



VISUALSIM TRAINING

Agenda- Part 4: Hardware Modeling

Architecture Library Overview 391-401

Configuring Hardware Blocks 402-421

Architecture Setup and Device Interface 422-429

Bus-Cache-RAM 430-452

Processor Modeling 453-480

Technology-specific Hardware Modeling 481-520

Power Modeling 521-576

Architecture Library Overview

Basic Components to define

Hardware modeling requires Architecture_Setup block

If using Traffic or other custom block to connect to a Bus, use the Device_Interface block in front of the Bus port

- Also true for AXI, PCIe and NoC

Processor block requires InstructionSet

Parts of a System

Anything in Electronics

- Network
- Protocols
- DSP
- Processor, memory, bus, cache and DMA
- Accelerators, AI pipeline
- Software task graph, trace file, instruction sequence or software code

Support

- Power Table
- Statistics and Report generation
- Traffic, trace, trigger, interrupt

Exploration

- Parameters

Electronics

Master

- Processor, DMA, Master-Generic

Non-Master

- Bus- Linear (AHB), Crossbar(AXI), Switch and Network-on-chip(CMN600)
- Memory (SRAM, DRAM, Flash)
- Slave- Generic

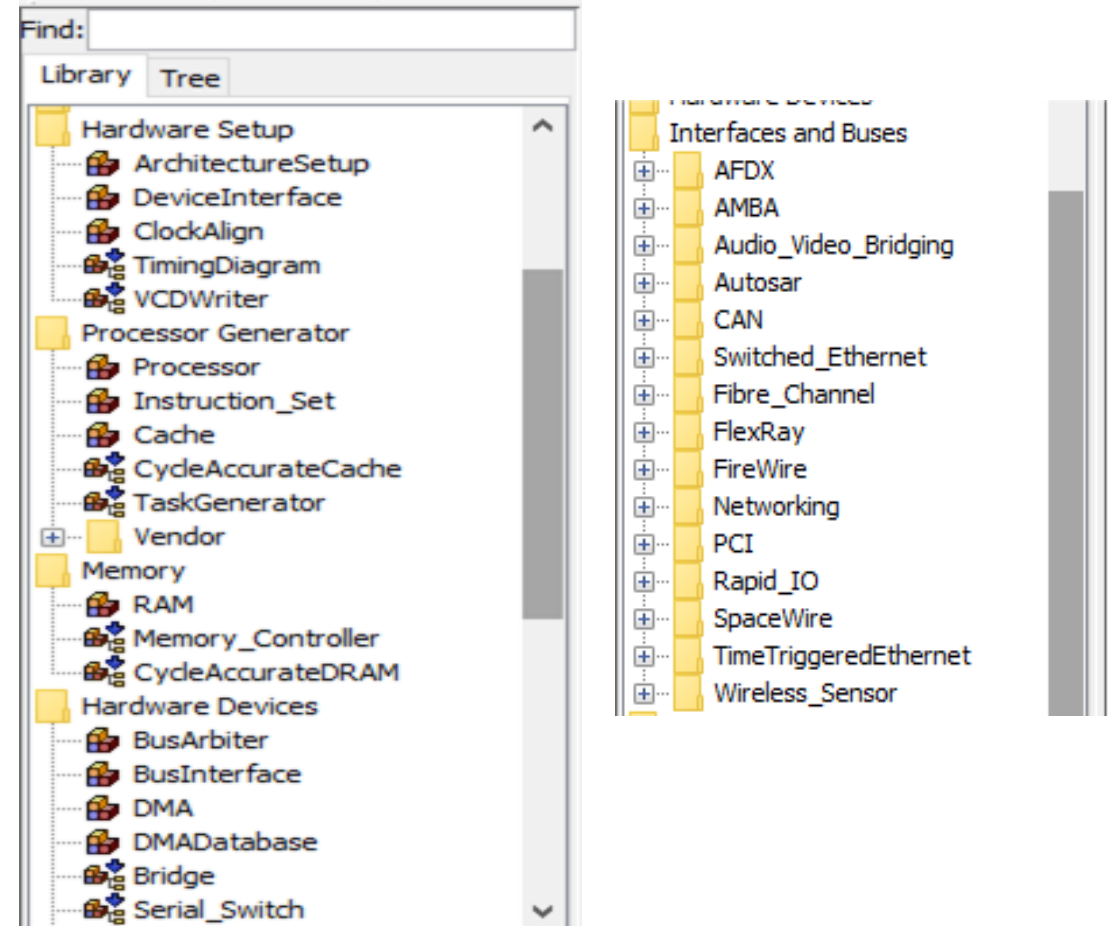
Software

- Traffic, Workflow, Profile-driven, Trace-Driven, Generated profile

Architecture Library Overview

- Generate architecture with parameterized blocks
- Define hardware and software components
- Create proposed or derivative architectures in few minutes
- Rapidly define application flow diagram and behavior
- Optimize architecture and functionality mapping combination

Hardware Modeling Library



Hardware Architecture Exploration

- Size processor, memory, cache, bus & RTOS for target application(s)
- Select arbitration algorithms for switches, custom buses and controller
- Optimize pipeline flows, control operation and input stacks/queue
- Partition applications between processors, ASICs and on FPGAs
- Explore potential architectures against wide-range of traffic stimuli

Software and Algorithm Performance Optimization

- Select or validate target architecture performance for application execution
- Trade-off thread distribution across multi-core or multi-processor systems
- Create stimulus models of the software to handoff to hardware design teams
- Evaluate different arithmetic flow and coefficients for performance on target architectures

