



# VISUALSIM TRAINING

---

# Agenda- Part 1: Basic Concepts

---

Overview

Areas of Modeling

Models of Computation

Modeling, Simulation, Analysis and Recommendation using VisualSim

Performance, Power and Functional Analysis

# Overview

# About Mirabilis Design

---

Started in 2007 and based in Santa Clara, CA, USA.

Development and support centers in US, India, Germany, China, Japan, Taiwan and Czech

System architecture exploration of electronics, semiconductors and software

Over 250 products worldwide across Semiconductors, Aerospace, Computing and Automotive

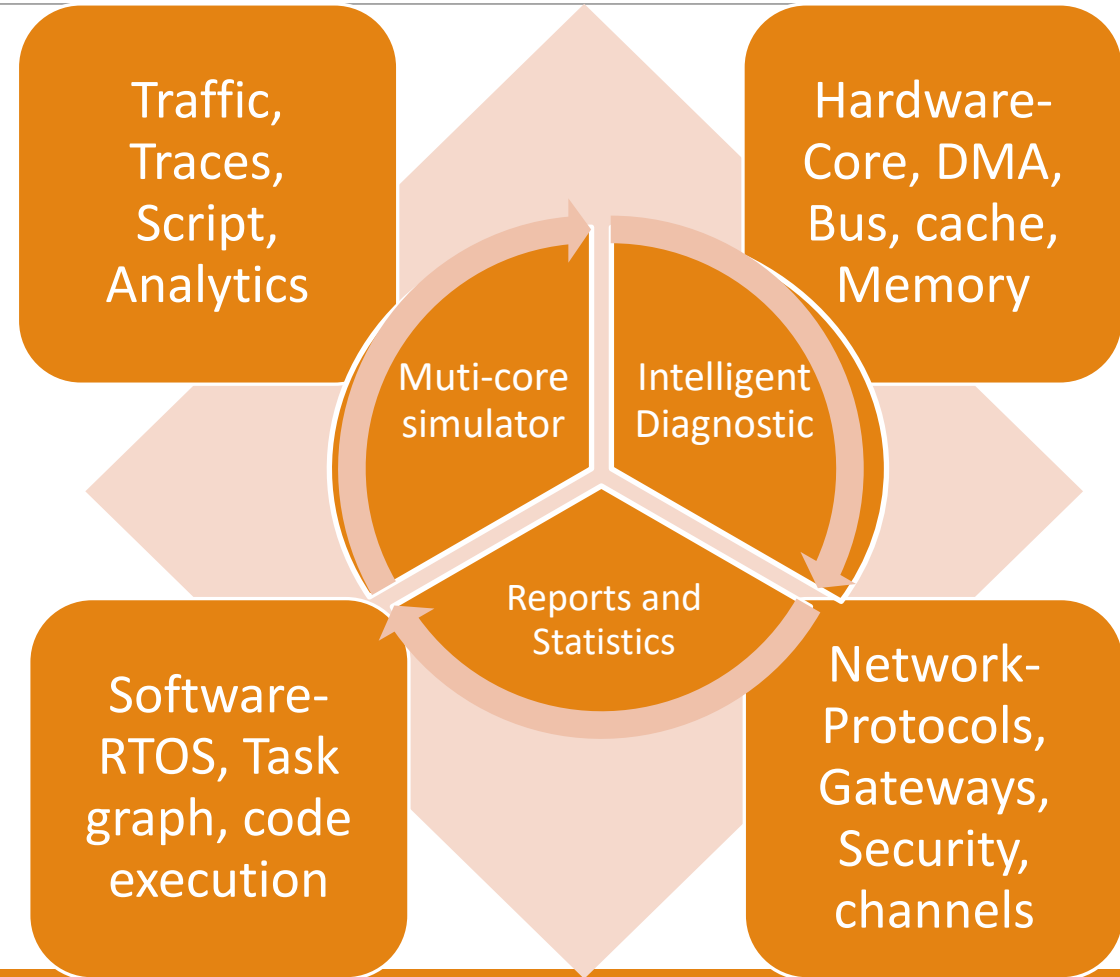
VisualSim- Modeling and simulation software

Largest source of system modeling IP with embedded timing and power

100's of man years experience in system design and exploration of digital electronics

# VisualSim Architect

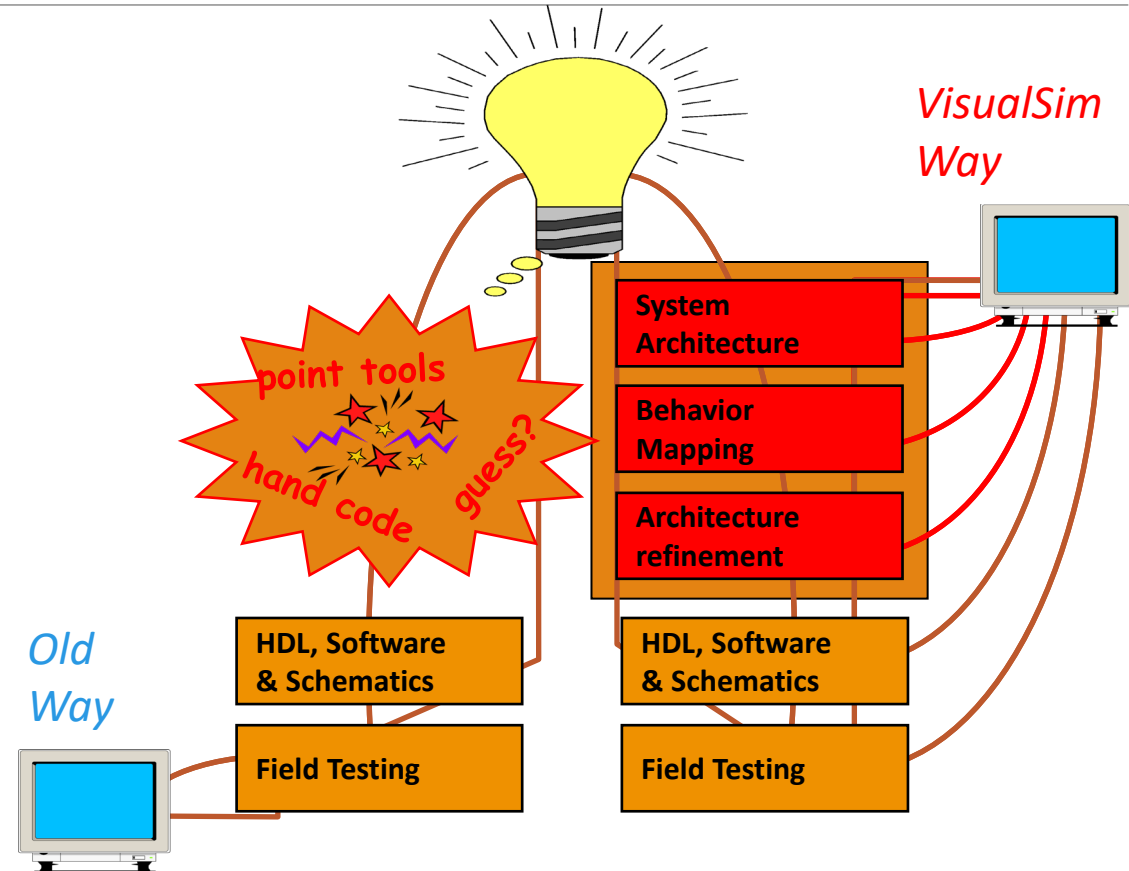
- Graphical and Hierarchical modeling
- Large library of parameterized components
- Integrated Discrete and Continuous Simulator
- Integrate API for simulators, programs and traces
- Optimizer to detect the best configuration



# Introduction to Conceptual Design

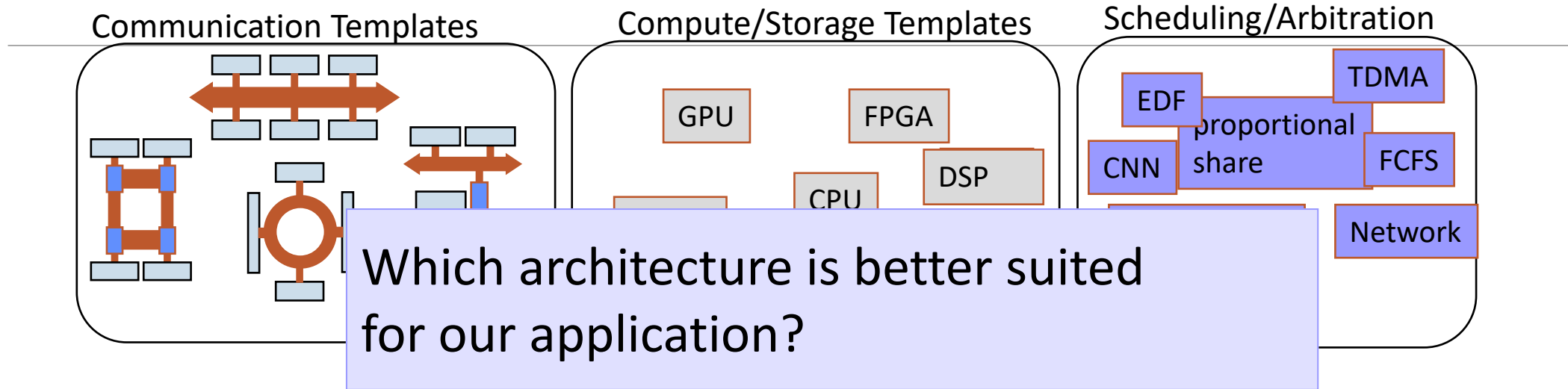
## ”Is this the Right Design?”

- Translation from product concept to implementation is critical link
- Traditional methods are “ad-hoc”
- VisualSim approach reduces risk and speed’s design

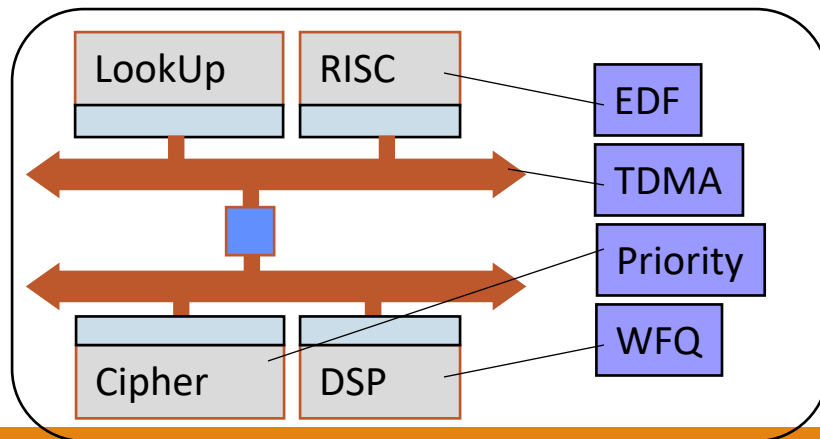


Errors in early stage cannot be rectified with optimized manufacturing

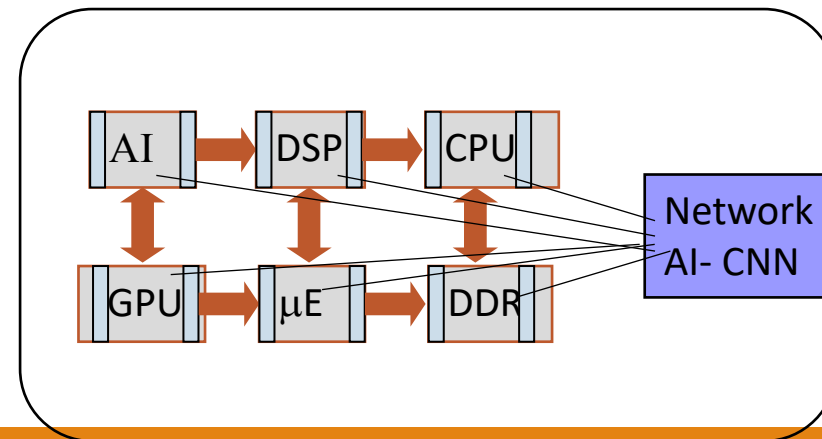
# Why is Architecture Exploration needed?



Architecture # 1



Architecture # 2



Run use-cases, workloads and traffic to decide on the best architecture

# Types of Model- For Architecture Exploration

---

## Mission or Network level

- Flow model- xOn\_xOff.xml
- Network- demo/networking folder or demo/automotive

## Full System including boards and boxes

- Embedded System- demo/System\_Architecture/Video\_Processing/Video\_Processing\_Model2.xml, demo/System\_Architecture/software\_Radar\_System/Large\_Radar\_System.xml. Demo/Bus\_Std/PCI\_Rad/PCI\_RAD\_Demo.xml
- Software:demo/software\_devl/software\_methodology/Software\_tasks\_w\_Power.xml, demo/automotive/Autosar/AUTOSAR\_Scheduler/Autosar\_Model4a\_noCCode.xml
- Evaluating software code- demo/automotive/Autosar/AUTOSAR\_Scheduler/Autosar\_Model4a.xml (Requires compile)

## SoC

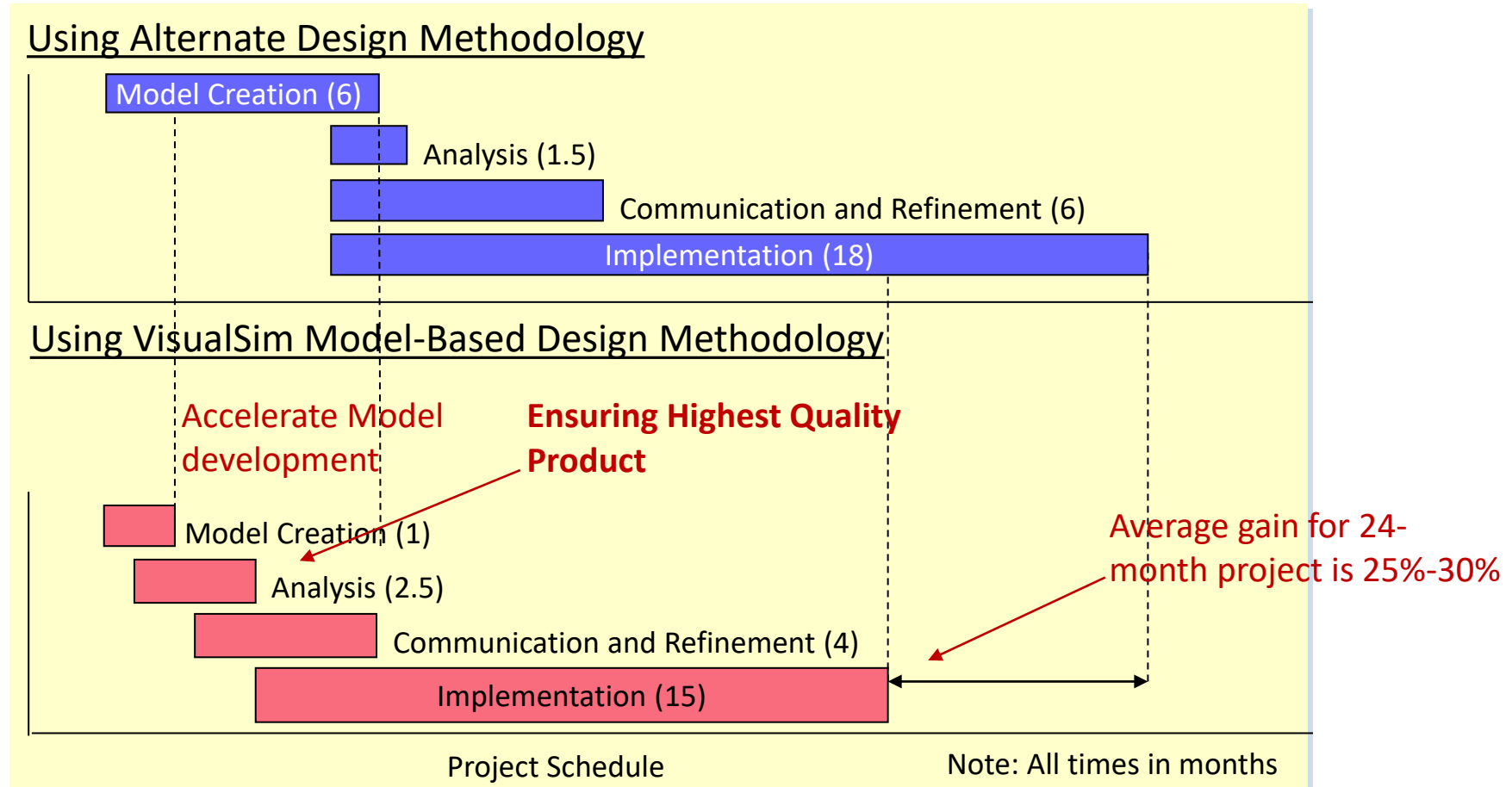
- Hardware+Software- early exploration- Multi\_Core\_Soc\_V43.xml
- Hardware+Software partitioning- Timing approximate- demo/partitioning/SoC/Power\_Perf\_Example.xml
- Hardware-software-cycle-accurate- CMN600\_with\_DDR5\_DynamicRT.xml
- Individual component- Bus and Memory-demo/complex\_system/Multi\_AXI\_to\_Memory\_Access.xml



