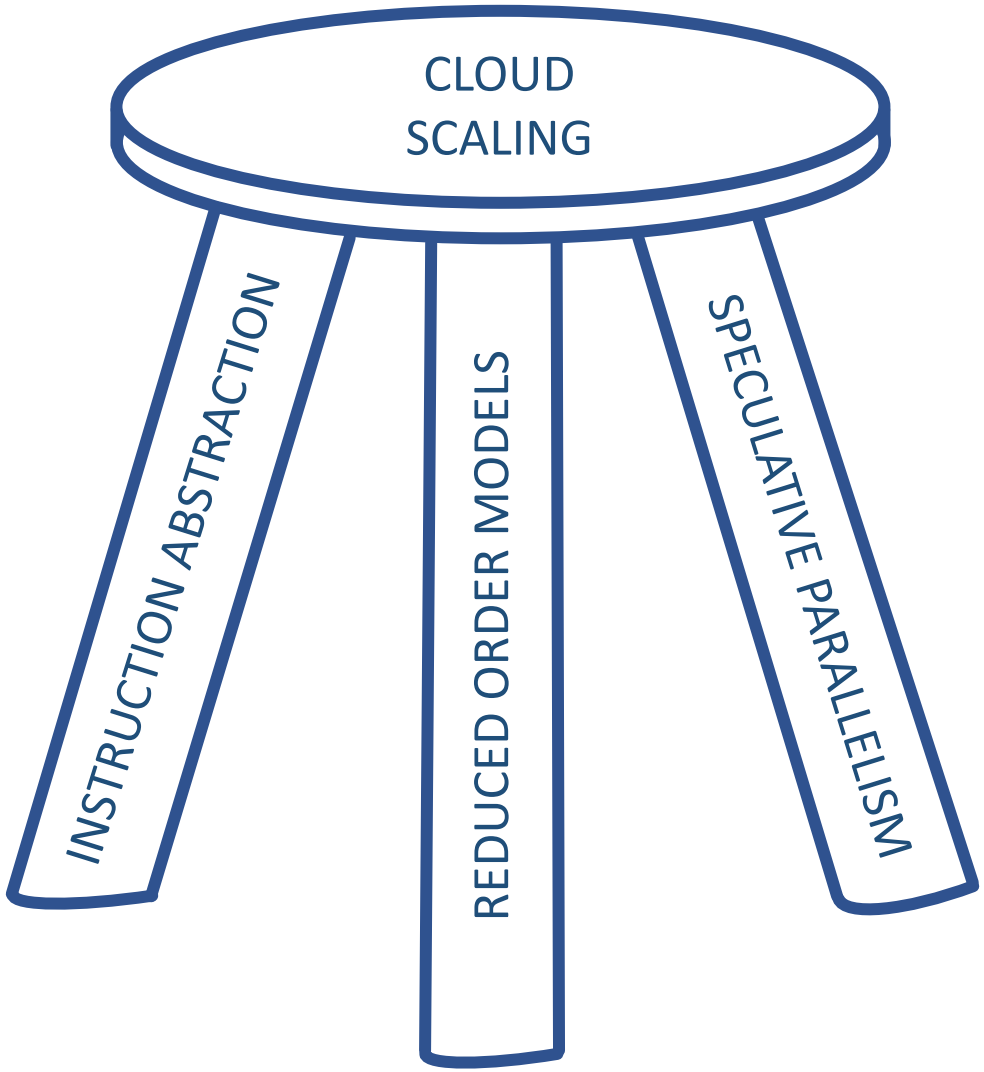
Source: https://github.com/karlrupp/microprocessor-trend-data

Convergence of Technologies to Drive Next-Gen Simulation Advances

- Combine limitless and elastic compute and storage available on the cloud with one or more of the following innovations:
  - Simulation engine novel algorithms
  - Parallel partitioning schemes
  - High-performance-compute (HPC) architectures and programmable cloud based FPGA fabric
  - ML-driven simulation partitioning