Assumptions

- Instruction list is for the timing, branch prediction and load/store operation only
- Behavior of the instruction is ignored except for branches and Load/Store operations
- Sequence of instructions is managed with the Reorder Buffer after the Execution Unit
- Pipeline is a single uninterrupted sequence
 - ✓ Can send request to external devices but there are no intermediate queues
 - ✓ Adding queues or special operations will have an impact on the cycle count. Intelligent planning can minimize the extra cycles
- All instructions read from two registers and write to a third register
- Number of instructions per DS must be less than 4,000 to evaluate the performance

Source of Data

- Microprocessor User's manual
- Programmer Reference Guide
- Key information
 - ✓ Features
 - ✓ List of peripherals and interfaces
 - ✓ Cache and memory
 - ✓ Pipeline
 - ✓ Instruction list by Execution Unit
 - ✓ Timing for each Instruction
 - ✓ List of Execution Units
 - ✓ Clock speed
 - ✓ Minimum peripheral information- clock speed, width, FIFO buffer, line width, memory/cache size, burst size(DMA and Bus), number of channels (DMA)

Model Routing Table

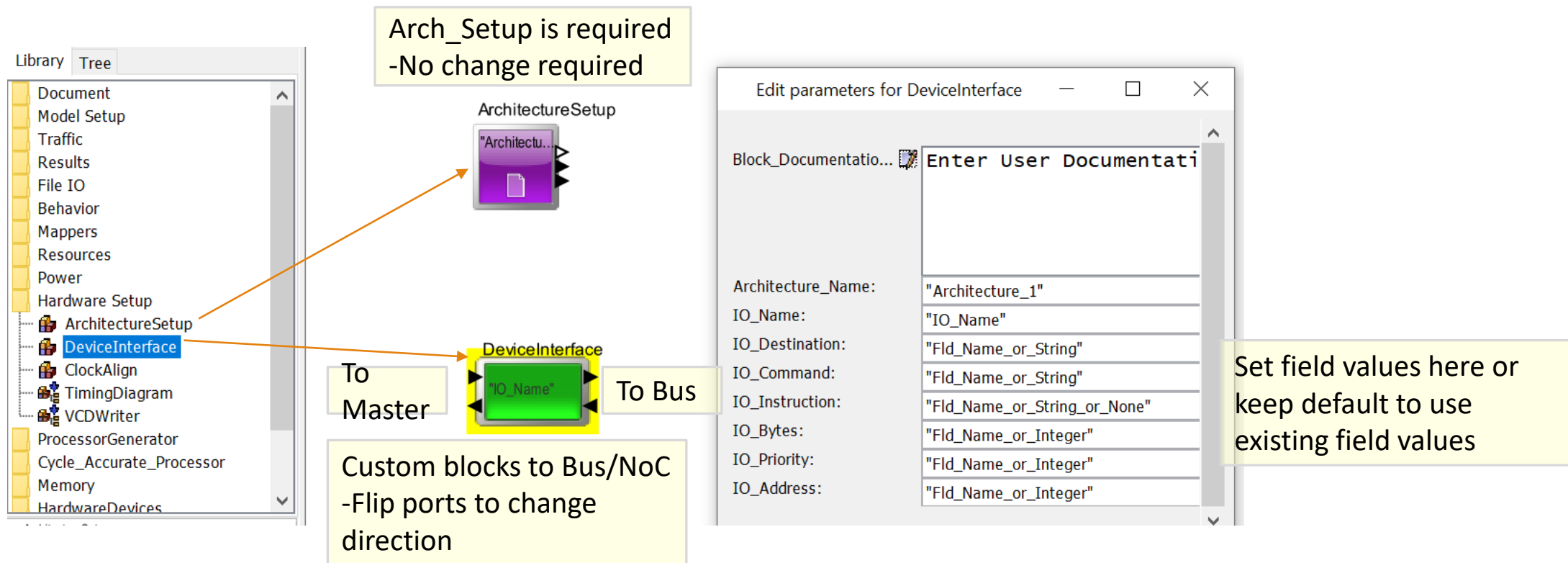
- Provides connectivity information for the model
- Hello messages are sent by all masters and Slaves to add to the Routing Table
- Maintained in the ArchitectureSetup block
- Currently used by
 - ✓ Req-Ack, Linear, AHB and PCI buses
 - ✓ Switches, Processor, DRAM and Cache
- Standard VisualSim hardware blocks get added to the Bus
- Custom blocks can be added by
 - ✓ Connecting the custom components to a DeviceInterface (HardwareSetup -> DeviceInterface) which is in-turn connected to the Bus
 - ✓ Using the Utility Function- addDeviceToBus to enter the connectivity information for this device in the Routing Table

Analysis and Results

- Common Statistics for all devices
 - ✓ End-to-end latency and Task Delay
 - ✓ Throughput (MIPS or MB/s), Utilization (%), Task Delay
 - ✓ Minimum, maximum, mean and standard deviation
- Processor
 - ✓ Individual statistics for Internal Caches, Execution Units, registers, Pipeline
 - ✓ Flush Time, Stall (%), Thread swaps and Context switching
 - ✓ Detailed pipeline activity
- Cache
 - ✓ Hit-miss Ratio

Configuring Hardware

Architecture_Setup and Device_Interface



The image shows a software interface with a Library Tree on the left, two block icons in the center, and a parameter dialog on the right.