# Complete Systems-Level Library

<p><b><u>Traffic</u></b></p> <ul style="list-style-type: none"> <li>• Distribution</li> <li>• Sequence</li> <li>• Trace file</li> <li>• Instruction profile</li> </ul> <p><b><u>Reports</u></b></p> <ul style="list-style-type: none"> <li>• Timing and Buffer</li> <li>• Throughput/Util</li> <li>• Ave/peak power</li> <li>• Statistics</li> </ul>	<p><b><u>Power</u></b></p> <ul style="list-style-type: none"> <li>• State power table</li> <li>• Power management</li> <li>• Energy harvesters</li> <li>• Battery</li> <li>• RegEx operators</li> </ul>	<p><b><u>SoC Buses</u></b></p> <ul style="list-style-type: none"> <li>• AMBA and Corelink</li> <li>• AHB, AB, AXI, ACE, CHI, CMN600</li> <li>• Network-on-Chip</li> <li>• TileLink</li> </ul>	<p><b><u>System Bus</u></b></p> <ul style="list-style-type: none"> <li>• PCI/PCI-X/PCIe</li> <li>• Rapid IO</li> <li>• AFDX</li> <li>• OpenVPX</li> <li>• VME</li> <li>• SPI 3.0</li> <li>• 1553B</li> </ul>	<p><b><u>Processors</u></b></p> <ul style="list-style-type: none"> <li>• GPU, DSP, <math>\mu</math>P and <math>\mu</math>C</li> <li>• RISC-V</li> <li>• Nvidia- Drive-PX</li> <li>• PowerPC</li> <li>• X86- Intel and AMD</li> <li>• DSP- TI and ADI</li> <li>• MIPS, Tensilica, SH</li> </ul>	<p><b><u>ARM</u></b></p> <ul style="list-style-type: none"> <li>• M-, R-, 7TDMI</li> <li>• A8, A53, A55, A72, A76, A77</li> </ul>
<p><b><u>Custom Creator</u></b></p> <ul style="list-style-type: none"> <li>• Script language</li> <li>• 600 RegEx fn</li> <li>• Task graph</li> <li>• Tracer</li> <li>• C/C++/Java</li> <li>• Python</li> </ul> <p><b><u>Support</u></b></p> <ul style="list-style-type: none"> <li>• Listener and Trace</li> <li>• Debuggers</li> <li>• Assertions</li> </ul>	<p><b><u>Stochastic</u></b></p> <ul style="list-style-type: none"> <li>• FIFO/LIFO Queue</li> <li>• Time Queue</li> <li>• Quantity Queue</li> <li>• System Resource</li> <li>• Schedulers</li> <li>• Cyber Security</li> </ul> <p><b><u>RTOS</u></b></p> <ul style="list-style-type: none"> <li>• Template</li> <li>• ARINC 653</li> <li>• AUTOSAR</li> </ul>	<p><b><u>Memory</u></b></p> <ul style="list-style-type: none"> <li>• Memory Controller</li> <li>• DDR DRAM 2,3,4, 5</li> <li>• LPDDR 2, 3, 4</li> <li>• HBM, HMC</li> <li>• SDR, QDR, RDRAM</li> </ul> <p><b><u>Storage</u></b></p> <ul style="list-style-type: none"> <li>• Flash &amp; NVMe</li> <li>• Storage Array</li> <li>• Disk and SATA</li> <li>• Fibre Channel</li> <li>• FireWire</li> </ul>	<p><b><u>Networking</u></b></p> <ul style="list-style-type: none"> <li>• Ethernet &amp; GiE</li> <li>• Audio-Video Bridging</li> <li>• 802.11 and Bluetooth</li> <li>• 5G</li> <li>• Spacewire</li> <li>• CAN-FD</li> <li>• TTEthernet</li> <li>• FlexRay</li> <li>• TSN &amp; IEEE802.1Q</li> </ul>	<p><b><u>FPGA</u></b></p> <ul style="list-style-type: none"> <li>• Xilinx- Zynq, Virtex, Kintex</li> <li>• Intel-Stratix, Arria</li> <li>• Microsemi- Smartfusion</li> <li>• Programmable logic template</li> <li>• Interface traffic generator</li> </ul> <p><b><u>Software</u></b></p> <ul style="list-style-type: none"> <li>• GEM5</li> <li>• Software code integration</li> <li>• Instruction trace</li> <li>• Statistical software model</li> <li>• Task graph</li> </ul>	<p><b><u>Interfaces</u></b></p> <ul style="list-style-type: none"> <li>• Virtual Channel</li> <li>• DMA</li> <li>• Crossbar</li> <li>• Serial Switch</li> <li>• Bridge</li> </ul> <p><b><u>RTL-like</u></b></p> <ul style="list-style-type: none"> <li>• Clock, Wire-Delay</li> <li>• Registers, Latches</li> <li>• Flip-flop</li> <li>• ALU and FSM</li> <li>• Mux, DeMux</li> <li>• Lookup table</li> </ul>

# Profit Upside using VisualSim



Quick Model Bring-UP and Meticulous Analysis

# Modeling Examples

# Modeling Examples

---

[Circuit and Mixed-Signal](#)

[Traffic, Queueing, Flow Control and Stochastic](#)

[Systems Engineering Process](#)

[Distributed Computing Systems](#)

[Software Design](#)

[Sub-system](#)

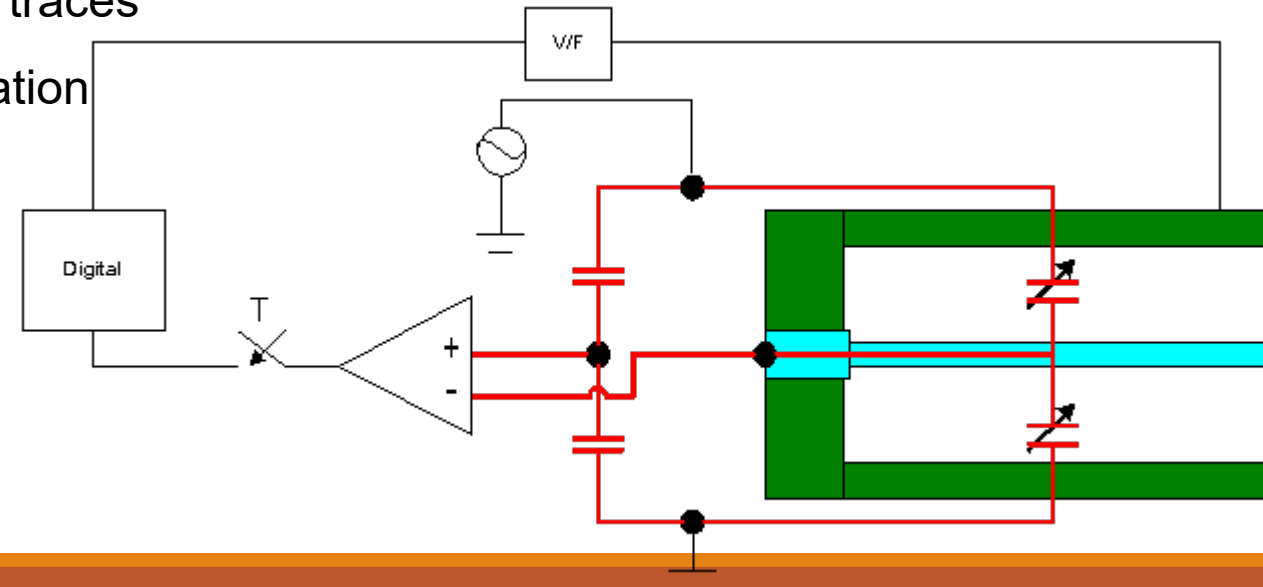
[FPGA Design](#)

[Hardware-Software Partitioning](#)

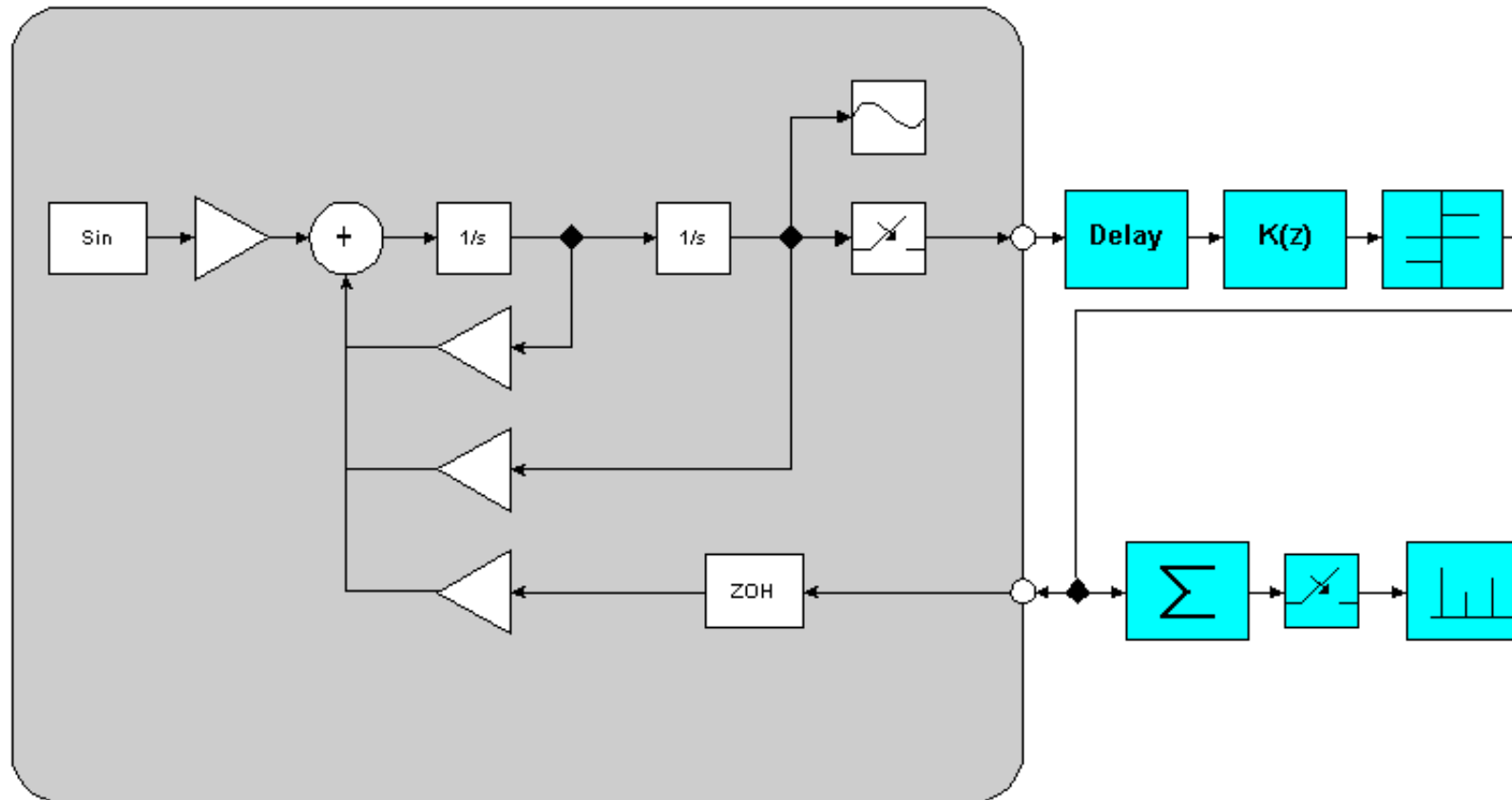
[Semiconductors](#)

# Circuit Example

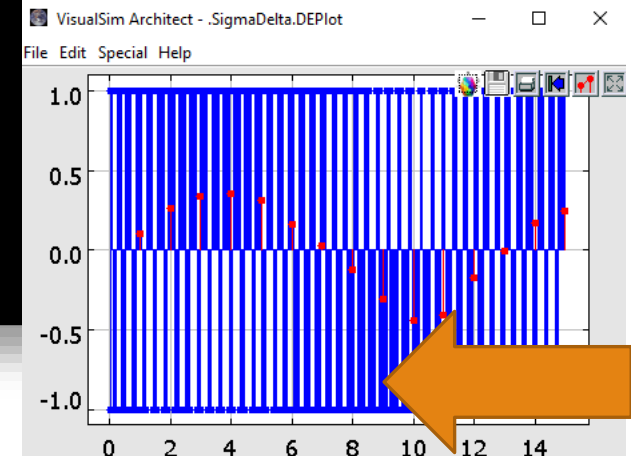
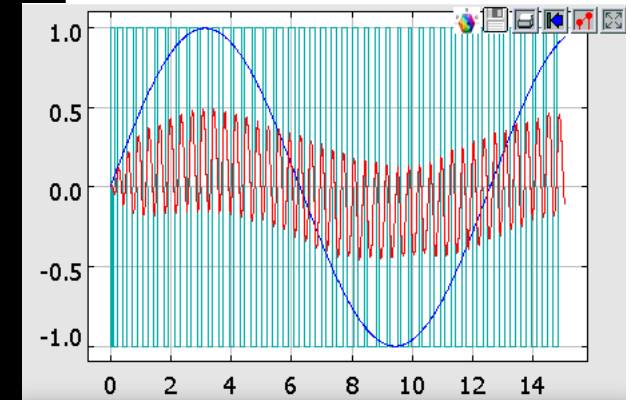
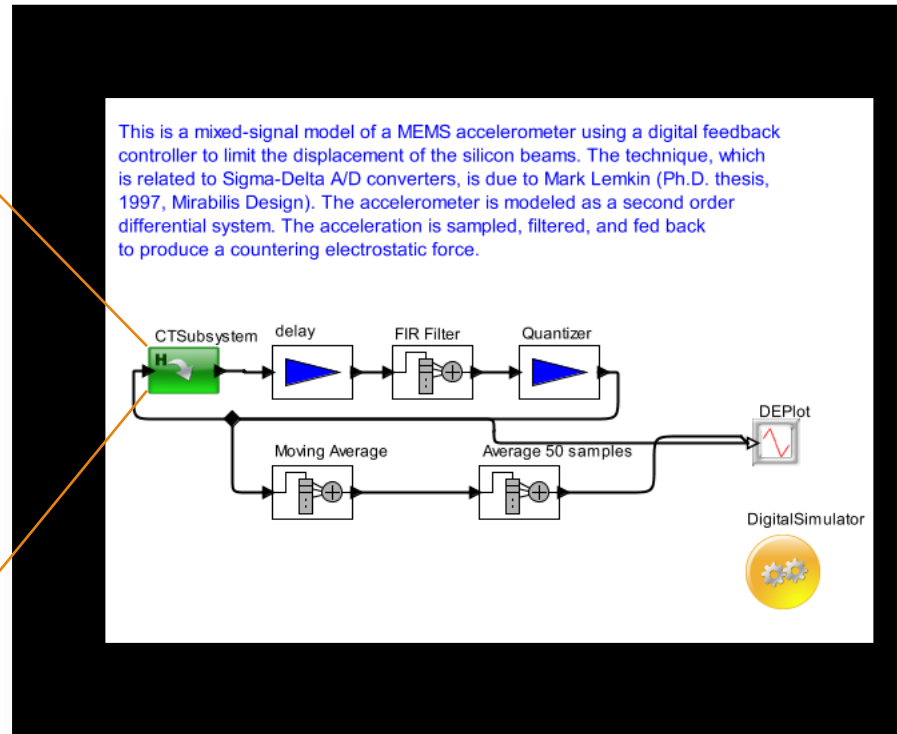
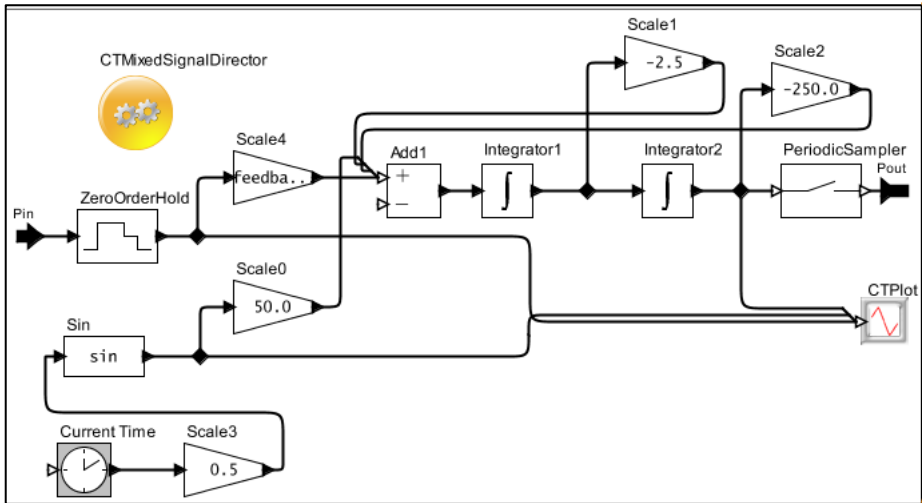
- Evaluate junctions and transistors
- Used to get global or system-wide perspective
- Semiconductor IC maybe a delay or a queue + processing element
- Traffic will be distribution-driven or network traces
- Traceability is the most important consideration



# Converting Sigma-Delta A/D to Behavioral Block Diagram



# Sigma-Delta A/D in VisualSim using Continuous and Discrete-Event Simulators



# Traffic or Queuing Analysis

---

## Analysis

- Quick and extensive feasibility study
- Identify areas for detailed investigation
- Scheduling and priority analysis

## Hardware

- Use a SystemResource to define each device
- Routing table to determine path for each task

## Software

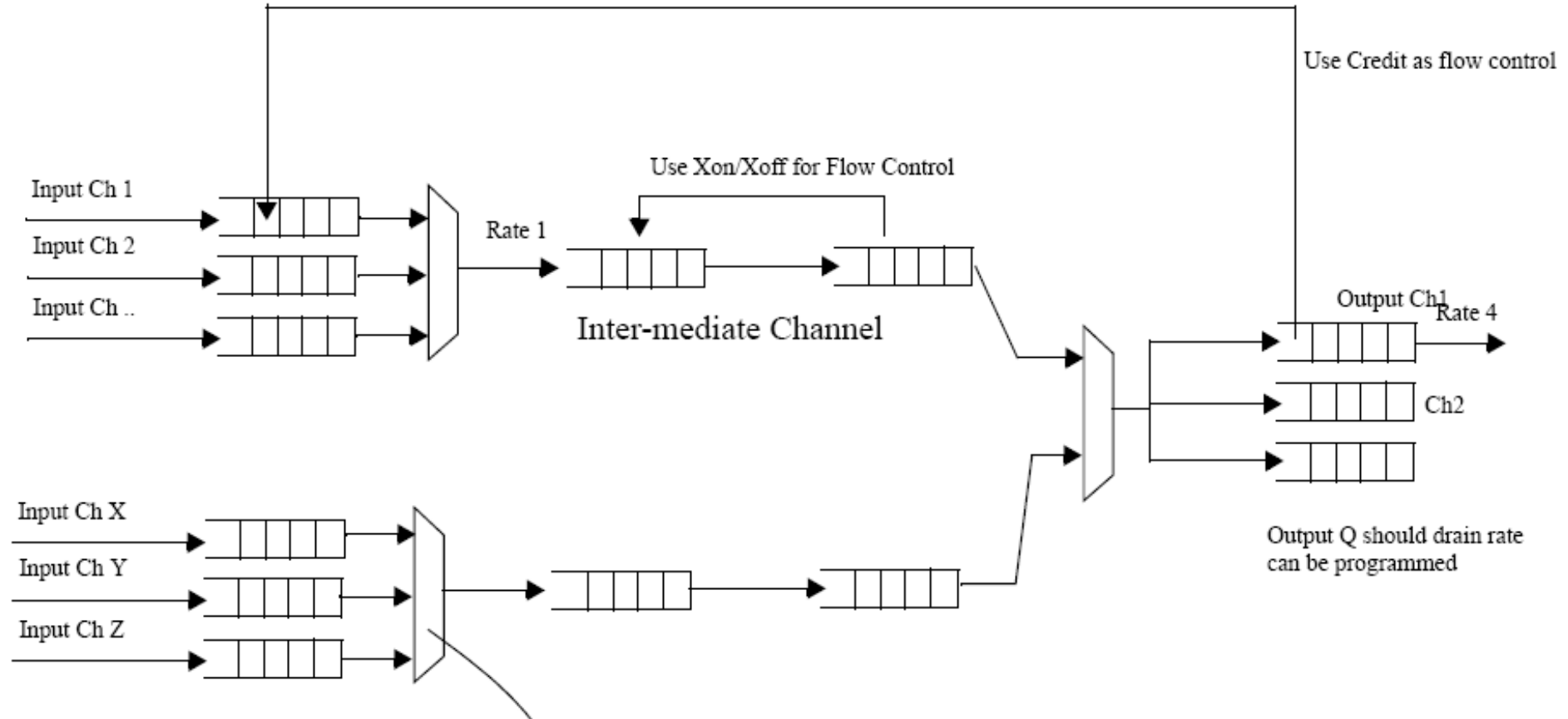
- Generate a statistical profile for the software code to each target
- Distribution-based traffic to emulate the task and message-passing
- Add priority, target processor and task ordering

## Throughput and feasibility study of Network of aircrafts, Ground Station and satellites

- Send and Receive data buffering, link margin and retransmission
- Size data packets to reduce power consumption for IoT devices
- Performance degradation due to retransmission or data loss
- Evaluate new protocol standards
- Exploring the impact of data center switches, bandwidth and routing paths



# Flow Control and Scheduling



# Modeling the Dynamic Behavior

The screenshot displays the VisualSim Architect interface with a network flow control model. The model includes traffic generators, ingress queues (Queue1, Queue2), Xon and Xoff queues (Queue4, Queue5, Queue6, Queue7), a multiplexer (Mux), and egress queues (Queue8). It also features smart controllers (RR1, RR2), statistics, and various monitors like TextDisplay and Latency/Histogram.

