VISUALSIM TRAINING

Training: Planning, Modeling, Simulation, Advanced Features

MIRABILIS design

Planning

- Goals & Requirements
- Tasks or Operations
- Workloads or Traffic
- Hardware or Platform
- Collaboration & Sharing
- Data Management
- Support
- Look & feel

Modeling

- Parameters/ Variables/ DS/ Fields
- Flow- System, Beh-Arch, Data flow
- Performance Power Functionality
- Hierarchical/ Class/ DI Blocks
- Traffic/ Results
- Hardware/ Software/ Network
- Abstraction
- Algorithms
- Task Graph
- Checking

Simulation and Analysis

- Types of Studies
- Parameter Ranges
- Graphical/ Batch/ Multi-Core
- Debugging/ Verification
- Requirements
- Post-Simulation
- Types of Plots
- Advanced Studies
- Accuracy
- Compare

Advanced Features

- Failure
- Integration
- Generation
- Encryption

Agenda- Part 3: Simulation and Analysis

- Types of Studies
- Batch Mode/Multi-Core Simulation
- Debugging
- Verification
- Requirements
- Advanced Studies
- Accuracy

Types of Studies

Behavioral Studies

Behavioral Studies: When studying the behavior of a system, we might focus on functional aspects such as how different components interact with each other to achieve specific tasks. Blocks representing functionalities or modules within the system could be analyzed to understand their behavior under different conditions or inputs.

Architectural Studies

In architectural studies, the emphasis is on the structure and organization of the system. This involves selecting appropriate components and arranging them to meet the system's requirements while optimizing for performance, power, cost, etc. Components like processors, memory, and interconnects are essential in architectural studies as they define the overall system architecture.

Network Studies

Network studies involve analyzing the communication infrastructure within a system, including data flow, bandwidth requirements, latency, etc. Components such as interconnects, interfaces, are critical in network studies as they determine how data is transmitted and routed within the system.

Workload Analysis

Workload analysis focuses on understanding the patterns of resource utilization within a system. This includes analyzing the types of tasks or processes being executed, their frequency, and their resource requirements. Traffic patterns and processing activities on components like processors and memory are important considerations in workload analysis.

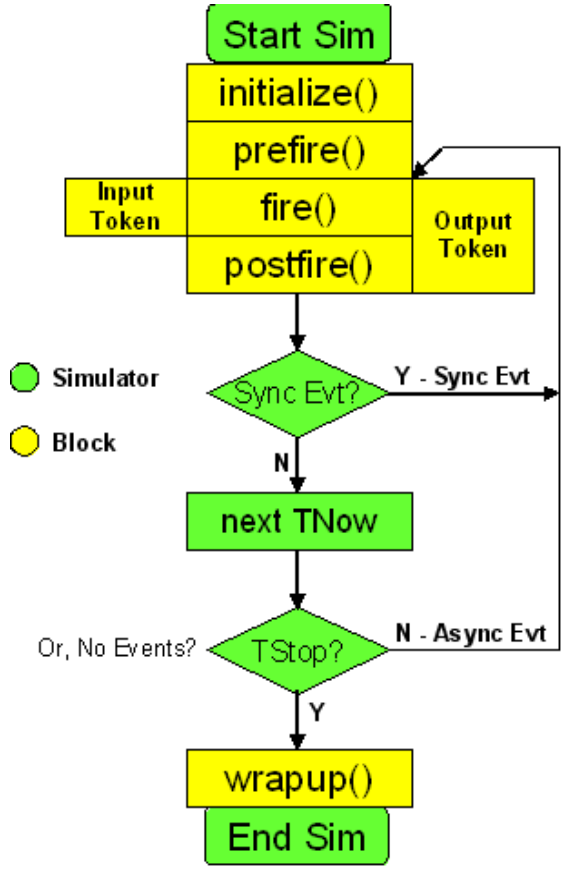
Performance Analysis

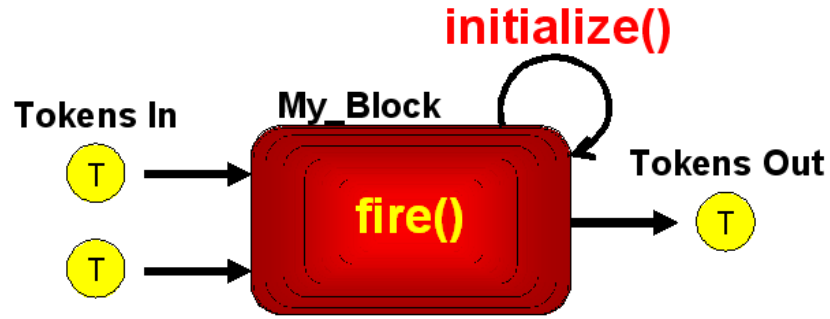
Performance analysis involves evaluating the efficiency and effectiveness of the system in terms of its speed, throughput, latency, etc.

Components such as processors, memory, and interconnects are analyzed to identify potential bottlenecks and optimize system performance.

Event Driven Simulation

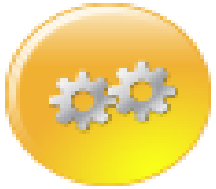
Discrete-Event Simulator and Block Execution