Library Tree: A list of components including Document, Model Setup, Traffic, Results, File IO, Behavior, Mappers, Resources, Power, Hardware Setup, ArchitectureSetup, DeviceInterface, ClockAlign, TimingDiagram, VCDWriter, ProcessorGenerator, Cycle_Accurate_Processor, Memory, and HardwareDevices.

ArchitectureSetup Block: A purple block with a document icon and a right-pointing arrow. A callout box above it says: "Arch_Setup is required -No change required".

DeviceInterface Block: A green block with a document icon and two arrows pointing left and right. A callout box below it says: "Custom blocks to Bus/NoC -Flip ports to change direction".

Callouts for DeviceInterface:

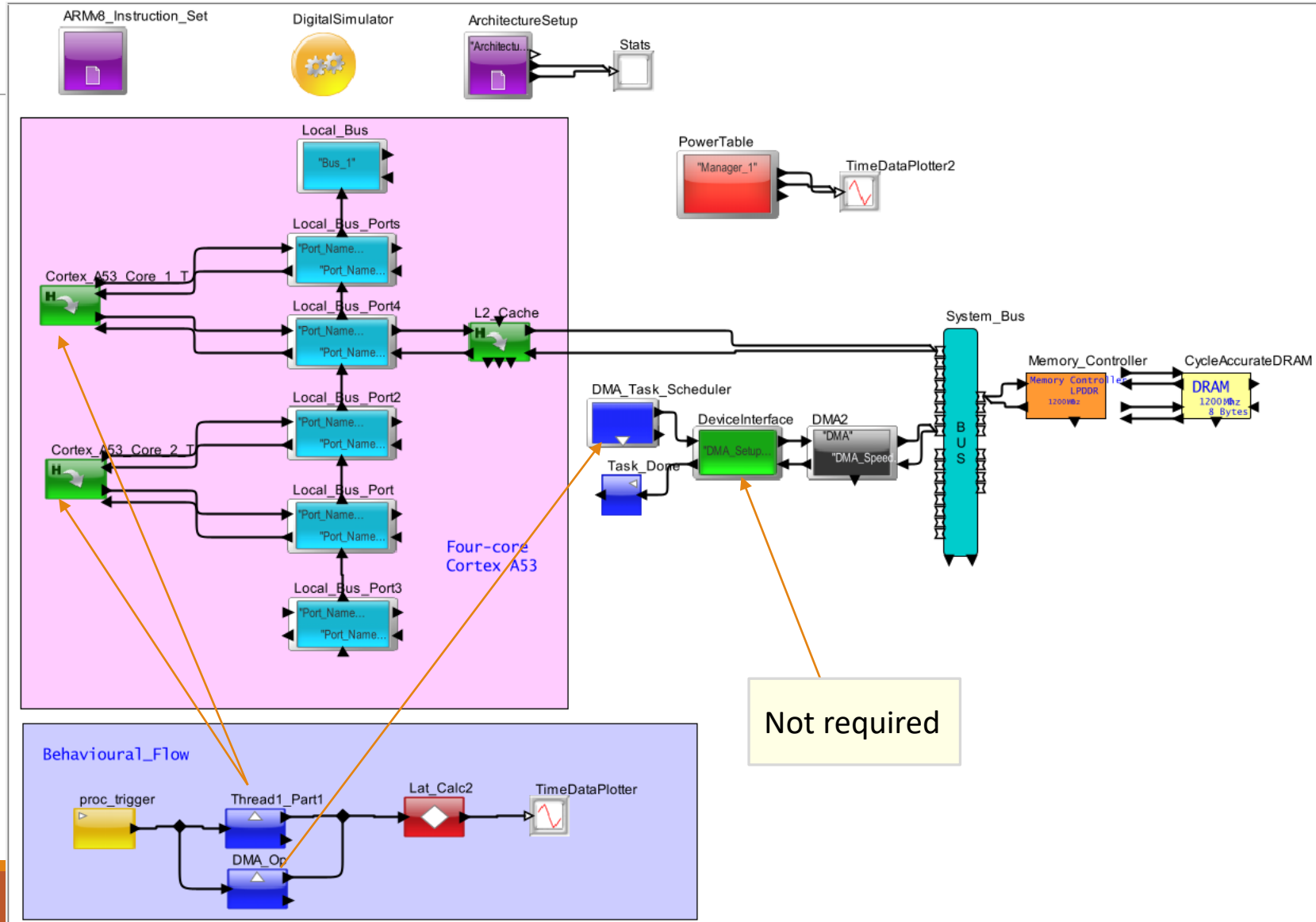
- "To Master" with an arrow pointing to the left port.
- "To Bus" with an arrow pointing to the right port.

Edit parameters for DeviceInterface Dialog: A window with a title bar and a scrollable list of parameters:

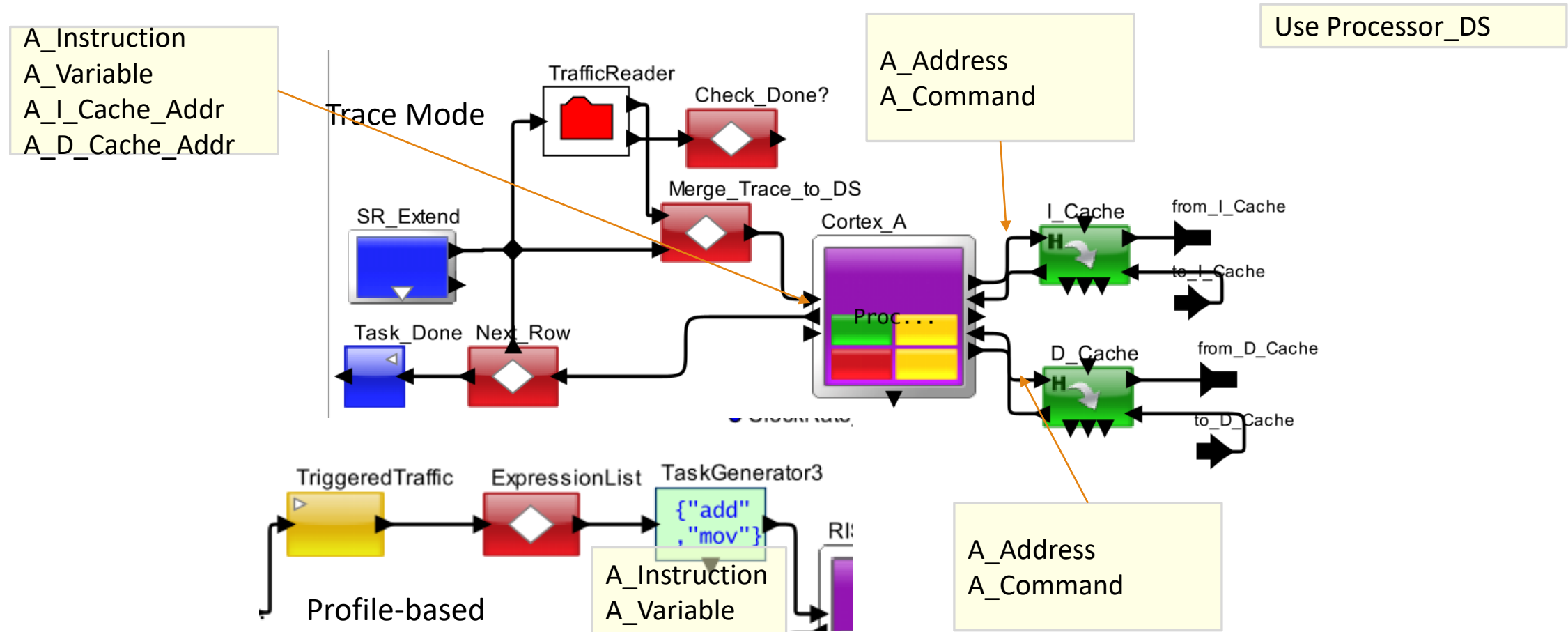
Block_Documentatio...	Enter User Documentati
Architecture_Name:	"Architecture_1"
IO_Name:	"IO_Name"
IO_Destination:	"Fld_Name_or_String"
IO_Command:	"Fld_Name_or_String"
IO_Instruction:	"Fld_Name_or_String_or_None"
IO_Bytes:	"Fld_Name_or_Integer"
IO_Priority:	"Fld_Name_or_Integer"
IO_Address:	"Fld_Name_or_Integer"

 A callout box to the right of the dialog says: "Set field values here or keep default to use existing field values".

Major Blocks and Location



Processor and L1 Cache



Trace Input

Instruction array

Instruction address array

Data cache address array for Load and Store operations

}
Derived from
Processor Fast models

```
I_Cache_Address,A_Instruction,D_Cache_Address  
array,array,array  
{"0x1044c","0x10450","0x10454"},{"mov","mov","ldr"},{"0x00","0x00","0xbeffe50"}
```

Processor

Block_Documentation: Enter User Documentation Here

Architecture_Name: Architecture_Setup_Name

Processor_Name: Processor_Name **Unique Name**

Processor_Setup: /* First row contains Column Names. */

Parameter_Name	Parameter_Value
Processor_Instruction_Set:	ARM_INSTR
Number_of_Registers:	32
Processor_Speed_Mhz:	ClockRate
Context_Switch_Cycles:	4 /* switch between threads */
Instruction_Queue_Length:	128
Instructions_per_Cycle:	2
Number_of_Pipeline_Stages:	10
Number_of_INT_Execution_Units:	5
Number_of_FP_Execution_Units:	1
ROB_Size:	40 /* Re order Buffer size */
Number_of_Cache_Execution_Units:	2 /* I-cache and D-cache */
External_I_Cache:	{}
External_D_Cache:	{Outstanding_Req_Count=2}

Pipeline_Stages: /* First row contains Column Names. */

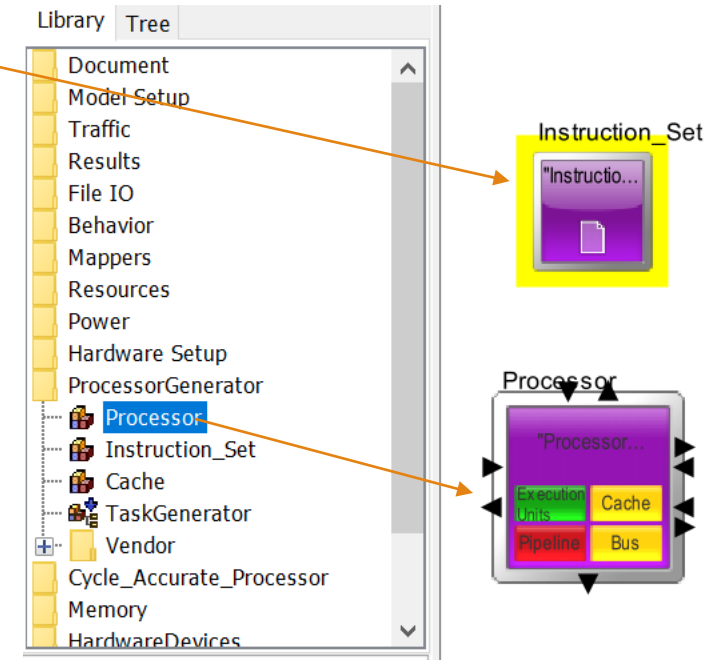
Stage_Name	Execution_Location	Action	Condition
1_INSTRFETCH	I_Cache	instr	none
2_INSTRFETCH	I_Cache	wait	none
3_INSTRFETCH	D_Cache	read	none
4_DECODE	none	exec	none
4_RENAME	D_Cache	wait	none
5_DISPATCH	none	issue	3
6_ROBL	none	exec	none
6_IW	none	exec	none
7_II	none	exec	none
8_EXECUTE	ARM	exec	none
9_EXECUTE	ARM	wait	none
10_STORE	D_Cache	write	none