Performance metrics for the model are shown in the right-hand window:

Metric	Value
Total_Delay_Min	= 0.0,
Total_Delay_StDev	= 3.7983044519159E-9,
Utilization_Mean	= 0.0}, {BLOCK
DELTA	= 0.0,
DS_NAME	= "Queue_Common_Stats",
ID	= 298,
INDEX	= 0,
Number_Entered	= 130,
Number_Exited	= 130,
Number_Rejected	= 0,
Occupancy_Max	= 1.0,
Occupancy_Mean	= 0.4980694980695,
Occupancy_Min	= 0.0,
Occupancy_StDev	= 0.4999962731484,
Queue_Number	= 13,
TIME	= 1.0E-4,
Total_Delay_Max	= 4.3399299999992E-8,
Total_Delay_Mean	= 4.2673233846154E-8,
Total_Delay_Min	= 0.0,
Total_Delay_StDev	= 3.7666784899562E-9,
Utilization_Mean	= 0.0}, {BLOCK
DELTA	= 0.0,
DS_NAME	= "Queue_Common_Stats",
ID	= 299,
INDEX	= 0,
Number_Entered	= 127,
Number_Exited	= 127,
Number_Rejected	= 0,
Occupancy_Max	= 1.0,
Occupancy_Mean	= 0.498023715415,
Occupancy_Min	= 0.0,
Occupancy_StDev	= 0.499996094284,
Queue_Number	= 14,
TIME	= 1.0E-4,
Total_Delay_Max	= 4.3398800000008E-8,
Total_Delay_Mean	= 4.2730402362204E-8,
Total_Delay_Min	= 0.0,
Total_Delay_StDev	= 3.8166368756054E-9,
Utilization_Mean	= 0.0}, {BLOCK
DELTA	= 0.0,
DS_NAME	= "Queue_Common_Stats",
ID	= 300,
INDEX	= 0,
Number_Entered	= 132,
Number_Exited	= 132,
Number_Rejected	= 0,
Occupancy_Max	= 1.0,
Occupancy_Mean	= 0.4980988593156,
Occupancy_Min	= 0.0,
Occupancy_StDev	= 0.499996385651,
Queue_Number	= 15,
TIME	= 1.0E-4,
Total_Delay_Max	= 4.3396600000006E-8,
Total_Delay_Mean	= 4.2686459090909E-8,
Total_Delay_Min	= 0.0,
Total_Delay_StDev	= 3.7380146955213E-9,
Utilization_Mean	= 0.0}}

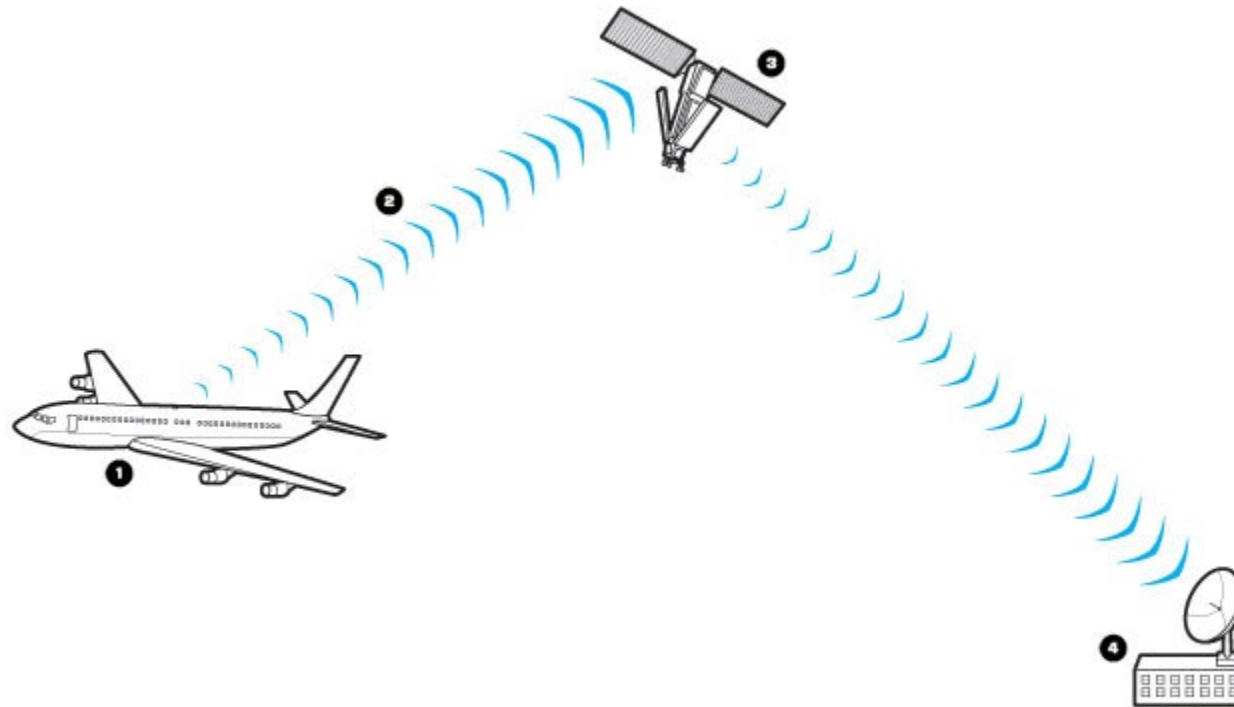
Two performance graphs are also visible:

- Latency:** A line graph showing latency over simulation time (0.0 to 1.0 x 10<sup>-4</sup> seconds). The y-axis ranges from 0 to 8 x 10<sup>-5</sup>.
- Buffer Occupancy:** A line graph showing buffer occupancy (Packets) over simulation time (0.0 to 1.0 x 10<sup>-4</sup> seconds). The y-axis ranges from 0 to 30 packets.

# Traffic Engineering Example

## Top-Level of the Aerial System

---



# Mission-Level Analysis

**Find:**

Library Tree

- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Memory
- HardwareDevices
- Interfaces and Buses
- Full Library
- UserLibrary

## Aerial Common Sensor

ACS will satisfy the Army and Navy's Critical Need for a worldwide, self-deployable, airborne reconnaissance, intelligence, surveillance, and target acquisition capability

**Parameters**

- Sim\_Time: 10.0 /\*in seconds\*/
- Ref\_DB\_Size: 4000
- Processors\_AirLink: 5
- Speed\_Processor: 10000000.0
- LinkSpeed: 12000.0
- RTOS\_Buffer: 200
- Cache\_Seek\_Time\_Low: 10.0
- Cache\_Seek\_Time\_High: 100

**Modify:**

- Number of processors
- Processor Speed
- Link Speed
- Reject\_threshold at the DC

**Find:**

Library Tree

- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Memory
- HardwareDevices
- Interfaces and Buses
- Full Library
- UserLibrary

### DCGS Ground Vehicle

DCGS can be on a Navy Ship, Battle Tank and Ground Vehicles. These systems will communicate with reconnaissance planes and satellites to gather deep strike data, process these and determines concurrent location of the target. This models looks at the processing requirements, the data transmission and processing latency and verify that arriving data is devoid of errors. If the checksum does not match, a request will be made for retransmission.

**DCGS Parameters**

- Max\_Distance: 6.5
- Ground\_Station\_Name: "Ship1"
- Recon\_Planes: "Data\_Link\_1, Data\_Link\_2, Data\_Link\_3"
- Number\_of\_Sources: "1, 2, 3"
- Frame\_ID\_DB: Ref\_DB\_Size
- LinkSpeed: LinkSpeed
- Reject\_Threshold: 1 /\*Range is 1-5, 5 being the highest! Digital

**Find:**

Library Tree

- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Memory
- HardwareDevices
- Interfaces and Buses
- Full Library
- UserLibrary

## Inside the Reconnaissance Plane (Sensors, Data Link and Processing)

Sensor Packages and data link equipment that communicates sensor data to ground stations

**Parameters**

- Cache\_Seek\_Time\_Low: Cache\_Seek\_Time\_Low
- Sen\_Start\_Time: 0.01
- Sen\_Mean\_Time: 0.01
- Buf\_Plot\_Name: "Data\_Link\_1\_Buf"
- RTOS\_Bu
- Cache\_Se
- ACS\_Nun
- ACS\_ID: "

**Find:**

Library Tree

- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Memory
- HardwareDevices
- Interfaces and Buses
- Full Library
- UserLibrary

VisualSim Architect - file://

File View Edit Graph Debug Interface Help

**Source Parameters**

- Final\_Destination: "Broadcast"
- Data\_Link\_ID: ACS\_ID
- Source: "Infrared" + Data\_Link\_ID
- Link\_Speed: 7500
- Source\_Num: 2
- X\_Distance: 100.0
- Start\_Time: Sen\_Start\_Time
- Y\_Distance: 75.0
- Mean\_Time: Sen\_Mean\_Time
- Link\_Delay: sqrt(X\_Distance ^ 2 + Y\_Distance ^ 2) / Link\_Speed

**Annotations:**

- Generate Sensor data (Uniform Distribution)
- Populate DS Fields with random distribution values
- Delay for Link Transmission
- Send to Platform Hub

# End-Product Study

## Aerial Common Sensor- Challenges

---

How many sensor can the system handle?

Compute, Storage and Communication requirements

Timing and power consumption

Project scrapped?

- Weight and battery capacity issues

# End-Product Study

## Aerial Common Sensor

File View Edit Graph Debug Interface Help



Find:

- Library
- Tree
- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Memory
- HardwareDevices
- Interfaces and Buses
- Full Library
- UserLibrary

### Aerial Common Sensor

ACS will satisfy the Army and Navy's Critical Need for a worldwide, self-deployable, airborne reconnaissance, intelligence, surveillance, and target acquisition capability



Digital



#### Parameters

- Sim\_Time: 10.0 /\*in seconds\*/
- Ref\_DB\_Size: 4000
- Processors\_AirLink: 5
- Speed\_Processor: 10000000.0
- LinkSpeed: 12000.0
- RTOS\_Buffer: 200
- Cache\_Seek\_Time\_Low: 10.0 / 11000000.0
- Cache\_Seek\_Time\_High: 100.0 / 11000000.0

Statistics



- Modify:
- Number of processors
  - Processor Speed
  - Link Speed
  - Reject\_threshold at the DCGS

execution finished.

# End-Product Study

## Aerial Common Sensor

VisualSim Architect - .ACS\_Co...

```

DISPLAY AT TIME          ----- 10.00000000000 sec -----
{BLOCK                   = "ACS_Data_Link.RTOS_Sched.ACS1",
DELTA                     = 0.0,
DS_NAME                   = "Queue_Common_Stats",
ID                        = 1,
INDEX                    = 0,
Number_Entered            = 689,
Number_Exited            = 580,
Number_Rejected           = 0,
Occupancy_Max            = 110.0,
Occupancy_Mean           = 55.4767063921993,
Occupancy_Min            = 0.0,
Occupancy_StDev          = 31.6536006244012,
Queue_Number             = 1,
TIME                     = 10.0,
Total_Delay_Max          = 7.1294790495,
Total_Delay_Mean         = 0.266355335535,
Total_Delay_Min          = 0.0107664315,
Total_Delay_StDev        = 0.9029137801163,
Utilization_Mean         = 99.579942098}
        
```

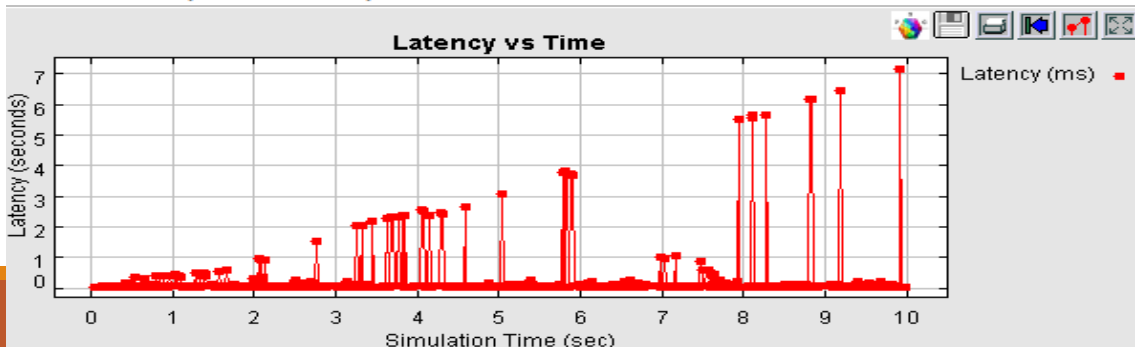
VisualSim Architect - ...

```

Number_Rejected           = 0,
Occupancy_Max            = 1.0,
Occupancy_Mean           = 0.4995693367786,
Occupancy_Min            = 0.0,
Occupancy_StDev          = 0.4999998145292,
Queue_Number             = 1,
TIME                     = 10.0,
Total_Delay_Max          = 0.0049637627,
Total_Delay_Mean         = 0.0013561969134,
Total_Delay_Min          = 8.2282199999284E-5,
Total_Delay_StDev        = 9.2685786241535E-4,
Utilization_Mean         = 8.1786884890511}

DISPLAY AT TIME          ----- 10.00000000000 sec -----
{BLOCK                   = "Flash_Memory",
DELTA                     = 0.0,
DS_NAME                   = "Queue_Common_Stats",
ID                        = 1,
INDEX                    = 0,
Number_Entered            = 581,
Number_Exited            = 580,
Number_Rejected           = 0,
Occupancy_Max            = 3.0,
Occupancy_Mean           = 1.5719207579673,
Occupancy_Min            = 0.0,
Occupancy_StDev          = 0.6213573377532,
Quantity_Used_Max        = 5,
Quantity_Used_Mean        = 2,
Quantity_Used_Min        = 0,
Quantity_Used_StDev      = 1,
Queue_Number             = 1,
TIME                     = 10.0,
Total_Delay_Max          = 0.025,
Total_Delay_Mean         = 0.025,
Total_Delay_Min          = 0.025,
Total_Delay_StDev        = 2.5172580344751E-9,
Utilization_Mean         = 0.0}
        
```

File Edit Special Help

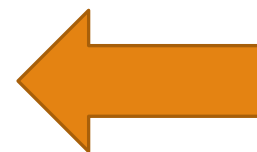


**Latency vs Time**

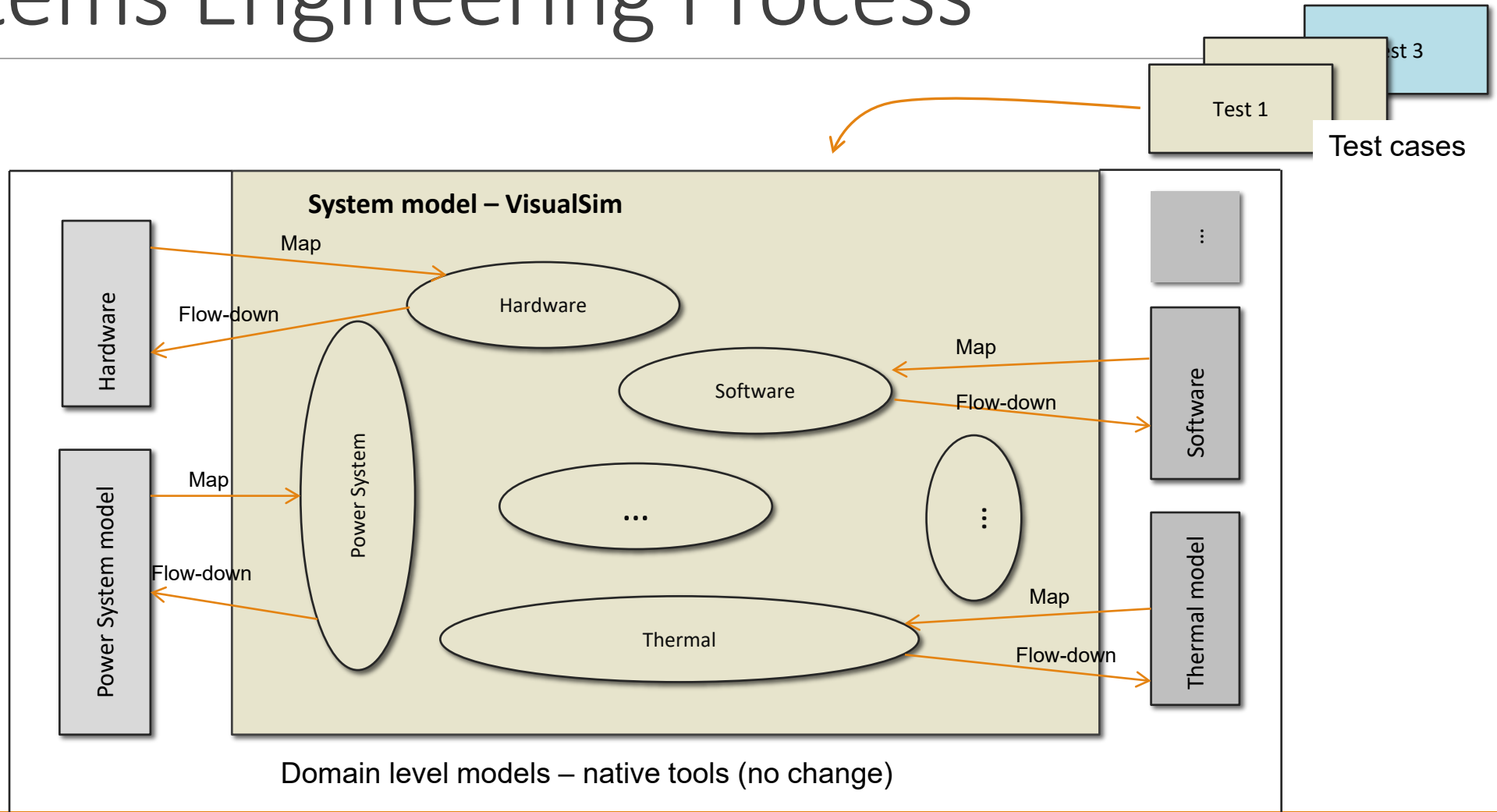
Y-axis: Latency (seconds) [0 to 7]