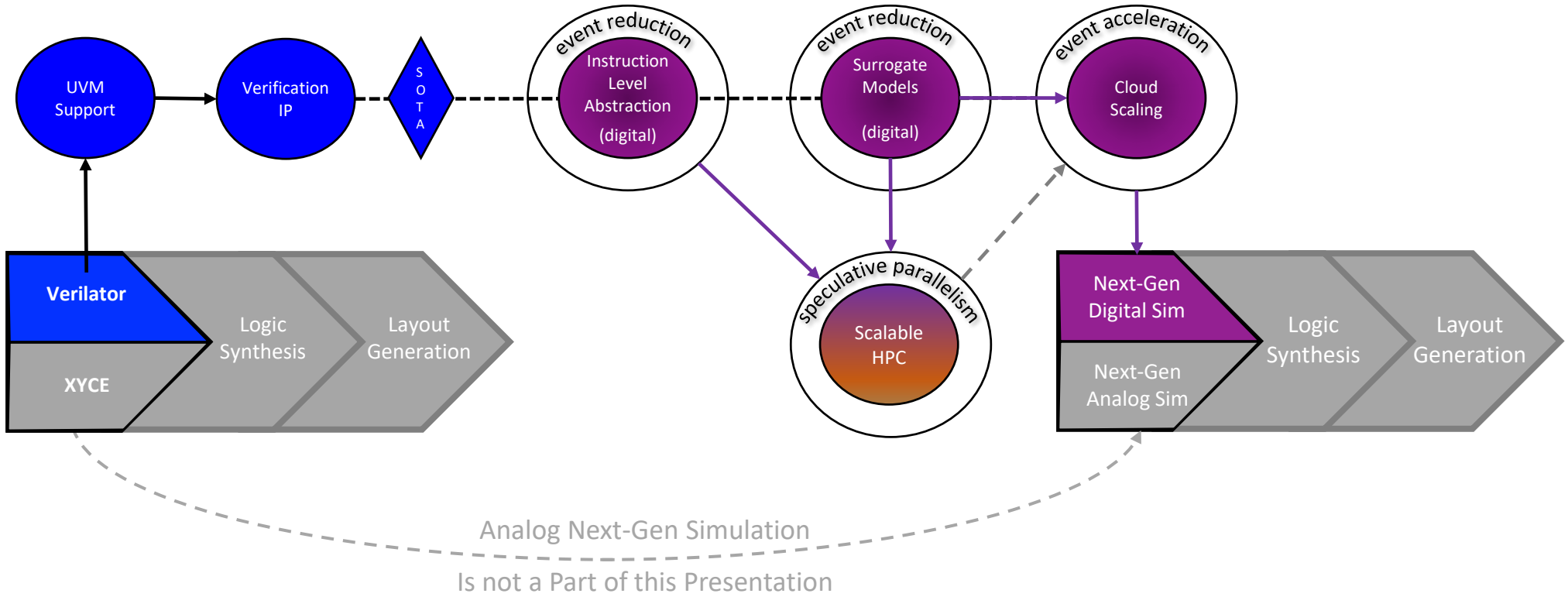


Source: http://users.ece.utexas.edu/~valvano/Volume1/



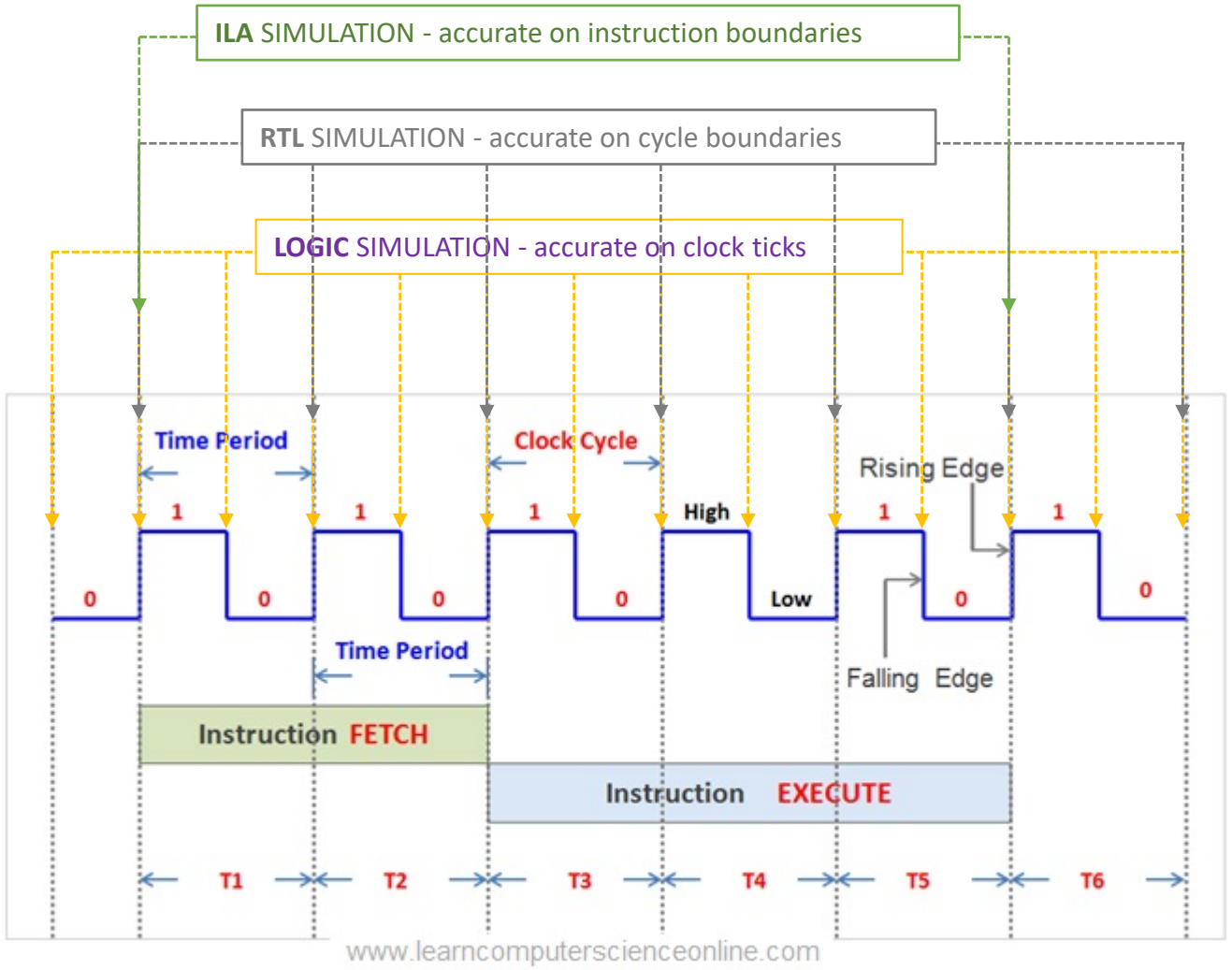
Abstraction + Reduced Order Models + HPC + Cloud Scaling

- Simulation physics limits reached decades ago and rely on Moore's law for speedups
- Parallel simulation has not been re-visited since emergence of the cloud computing
- Emergence of Machine Learning has not been employed to drive creation of faster models
- Need to re-think simulation in light of advances in ILA, ML, HPCs, and Cloud

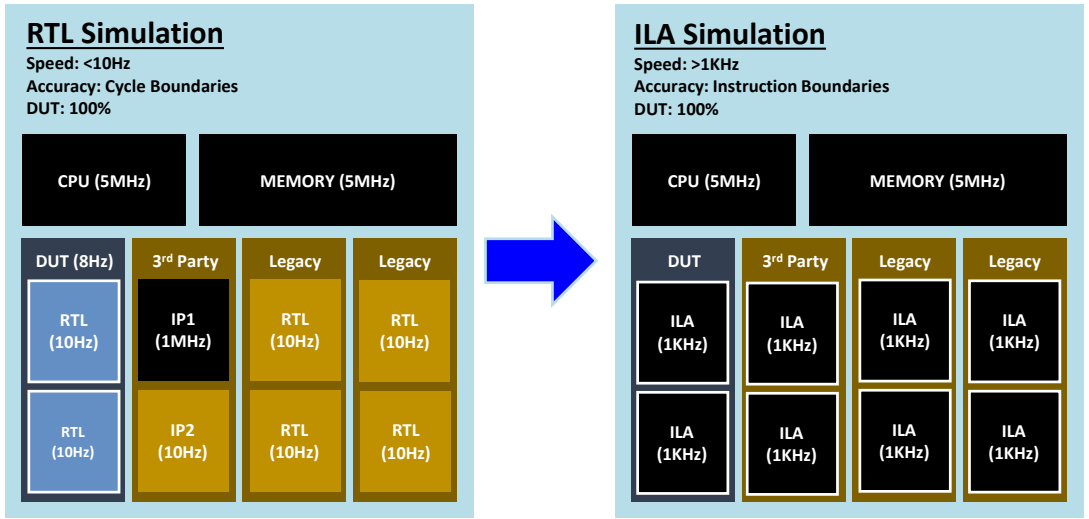


Raising Abstraction to Instruction Level for ALL IP Blocks

- Instruction level simulation is fast and the accuracy is adequate for numerous SoC system simulation use cases

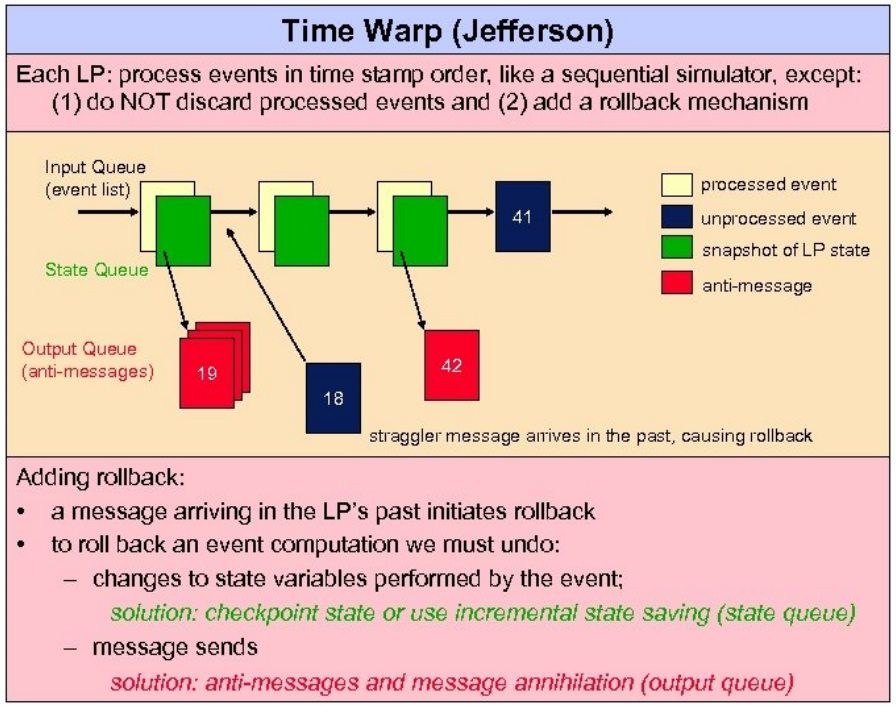


- This approach treats bus-based peripherals (IP) as if they were processors
- Memory mapped communication protocols are modeled as "machine instructions"
- Performance gain of 100X have been demonstrated at the block level
- Raise the abstraction of the entire SoC to the level where instruction based simulation benefits can be realized