Enable_Hello_Messages:

Processor_Bits: 64

Required

Key Parameters to modify



```

arm_isa_a65_ae_gem5.txt - Notepad
File Edit Format View Help
/* Instruction Set or File Path. */

Mnew Ra Rb Rc Rd Re Rf Rg Rh ; /* Label */
ARM INTG ALU FPNEOSIMD LDSTR ;

INTG      INT_1 INT_2 ; /* Integer_add */
ALU       INT_3 INT_4 ; /*ALU Branch*/
FPNEOSIMD FP_1      ; /*Floating point/NEON/ASIMD instructions*/
LDSTR     INT_5      ; /*Load/Store instructions*/

begin size_config
  Read   6      64      INT_5[1:163] ;
  Write  6      128     INT_5[164:500] ;
end size_config

begin execUnit_config
  Queue_Size INT_1 16 ;
  Queue_Size INT_2 16 ;
  Queue_Size INT_3 16 ;
  Queue_Size INT_4 16 ;
  Queue_Size FP_1 16 ;
  Queue_Size FP_2 16 ;
  Queue_Size INT_5 12 ;
end execUnit_config

```

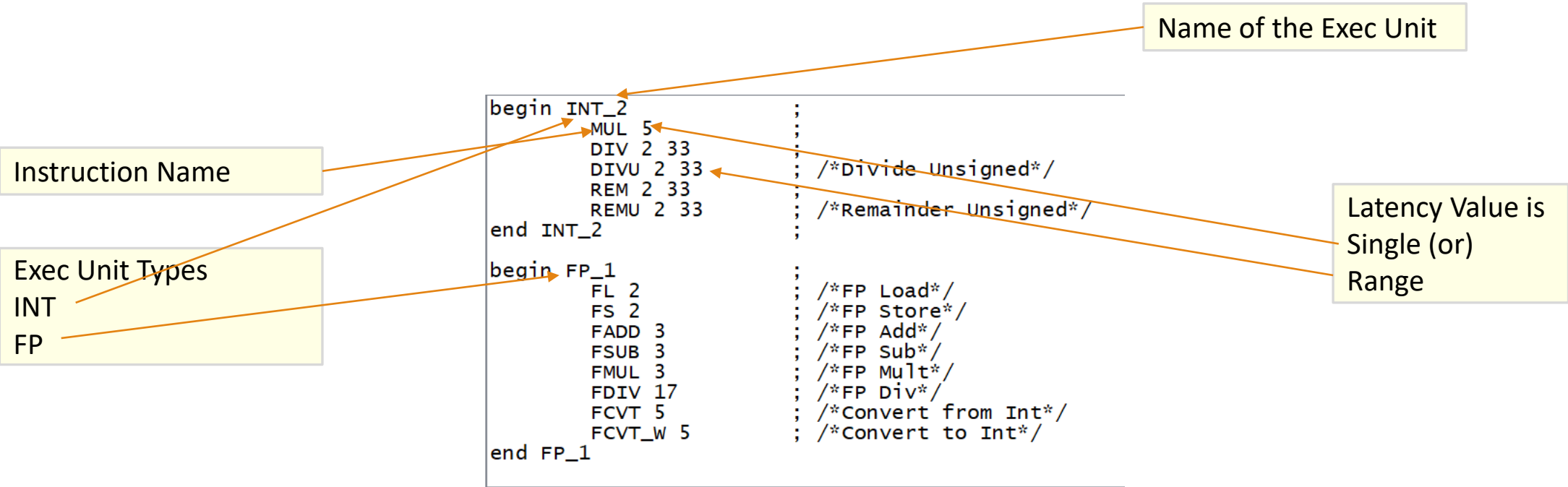
Instruction Set

Execution Units

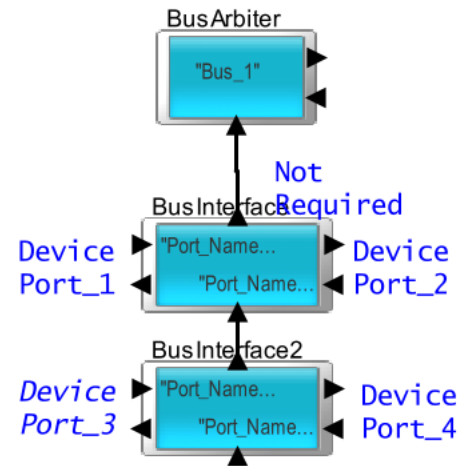
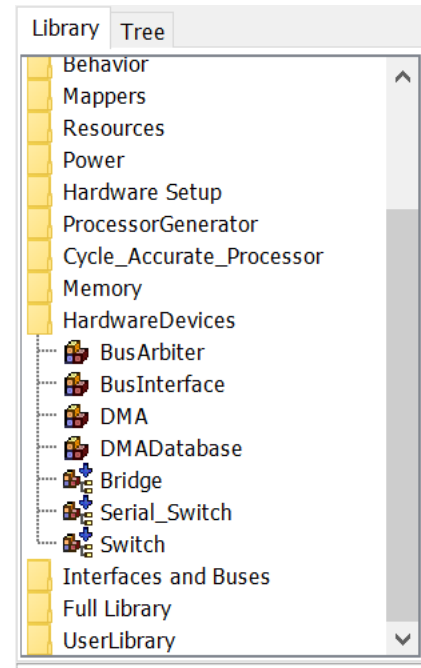
Datapath to D-cache