X-axis: Simulation Time (sec) [0 to 10]

Legend: Latency (ms) [red line]



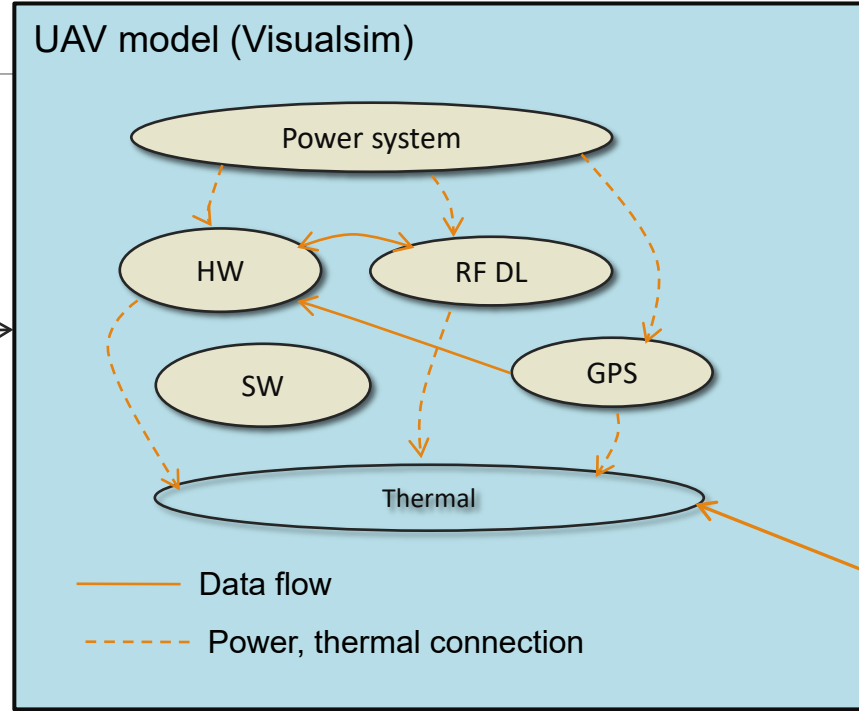
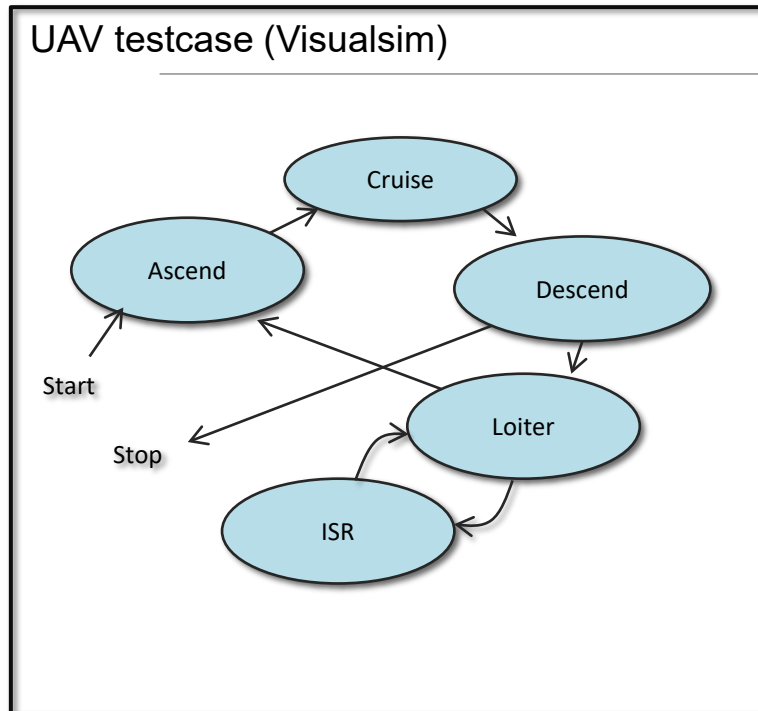
# Systems Engineering Process



VisualSim has capability to model most sub-systems/domains



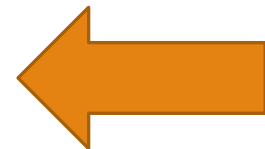
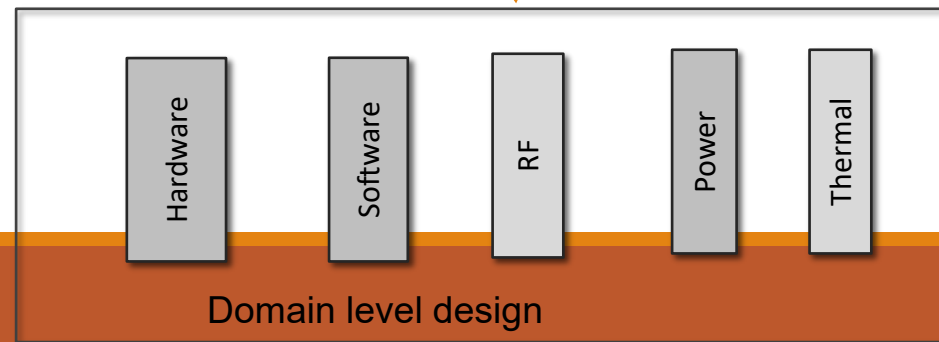
# Aero Example



- HW and SW blocks form Avionics
- RF Datalink include transmitter and receiver
- Key performance of Avionics and RF Datalink are constrained by input power and temperature
- GPS performance (sensitivity) is constrained by RF interference caused by RF Datalink and temp

Mechanical structure (fixed for given model)

Testcase determines power available with respect to time to functional blocks under simulation



# Distributed System Analysis

---

## System details

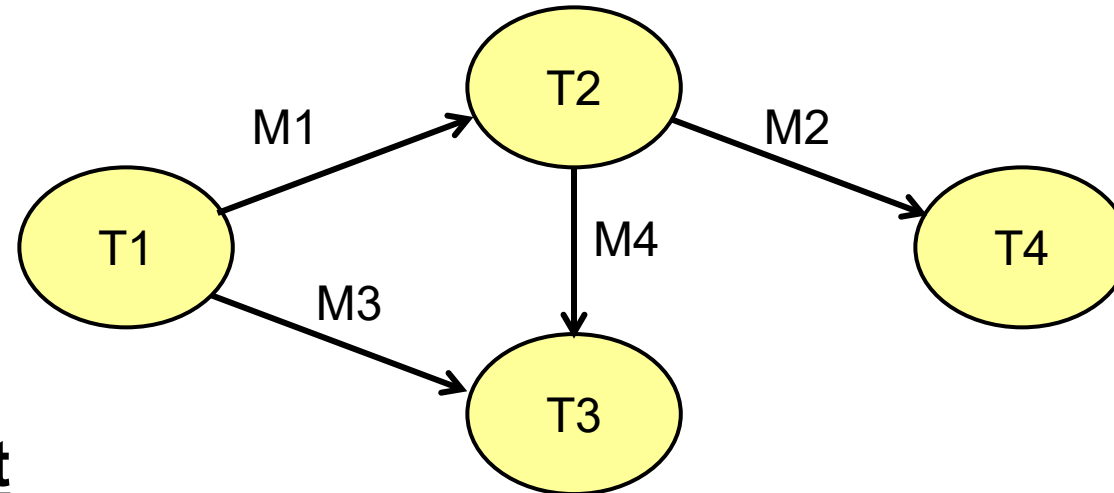
- Multi-independent processing computers
- Software tasks distributed across these computers
- Connectivity across multiple shared networks

## Analysis

- Optimal Routing Table configuration
- Capacity planning
- Software tasks and thread distribution
- Resource allocation

# Distributed System Logical Task Flow

---



## Initial Assignment

List of Tasks: T1, T2, T3, T4

T1, T2 -> ECU\_1

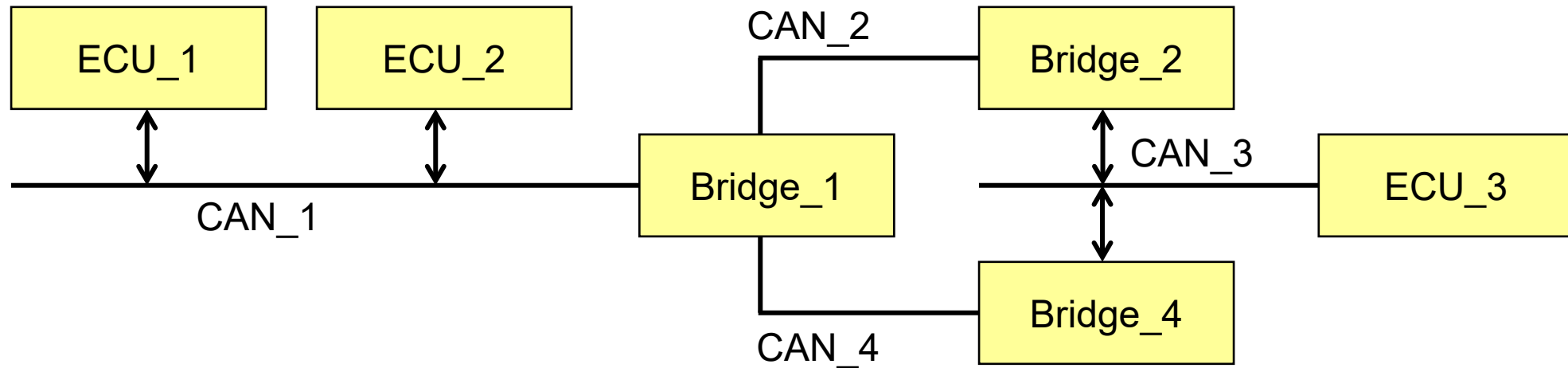
T3 -> ECU\_2

T4 -> ECU\_3

Messages: M1, M2, M3, M4

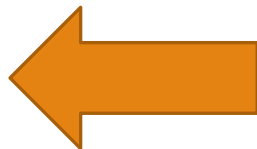
# Distributed System

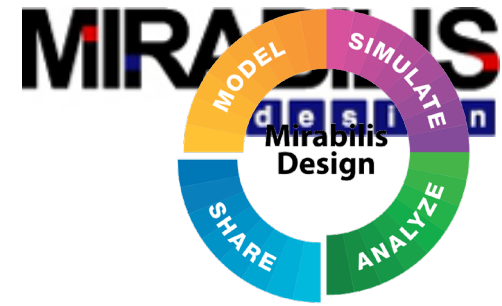
## uP-Computer/Bus Physical Mapping



### Message Assignment

- Messages: M1, M2, M3, M4
- Internal to ECU, if Source ECU == Destination ECU
- CAN\_N bus segments selection based on Source:Destination of Task,
- Routing is selected for shortest hops
- Dynamic allocation based on topology selection





# Applications of VisualSim- Software

---

## Software, RTOS and Scheduling

- Selecting the right scheduler and parameter definition
- Allocation of resources for the software task
- Validating the behavior of the security and safety features
- Parameter tuning or modifying software architecture

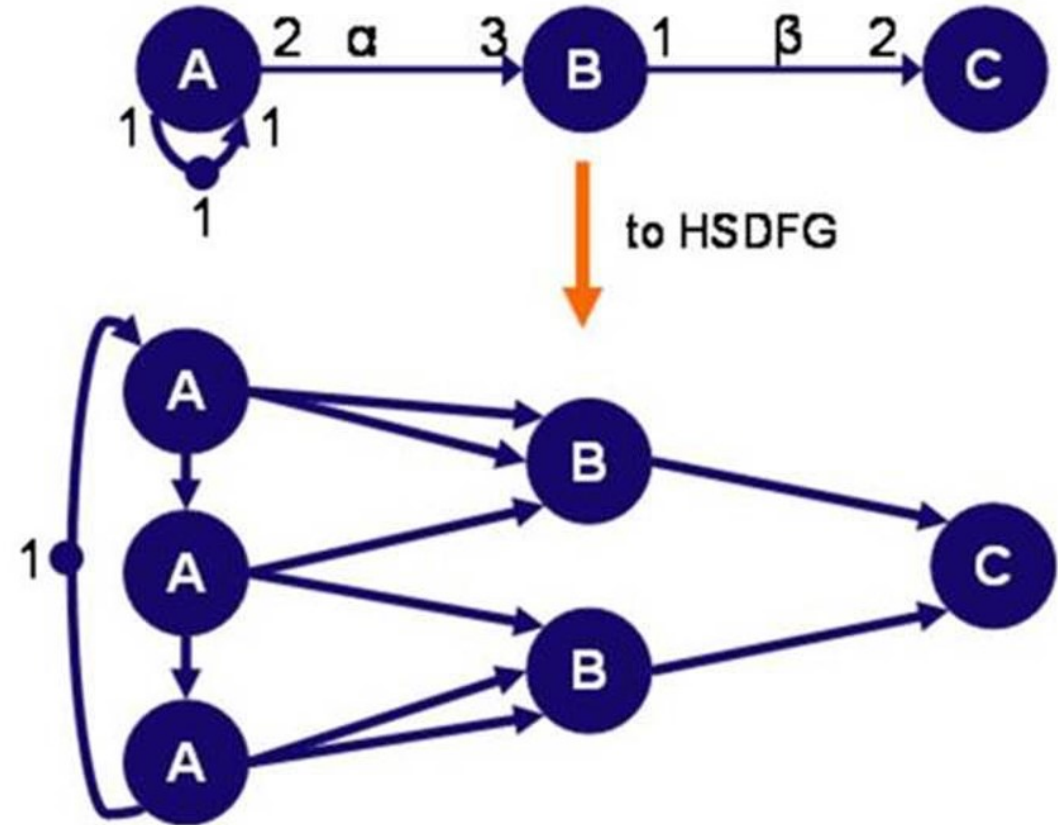
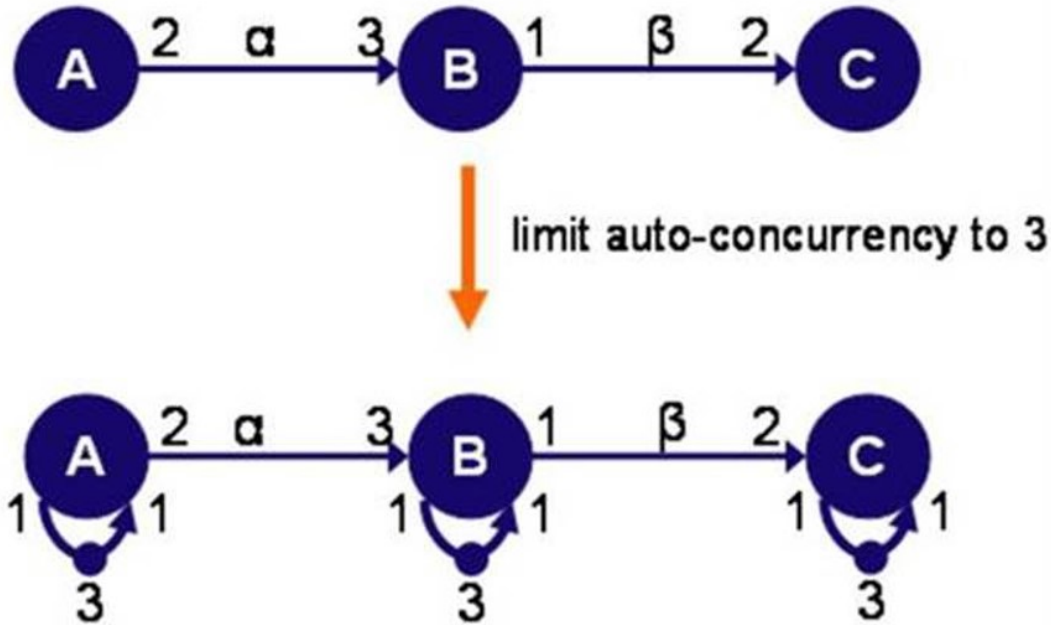
## Exploring the Firmware

- Timing deadline for the firmware
- Power reduction techniques
- Validating correctness of the algorithm
- Allocating tasks across multi-core platforms

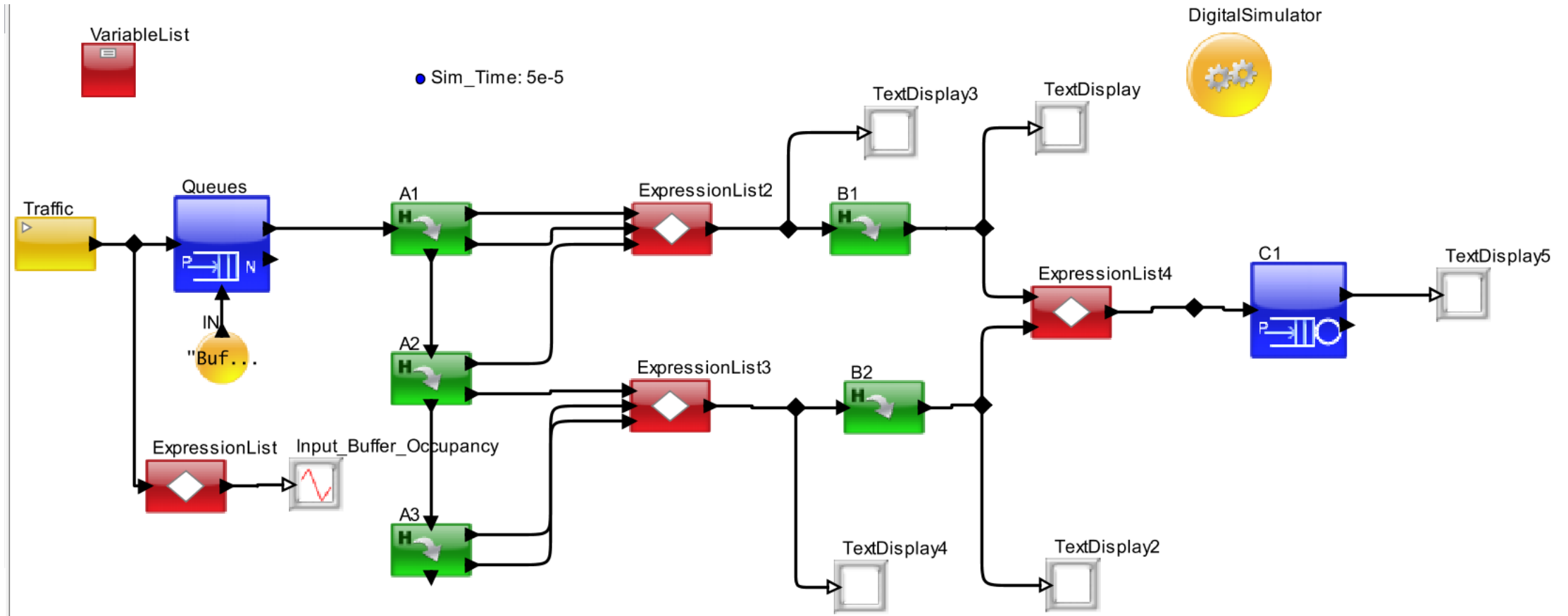
# Application Task Graph (Implementation can be in HW or SW)

Example

In this example, the auto-concurrency of the actors is limited to 3. This means that only three instances of each actor can be firing simultaneously.