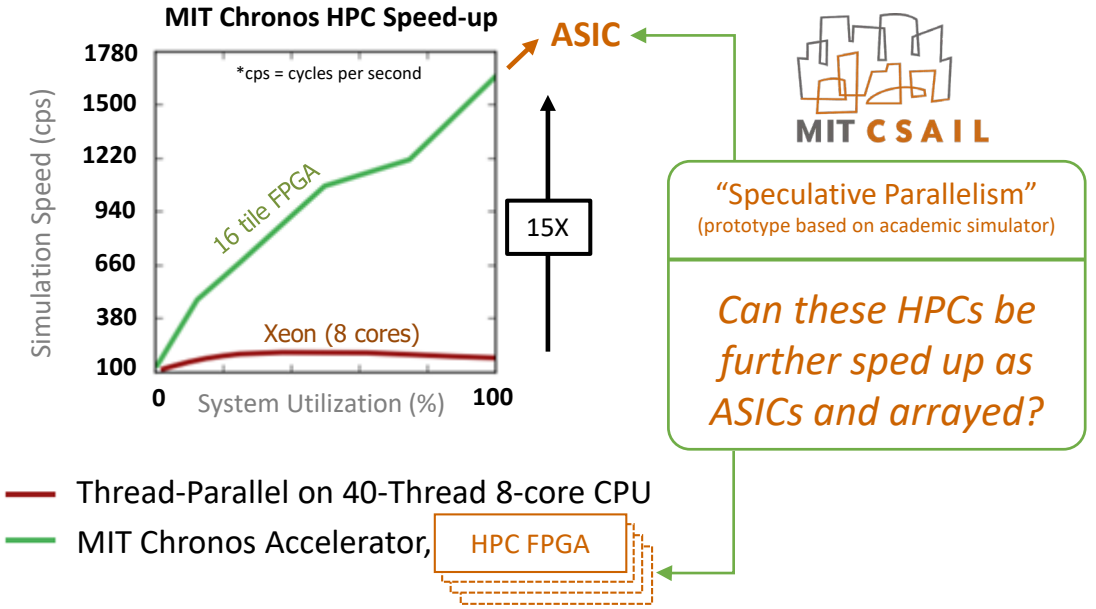
HPC to Exploit Speculative Parallelism

- ASIC verification relies on single-threaded simulation algorithms (100 cps) and/or high-cost, inflexible, dedicated hardware emulators (1.5M cps)
- Concept of speculative parallelism dates back to 1987, but since then the computational fabric and opportunities have substantially evolved (Cloud, eFPGA, GPUs, etc.)

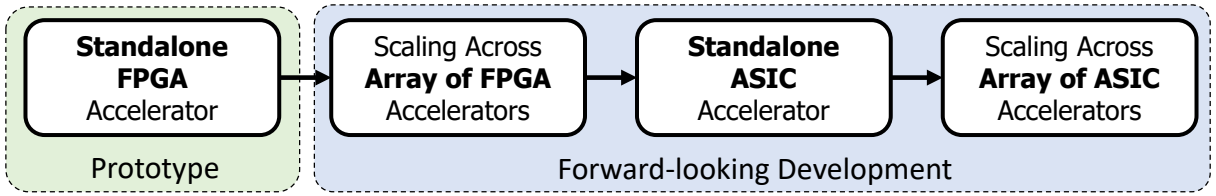


Source: R. Fujimoto, Georgia Institute of Technology

- Single-FPGA-based digital simulation accelerators have been prototyped, but have not demonstrated scaling off-chip or ASIC implementation
- Can we do better and achieve smooth system-level scaling?

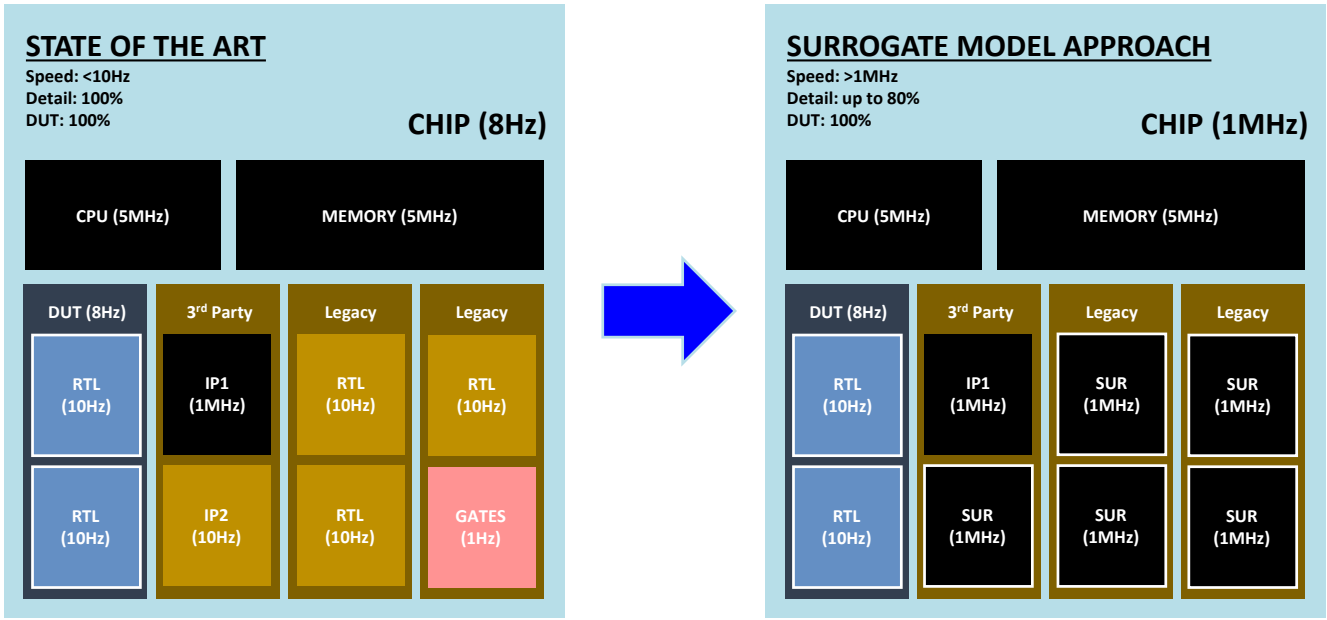


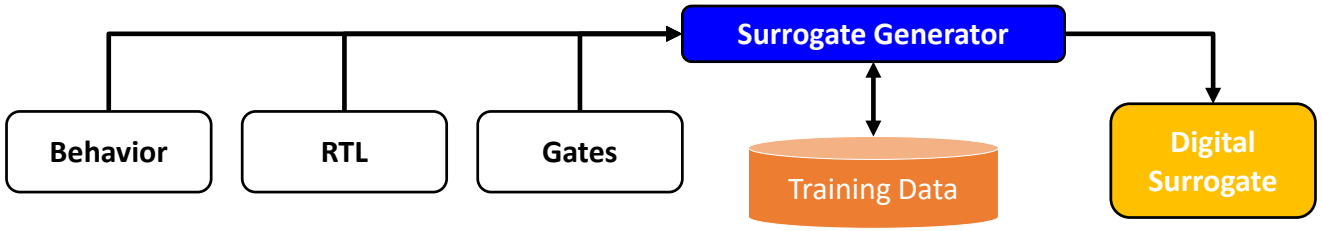
Source: MIT CSAIL



Auto-generation of Reduced Order Models

Can we drastically improve simulation performance in digital and mixed-signal SoCs by selectively substituting complex, high-fidelity circuit models with auto-generated, simplified, approximate surrogates?