Issue Queue

Instruction Set




Bus Arbiter and Interface



Port Names in order
 Port_1, Port_2....
 Order match Device List
 Use any num of Interface

Bus Arbiter

Edit parameters for BusArbiter2

Block_Documentation:	 Enter User Documentation Here
Architecture_Name:	"Architecture_1"
_explanation:	HardwareDevices->BusArbiter
Bus_Name:	Cluster_Name+"_DSU_Cache_Bus" Unique Name
Bus_Speed_Mhz:	Core_Speed_Mhz
Burst_Size_Bytes:	100
Round_Robin_Port_Array:	{"Port_1", "Port_2"}
Devices_Attached_to_Slave_by_Port:	{1}, {"Device_2"}, {"Device_3"}, {"Device_4"}, {"Device_5"}, {"Device_6"}, {"Device_7"}, {"Device_8"}}
Width_Bytes:	{16,64} {Read, Write}
Arbiter_Mode:	FCFS
Split_Retry_Flag:	<input checked="" type="checkbox"/>
Enable_Plots:	<input type="checkbox"/>

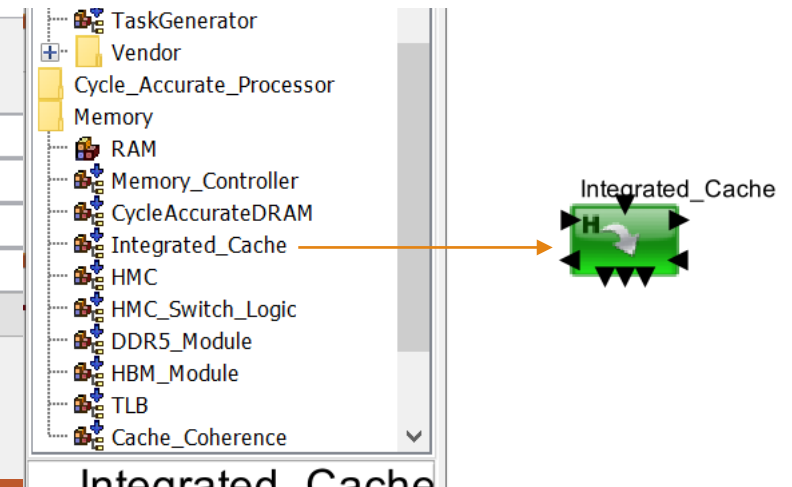
Required

List all devices connected via this port. Index is in order of name- Port_1, Port_2 etc