Models of Computation

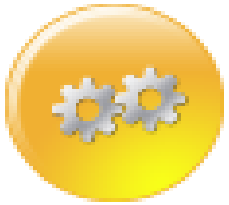
DigitalSimulator



Discrete Event Simulator Object

Timed Simulation

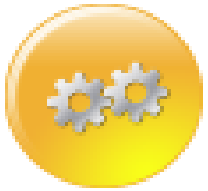
Untimed Digital



Synchronous Data Flow Simulator Object

Untimed DSP Algorithms

Continuous Time

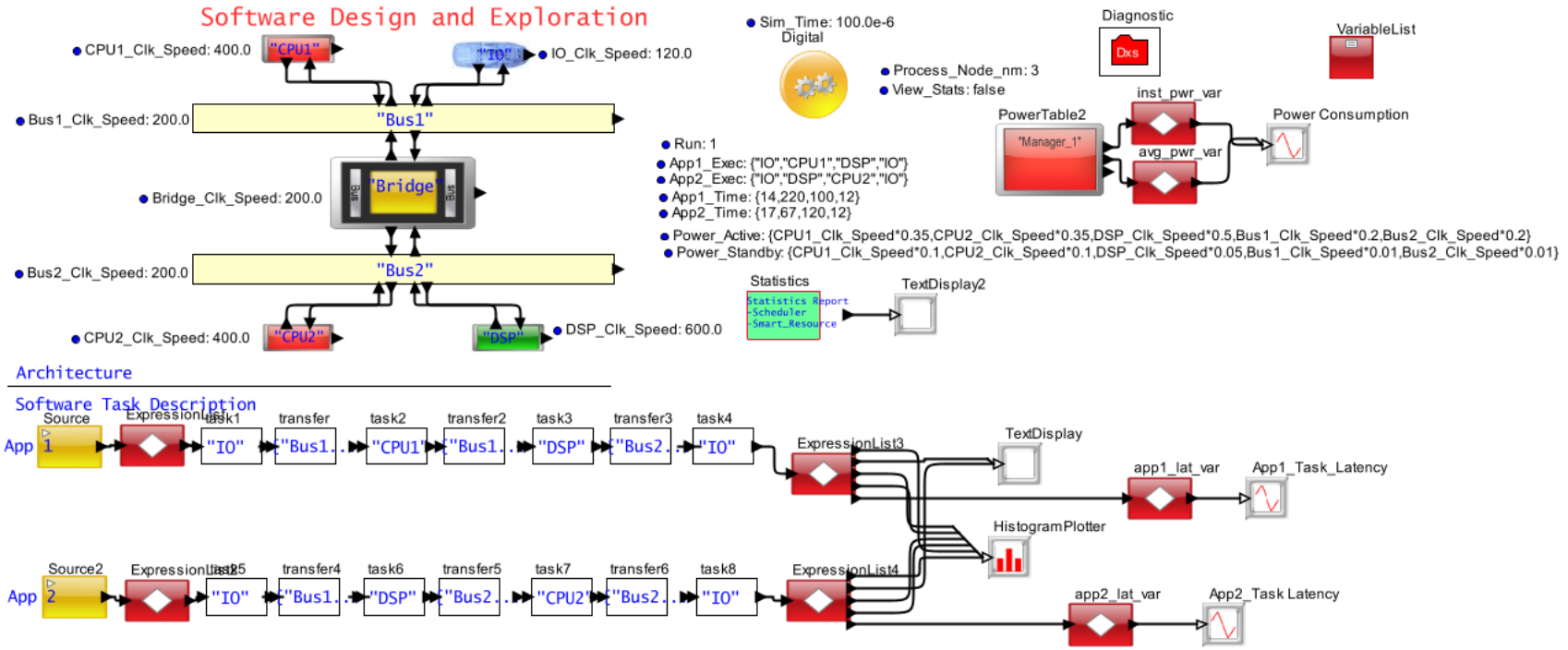


Analog Simulator Object

Analog Simulation

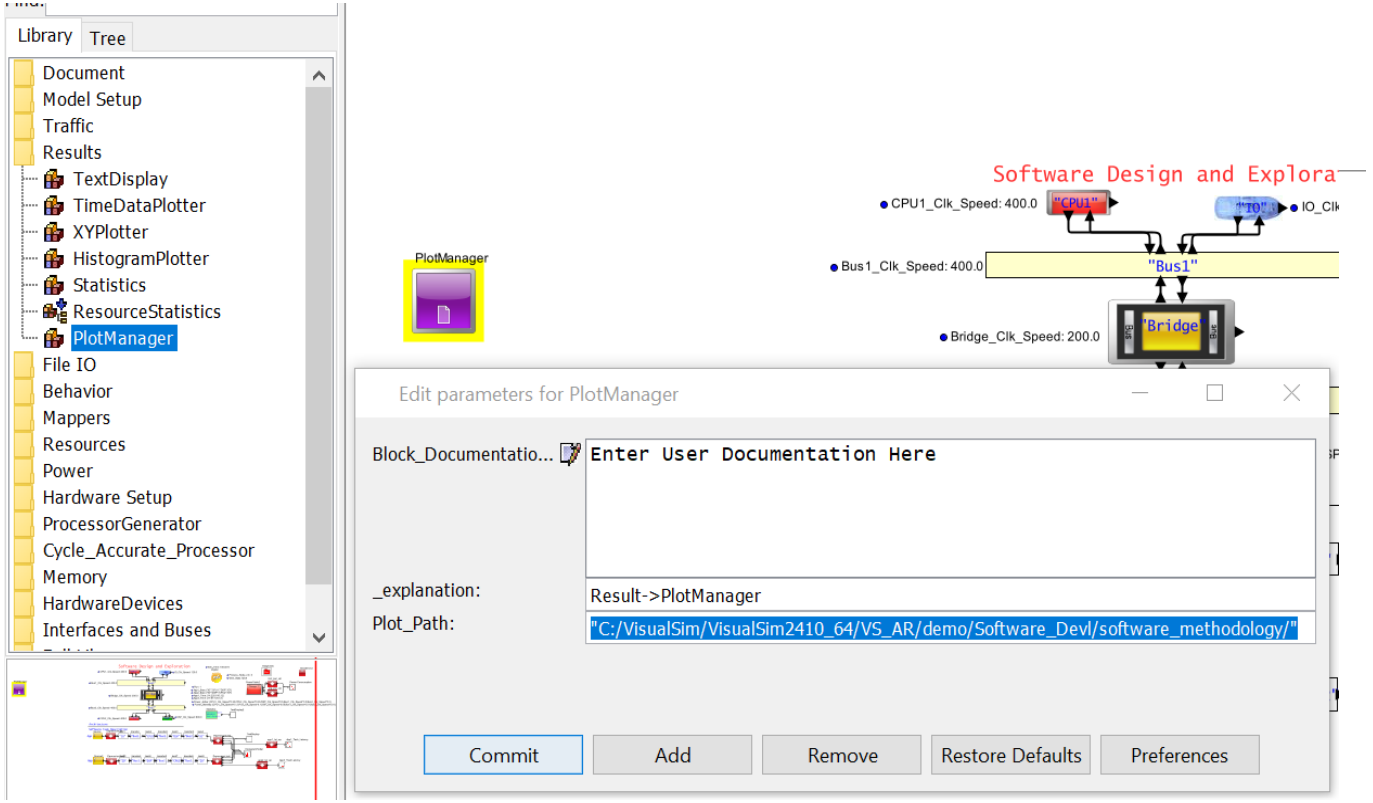
Batch/Multi-core Simulation

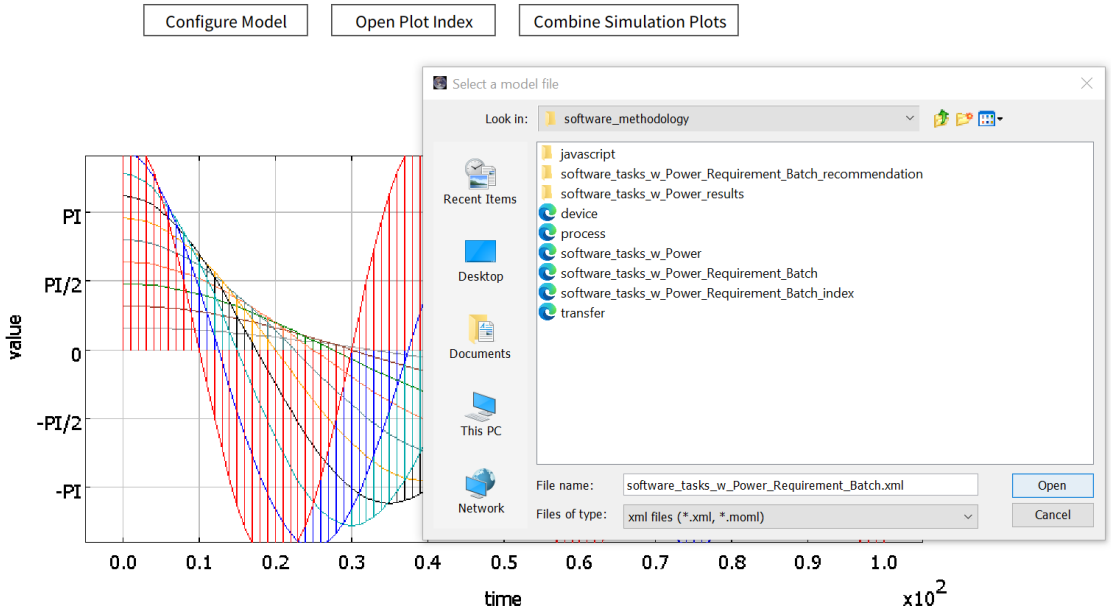
Assemble a Model with Architecture and Behavior



Model Location:
$VS/demo/Software_Devl/software_methodology/software_tasks_w_Power_Requirement_Batch.xml

Post Processor to run multi core

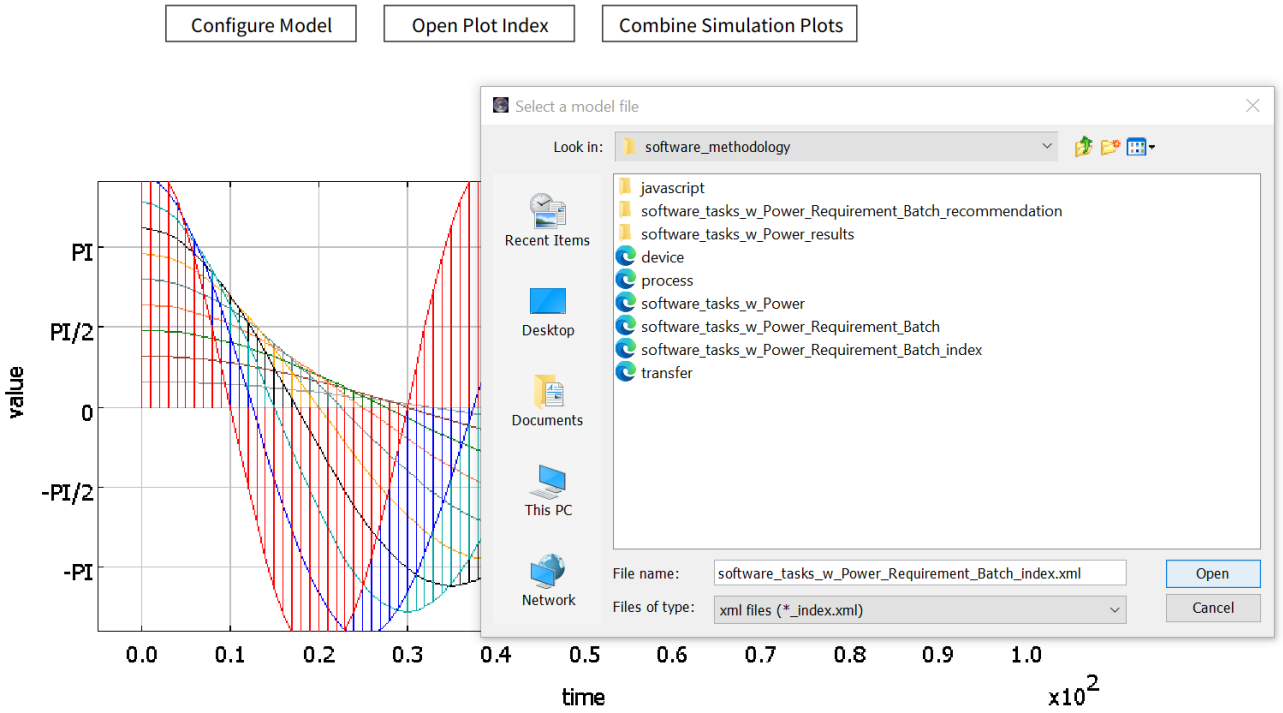




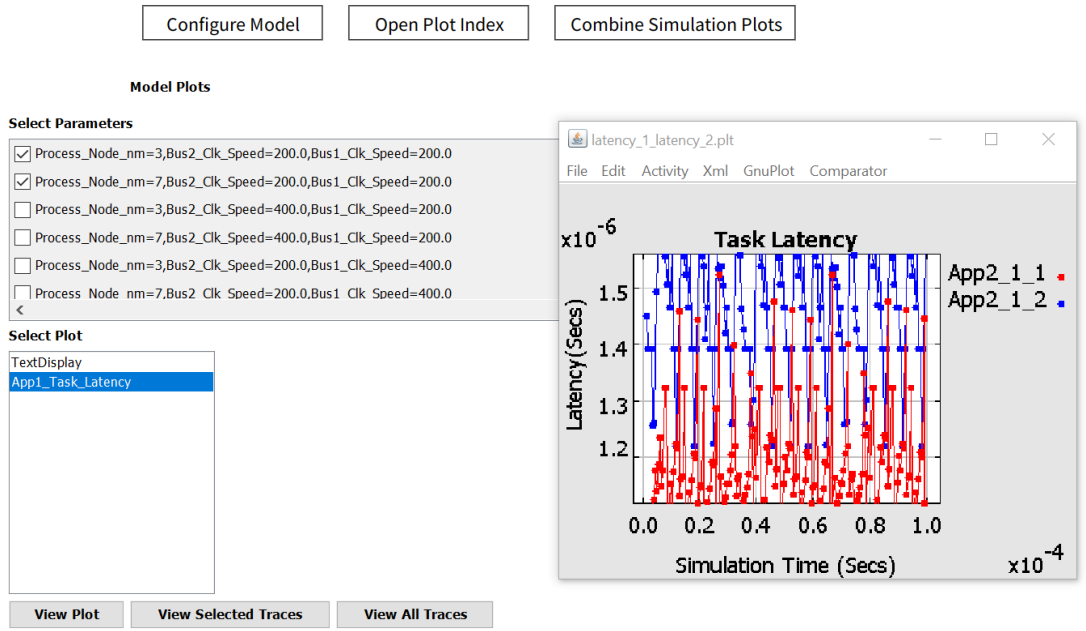
Configure Model → select model with PlotManager

Add PlotManager -- > Set Plot_Path

w Simulated Results



Open Plot Index



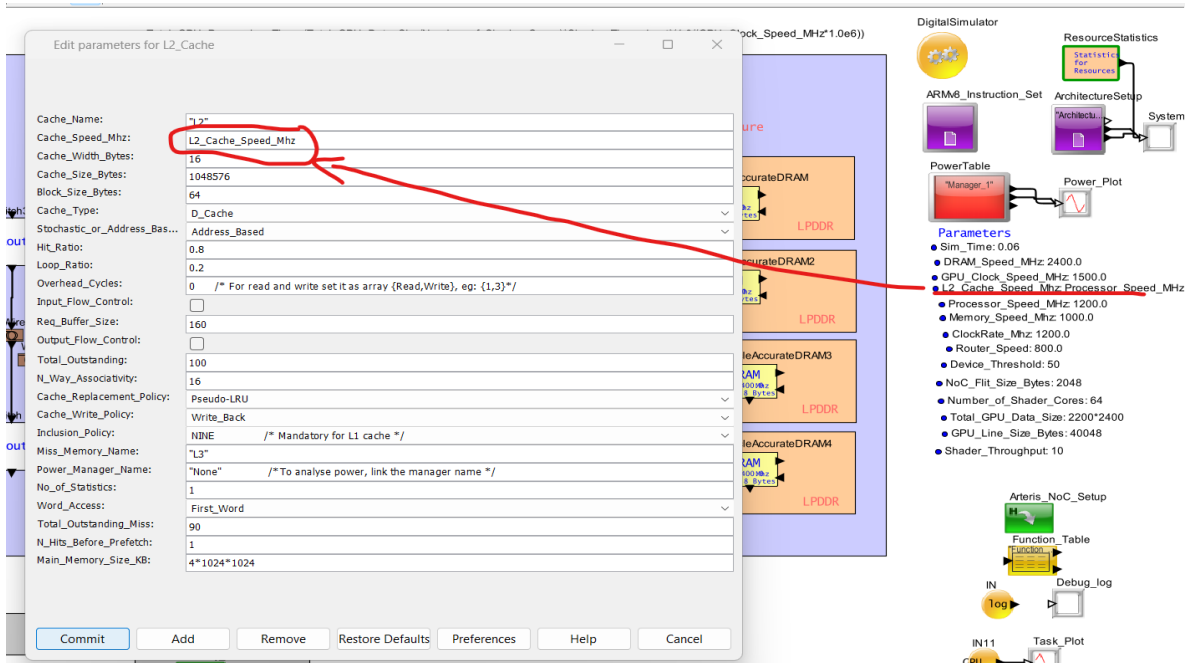
Select Parameters and Select plot

FAQS regarding batch simulation

1. How do we run a batch simulation by changing parameters that are not at the top level?

The Post Processor only recognizes the top level parameters. This is to make sure only the key or parameters of importance are used. User has 2 options