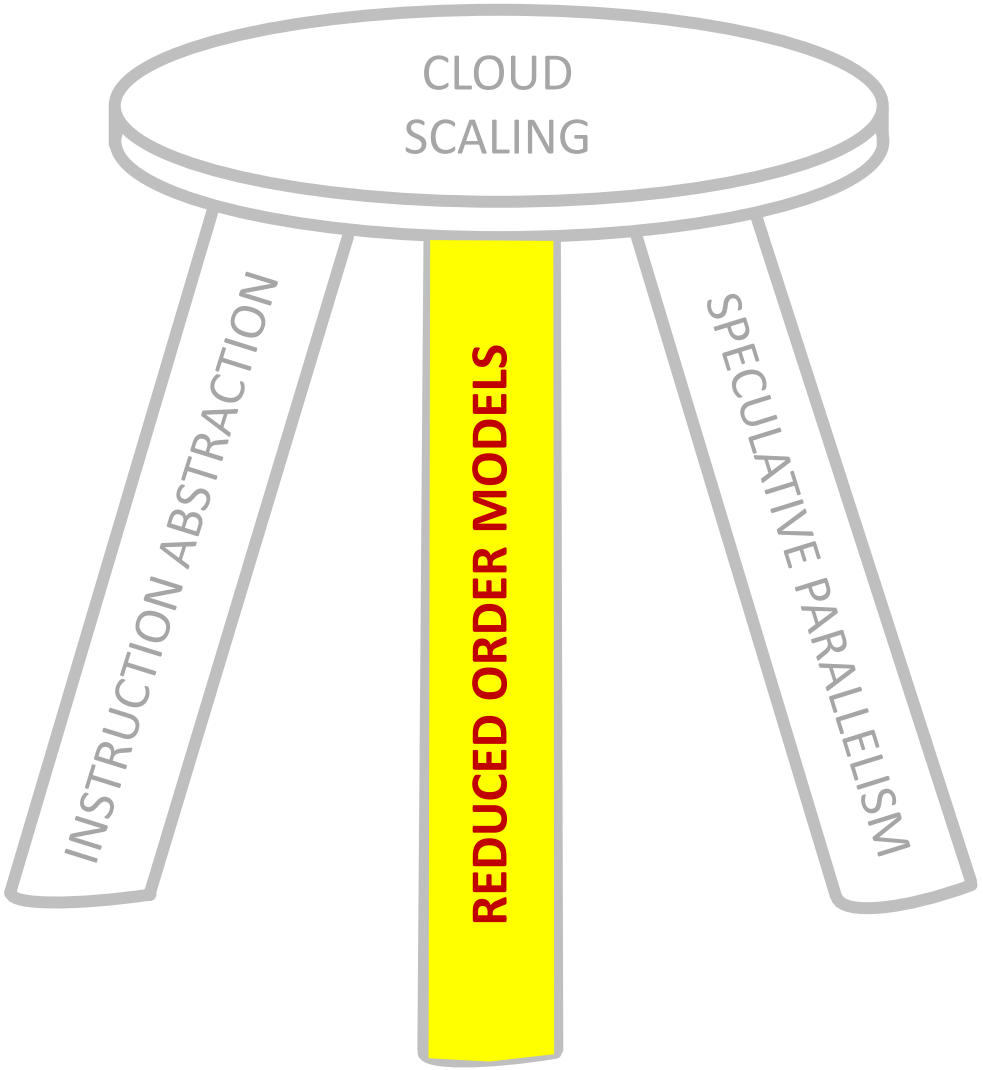


Rationale

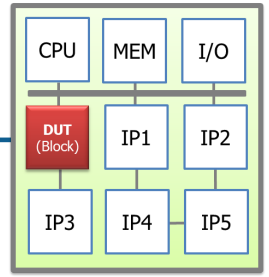
- Deploying state-of-the-art artificial intelligence techniques as the fundamental basis for generating simplified SoC models

- Develop machine learning models that can collapse/expand into different levels of hierarchy & have an awareness of their role within a larger system

- Make intelligent trade-offs between model accuracy and speed to achieve meaningful simulation



System* Modeling



Current Situation

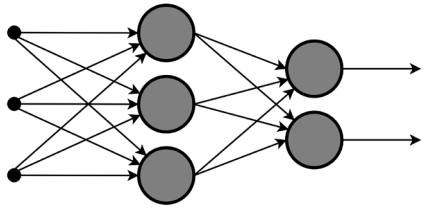
- Any system can be viewed as consisting of two partitions: Original Content or Design Under Test (DUT) and pre-existing design modules
- Non-DUT models consume significant compute resources and time to negatively impact overall system simulation speed
- Models that represent system sub-components can be slow, highly complex functions
- Models map input/output with maximum accuracy supported by the simulator
- Comprehensive full system simulation is slow, and often impractical

What We Would Like

- Non-DUT components consume a significantly smaller portion of total simulation speed
- Faster simulation models that trade-off an acceptable loss of accuracy for significant speed-up
- Comprehensive full system simulation is fast
- Simulation speed that drives greater acceptance and utilization of full system simulation that, in turn, increases fault detection, provides earlier insights into system designs, and mitigates risk

*__System__ in this context represents a complex of capabilities realized at chip, board or distributed compute/control realizations

AI/Machine Learning



Source: Wikipedia

Current Situation

- Neural network models that are excellent function approximators
- Models exist in isolation – they have no awareness of what they represent or how they exist in a larger system context
- Models are static and there is no ability to functionally aggregate them

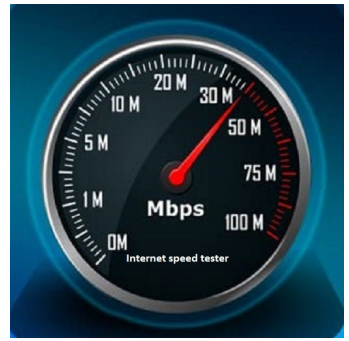
What We Would Like

- Meta-aware ML models that incorporate knowledge of domain and data semantics and understand their role in the larger system context
- Composable ML models that can collapse/expand into different levels of system hierarchy while maintaining acceptable accuracy levels for relevant input/output pairings

Objectives

1) Speed up simulation of microelectronic systems

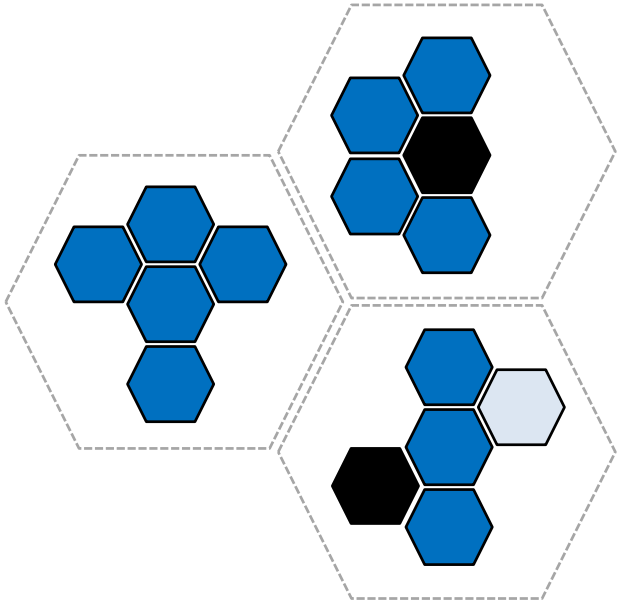
- This will be achieved by intelligently generating less-accurate, faster surrogate models of non-DUT components
- These models need to be able to achieve user-required target accuracy while still providing drastic speed-up