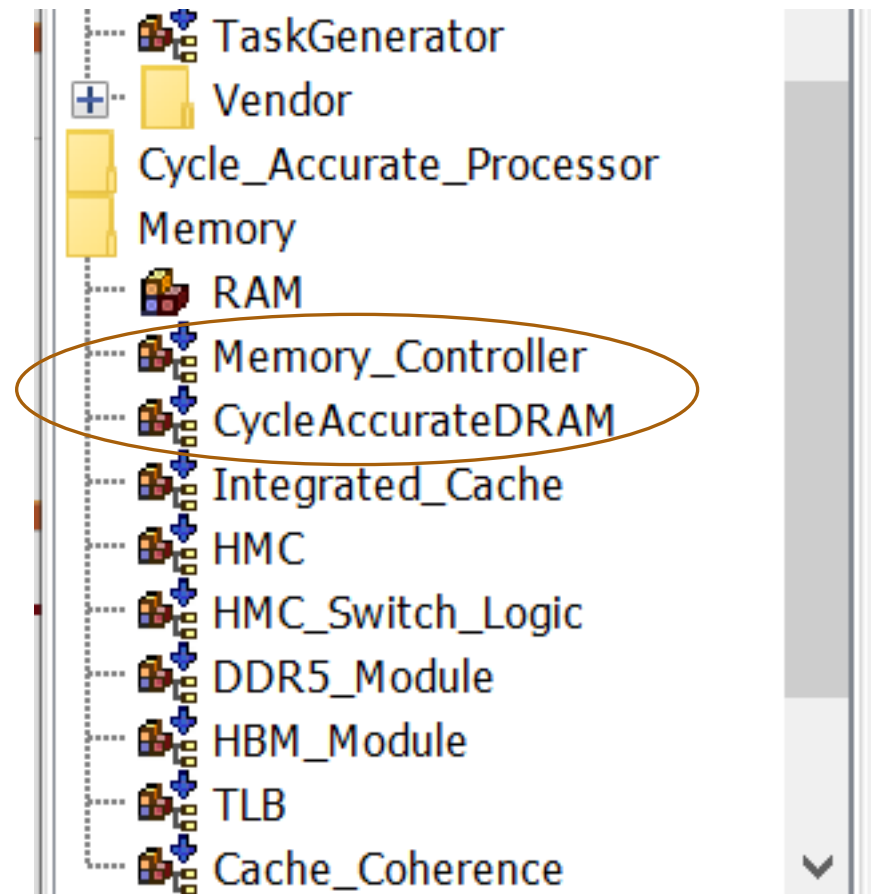
Cache- Key Parameters

Edit parameters for I_Cache

Cache_Name:	Core_Name+"_I_Cache"	Unique name
Cache_Speed_Mhz:	Core_Speed_Mhz	}
Cache_Width_Bytes:	4	
Cache_Size_Bytes:	I_Cache_Size_Bytes	
Block_Size_Bytes:	Cache_Block_Size_Bytes	
Hit_Ratio:	0.8	} Stochastic
Loop_Ratio:	0.2	
Overhead_Cycles:	0	Extra cycle to match hardware
Cache_Replacement_Policy:	Pseudo-LRU	} Minimum Required field
Cache_Write_Policy:	Write_Back	
Stochastic_or_Address_Based:	Address_Based	} Optional but increase accuracy
Miss_Memory_Name:	Core_Name+"_L2_Cache"	
Power_Manager_Name:	"none" /*To analyse power, link the manager name */	
First_Word:	true	
No_of_Statistics:	2	
Req_Buffer_Size:	16	
Architecture_Name:	"Architecture_1"	
Enable_Hello_Messages:	<input checked="" type="checkbox"/>	
Output_Flow_Control:	<input type="checkbox"/>	
N_Way_Associativity:	4	
Main_Memory_Size_KB:	2048	
Main_Mem_Start_Addr:	0	
Fill_Buffer_Size:	10	
Cache_Type:	I_Cache	Used for stochastic
VIPT_Mode:	<input type="checkbox"/>	
Input_Flow_Control:	<input type="checkbox"/>	
Debug:	<input type="checkbox"/>	



Memory Controller and HW_DRAM

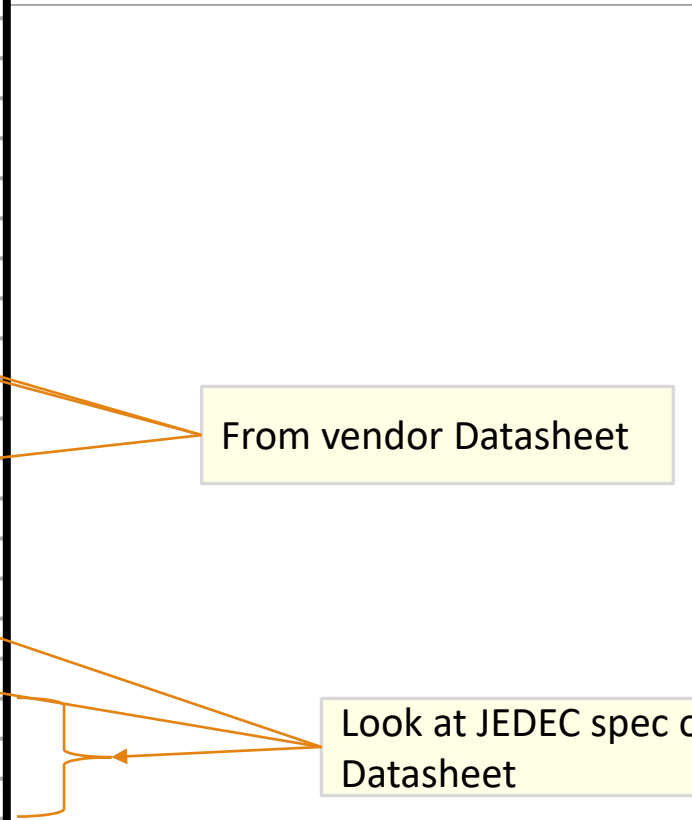


Memory Controller

Edit parameters for Memory_Controller	
Architecture_Name:	"Architecture_1"
Controller_Name:	"MC"
Controller_Speed_Mhz:	1000.0
Memory_Width_Bytes:	4
Bus_Width_Bytes:	4
Command_Buffer_Length:	8
Commands_in_a_Row:	8
Memory_Column:	{0,9}
Memory_Row:	{10,24}
Memory_Bank:	{25,28}
DRAM_Return_Cycles:	0
DEBUG:	true
_explanation:	Hardware_Modeling->Memory->Memory_Controller
Memory_Rank:	{29}
Burst_Length:	4 /* 2, 4, 8 */
First_Word_Flag:	true
HW_DRAM_Name:	"DRAM"
Power_Manager_Name:	"none" /* Default */
Mfg_Suggest_Timing:	{16,16,16,32} /* tCL, tRCD, tRP, tRAS */
Extra_Timing:	{0,2,1,1,3,1,0,1,0,30} /* DQSS, tWTR, tRRD, tWR, tRL, tWL, tDQCK, tRTP, tHWpre, tFAW */
DDR4_Timing:	{6.0,4.0,5.0,6.0,4.0,5.0,30.0} /* CCD_L, CCD_S, CCD, RRD_L, RRD_S, RRD, FAW; units clks, except FAW ns */
Number_of_Samples:	10
DRAM_Type:	DDR4
Number_of_Ranks:	2
Number_of_Bank_Groups:	2
writeStats_to_File:	false