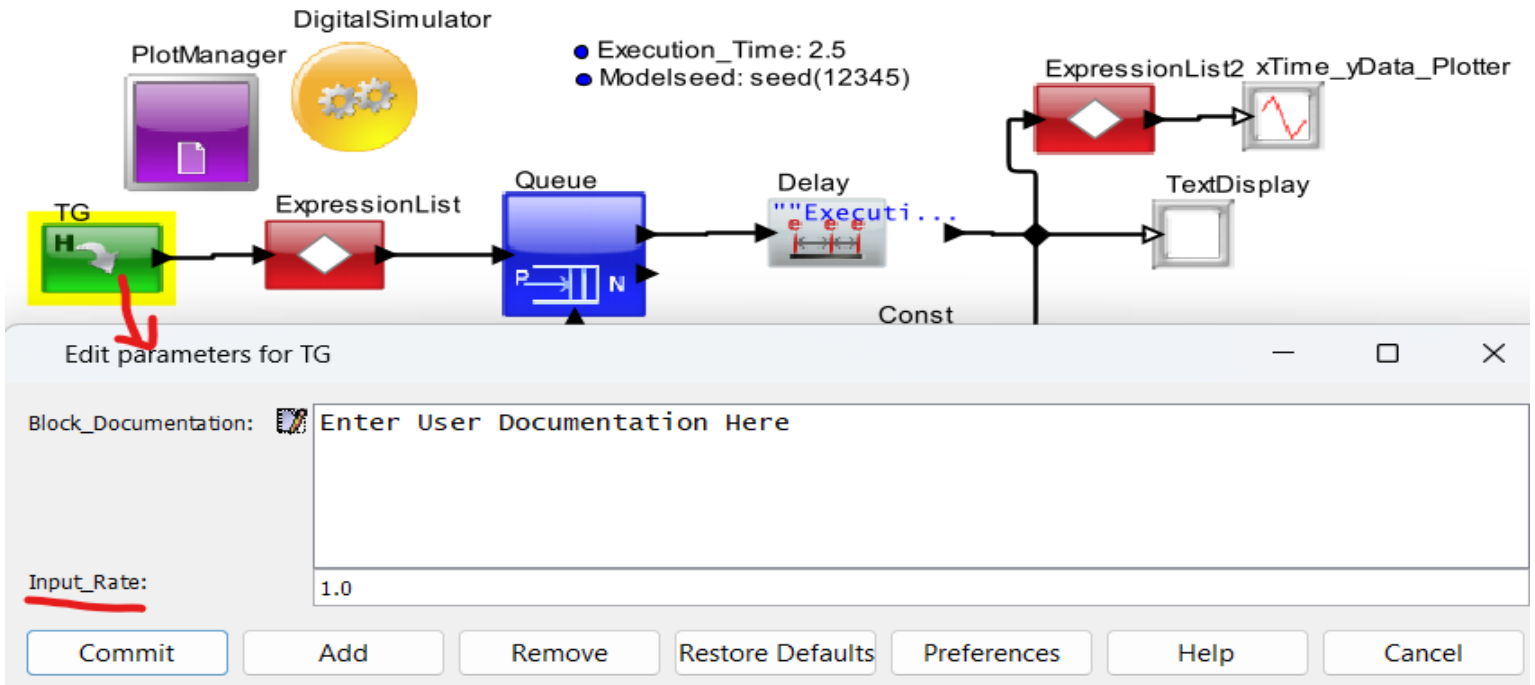
a.Bring the hierarchical parameters to the top level. This can be done by creating new parameters at the top level and linking them to the hierarchical parameter.



Continued….

b. Manually edit the batch script (Sim_Batch_Run.bat/.sh) which was created by post processor. We can configure the parameters within hierarchical blocks or even block parameters by following the format shown below:

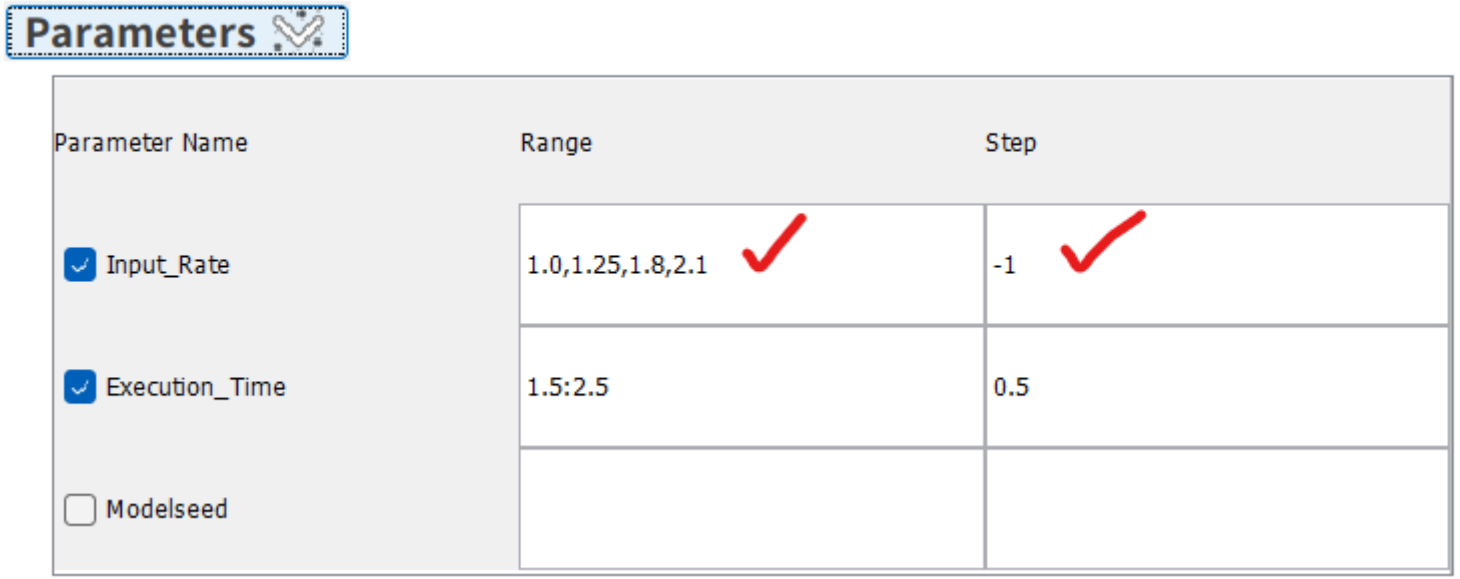
- Hierarchical Block Name.Parameter Name (Eg: -TG.Input_Rate)



Continued …

2. The post processor seems to only have options for Range and step. How do I specify discrete set of values?

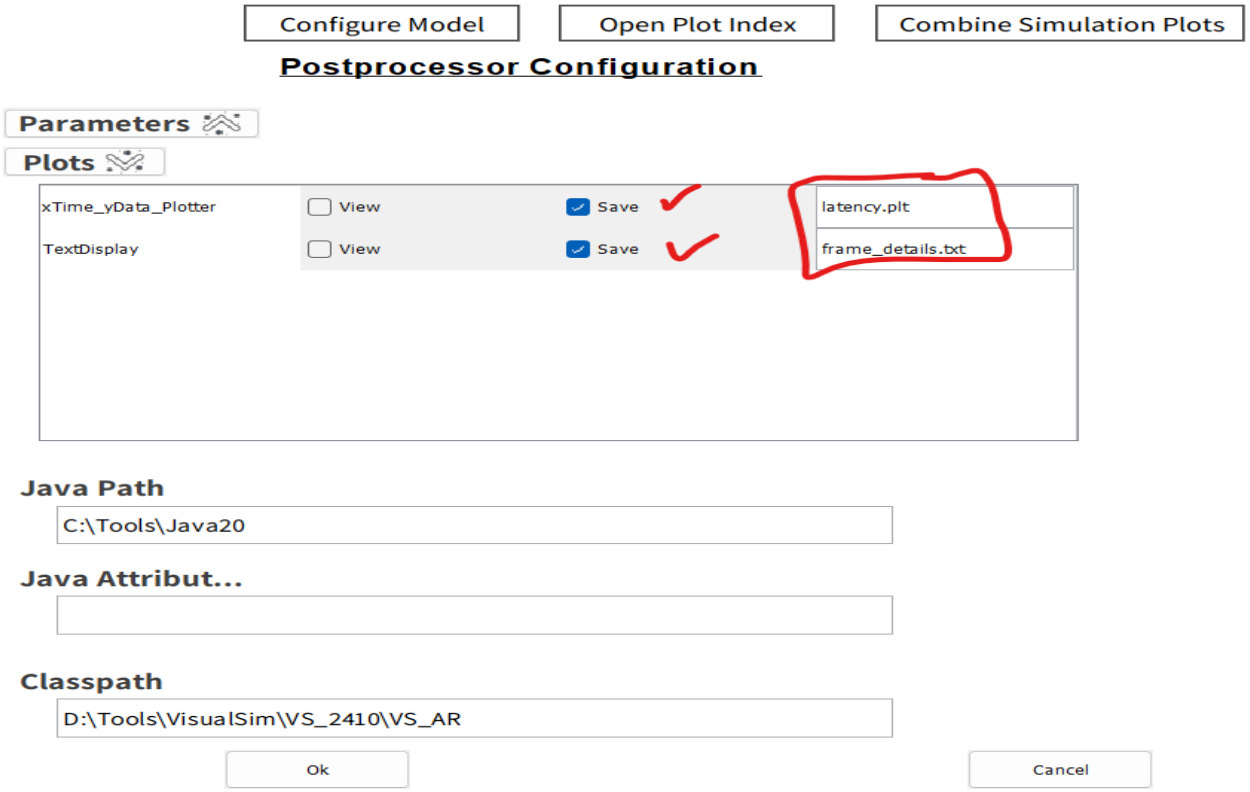
For defining discrete set of values, user can define the values separated by comma in the “Range” and keep “Step” as -1



Continued …

3. Batch simulation does not print out the system stats from all blocks like DRAM controller. How do we enable this?

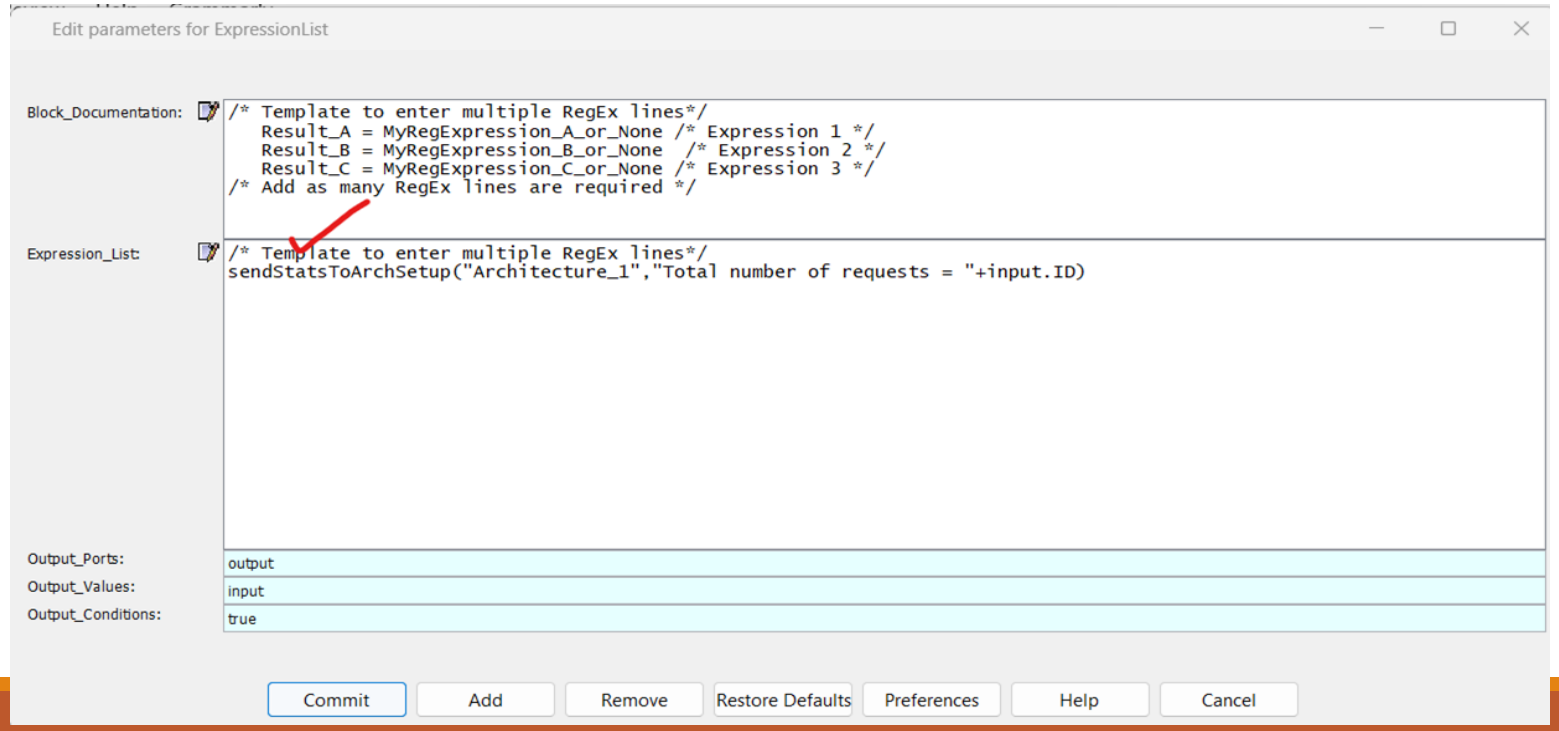
When we configure the model using Post Processor, user need to enable “Save” option for the text displays and plotters that are required.