Source: Amazon

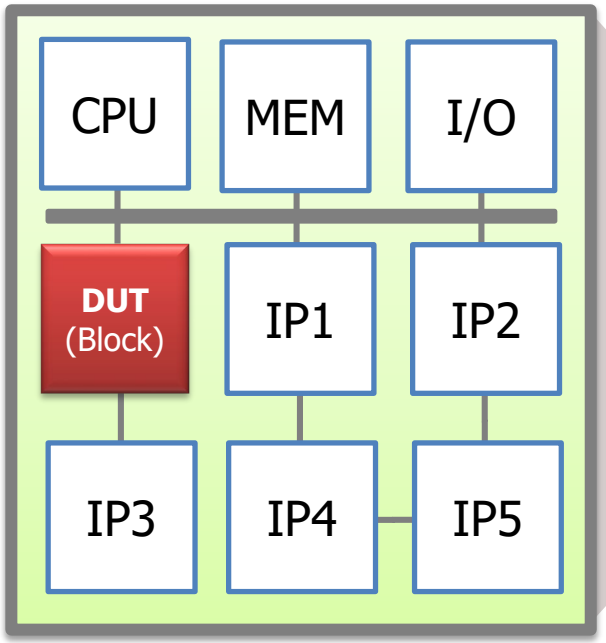
2) Develop novel 3rd-wave AI techniques that address drawbacks to 2nd wave AI solutions

- This will be achieved by creating AI surrogates that are:
  - *Composable:* Able to collapse/expand into different levels of hierarchy while maintaining acceptable accuracy and input/output coverage for all sub-components
  - *Meta-aware:* Able to maintain an awareness of how their representative real-world component exists and interacts within the overall system structure



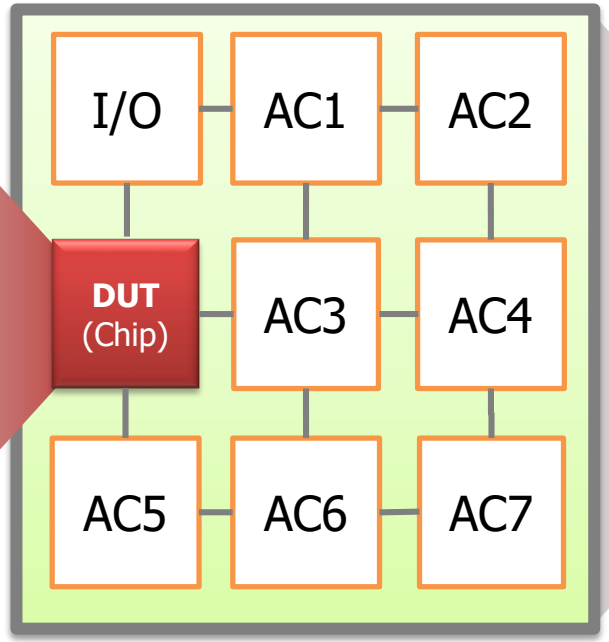
Does the DUT work correctly in its operational context?

Chip = DUT, CPU, Mem, I/O, DIP
Software = drivers
Surrogate candidate: IP



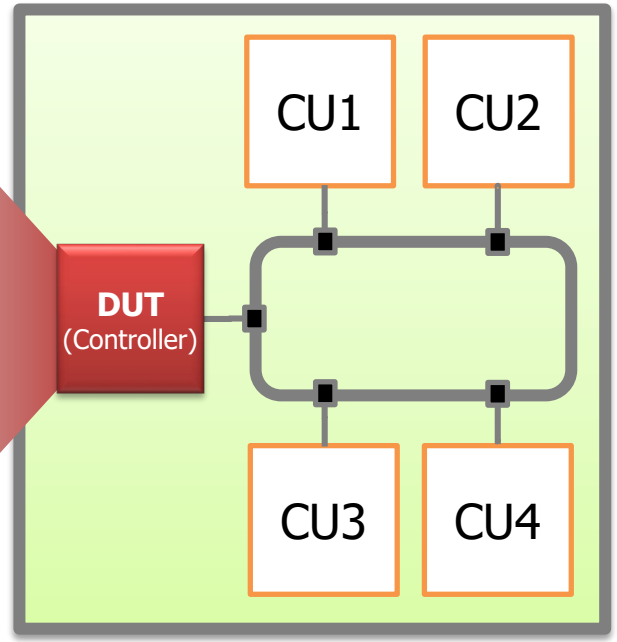
DUT = Block in chip context
IP = Digital IP
Model = RTL

Controller = Chip, I/O, AC
Software = drivers, OS, NM
Surrogate candidate: AC



DUT = Component in controller context
AC = Analog Component
Model = SPICE

System = Controllers, Network
Software = drivers, OS, NM, application
Surrogate candidate: CU



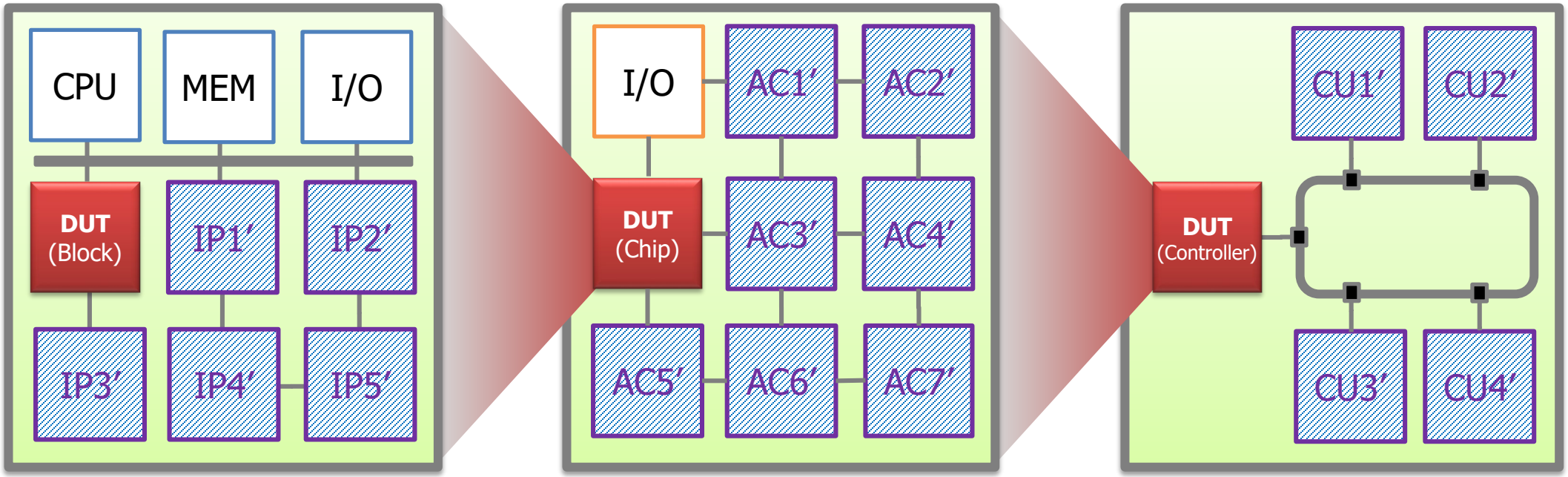
DUT = Controller in system context
CU = Control Unit
Model = Host Code or Simulink