# VisualSim Model of the Task Graph



# Types of Exploration

---

## 1. Task Flow through system

- a. Source and intermediate nodes
- b. Latency for each flow per type
- c. Usage- processing, buses and memory

## 2. Loading on individual modules

- a. Memory throughput and consumption
- b. Bus and switch usage
- c. Failure and loss of data
- d. Non-availability of resources and memory
- b. Any starved flows

## 1. Task Flow internals

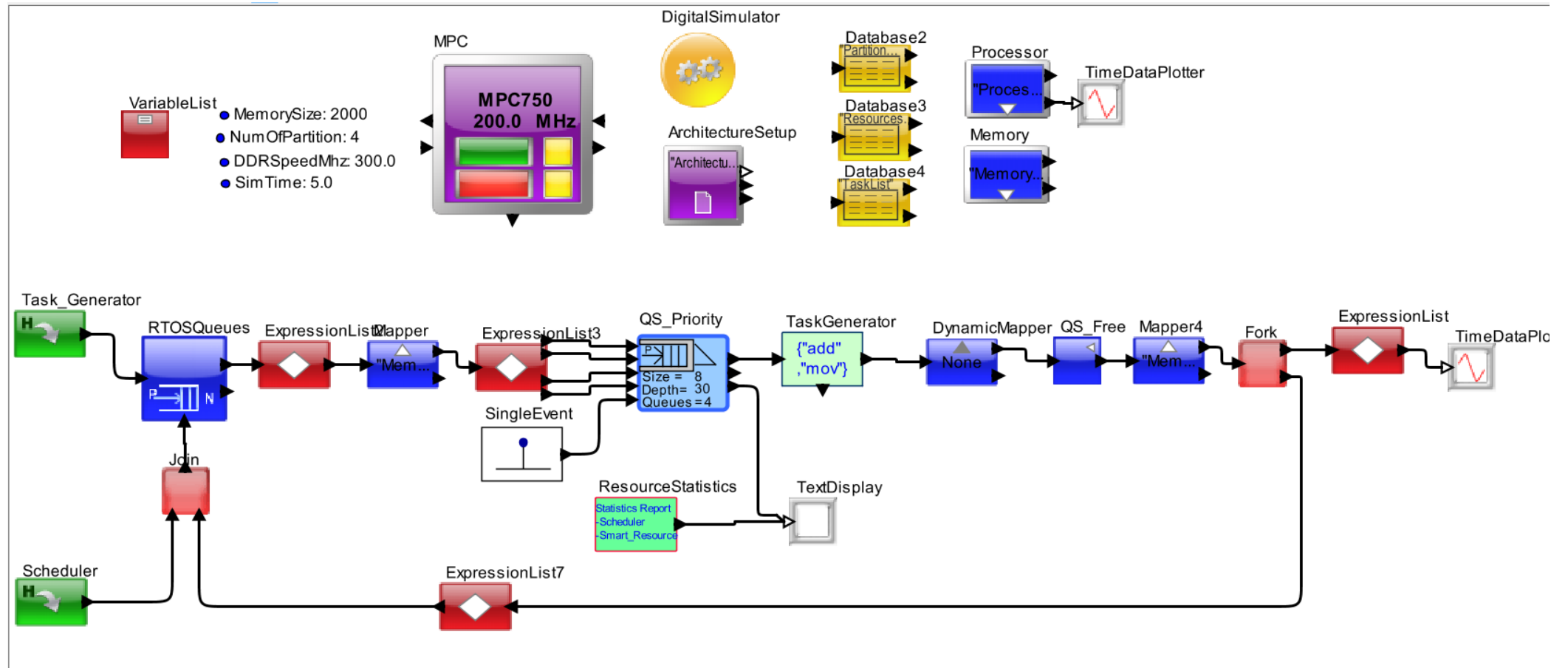
- a. Number of accesses per Node or Process
- b. Correctness of access
- c. Missing timing deadlines

## 2. Buffer and Scheduler

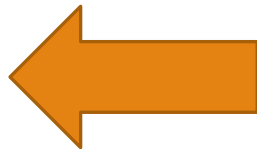
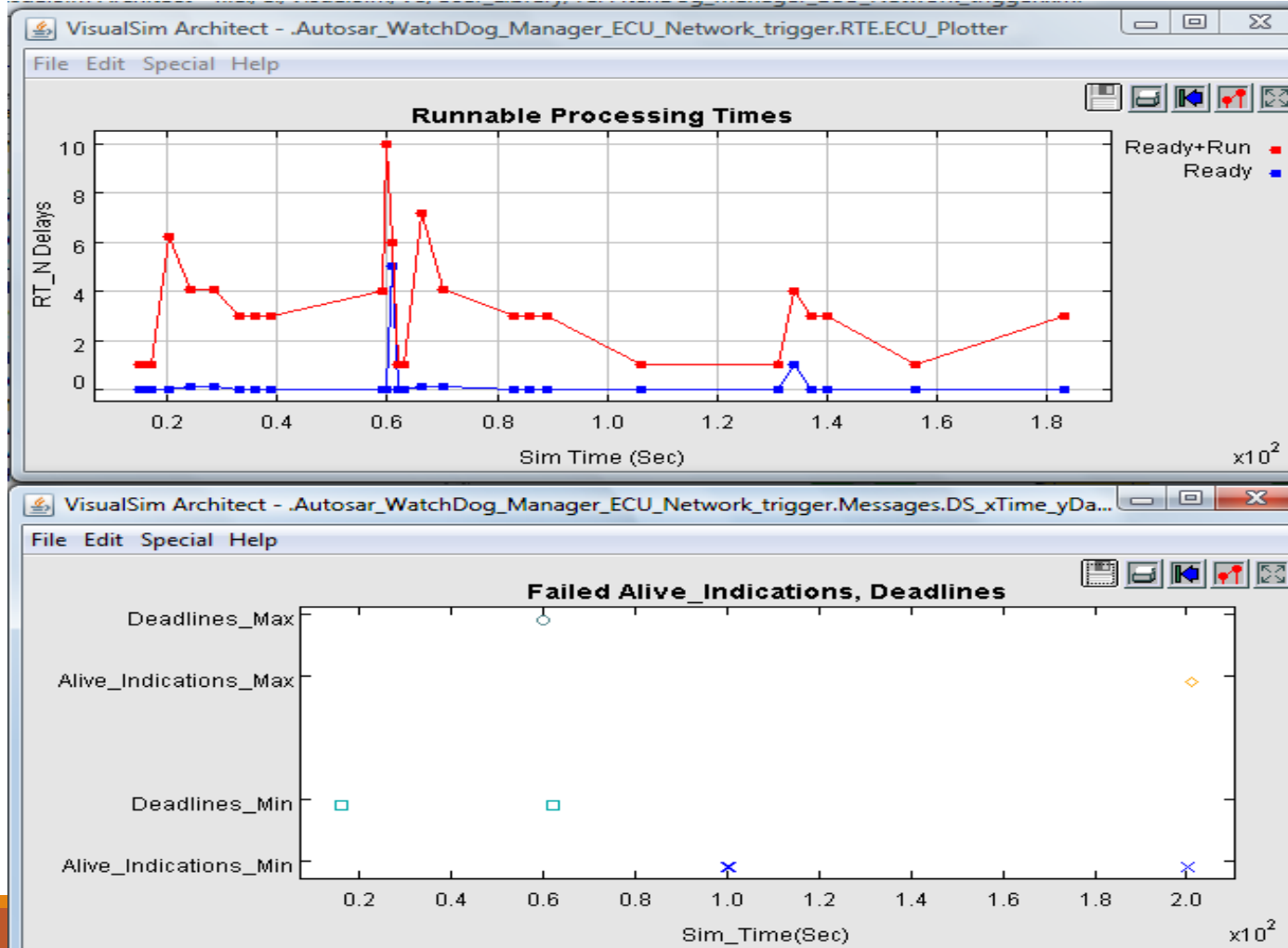
- a. Buffer size impacts
- b. Scheduler efficiency
- c. Waiting or stalled time
- d. Impact of task schedule offsets



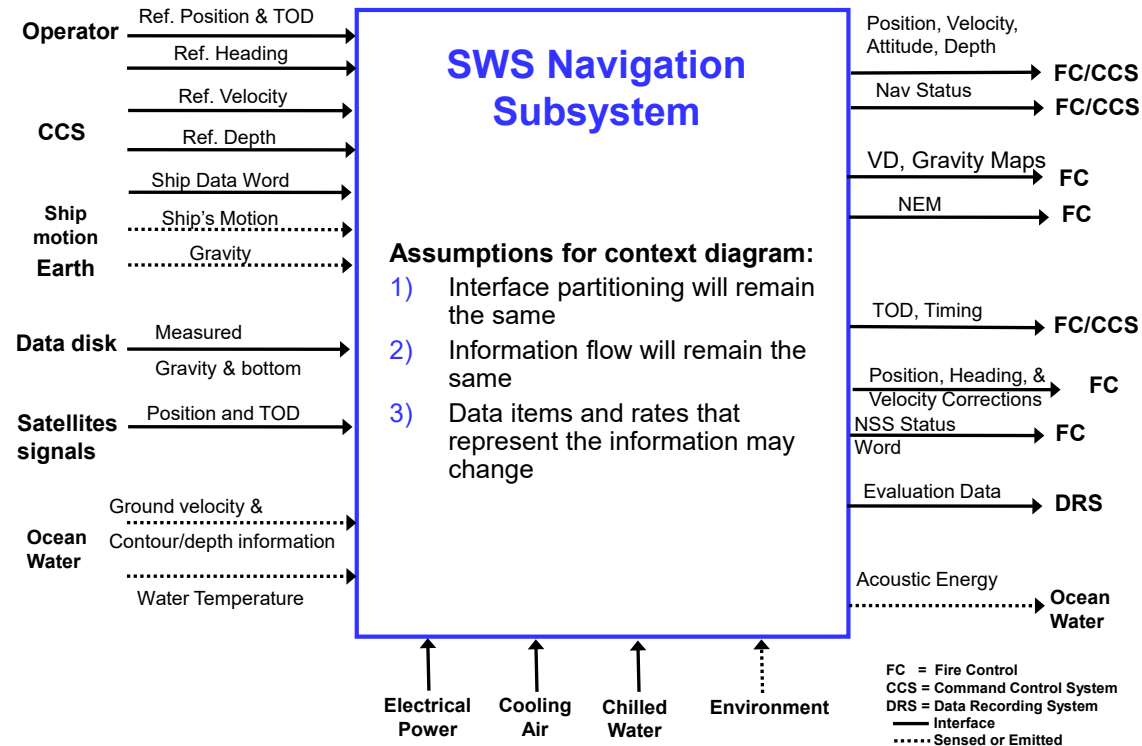
# VisualSim Software Modeling- RAD750



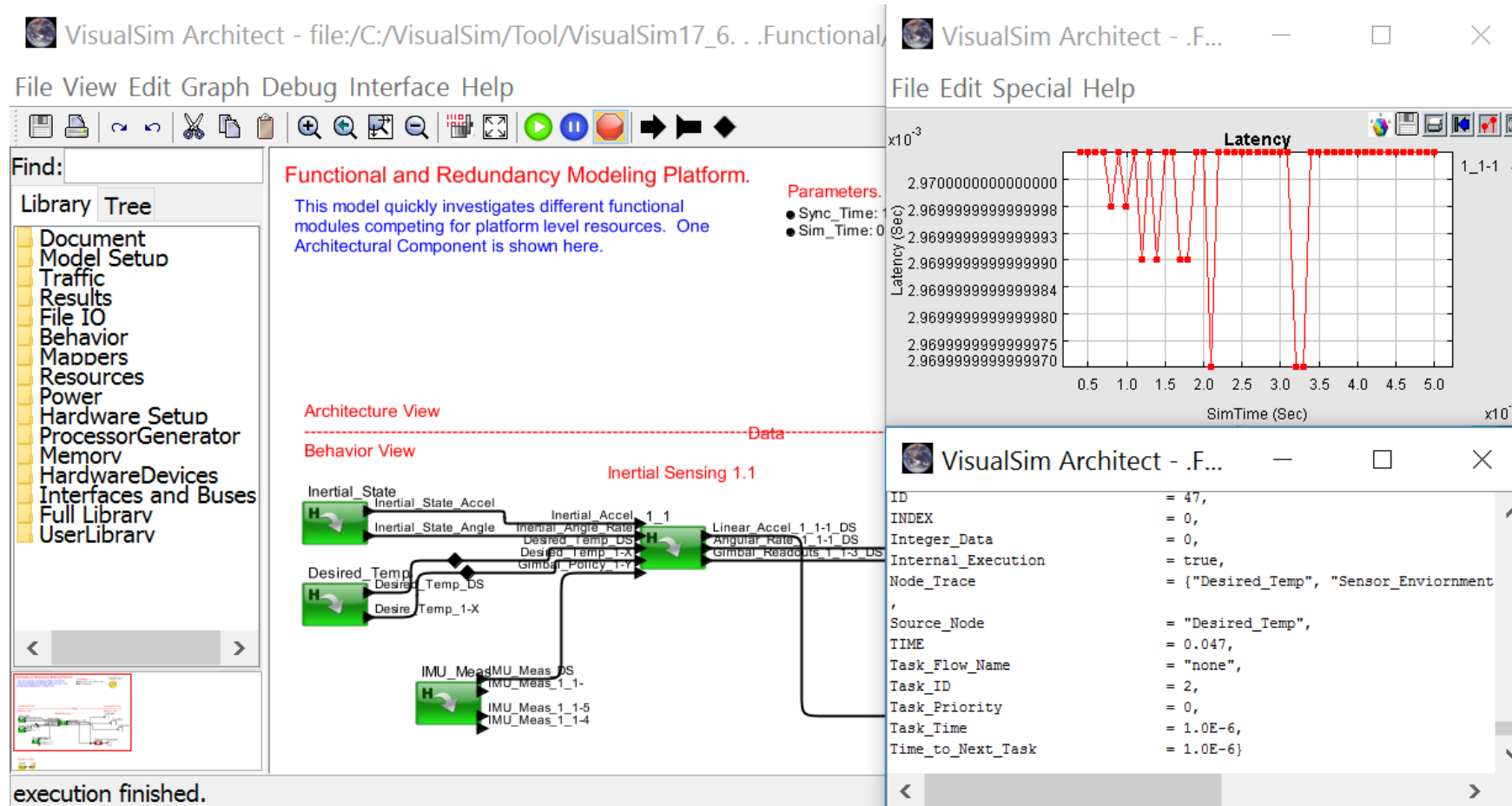
# VisualSim Latency and Functional Exploration Reports



# Designing Sub-systems- Concept and Functional



# VisualSim Conceptual Model

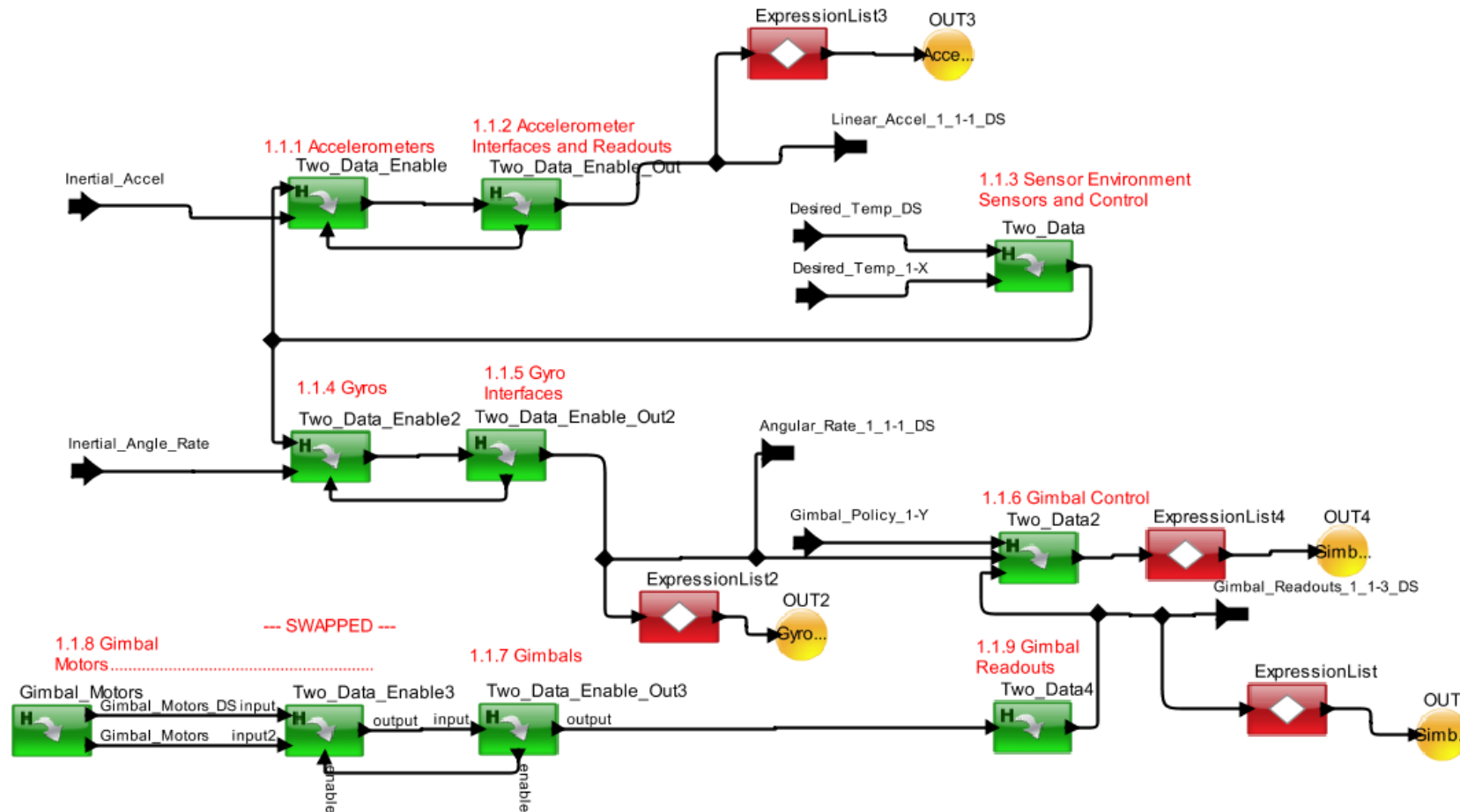


The screenshot displays the VisualSim Architect interface, divided into several panels:

- Left Panel (Library Tree):** A hierarchical list of components including Document, Model Setup, Traffic, Results, File IO, Behavior, Mappers, Resources, Power, Hardware Setup, ProcessorGenerator, Memory, HardwareDevices, Interfaces and Buses, Full Library, and UserLibrary.
- Top Panel (Architecture View):** Titled "Functional and Redundancy Modeling Platform." It contains a description: "This model quickly investigates different functional modules competing for platform level resources. One Architectural Component is shown here." Below this, a "Behavior View" shows a block diagram for "Inertial Sensing 1.1". The diagram includes blocks for "Inertial State", "Desired Temp", "IMU Meas", and "Linear Accel".
- Right Panel (Graph):** A line graph titled "Latency". The y-axis is "Latency (Sec)" ranging from  $2.9699999999999970 \times 10^{-3}$  to  $2.9700000000000000 \times 10^{-3}$ . The x-axis is "SimTime (Sec)" ranging from 0.5 to 5.0. The graph shows a red line with square markers that fluctuates between approximately 2.9699999999999975 and 2.9700000000000000 seconds over time.
- Bottom Right Panel (Code View):** A list of parameters for a task, including ID, INDEX, Integer\_Data, Internal\_Execution, Node\_Trace, Source\_Node, TIME, Task\_Flow\_Name, Task\_ID, Task\_Priority, Task\_Time, and Time\_to\_Next\_Task.