Continued ….

4. If we want to send statistics or any data to the system stats module globally, how do we do this?

Send the stats to Architecture Setup block
We need to have Architecture_Setup block in the demo model. We can send the statistics to the Architecture_Setup block by using a RegEx called sendStatsToArchSetup(). The format is:
sendStatsToArchSetup(<Architecture Setup Block Name>, Statistics)



Debugging

Various methods to Debug a particular model

- Trace Tracking
- Animation
- Listen to Port
- Listen to Block
- Listen to Simulator
- Digital Debugger
- Error Messages

- Variable Dump
- RegEx
- Script Debugging
- Data Structure Fields
- Plotters & Text Display

Debugging Procedure

During and End of Simulation

1. Error Message
   - Identify the block listed in the Error Description and the Block Highlight in the block diagram view
   - Review Possible Solution description to resolve
2. Listen to Block to see if the sequence of execution is correct
3. Listen to Port to see if the input and output field values are correct
4. Variable Dump block and see the values of the memory
5. View Command Line at the end of the simulation for the summary of total Simulator events, synchronous event, asynchronous mix events, time taken and memory used

Follow the Data Flow

If TextDisplays has no output

- Check the block before it and follow up with each prior block
- Listen to Port to check the output values

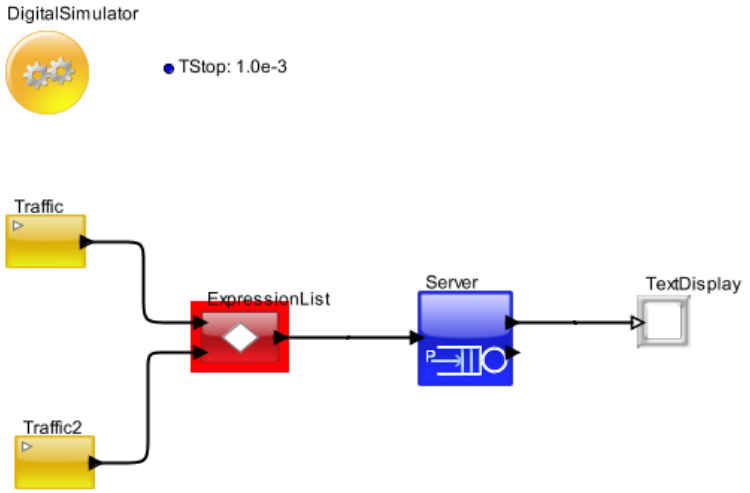
Make sure the transaction is arriving from the correct source and going to expected destination

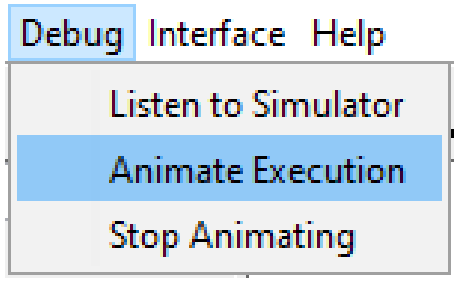
Does the Transaction ID sequence make sense?

Any special Transaction flags set, indicating mode of operation that is inconsistent with current block

Animate Execution

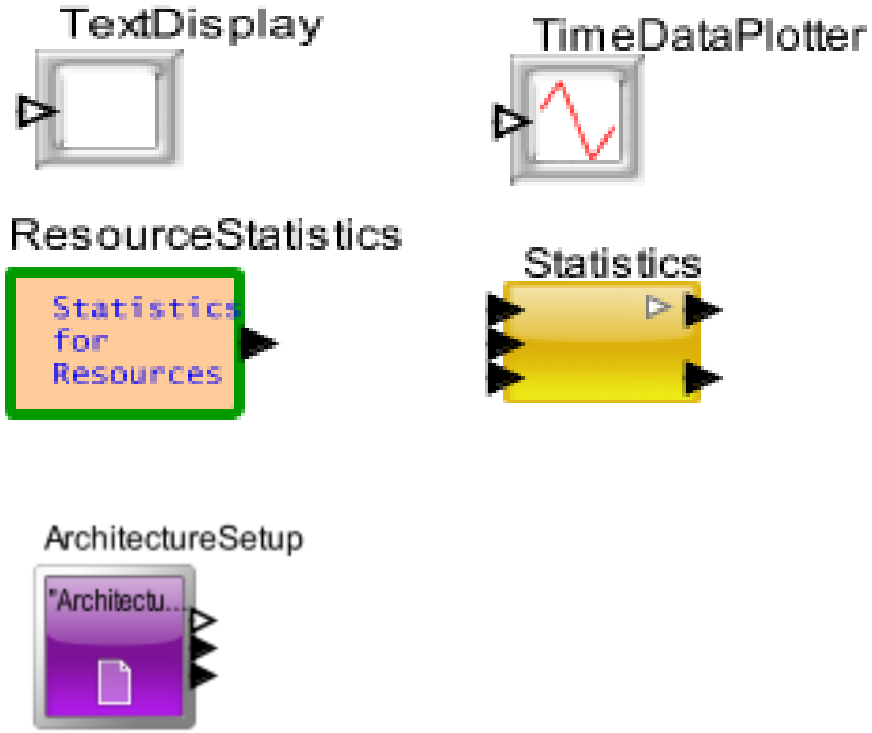
- To view the dynamic operation of the model
- The Executing Block gets highlighted
- The time to highlight a block is in **milliseconds**
- To Start Animation:
  Debug -> Animate Execution
- To Stop Animation:
  Debug -> Stop Animating





Text Display, Plotters, Statistics

- To graphically display and analyze data collected from the simulation.
- Helps to detect any errors in the behavior
- Statistics Generators - Generates statistics of all resources and hardware blocks
- Extracts the appropriate fields in the data structure or the entire object and display them.
- Plotter – Latency, throughput, etc
- Text Display – Entire Data Structure, any value coming in the input port



Statistics to Identify Behavior Errors

Resource statistics

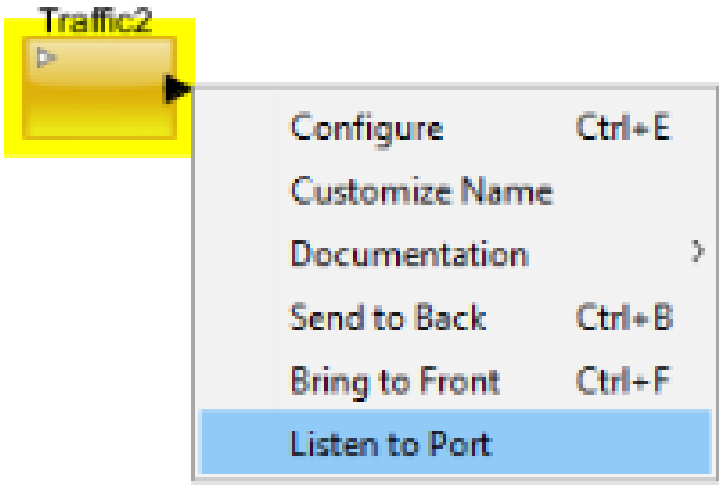
```
DISPLAY AT TIME                  ------ 100.00000000000 sec --
{BLOCK                        = "Resource_Statistics.Queue",
DELTA                         = 0.0,
DS_NAME                       = "Queue_Common_Stats",
ID                            = 6,
INDEX                         = 0,
Number_Entered                = 199,
Number_Exited                 = 12,
Number_Rejected               = 157,
Occupancy_Max                 = 30.0,
Occupancy_Mean                = 20.0754716981132,
Occupancy_Min                 = 0.0,
Occupancy_StDev               = 10.1402412546265,
Queue_Number                  = 1,
TIME                          = 100.0,
Total_Delay_Max               = 91.0182813545,
Total_Delay_Mean              = 26.2911000907667,
Total_Delay_Min               = 0.0,
Total_Delay_StDev             = 23.8057259325954,
Utilization_Mean              = 0.0}
```

Architecture Setup

```
DISPLAY AT TIME                    ------ 11.42860 us ------
{AHB_Bus_Delay_Max              = 1.8E-8,
AHB_Bus_Delay_Mean              = 1.8E-8,
AHB_Bus_Delay_Min               = 1.8E-8,
AHB_Bus_Delay_StDev             = 0.0,
AHB_Bus_IOs_per_sec_Max         = 524998.6875032812,
AHB_Bus_IOs_per_sec_Mean        = 524998.6875032812,
AHB_Bus_IOs_per_sec_Min         = 524998.6875032812,
AHB_Bus_IOs_per_sec_StDev       = 0.0,
AHB_Bus_Input_Buffer_Occupancy_in_Words_Max       = 8.0,
AHB_Bus_Input_Buffer_Occupancy_in_Words_Mean      = 3.4736842105263,
AHB_Bus_Input_Buffer_Occupancy_in_Words_Min       = 0.0,
AHB_Bus_Input_Buffer_Occupancy_in_Words_StDev     = 3.4084789176194,
AHB_Bus_Preempt_Buffer_Occupancy_in_Words_Max     = 0.0,
AHB_Bus_Preempt_Buffer_Occupancy_in_Words_Mean    = 0.0,
AHB_Bus_Preempt_Buffer_Occupancy_in_Words_Min     = 0.0,
AHB_Bus_Preempt_Buffer_Occupancy_in_Words_StDev           = 0.0,
AHB_Bus_Throughput_MBs_Max      = 8.3999790000525,
AHB_Bus_Throughput_MBs_Mean     = 8.3999790000525,
AHB_Bus_Throughput_MBs_Min      = 8.3999790000525,
AHB_Bus_Throughput_MBs_StDev    = 0.0,
BLOCK                           = ".Processor_Power_model.ArchitectureSetup",
Cache_Delay_Time_Max            = 5.0E-8,
Cache_Delay_Time_Mean           = 5.0E-8,
Cache_Delay_Time_Min            = 5.0E-8,
Cache_Delay_Time_StDev          = 0.0,
Cache_Hit_Ratio_Max             = 100.0,
Cache_Hit_Ratio_Mean            = 100.0,
Cache_Hit_Ratio_Min             = 100.0,
Cache_Hit_Ratio_StDev           = 0.0,
Cache_Memory_Used_By_MAC_ARM9_MB_Max      = 9.6E-5,
Cache_Memory_Used_By_MAC_ARM9_MB_Mean     = 9.6E-5,
Cache_Memory_Used_By_MAC_ARM9_MB_Min      = 9.6E-5,
Cache_Memory_Used_By_MAC_ARM9_MB_StDev    = 0.0,
Cache_Memory_Used_By_Total_MB_Max         = 9.6E-5,
Cache_Memory_Used_By_Total_MB_Mean        = 9.6E-5,
Cache_Memory_Used_By_Total_MB_Min         = 9.6E-5,
```

Listen to Port

- Displays each token passing through the port
- Used to check whether the data is flowing through the particular port
- If no data on the "Listen to Port" window , it indicates that the model did not generate any output from that block.
- If there is no data, then one needs to check the ports, or virtual connections driving this block to see that they are being activated correctly.
- Helps to debug errors in connections, routing and conditional branches.
- Usage
  - Right click the required port and select Listen to Port