Individual Surrogate Strategy *(each surrogate is individually trained)*

Chip = DUT, CPU, Mem, I/O, DIP
Software = drivers
Surrogate candidate: IP

Controller = Chip, I/O, AC
Software = drivers, OS, NM
Surrogate candidate: AC

System = Controllers, Network
Software = drivers, OS, NM, application
Surrogate candidate: CU



DUT = Block in chip context
IP = Digital IP
Model = RTL

DUT = Component in controller context
AC = Analog Component
Model = SPICE

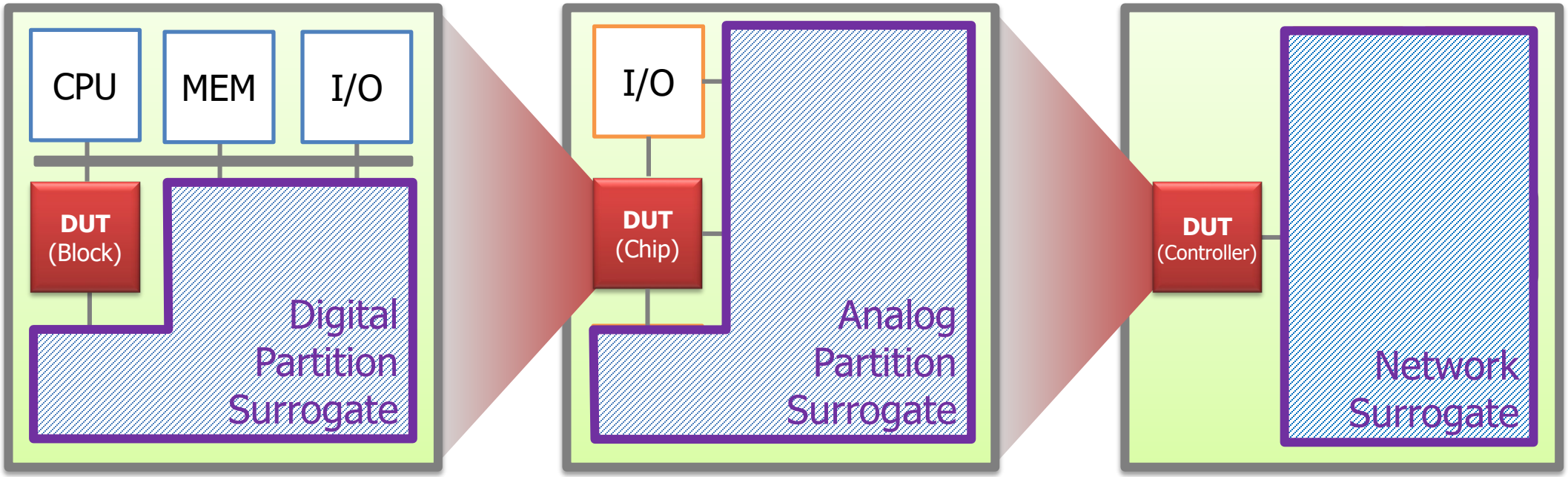
DUT = Controller in system context
CU = Control Unit
Model = Host Code or Simulink

Aggregate Surrogate Strategy *(each surrogate is trained as a cluster)*

Chip = DUT, CPU, Mem, I/O, DIP
Software = drivers
Surrogate candidate: IP

Controller = Chip, I/O, AC
Software = drivers, OS, NM
Surrogate candidate: AC

System = Controllers, Network
Software = drivers, OS, NM, application
Surrogate candidate: CU



DUT = Block in chip context
IP = Digital IP
Model = RTL

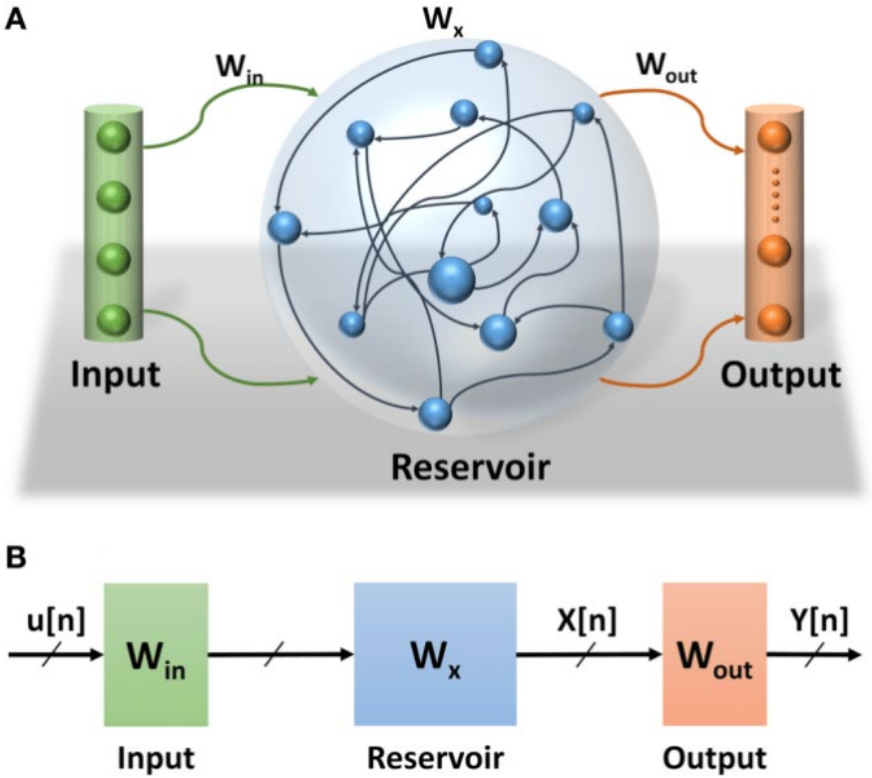
DUT = Component in controller context
AC = Analog Component
Model = SPICE

DUT = Controller in system context
CU = Control Unit
Model = Host Code or Simulink

Julia Computing and QuEra Computing
An ML-accelerated Mixed Signal Simulation Framework

Summary: Extending Julia's existing machine learning technology to cover analog/mixed-signal circuits

- JuliaSim: Unified simulation stack from (multi-)physics to digital
- JuliaSPICE: SPICE-compatible frontend for Analog Simulation
- Neural Surrogates provide allow trading small amounts of accuracy for high performance gains
- Fully Differentiable (unlike SOA simulators)
  - Suitable for white-box surrogatization
  - Enables other use cases also, e.g. gradient-based design optimization
- Model coverage: Basic SPICE models, Verilog-A import
  - Support for BSIM-CMG (ASAP7 7nm PDK)



Source: Julia Computing

Automatic creation of CTESN and IIR-CTELM surrogates

A modern, fully-differentiable, SPICE-compatible analog simulator using Ditto surrogates