At the bottom of the interface, the text "execution finished." is visible.

# View Internals of the Inertial Sensing



# Designing with FPGAs

---

Designers of FPGA systems can easily

- Profile interaction between multiple FPGA, Processors and external memory
- Improve product quality, performance, cost and power with accurate predictions

Eliminate performance bottlenecks with FPGA system models

- Accurate analysis of memory interfaces and processor-to-FPGA Fabric interconnect
- Easy task partitioning to  $\mu$ Blaze, PPC and hardware accelerators for best system performance

# Analysis using Xilinx Zynq-7000

---

## Buffering, Latency and throughput statistics

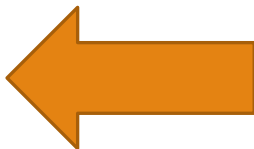
- For all buses
- Minimum, maximum, and mean
- DMA I/Os per second

## Memory

- Number of Reads and Writes to Memory
- Effective throughput to Memory

## Processor

- Hit-ratio, pipeline throughput, buffering and memory required



# Hardware-Software Partitoning

VisualSim Architect - file://E:/VisualSim/VisualSim16\_64\_Jun...artitioning/SoC/Power\_Perf\_example.xml

File View Edit Graph Debug Interface Help

Find: Library Tree

- Document
- Model Setup
- Traffic
- Results
- File IO
- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- Processor Generator
- Memory
- Hardware Devices
- Interfaces and Buses
- Full Library
- UserLibrary

### Multimedia System Design

**Scenarios**

- Scenario (1) : Simple application (traffic model)
- Scenario (2) : HW-SW partitioning (rotation algo)

**Top Level Parameters**

- Board\_Name: "Streaming\_Board"
- Select\_Scenario: 2
- Sim\_Time: 0.02

**Simulator Engine**

- Digital\_Simulator
- Init\_Memory
- Bus\_Name: "TOP\_LEVEL\_AXI"
- AXI\_Speed\_Mhz: 166.0
- Architecture\_Name: Board\_Name

**HW Architecture**

SingleEvent → Processing → Power Usage (Power Percent)

Power\_Manager ("Manager\_1") → InstPowerPlot → Battery\_Charge → TimedPlotter

MOVE\_Instruction\_Set, ARM9\_Instruction\_Set

ARM\_SUBSYSTEM → Mem\_Select (Memory\_Select) → HW\_ACC (Accelerator)

● Enable\_Pwr\_Gating: false

AMBA\_AXI BUS → SDRAM, Flash, OUT4 (Arch...)

**SW Architecture**

**Scenario 1**

Trans\_Source → SoftGen ("add", "mov") → Scenario → call\_func() → Delay (0.00000) → Software\_Mapper (None) → Software\_Mapper2 (None)

use case A, application layer, driver layer

Defining\_Flow->Mapping->Software Mapper

**Scenario 2**

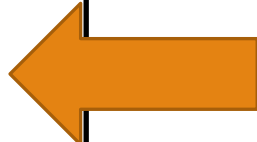
Sensor → Read Frame → Decode → Frame Video\_post\_procRender → Frame Format conv → partitioning candidate (Rotate Frame) → display → ANALYSIS\_PLOT\_SW

use case B

**Bus Activity and Statistics**

Timing\_Diagram (Arch State Plot)

Architecture\_Setup2 → Utilization → Display\_Text → Stati...

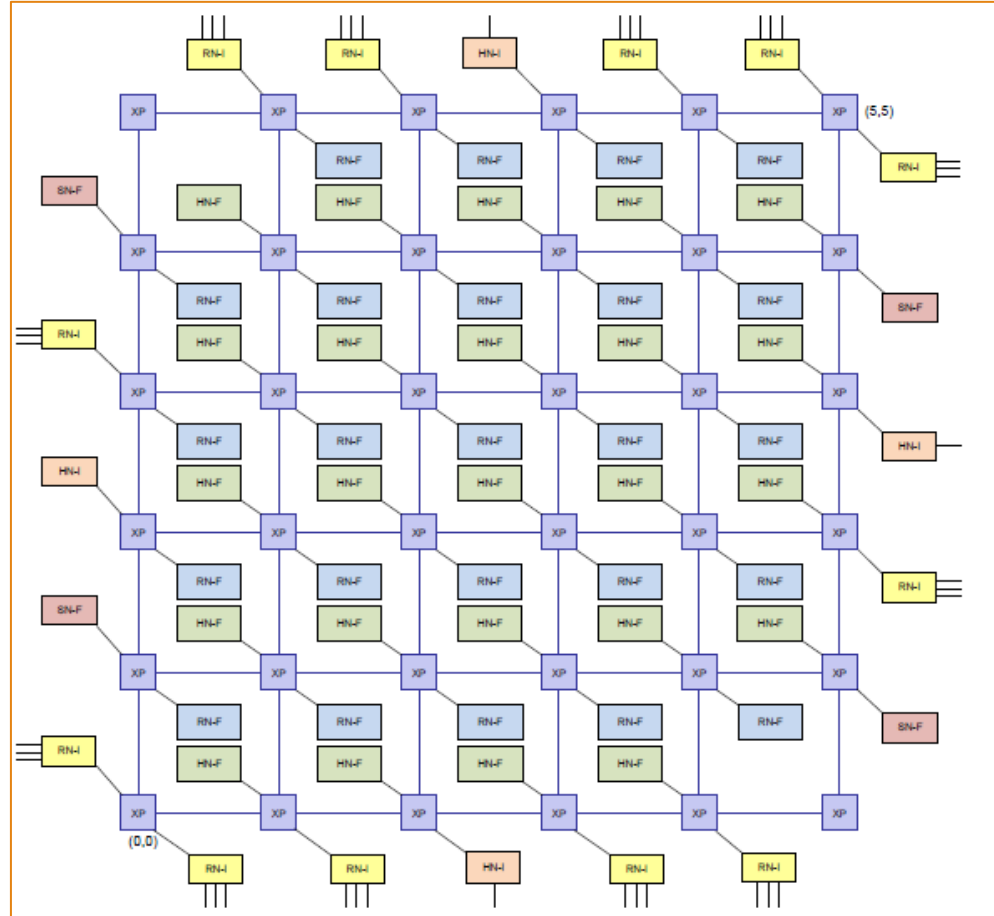




# Creating Extremely Large Systems

## Devices in the SoC

CMN600 NoC 6\*6 and 8\*8  
 8 AI Processor Cores  
 64 ARM Z1 Cores  
 16 GPU  
 4 DRAM5 and 2 HBM2.0  
 DMA  
 Legacy devices via AMAB AXI  
 Coherent PCIe to 2<sup>nd</sup> SoC  
 DMA and peripherals



## Target to achieve

Power <55.0W  
 Number of frames in 20 ms > 28K  
 Memory Bandwidth: 1 terabyte  
 Three Explorations

1. AI and Tasks deployed in Software
2. AI tasks to custom AI processor
3. Power management to reduce power

# Scalable SoC Model

**Parameters**

- X\_dimension: 2\*Row!
- Y\_dimension: 2\*Column!
- HNF\_QoS\_Register\_Value\_Selection: Generic
- SimTime: 150.0e-6
- Device\_Threshold: 50
- Flit\_Size: 256
- Router\_Frequency: 800.0e6

**Parameters - Details**

X\_dimension - number of Rows -> Linked with XP Module  
 Y\_dimension - number of Columns -> Linked with XP Module  
 HNF\_QoS\_Register\_Value - Value selection from database-> Used in HNF  
 SimTime - simulation runs for specified time ( in seconds )-> Linked with Digital Simulator

**Databases**

Database8, Database9, Database5, Database7, Database10, Database6, Database2, Database4, Database3, Database6

**Edit parameters for XP**

Block\_Documentatio... Enter User Documentation Here

nInstances:  $X\_dimension * Y\_dimension / \text{Total number of XP modules that we}$

instance: 0

showClones:

Ingress\_Buffer\_Size: 1000/\*The maximum number of packets which can be stored at in

VC\_Buffer\_Size: 10/\*The maximum number of packets which can be stored at egr

Router\_Frequency: Router\_Frequency/\*Top level parameter\*/

Node\_Name: "R\_"+x\_val+"\_"+y\_val/\*DO NOT MODIFY\*/

Power\_Manager\_Name: "Manager\_1"/\*Power Manager name\*/

Single\_bit\_error\_ratio: 0.4/\*Error ratio for Single bit Corruption\*/

Double\_bit\_error\_ratio: 0.2/\*Error ratio for Double bit Corruption\*/

VLAN\_Q: 4/\*Total number of VLAN\_Q available\*/

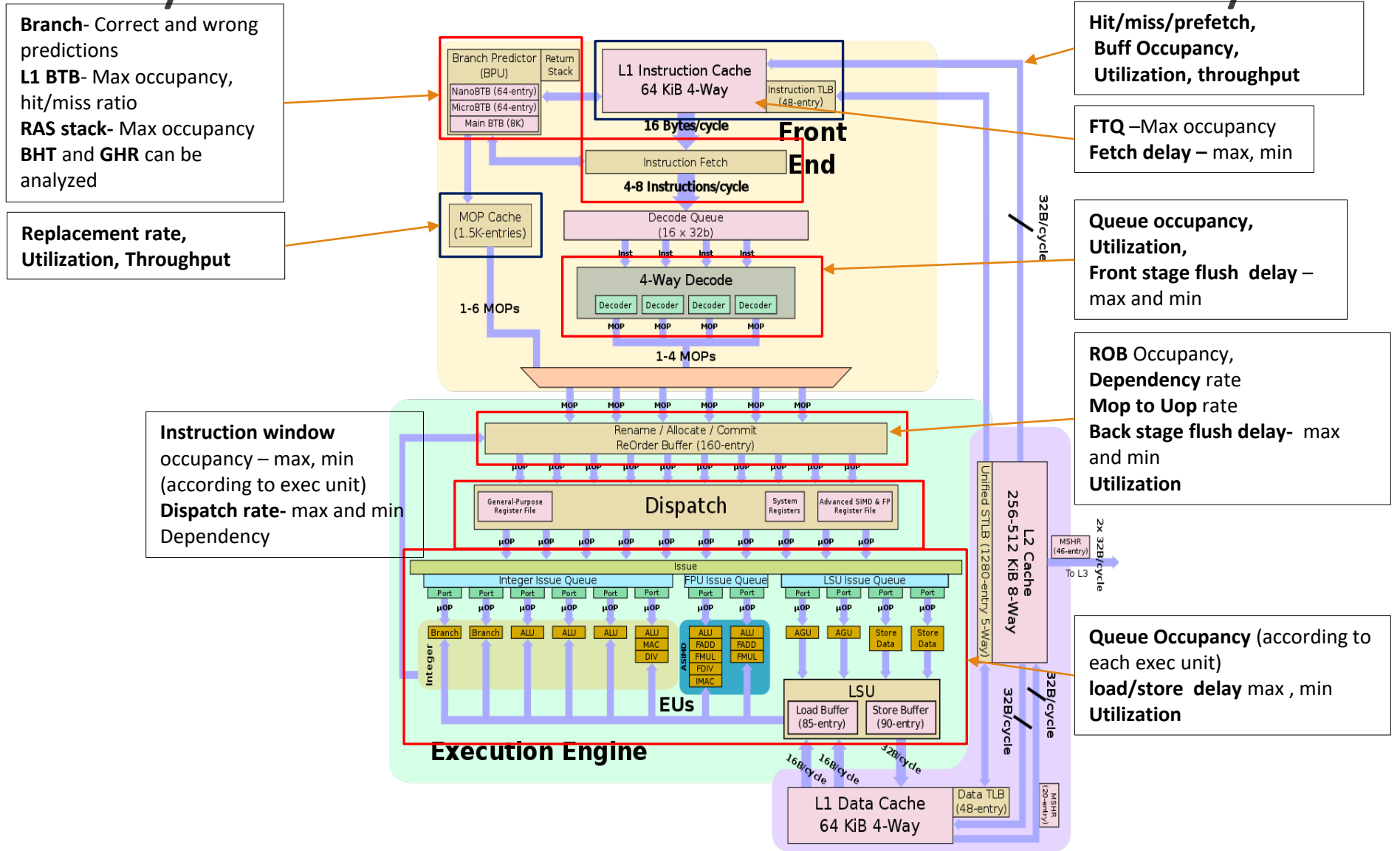
y\_val: instance\*Y\_dimension/\* DO NOT MODIFY \*/

x\_val: instance/Y\_dimension/\* DO NOT MODIFY \*/

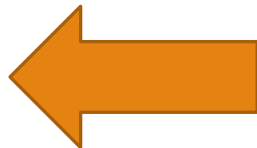
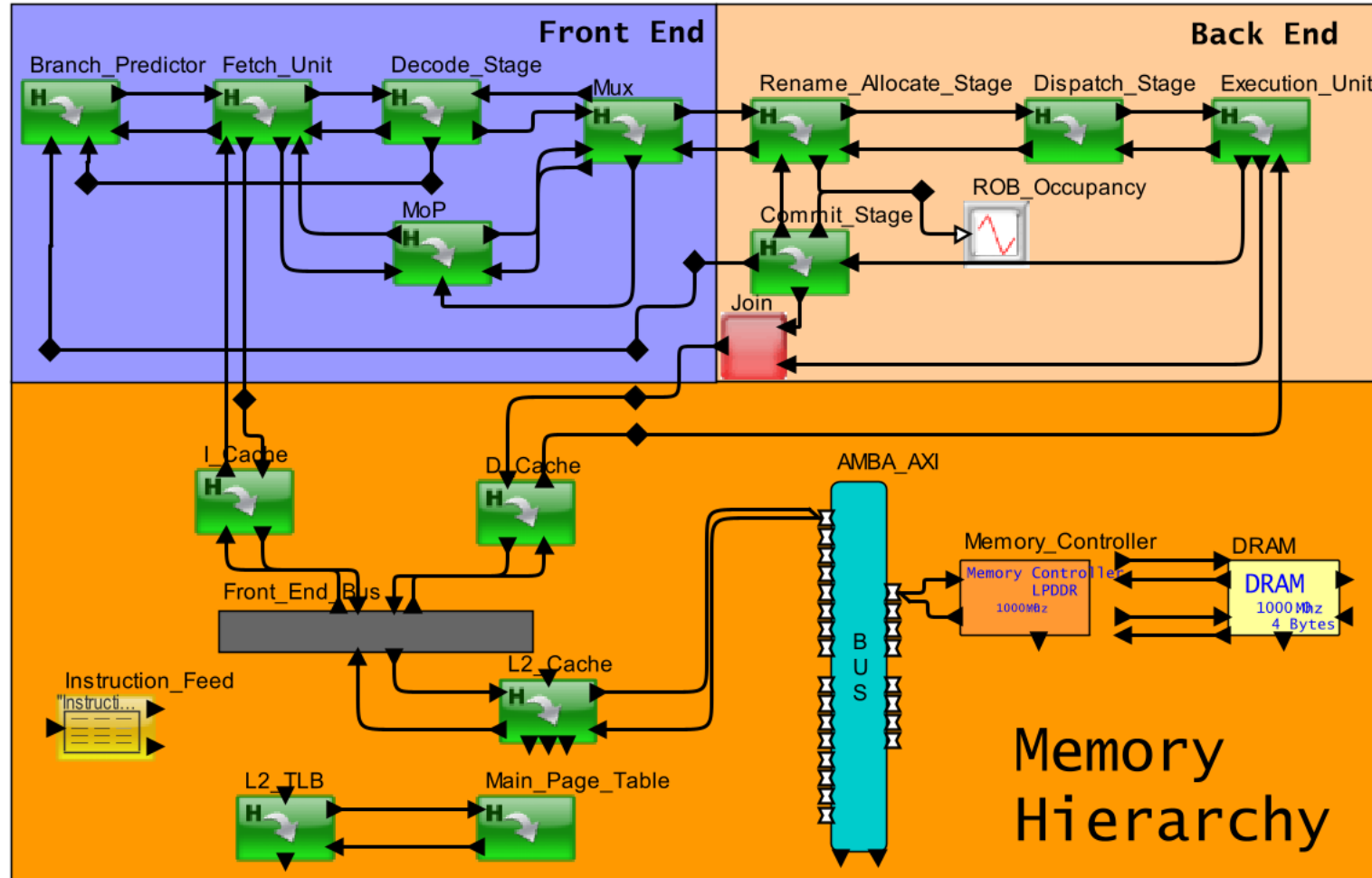
Commit Add Remove Restore Defaults Prefe

Define at run-time number of instances

# Cycle-Accurate Processor Analysis



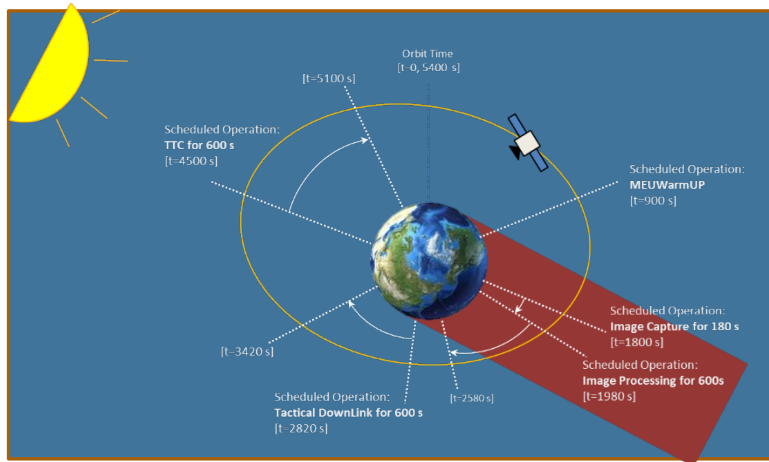
# Using VisualSim Cycle-Accurate Library to Create an ARM A77