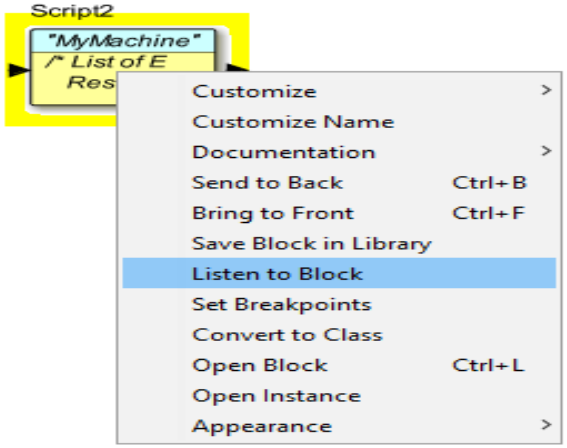

```
INPUT AT TIME                  ------ 0.0 ps ------
{BLOCK                      = "Traffic",
DELTA                       = 0.0,
DS_NAME                     = "Header_Only",
ID                          = 1,
INDEX                       = 0,
Priority                    = 3,
TIME                        = 0.0,
Task_Latency                = 0.0,
Time_Array                  = {0.0, 0.0},
Trace_Array                 = {"Queue_in", "Queue_out"},
length                      = 1.9769883723326}

INPUT AT TIME                  ------ 1.9769883722330 sec ------
{BLOCK                      = "Traffic",
DELTA                       = 0.0,
DS_NAME                     = "Header_Only",
ID                          = 2,
```

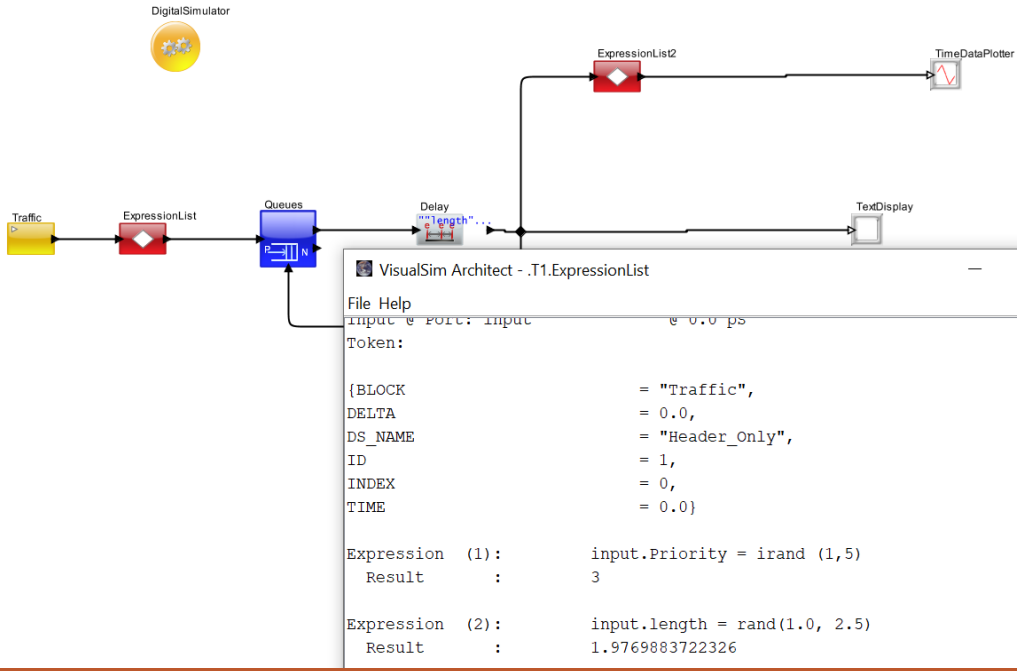
Listen to Block

- Gives more insight into the internal block operation
- Shows the sequence of execution, entry/exit, virtual Send, threads

Support coverage

- All blocks except
  - Hierarchical blocks
  - instantiated hierarchical class
- Provides simulator level information relative to methods being fired, and so on. Simulator information is most useful for evaluating custom blocks in the simulation environment.
- Right click on the block and select Listen to Block

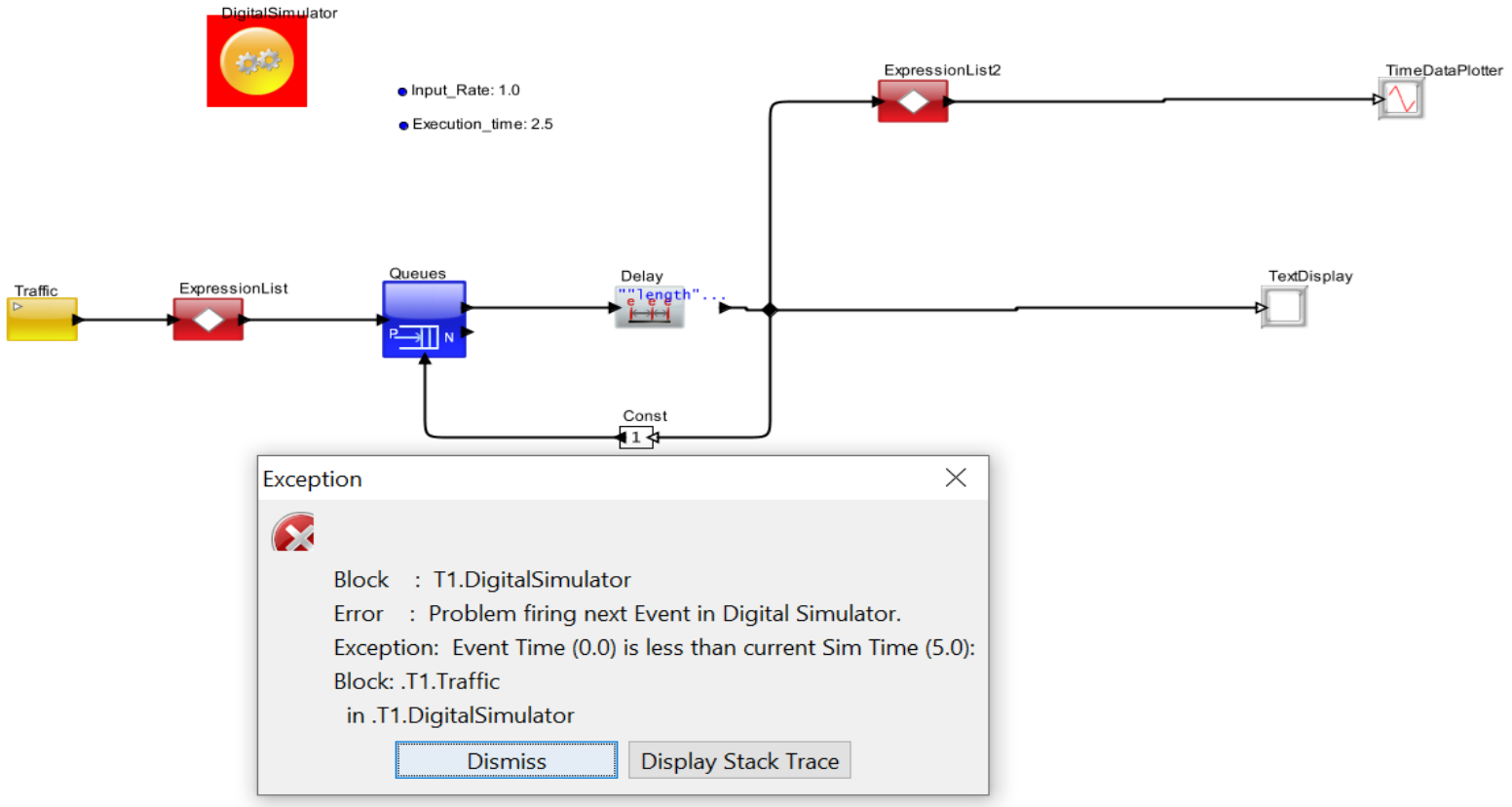




Error Message

Message

- Highlight Error Block
- And Error Message



Fields of the Data Structure

- Each Data Structure (DS) has header fields that provide information as to its source, ID increment, and time created.
  - Traffic field list the originating block
  - ID indicates the sequence of generation
  - TIME is a generation time stamp
- These are valuable clues if a request does not arrive or arrives out of order.

```
DISPLAY AT TIME                  ------ 0.0 ps ------
{BLOCK                         = "Traffic",
DELTA                          = 0.0,
DS_NAME                        = "Header_Only",
ID                             = 1,
INDEX                          = 0,
TIME                           = 0.0}
```

Time and Task Tracer fields

- Set of arrays that are updated with the **name and time stamp** every time the Data Structure enters or departs a Resource or Architecture block.

```
Time_Array              = {0.0, 5.0E-4, 5.0E-4, 1.0E-3},
Time_In_Resource        = 5.0E-4,
Trace_Array             = {"Server_in", "Server_out", "Scheduler_in", "Scheduler_out"}}
```

Memory Dump/Variable Dump

- This outputs the current value of all the global and local variables in the model

- The output is a data structure with each field representing one of the memories.

- Full Library ->Model-> Utility-> Checkers-> Variable_Dump

RegEx

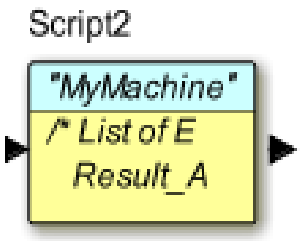
- Provides status and visibility into resource and hardware blocks in the execution flow.

✓readAllVirtual( ) - Provides the List of virtual Blocks in the model.

✓readAllMemory( ) - Provides a output of all local and global memories and their current value.

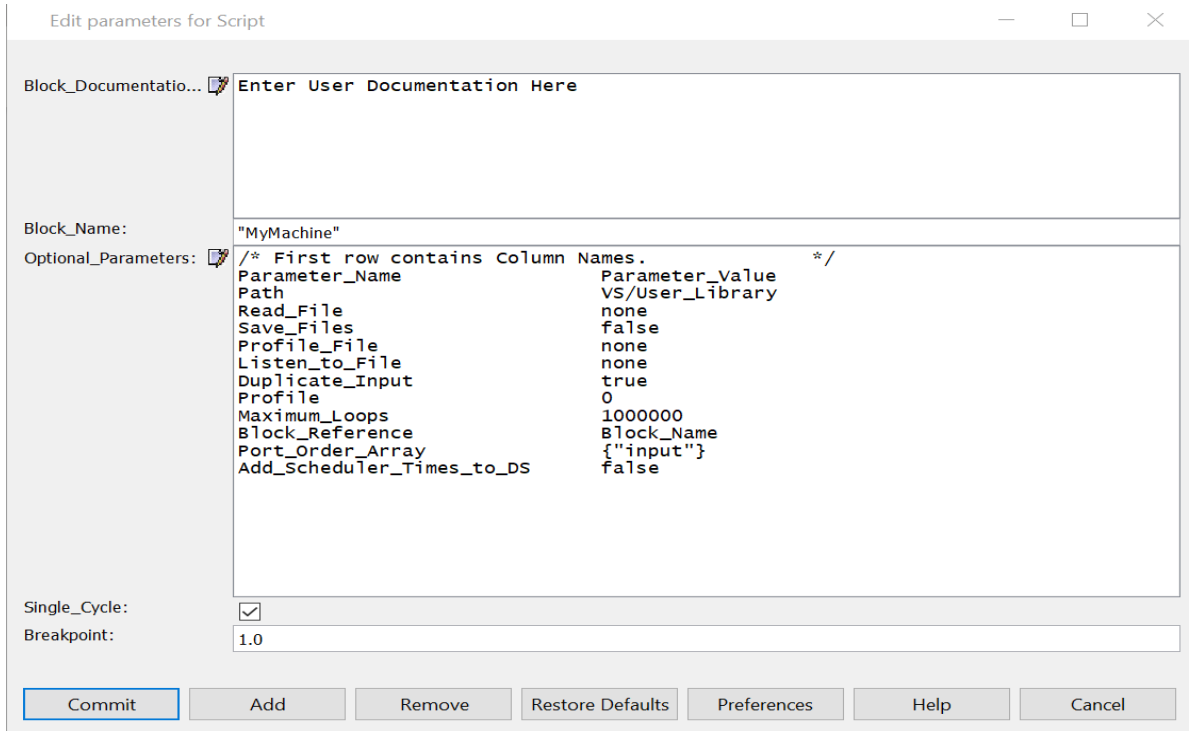
✓getBlockStatus( ) - The statistics for the blocks are generated using the getblockStatus RegEx function with the type, length, stats etc.,

✓getResourceActivity( ) - Used to access the information in the database block. It returns an array of the current queue length of the resources listed in the named column.