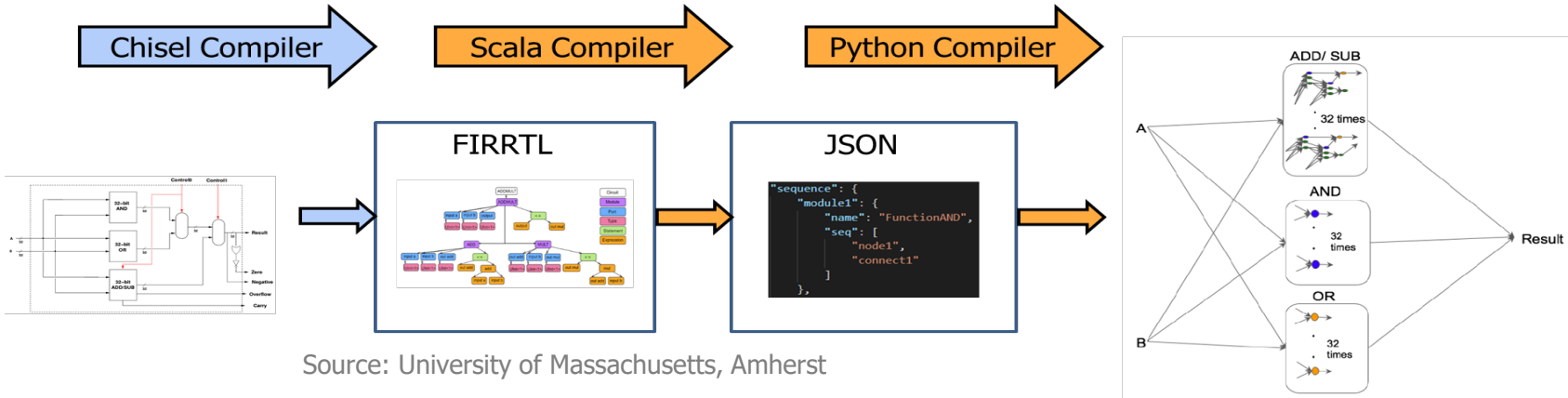
University of Massachusetts, Amherst and Lockheed Martin
MOKA - Modular Knowledgeable AI

Summary: A design-centric approach that hierarchically converts circuit structures into neural networks

- First, Programming Neural Networks (PNNs) are generated without training, within seconds, having full accuracy, and are usable as a reliably accurate simulation skeleton for any size IC

- Then, select PNNs can be converted to Masked Neural Networks (MNNs) via a method where weights and masks are learned simultaneously, leading to great speedup of simulation and higher accuracy; requires training data

- The resulting hybrid neural networks combine accurate PNNs with fast MNNs in strategic places – achieving user-defined trade-offs in modularity, speed, accuracy, and training time

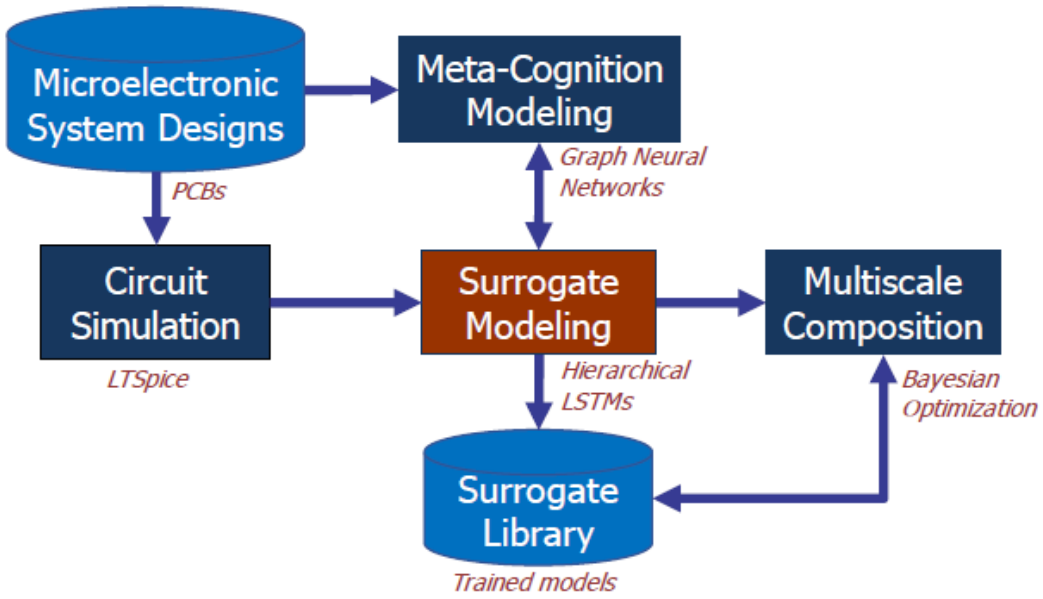


Source: University of Massachusetts, Amherst

A modular Ditto approach allowing for trade-offs in speed, accuracy, and training time

Summary: Applying a CPU-directed surrogate strategy to AMS

- Leverages Hierarchical Long Short-Term Memory networks (LSTMs), differentiable and good for processing not only single data points (such as images), but also entire sequences of data (such as speech or video).
- Graph Neural Network (GNN) approach combines graph structures and LSTMs for learning meta-cognitive representations of systems and their input/output environments.
- Bayesian Optimization further efficiently searches large configuration spaces with a small number of samples for automatic hierarchical multiscale composition of individual LSTM surrogates into larger systems.
- Benchmarked using real aerospace designs



Source: Aurora Flight Sciences

Aurora Ditto Modeling Framework

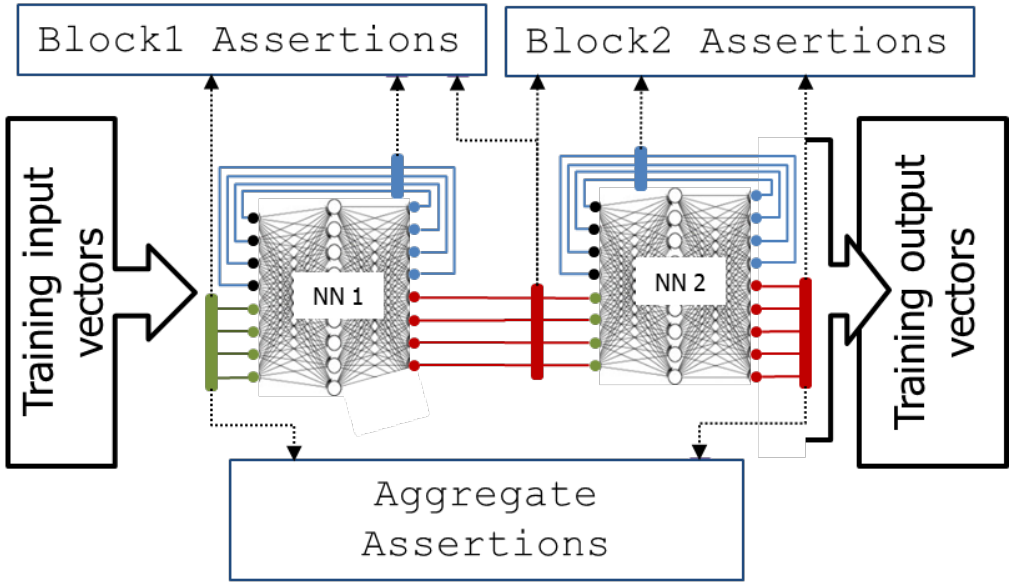
Differentiable surrogates allowing for optimization of systems with complex time-dependent behavior

SRI

System Understanding for Composable Cognitive Emulation of Electronic Devices (SUCCEED)

Summary: A traffic-centric approach that prunes the model around portions impacted by tests and assertions

- Neural networks train on novel inputs by using Deep Adaptive Semantic Logic (DASL) to back-propagate loss based on failure to satisfy formal assertions
- NNs then not only satisfy known input/output test vectors but also generalize beyond the small test vector set that may be available
- Training on random vectors without knowledge of correct outputs via the use of behavioral constraints obviates the need for use of manually-specified datasets and production of test data from slow simulators, speeding up the training process considerably