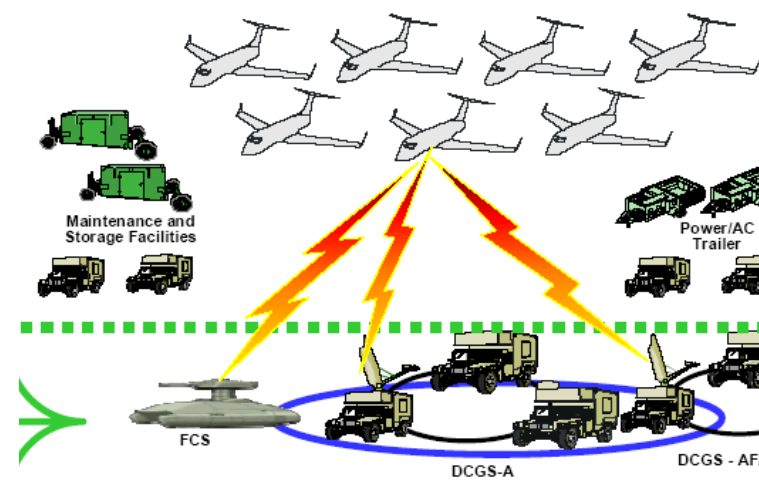
# Modeling, Simulation, Analysis and Recommendation using VisualSim

# Mission-to-Integration Analysis

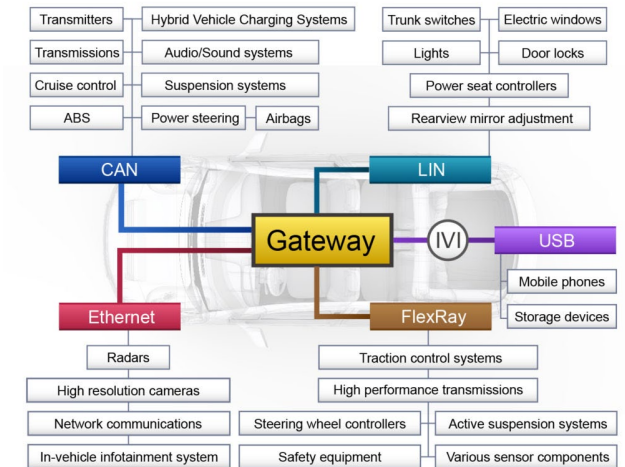
## Orbital-Level



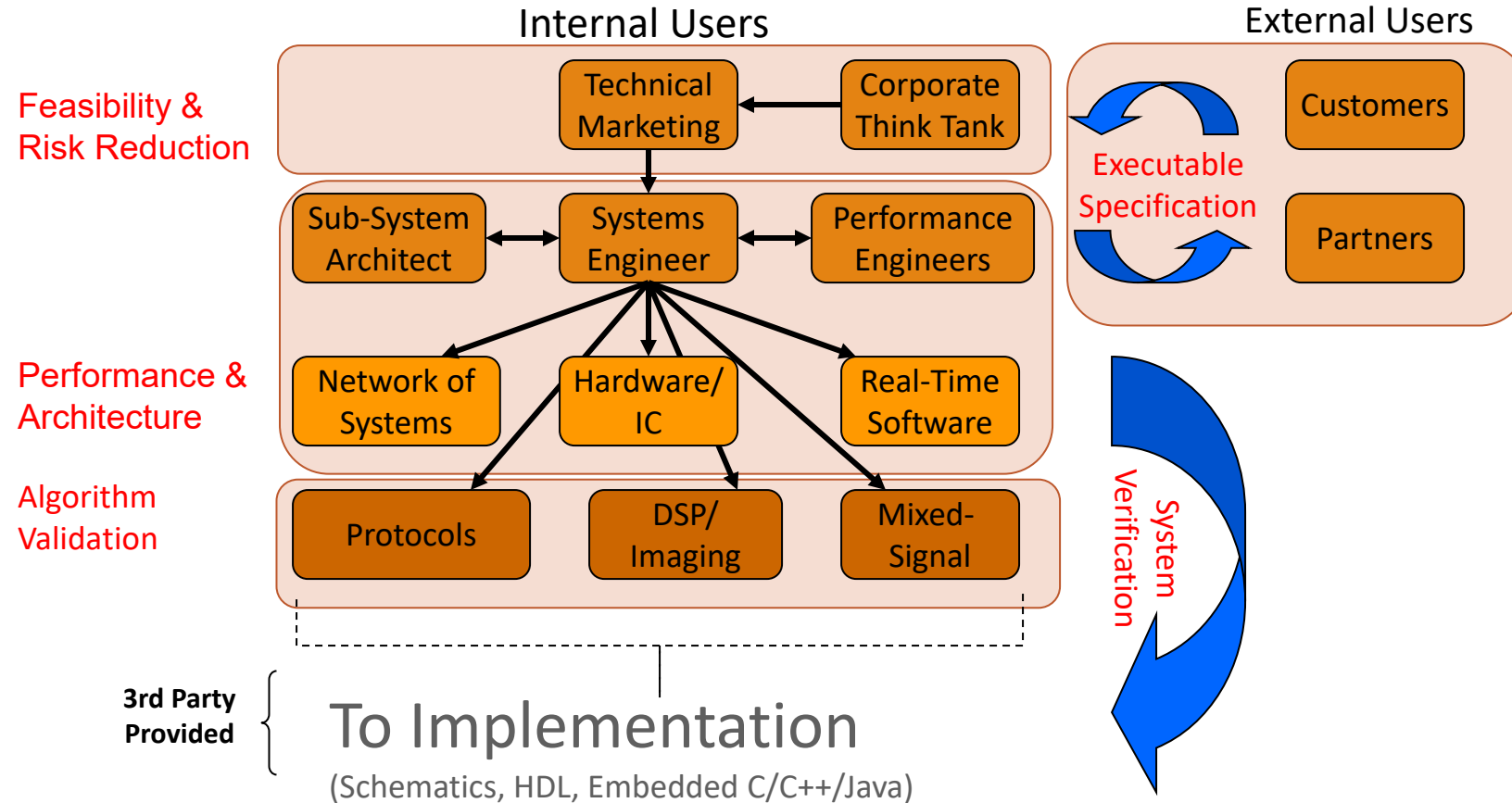
## Network-Level



## Integrated/Distributed System, Semi, Software And Boards



# Mirabilis Design Users and Technology

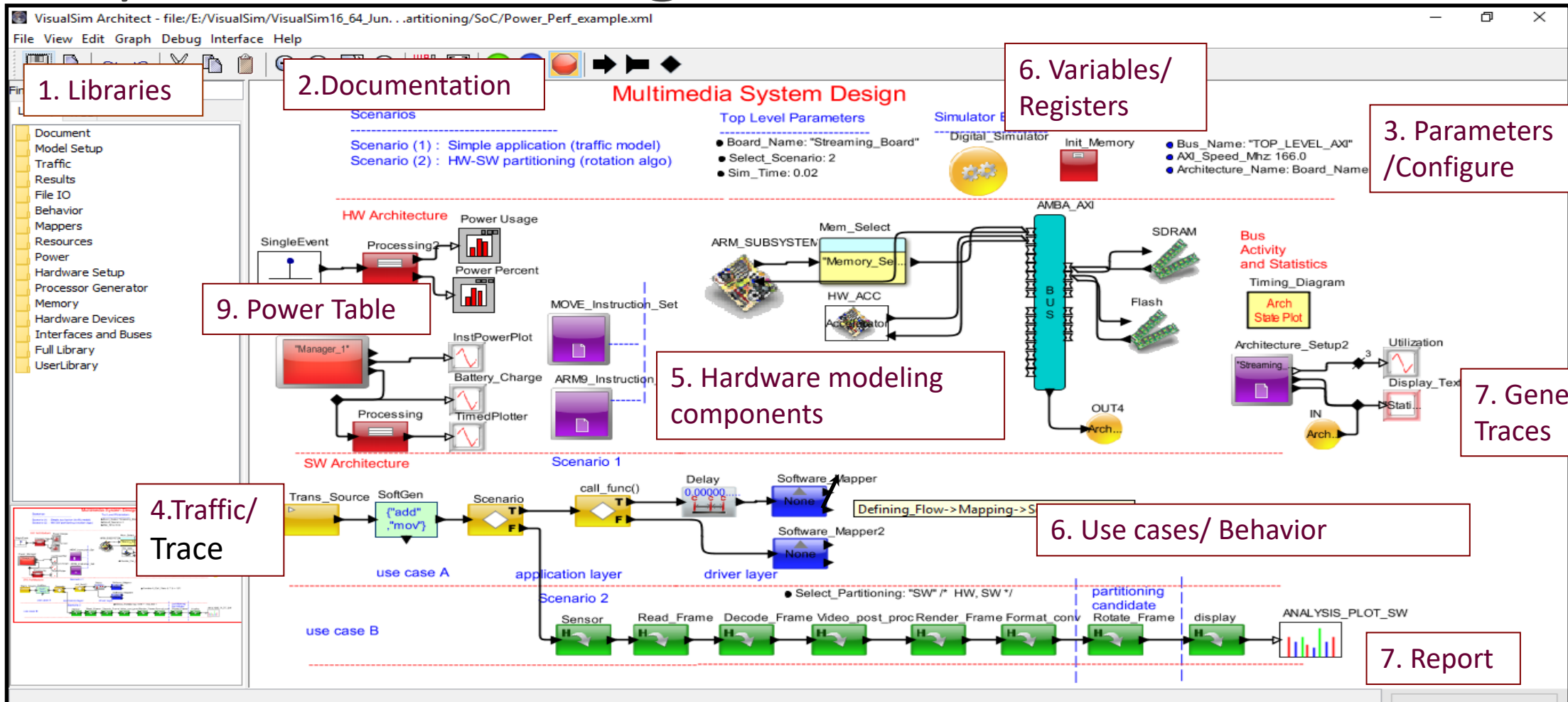


# Modeling Abstraction Determines the Level of Accuracy

Abstraction Level	Type of Library	Example	Accuracy	Limited by
Blocks	Generic	Resource-based	~85%	Available parameters
	Specific	AXI, DDR3, Processor	>97%	Branch prediction accuracy and minor proprietary details
Scripts	n/a	User's Hardware/ Software	95%~	None (can be exactly to the hardware)
Coding	n/a	C/C++/RTL/SystemC		



# System Modeling



# Simulate, Explore and Analyse

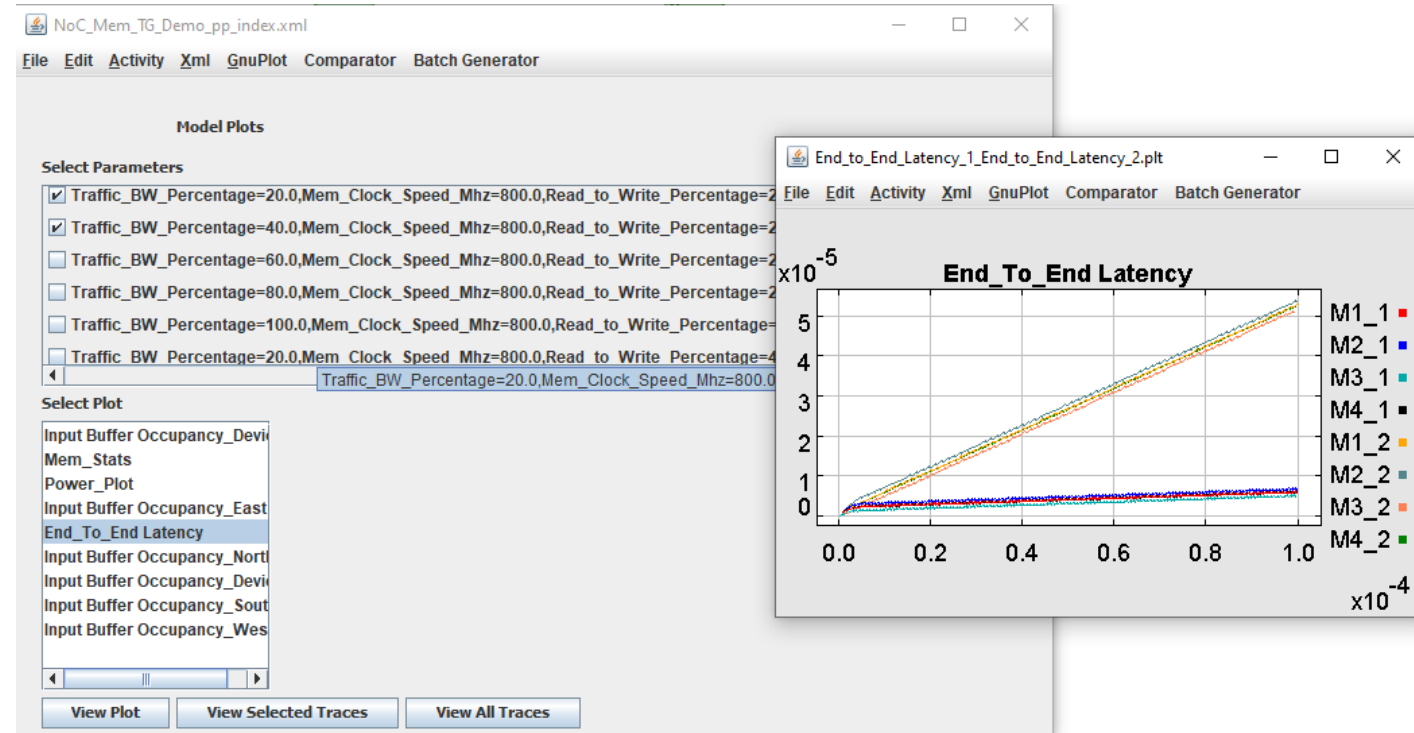
1. Generate parameter variations to simulate on multi-core

	A	B	C
1	Parameter	Range	Step
2	Traffic_BW_Percentage	{20.0;100.0}	20.0
3	Read_to_Write_Percentage	{25.0;100.0}	25.0

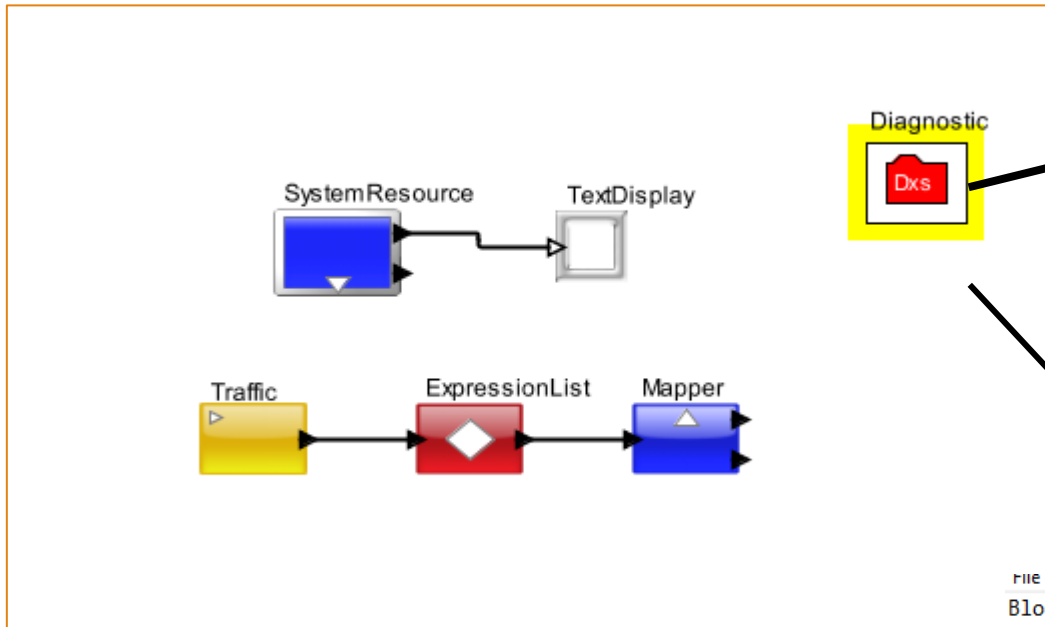
2. Setup the requirements and constraint file

	A	B	C	D	E
1	Block_Name	Stats_Name	Constraint	Value	
2	Smart_Resource	Latency	>	2	
3	Smart_Resource	Buffer_Occupancy	>	25	
4	Smart_Resource2	Utilization	>=	0	
5					

3. Compare Results across Multiple Runs



# Recommendations



Utilization	TIME	Occupancy	Occupancy	Occupancy	Occupancy	Latency	Buffer_Oc	Buffer_Ca	BLOCK_TaskID
50	4	0.745356	2	2.666667	4	3	4	30	SW_1
66.66667	6	0.943398	2	3.1	5	4	5	30	SW_1
75	8	1.17803	2	3.571429	6	5	6	30	SW_1
80	10	1.432644	2	4.055556	7	6	7	30	SW_1
83.33333	12	1.698322	2	4.545455	8	7	8	30	SW_1
85.71429	14	1.970567	2	5.038462	9	8	9	30	SW_1
87.5	16	2.246973	2	5.533333	10	9	10	30	SW_1
88.88889	18	2.526161	2	6.029412	11	10	11	30	SW_1
90	20	2.807292	2	6.526316	12	11	12	30	SW_1
90.90909	22	3.08983	2	7.02381	13	12	13	30	SW_1
91.66667	24	3.37342	2	7.521739	14	13	14	30	SW_1
92.30769	26	3.657814	2	8.02	15	14	15	30	SW_1
92.85714	28	3.942836	2	8.518519	16	15	16	30	SW_1
93.33333	30	4.228358	2	9.017241	17	16	17	30	SW_1
93.75	32	4.514285	2	9.516129	18	17	18	30	SW_1
94.11765	34	4.800544	2	10.01515	19	18	19	30	SW_1
94.44444	36	5.087078	2	10.51429	20	19	20	30	SW_1
94.73684	38	5.373844	2	11.01351	21	20	21	30	SW_1
95	40	5.660804	2	11.51282	22	21	22	30	SW_1
95.2381	42	5.947932	2	12.0122	23	22	23	30	SW_1
95.45455	44	6.235204	2	12.51163	24	23	24	30	SW_1
95.65217	46	6.522601	2	13.01111	25	24	25	30	SW_1
95.83333	48	6.810106	2	13.51064	26	25	26	30	SW_1
96	50	7.097707	2	14.0102	27	26	27	30	SW_1

file edit format view Help  
Block\_Recommendation 2.000000000000 sec

SW block is not sending out any transaction. Check the resource to make sure the logic is correct at 2.000000000000 sec  
 The loading on the SW is above the threshold at 4.000000000000 sec  
 The loading on the SW is above the threshold at 6.000000000000 sec  
 The loading on the SW is above the threshold at 24.000000000000 sec  
 SW is extremely busy and did not go below the threshold at 24.000000000000 sec  
 The loading on the SW is above the threshold at 26.000000000000 sec  
 SW is extremely busy and did not go below the threshold at 26.000000000000 sec