Script Debugging

- Right Click on Script -- > Customise -- > Configure
- This will list down the parameters



Script Tracer

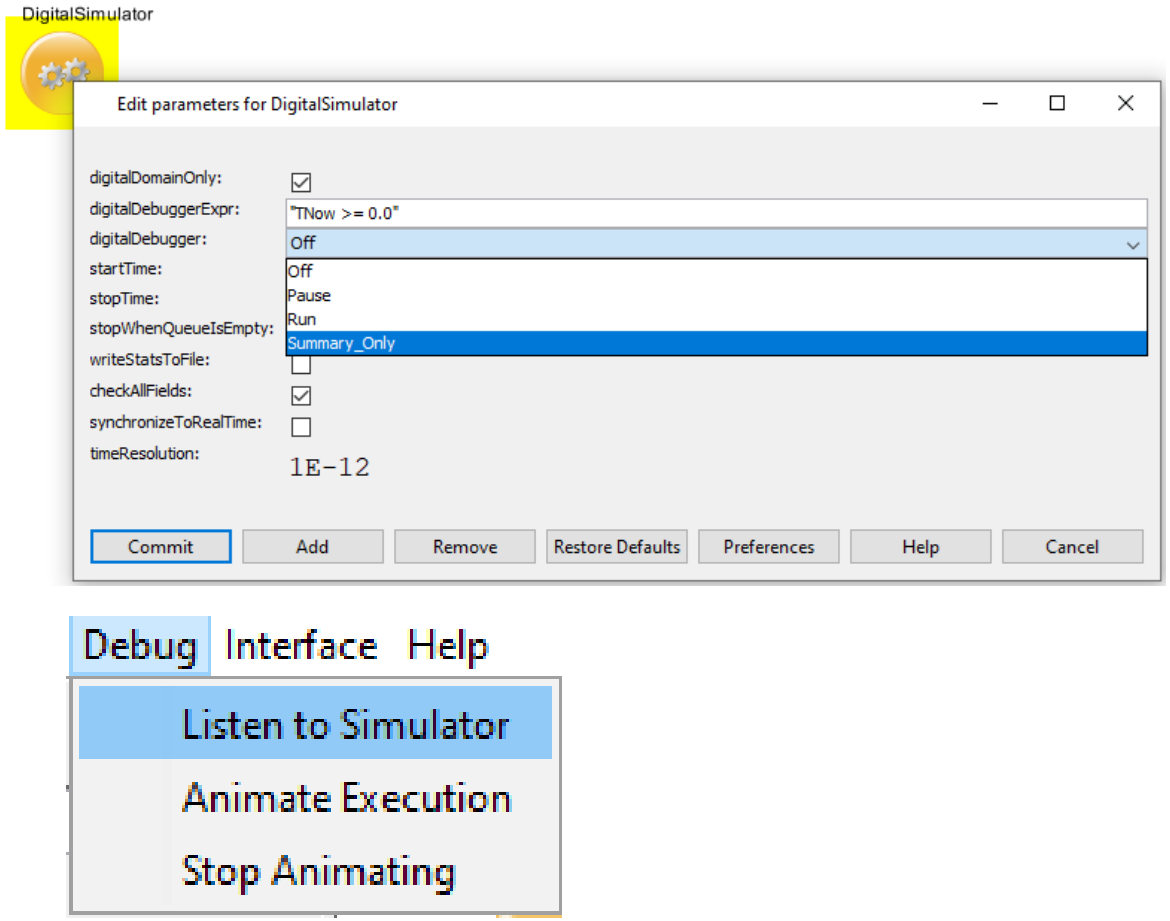
- Enables the capture of execution of multiple Scripts from one location and the content is written to a field.

- Helps to check the interaction between scripts and if the sequence of execution of instructions is correct.

- Data is written into a file called **VisualSimTraceLog.txt** which can be seen under **users/user folder/.VisualSim/**

Listen to Simulator- Digital Debugger

- Provides a sequence of execution for the selected simulation.
- Integrated with Digital Debugging utility in the Digital simulator.
- This window displays the usage statistics, current block execution, and model summary information.
- **Debug -> Listen to Simulator**



Digital Debugger parameters

- Off – Disables the Debugging Mode
- Pause – Turns the Debugger to Stop at every block in the model Flow. Provides summary at the end of Simulation
- Run – Records the order in which each block is fired in the model. Provides summary at the end of simulation.
- Summary Only – Generates the List of all the blocks at the current level of simulation and the levels below

For each Block

- ✓ Records the number of time each block is fired in the model
- ✓ Average execution time for each firing
- ✓ Total time spent in each Block
- ✓ It also lists the Blocks that are not executed in the model

Pause and Resume

- Saves the simulation data, events and status in a file
- Handy to debug simulations that run for a large period of time.
- User can analyze system behavior at various points in the simulation.
- User can pause at a timestamp and analyze the system response and continue simulation step by step from that point onwards.
- The system can be analyzed for required functionality and also helps the designer to identify if the crucial tasks are being executed within the deadlines.

Verification

Latency Measurement and identifying the bottleneck

Using Time Array and Trace Array :

1. The Time_Array gives the timestamps at which the packet reached a particular Block and the Block names are given by the trace array.
2. This information can be used to identify latency bottlenecks
3. The same information can be written to an excel and this process can be automated using a Script block

```
Task_Throughput          = 2.770038127491SE9,
Throughput_Array         = {8.5235818963176E6, 1.536E8, 3.072E7},
Time_Array               = {0.011111111111, 0.013382363603, 0.013382363603, 0.0151844229, 0.0151844229, 0.0152844229, 0.0152844229,
Time_In_Resource         = 0.0,
Trace_Array              = {"DMA_Task_in", "DMA_Task_Done", "Read_DMA_in", "Read_DMA_out", "Block1_in", "Block1_out", "Block3_in", '
```

Case Study – latency Induced bandwidth loss

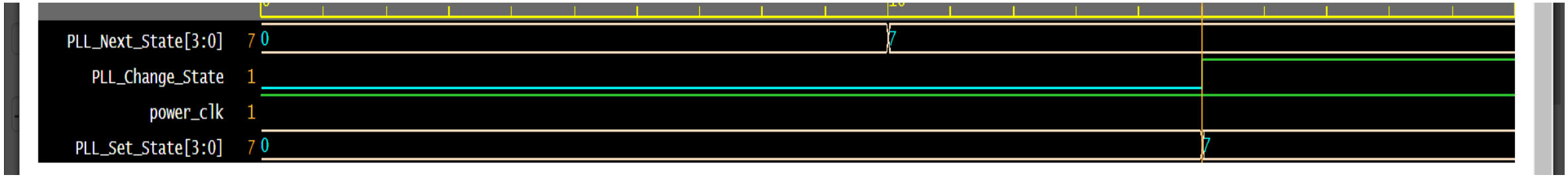
Packet_ID	Block 1	Block 2	Block 3	Block 4
0	1ns	2ns	1.5ns	3.7us
1	2ns	2ns	1.5ns	3.8us
2	4ns	2ns	1.5ns	3.5us
3	8ns	2ns	1.5ns	3.6us
4	16ns	2ns	1.5ns	3.2us
5	32ns	2ns	1.5ns	3.1us

- Consider a flow where the packets travel from Block 1 to Block 4.
- Although most part of the Latency seems to be due to Block4, Block1 has a linearly increasing latency.
- The Packets are getting buffered at Block 1 itself without being immediately transferred to the next nodes.
- There is a possibility to achieve a better throughput if can transfer the packets without buffering, depending on the destination data size, processing speed and buffer size.
- But buffering it at the source reduces the maximum achievable bandwidth

Power Verification

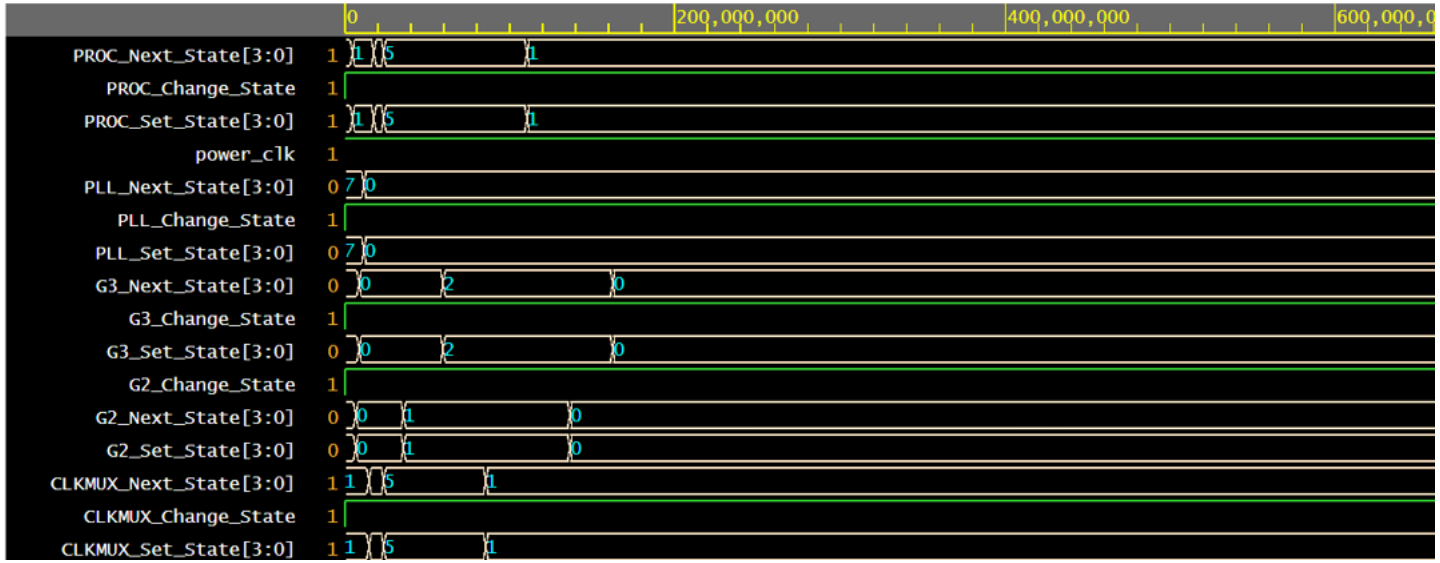
Using Power Log from third port of power table – explained in Power Modeling PPT

Using the VCD Dump obtained by executing the systemVerilog testbench generated out of VisualSim in an EDA tool that supports systemVerilog.