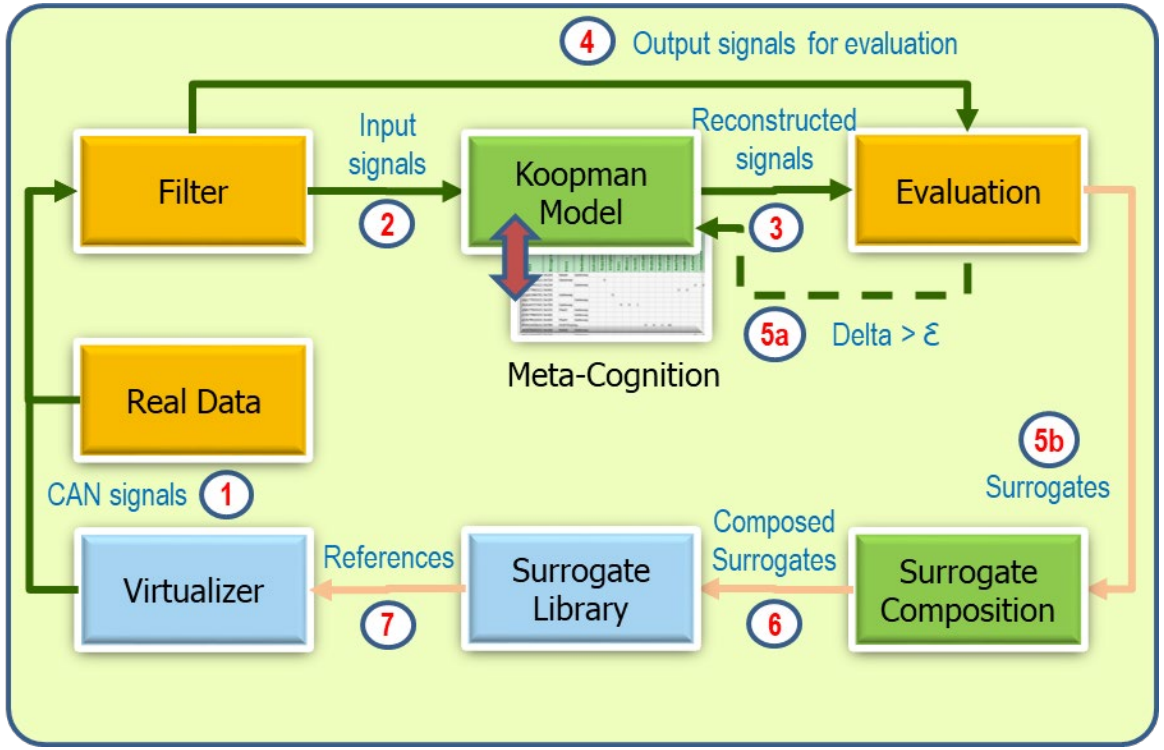
Source: SRI

SRI Ditto Modeling Framework

Surrogates which train quickly based on random input vectors rather than hand-selected datasets

Summary: Leverage knowledge of distributed systems to learn the dynamics and operating conditions of input/output traffic

- Data-driven synthesis, aggregation, and adaptation of surrogate models using Koopman operator
- Meta-cognition of Koopman surrogate models via discovered representations of operating conditions
- Extends model-based virtual prototyping
- Training data "topics" discover operating conditions such as acceleration or braking automatically
- Results show feasibility of using metacognition of distinct operating conditions to guide selection of adequate models

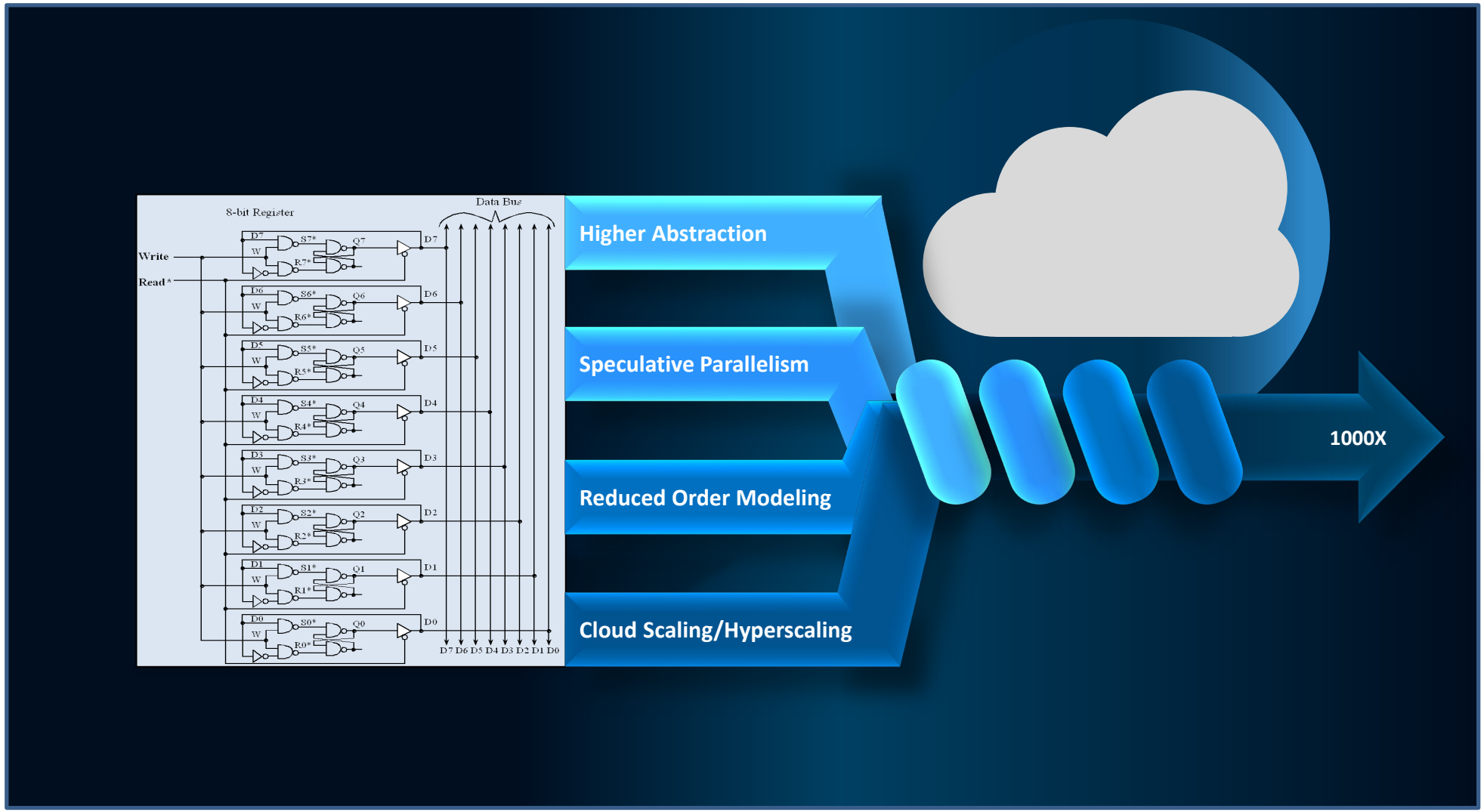


Source: BAE Systems

BAE Ditto Modeling Framework

Metacognitive surrogates apply novel math for fast, adaptive models learned with sparse data

Summary: Multiple Approaches Need to be Combined



Source: http://users.ece.utexas.edu/~valvano/Volume1/

www.darpa.mil