**Recommendation file**

# Software Performance Tuning

---

Process of accurately measuring the expected latency of a software code on a hardware or SoC platform using an architecture model

Used to plan the code sequencing and editing to maximize the processor efficiency

Eliminates the need for hardware boards for early testing and validation

# Methodology

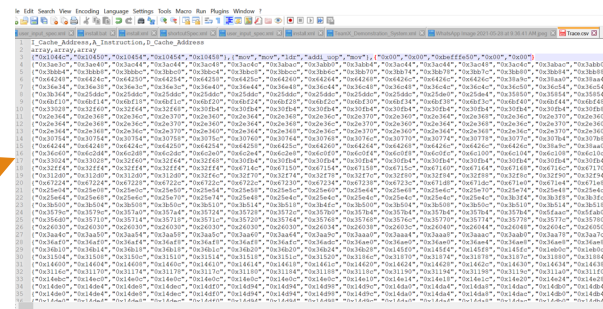
## AI and Software Code



Modify the code to improve performance  
And repeat loop

## Performance Reports

Delay_95_StDev_196s	= 4.39E-6
Latency_Value	= 7.46E-6
Mean_95_Confidence	= 3.61E-7
Mean_Value	= 4.01E-6
Min_Value	= 3.62E-7
StDev_Value	= 2.24E-6



Translate into Interim trace



Execute Software trace on SoC Platform

Machine A



Machine B



x86Machine

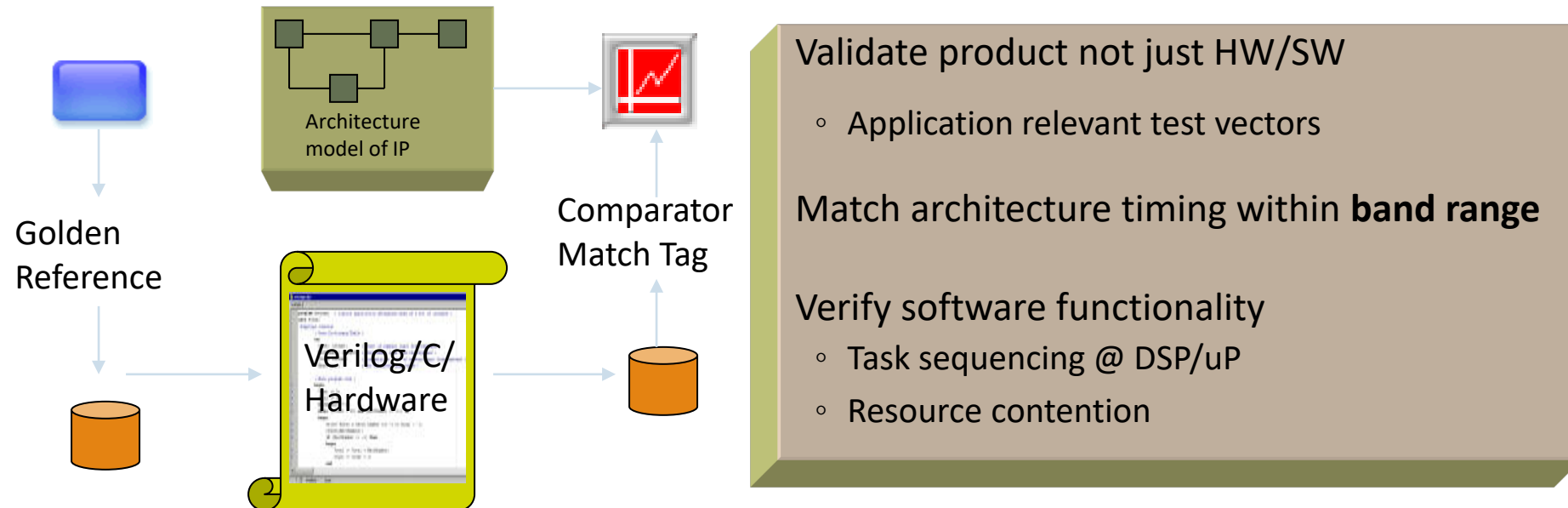
Generates output for ARM platform

ARM Machine

.ARM File

Compile software to target hardware

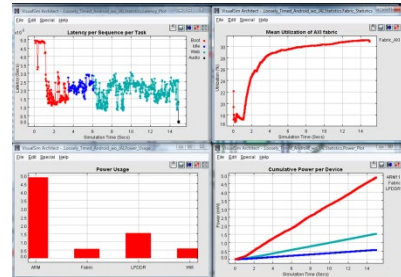
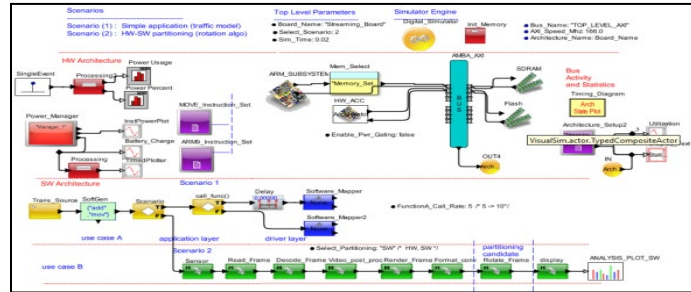
# System Verification



Eliminate product failure by maximizing relevant verification

# Creating Early Visual Demonstrator and Executable Specification

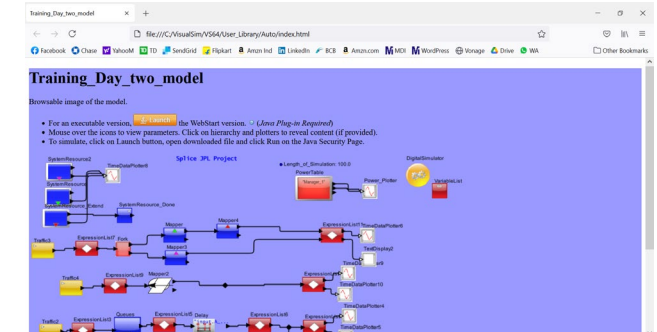
## System Architects



VisualSim Architect

- Architecture Exploration, power and Performance analysis
- Dynamic use case and application profiling
- Fault and Reliability analysis

## Client/Supplier



VisualSim Executable Specification

- Models can be viewed, modified and simulated within Web Page without a local install
- Experiment with different traffic and use cases.

# Performance, Power and Functional Analysis



# Purpose of System Modeling

---

- Select the right platform
  - ✓ Processor, FPGA or SoC
  - ✓ Hardware-Software partitioning
  - ✓ Trade-off power, performance and functionality
- Develop full system prototype
  - ✓ Visibility into complete system operations
  - ✓ View both implementation agnostic and effects
- When to perform system simulation
  - ✓ Identify capacity limitation and bottlenecks
  - ✓ Performance, Power or Functionality is non-deterministic

# Key Terms and Analysis

## Architecture Exploration

- Is the process of evaluating the specification prior to development

## System modeling

- Construct virtual model to represent functionality, timing and power without the implementation code

## Trade -off

- Select right configuration and parameters to meet requirements
- Evaluate task mapping, power vs timing, hardware vs software, distributed vs centralized etc.

## Performance Analysis

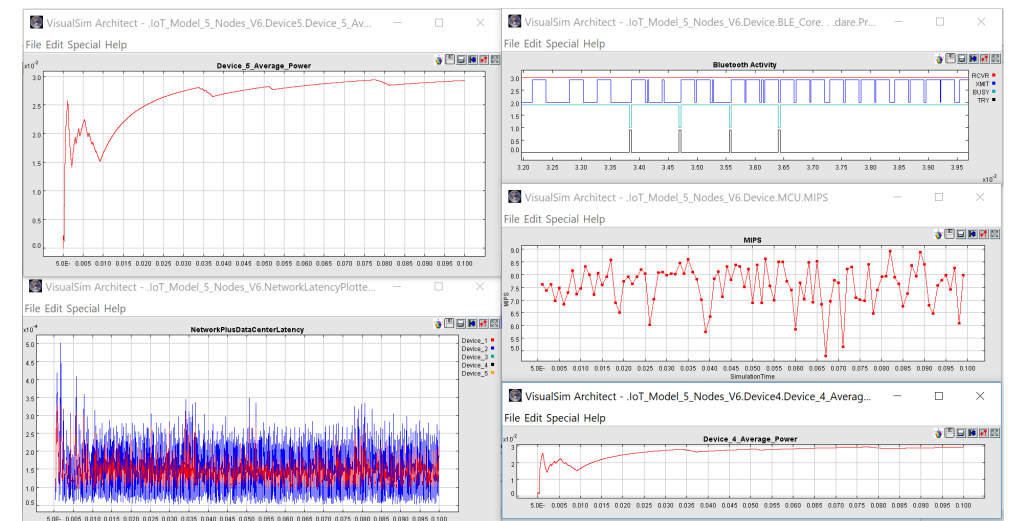
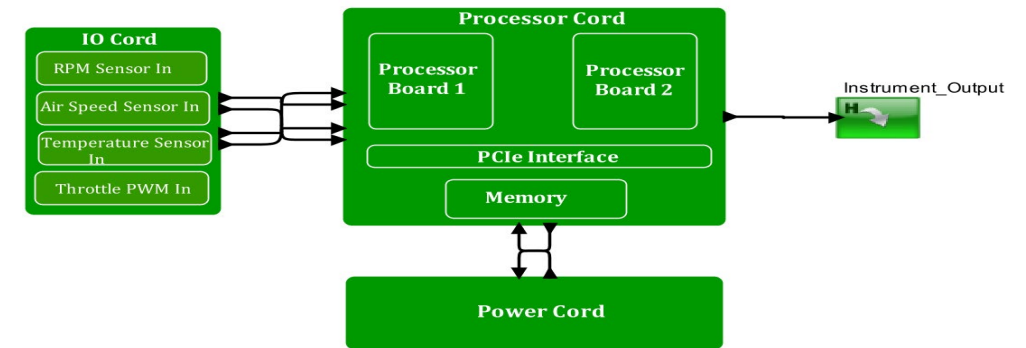
- Buffer usage, utilization, throughput and latency

## Power Measurement

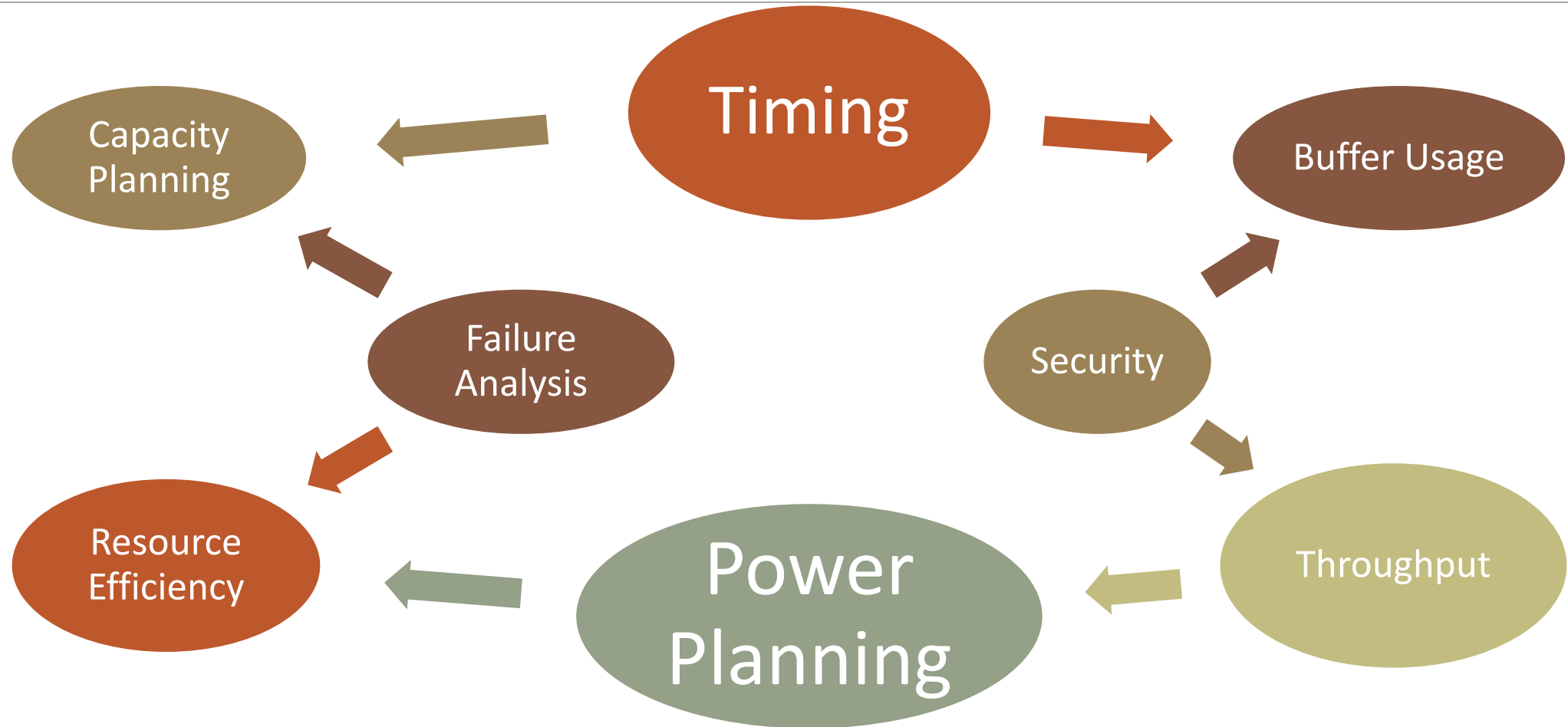
- Peak and average power, device and task energy consumption

## Functional Correctness

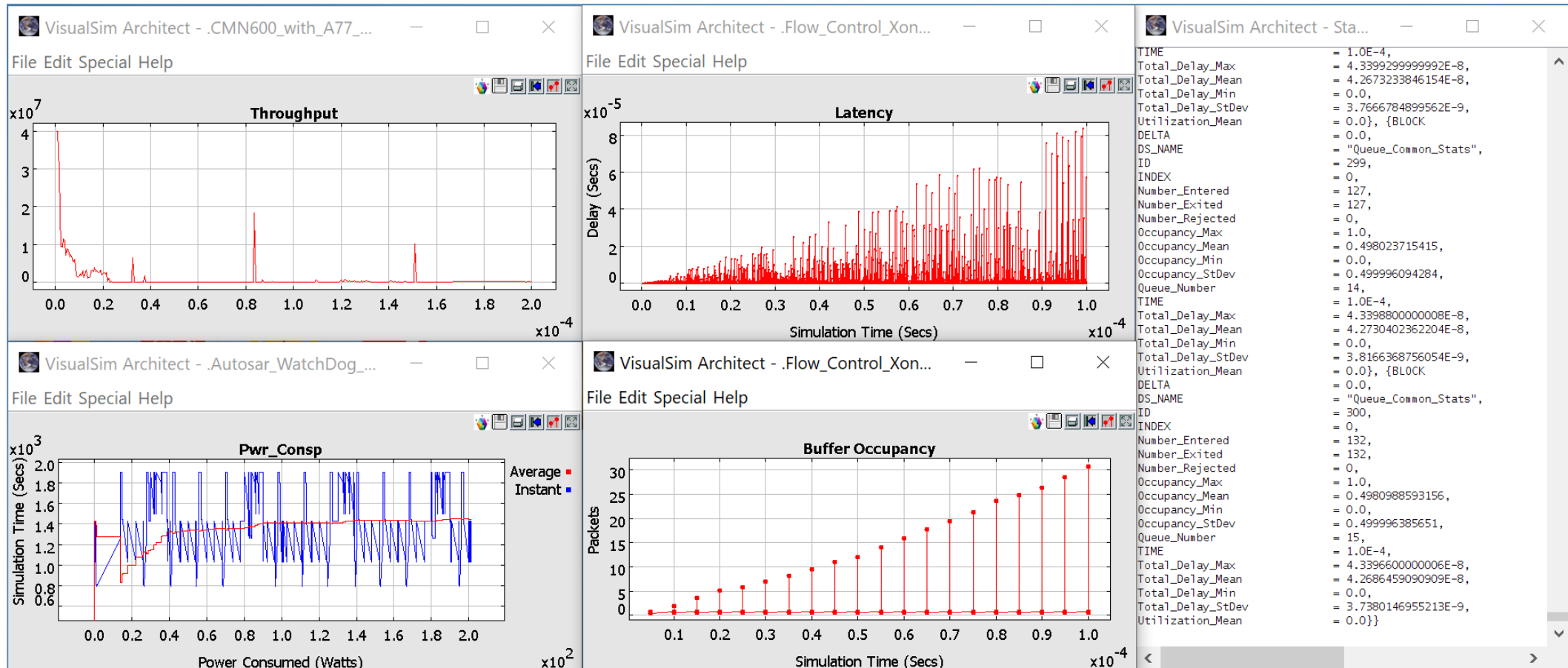
- Arbitration, failure analysis, task scheduling and task graph



# Analysis to Identify Bottlenecks and Optimize System Specification



# Statistics and Plots for Accurate Analysis



# VisualSim Solution Details

---

## Rapid prototype of systems

- Use Library of modeling components of pipeline, SoC, software and system
- Make decision with Reports and metrics for performance, power and functionality

## Select IP blocks and configure

- Partitioning algorithms onto cores/accelerators to meet requirements
- Intelligent Diagnostics provides recommendation on requirements-based feasibility based

## Performance testing of AI/Application software

- Run traces of the C/C++/Python code on cycle-accurate SoC architecture model

## Integration with design flow

- Generate dynamic documentation for use as specification
- Connect to FPGA boards for early system verification

## Dynamic failure analysis for requirements validation and functional safety

# Example of Exploration

---

## IP selection

- GDDR6 vs LPDDR5
- NoC vs custom fabric

## Topology to meet latency and throughput

- Master and slave placement
- Distance between Routers

## Hardware-software partitioning

- Distribute tasks on cores, FPGA and multi-ASIC

## Power-Timing trade-off

- Instant power vs latency

## Parameter selection

- Clock speeds
- Flit size, width

## Functional

- Trade-off arbitration, scheduling, dispatcher

## Protocol design

- Performance for algorithms and HW assignment
- Impact on network traffic and QoS

## Software design

- Algorithm and control evaluation
- Test behavior for variable and state change
- Interaction between software components



# VISUALSIM TRAINING

---