Case 1: VCD can be used to verify whether the Block is switching it's PowerState as expected and also the delay between tranisitons due to capacitance

Power Verification - Power State Coverage

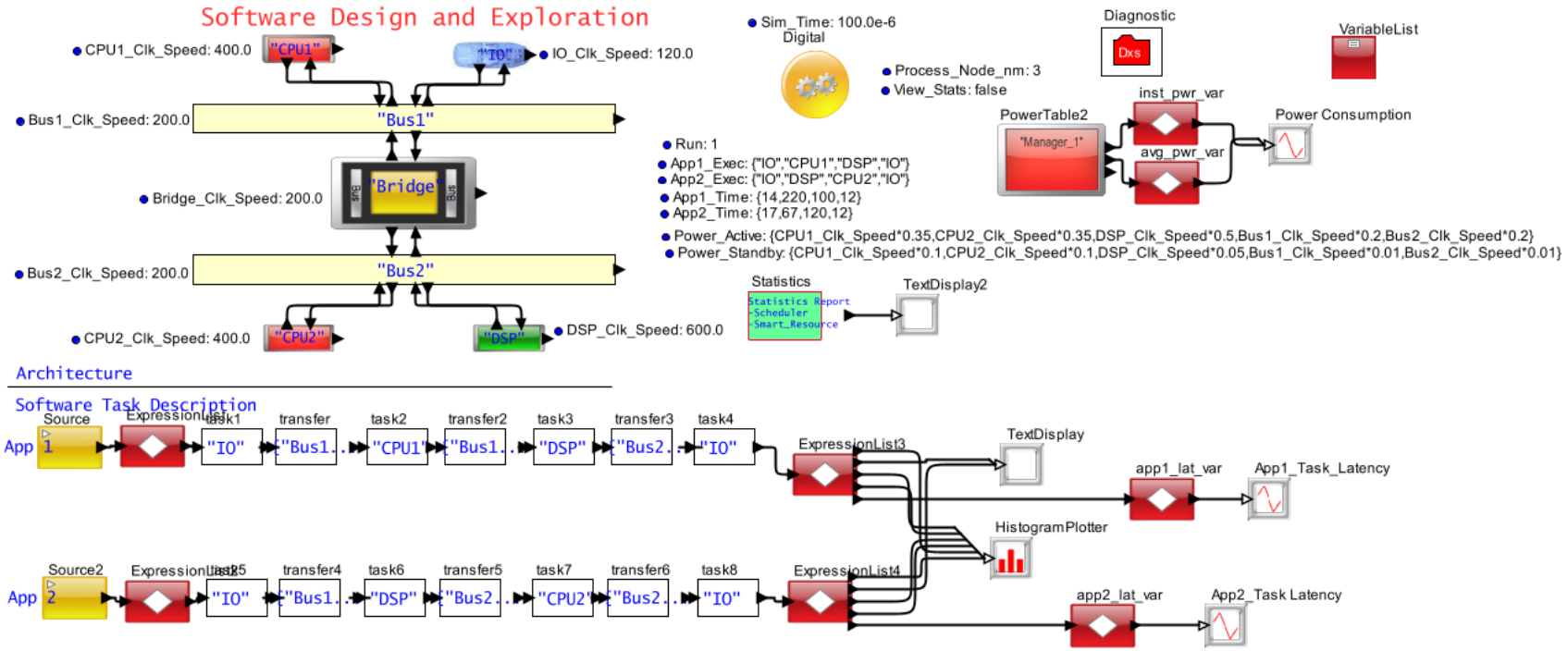


Case 2 : VCD can be used to verify whether the test case is covering all the possible States for a given domain/block. This is important because for the verification to be complete, all the power states with their power consumption and transitions have to be verified.

In this particular case, we can notice that the test case does not cover the states 3,4 and 6.

Requirements

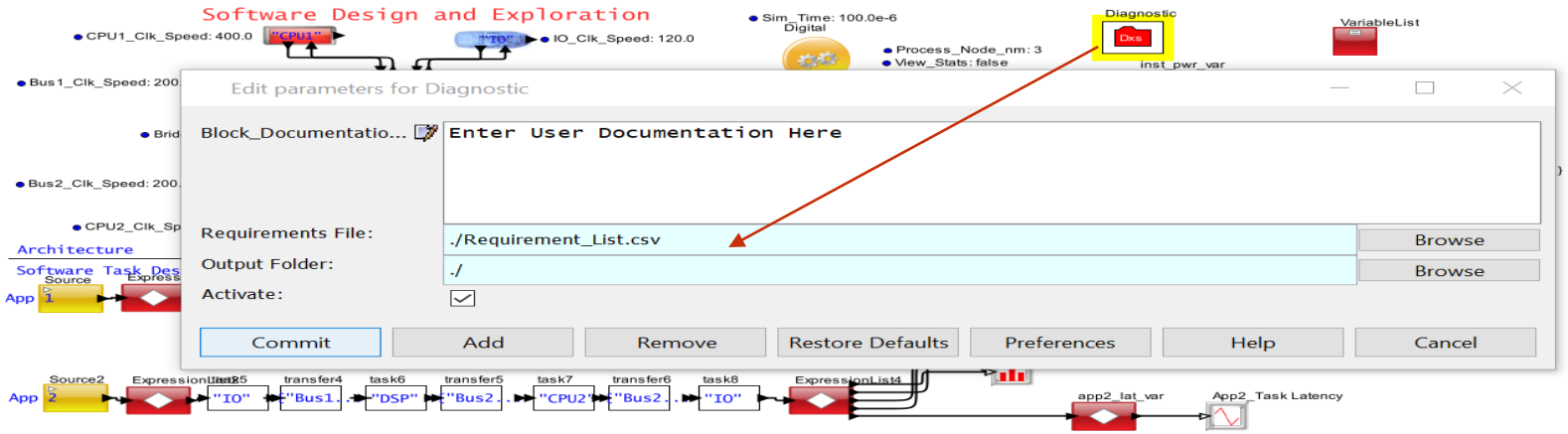
Diagnostic Block – Add Design Constraints



Model Location:

$VS/demo/Software_Devl/software_methodology/software_tasks_w_Power_Requirement_Batch.xml

Requirements/Design Constraints Input



Design Constraint Example

VisualSim Architect - file:/C:/VisualSim/VisualSim2340_64/V. . .tware_methodology/Requirement_List.csv

File Edit Help

```
1  BLOCK,METRICS,CONSTRAINT,CONSTRAINT VALUE,STATISTIC TYPE,REFERENCE VARIABLE
2  Global,avg_power,<,0.8,Mean,
3  Global,inst_power,<=,1,Max,
4  Global,app1_latency,<,1.70E-06,Max,
5  Global,app2_latency,> ,1.50E-06,Max,
6
```

Requirements CSV – How to Populate?

BLOCK	METRICS	CONSTRAINT	CONSTRAINT VALUE	STATISTIC TYPE	REFERENCE VARIABLE
Global	avg_power	<	0.8	Mean	
Global	inst_power	<=	1	Max	
Global	eTeLatency	<	6.00E-05	Max	
Global	MIPS	>	1.00E+02	Max	

Min, Max, Mean (Average), All (instantaneous) values are supported

Constraint can be any of these options

1. If Using Variables that are manipulated Script/ExpressionList, declare them as Global variables within the Model and use “Global” for BLOCK Column

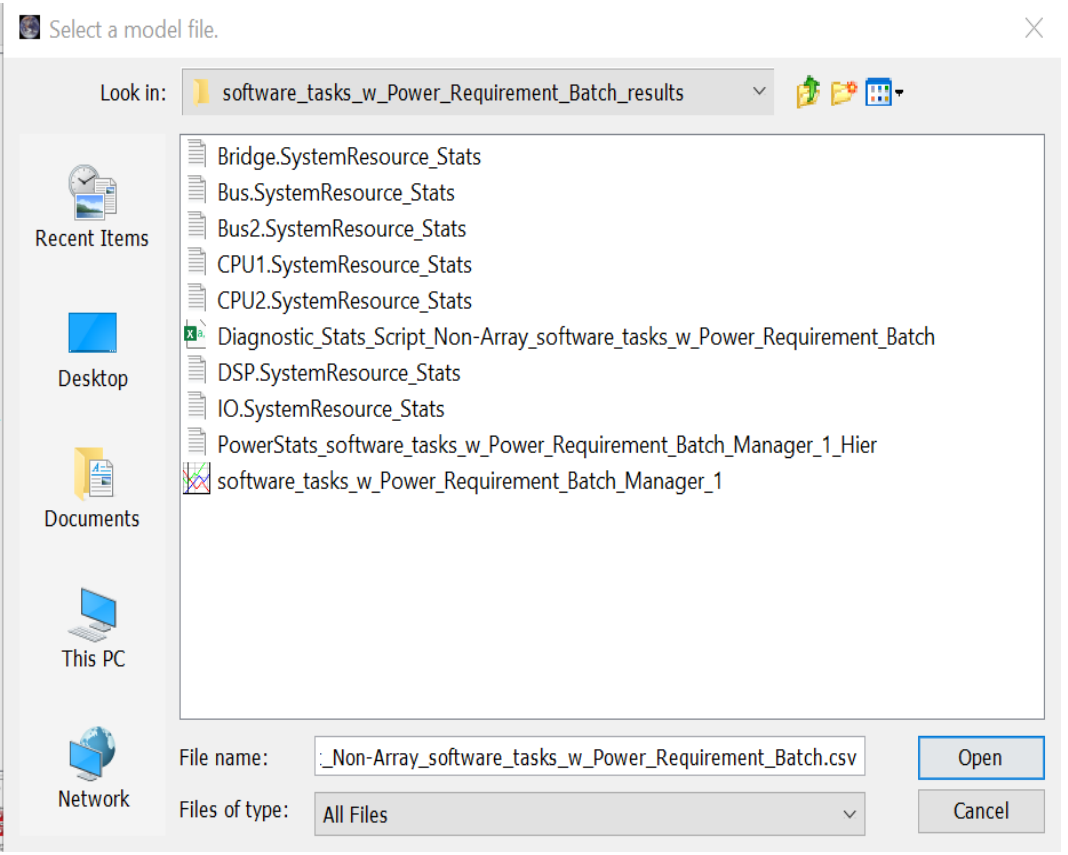
2. If Using Standard Block, add the Block name under BLOCK

1. All the Variables can be used here

2. Only Latency, Buffer Occupancy and Utilization are supported

Requirements Output

CSV Path :
<Model_path>_results



VisualSim Architect - file:/C:/VisualSim/VisualSim2320_64/V. . .re_tasks_w_Power_Requirement_Batch.csv

File Edit Help

```
Variable Tracker,
Model : software_tasks_w_Power_Requirement_Batch,
Input file : Requirement_List.csv

BLOCK,VARIABLE,MEASURED_VALUE,CONSTRAINT,CONSTRAINT_VALUE,STATISTIC TYPE,RESULT,
Global,app2_latency,1.5394000000000009E-6,>,1.5E-6,Max,True,
Global,inst_power,0.8720000000000055,<=,1.0,Max,True,
Global,avg_power,0.5780769047248477,<,0.8,Mean,True,
Global,app1_latency,1.7196999999999964E-6,<,1.7E-6,Max,False,

```

Failing Case :
BUS1_Clk_Speed= 200,
BUS2_Clk_Speed = 200

VisualSim Architect - file:/C:/VisualSim/VisualSim2320_64/V. . .re_tasks_w_Power_Requirement_Batch.csv

File Edit Help

```
Variable Tracker,
Model : software_tasks_w_Power_Requirement_Batch,
Input file : Requirement_List.csv

BLOCK,VARIABLE,MEASURED_VALUE,CONSTRAINT,CONSTRAINT_VALUE,STATISTIC TYPE,RESULT,
Global,app2_latency,1.524500000000002E-6,>,1.5E-6,Max,True,
Global,inst_power,0.8740000000000002,<=,1.0,Max,True,
Global,avg_power,0.586196889703679,<,0.8,Mean,True,
Global,app1_latency,1.6348000000000068E-6,<,1.7E-6,Max,True,

```

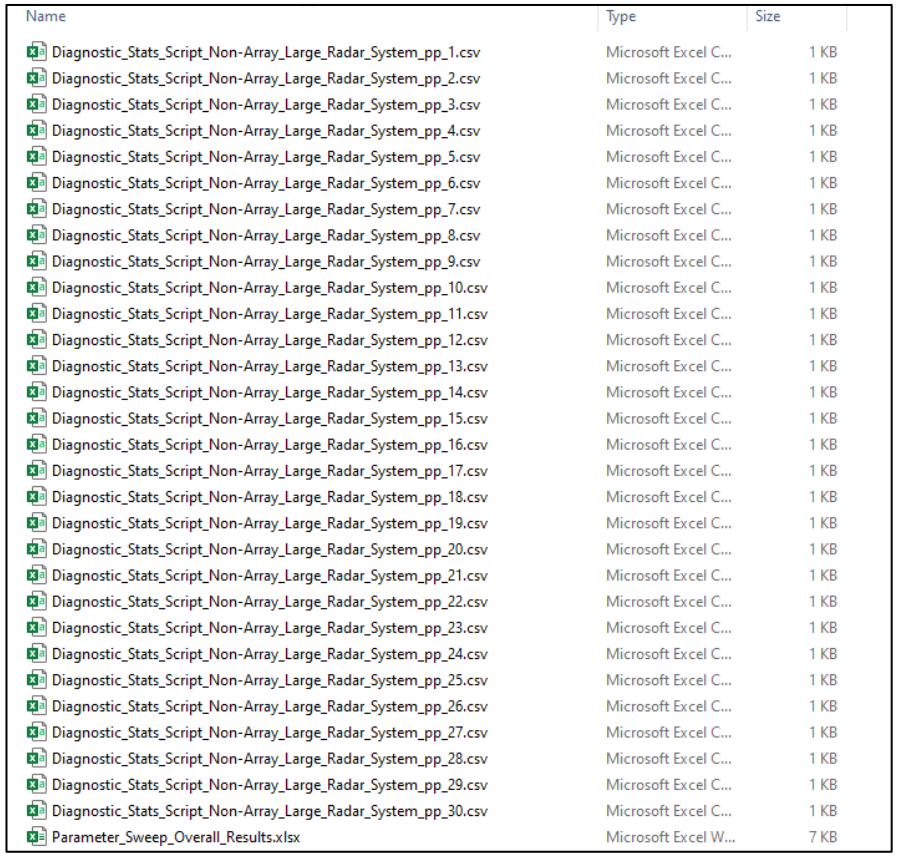
Passing Case:
BUS1_Clk_Speed= 400,
BUS2_Clk_Speed = 400

Conduct Parameter Sweep

```
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 1 -resultpath ./ -Buffer1_Threshold 1 -Mode 1 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 2 -resultpath ./ -Buffer1_Threshold 6 -Mode 1 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 3 -resultpath ./ -Buffer1_Threshold 1 -Mode 2 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 4 -resultpath ./ -Buffer1_Threshold 6 -Mode 2 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 5 -resultpath ./ -Buffer1_Threshold 1 -Mode 3 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 6 -resultpath ./ -Buffer1_Threshold 6 -Mode 3 -TS201_CCLK_MHz 400 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 7 -resultpath ./ -Buffer1_Threshold 1 -Mode 1 -TS201_CCLK_MHz 600 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 8 -resultpath ./ -Buffer1_Threshold 6 -Mode 1 -TS201_CCLK_MHz 600 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 9 -resultpath ./ -Buffer1_Threshold 1 -Mode 2 -TS201_CCLK_MHz 600 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 10 -resultpath ./ -Buffer1_Threshold 6 -Mode 2 -TS201_CCLK_MHz 600 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 11 -resultpath ./ -Buffer1_Threshold 1 -Mode 3 -TS201_CCLK_MHz 600 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 12 -resultpath ./ -Buffer1_Threshold 6 -Mode 3 -TS201_CCLK_MHz 600 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 13 -resultpath ./ -Buffer1_Threshold 1 -Mode 1 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 14 -resultpath ./ -Buffer1_Threshold 6 -Mode 1 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 15 -resultpath ./ -Buffer1_Threshold 1 -Mode 2 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 16 -resultpath ./ -Buffer1_Threshold 6 -Mode 2 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 17 -resultpath ./ -Buffer1_Threshold 1 -Mode 3 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 18 -resultpath ./ -Buffer1_Threshold 6 -Mode 3 -TS201_CCLK_MHz 800 /Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 19 -resultpath ./ -Buffer1_Threshold 1 -Mode 1 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 20 -resultpath ./ -Buffer1_Threshold 6 -Mode 1 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 21 -resultpath ./ -Buffer1_Threshold 1 -Mode 2 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 22 -resultpath ./ -Buffer1_Threshold 6 -Mode 2 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 23 -resultpath ./ -Buffer1_Threshold 1 -Mode 3 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 24 -resultpath ./ -Buffer1_Threshold 6 -Mode 3 -TS201_CCLK_MHz 1000 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 25 -resultpath ./ -Buffer1_Threshold 1 -Mode 1 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 26 -resultpath ./ -Buffer1_Threshold 6 -Mode 1 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 27 -resultpath ./ -Buffer1_Threshold 1 -Mode 2 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 28 -resultpath ./ -Buffer1_Threshold 6 -Mode 2 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 29 -resultpath ./ -Buffer1_Threshold 1 -Mode 3 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
%CLASS_PATH% VisualSim.actor.gui.VisualSimBatchModeSimulator -run 30 -resultpath ./ -Buffer1_Threshold 6 -Mode 3 -TS201_CCLK_MHz 1200 ./Large_Radar_System_pp.xml
```

Different parameter combinations based on the configured ranges are generated and simulated

Diagnostic Block – Requirement tracker output for parameter sweep



Name	Type	Size
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_1.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_2.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_3.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_4.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_5.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_6.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_7.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_8.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_9.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_10.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_11.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_12.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_13.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_14.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_15.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_16.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_17.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_18.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_19.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_20.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_21.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_22.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_23.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_24.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_25.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_26.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_27.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_28.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_29.csv	Microsoft Excel C...	1 KB
Diagnostic_Stats_Script_Non-Array_Large_Radar_System_pp_30.csv	Microsoft Excel C...	1 KB
Parameter_Sweep_Overall_Results.xlsx	Microsoft Excel W...	7 KB

Requirements being evaluated for each simulation run in the parameter sweep

Run Number	Requirement_1 - stFlowLatency	Requirement_2 - mtFlowLatency	RESULT
1	FAIL	FAIL	FAIL
2	FAIL	FAIL	FAIL
3	PASS	PASS	PASS
4	PASS	PASS	PASS
5	FAIL	FAIL	FAIL
6	FAIL	FAIL	FAIL
7	FAIL	FAIL	FAIL
8	FAIL	FAIL	FAIL
9	PASS	PASS	PASS
10	PASS	PASS	PASS
11	FAIL	FAIL	FAIL
12	FAIL	FAIL	FAIL
13	PASS	FAIL	FAIL
14	FAIL	FAIL	FAIL
15	PASS	PASS	PASS
16	PASS	PASS	PASS
17	FAIL	FAIL	FAIL
18	FAIL	FAIL	FAIL
19	PASS	PASS	PASS
20	PASS	FAIL	FAIL
21	PASS	PASS	PASS
22	PASS	PASS	PASS
23	FAIL	FAIL	FAIL
24	FAIL	FAIL	FAIL
25	PASS	PASS	PASS
26	PASS	PASS	PASS
27	PASS	PASS	PASS
28	PASS	PASS	PASS
29	FAIL	FAIL	FAIL
30	FAIL	FAIL	FAIL

Overall Results – We can identify the simulation runs which meet the requirements and select the right configuration after considering cost vs performance trade-offs

BLOCK	VARIABLE	MEASURED_VALUE	CONSTRAINT	CONSTRAINT_VALUE	STATISTIC TYPE	RESULT
Global	mtFlowLatency	0.001800255	<=	0.002	Max	TRUE
Global	stFlowLatency	0.002359969	<=	0.003	Max	TRUE

- Run number 19 – clock frequency at 1000 MHz satisfied the performance requirements we had set.
- Since the frequency was increased from 600 MHz, the total power consumption went up while running the system at 1000 MHz
  - Architect can evaluate different processing resources – DSP vs Fabric vs cores vs AI Tils if they have stringent power thresholds

Accuracy

VisualSim Accuracy (Performance-level)

Deficit RR-based Router

- (Simulated vs. expected)
- 100% for throughput, latency & algorithm
- Input and output rates matched

MPEG Encoder on TI DSP (Software model)

- Customer Feedback
- 100% matched DSP utilization
- 98% of time for end-to-end latency

VisualSim Bus Accuracy (PCI)

Burst 64 Data 256	Actual	VisualSim	Accuracy
Latency	2.16 us	1.97 us	91.29%
Throughput	107 MBps	111.3 MBps	95.92%

Burst 32 Data 128	Actual	VisualSim	Accuracy
Latency	1.20 us	1.06 us	88.33%
Throughput	96 MBps	102 MBps	93.75%

VisualSim Power Accuracy (Architecture-level)

Power Consumption

- MPEG II running on ARM 7
- Tested on Samsung KS32C50100
- With SRAM, accuracy was 86%
  - Actual= 217 vs. VisualSim=188 mW
- With internal Cache, accuracy was 89%
  - Actual=180 vs. VisualSim=199 mW

ARM Cortex M4

Run Detail	ARM Model – Latency (Cycles)	VisualSim Model - Latency (Cycles)	Delta (%)
M4 with AHB Cache – 2KB	150471	150121	0.232
M4 with 0 wait state SRAM	84254	81293	3.5
M4 with 4 wait state SRAM	296931	333203	12.2

ARM Cortex A53

Benchmark	FPGA	VisualSim	Difference	Comments
ED1	5.94ms	6.425ms	7.55%	Integer processing
MM	12.084ms	11.863ms	1.08%	Most load operations with random addresses
MM_st	13.984ms	14.65ms	4.5%	Most store operations with random addresses

Test System

Xilinx Ultrascale+ Zynq® UltraScale+™ XCZU9EG-2FFVB1156E MPSoC running on the ZCU102 board
Specification: 4 core ARM Cortex A53 at 1200Mhz; 32KiB i-cache; 32KiB d-cache, 1MiB L2; 2GB DDR4 DRAM 2400

MIRABILIS design

Comparing Power for ARM Cortex A53

Frequency	Simulated Power	Real (Measured) Power	Delta percentage
500.0 Mhz	0.037 W	0.038 W	2.63%
600.0 Mhz	0.053 W	0.051 W	-3.92%
700.0 Mhz	0.073 W	0.080 W	8.75%
800.0 Mhz	0.097 W	0.090 W	-7.77%
1000.0 Mhz	0.157 W	0.159 W	1.25%
1100.0 Mhz	0.193 W	0.188 W	-2.65%
1200.0 Mhz	0.233 W	0.227 W	-2.64%
1300.0 Mhz	0.277 W	0.269 W	-2.97%

Source: Anandtech.com

Over 97% accuracy

Comparing different Cores- Dhrystone

MIRABILIS design

Processor	Instructions	Latency	Max MIPS
ARM Cortex A53	~ 56,66,000	0.0055846	~ 1039
ARM Cortex A77	~ 44,78,000	0.0011795	~ 3960
RISC-V u74	~ 60,58,000	0.007726	~ 797

Processor	MoP Hit Ratio	MoP Mean Latency	I1 Hit Ratio	I1 Mean Latency	D1 Hit Ratio	D1 Mean Latency	L2 Hit Ratio	L2 Mean Latency	DSU Hit Ratio	DSU Mean Latency
ARM Cortex A53	-	-	99.97	1.93E-09	99.98	2.02E-09	18.75	9.33E-08	-	-
ARM Cortex A77	99.90	1.75E-09	67.22	6.25E-08	99.96	7.32E-10	14.19	1.82E-07	6.96	2.05E-09
RISC-V u74	-	-	99.98	4.15E-09	99.98	1.86E-09	39.58	5.25E-08	-	-

Cycle Accurate Vs Stochastic DRAM – Accuracy Comparision

	Cycle Accurate DRAM			Stochastic DRAM			Delta		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Flow_Latency	0.00236	0.009408	0.005789	0.00227	0.010197	0.005833	2%	8%	1%