From vendor Datasheet

Look at JEDEC spec or Datasheet



HW_DRAM- DDR, LPDDR, GDDR

Edit parameters for CycleAccurateDRAM

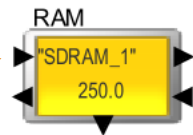
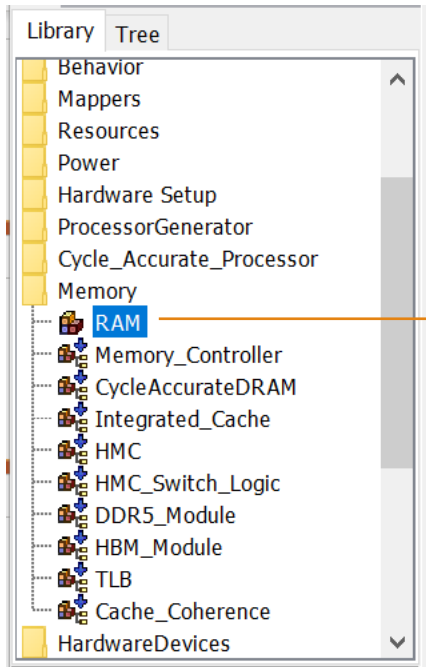
Architecture_Name:	"Architecture_1"
HW_DRAM_Name:	"DRAM"
HW_DRAM_Speed_Mhz:	1000.0
Sim_Time:	1.0E-05
_explanation:	Hardware_Modeling->Memory->HW_DRAM
Memory_Width_Bytes:	4
Burst_Length:	4 /* 2, 4, 8 */
Mfg_Suggest_Timing:	{16,16,16,32} /* tCL, tRCD, tRP, tRAS */
Extra_Timing:	{0,2,1,1,3,1,0,1,0,16} /* DQSS, tWTR, tRRD, tWR, tRL, tWL, tDQCK, tRTP, tHWpre, tFAW */
Retention_Time:	64.0E-03 /* 64.0 msec */
Enable_External_Data:	false
Address_Bit_Map:	{{0,9},{10,24},{25,27},{30}} /* col, row, bank, rank (min, max) Bit Position */
Standard_Name:	"none" /*reads DDR_Memory_Standards.txt */
Standard_File:	<input type="text"/> Browse
Power_Manager_Name:	"none" /* Default */
Memory_Controller:	"MC"
DEBUG:	false
State_Plot_Enable:	false
DRAM_Type:	DDR4
Fine_Granularity_Refresh:	FGR_1x
Fine_Granularity_Refresh_Tim...	166.0E-09
REFpb_T_REFab_F:	true
Refresh_Statistical:	true
Same_Bank_Refresh:	false

From vendor Datasheet

Look at JEDEC spec or Datasheet

JEDEC & User-setting

SRAM and Stochastic Memory



Architecture_Name:	"Architecture_1"
Memory_Name:	"SDRAM_1"
Memory_Speed_Mhz:	250.0
Memory_Size_MBytes:	64.0
Access_Time:	"Read 5.0, Prefetch 6.0, Write 7.0, ReadWrite 8.0, Erase 9.0"
FIFO_Buffers:	32
Refresh_Rate_Cycles:	16384
Refresh_Cycles:	32
Memory_Address:	"/ * Format: Min_Address,Max_Address. Example:201,300 */"
Controller_Time:	"Cycle_Time * 1.0"
Enable_Hello_Messages:	<input checked="" type="checkbox"/>
Width_Bytes:	4
Memory_Type:	SDR
Refresh:	true

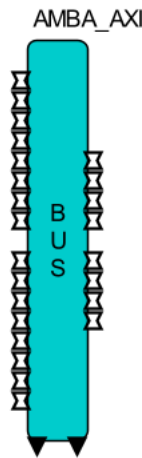
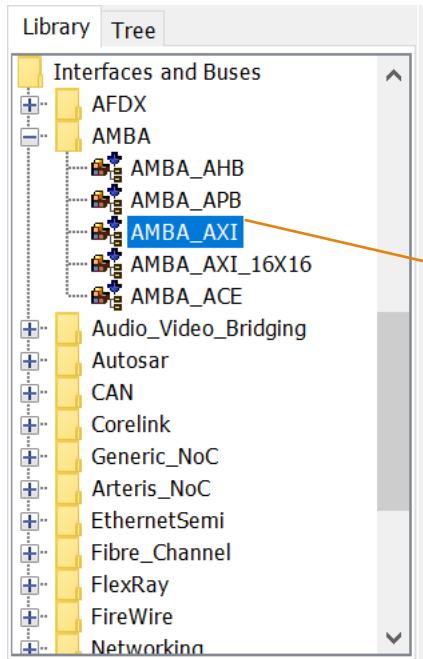
Required field

In ns for one width access

Activate & Overhead cycles
Value is expression, variable or field

For SRAM, set to false

AXI



Edit parameters for AMBA_AXI

Architecture_Name:	"Architecture_1"
Bus_Name:	"AXI_Top"
AXI_Speed_Mhz:	1000.1 Required
AXI_Cycle_Time:	1.0E-06 / AXI_Speed_Mhz
_explanation:	Interfaces and Buses->AHB->AXI_Bus
Bus_Width:	8 Required
Read_Threshold:	2 Required
Write_Threshold:	2 Slave Buffer
Master_Request_Threshold:	{2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2} Master Buffer (Not required)
Number_Masters:	16 Speed up simulation. List last
Number_Slaves:	8 connected number
Threshold_Trans_T_Bytes_F:	true
Arbiter_FIX_1_RR_2_CUSTOM_3:	1
Slave_Speeds_Mhz:	1z, AXI_Speed_Mhz, AXI_Speed_Mhz, AXI_Speed_Mhz, AXI_Speed_Mhz, AXI_Speed_Mhz
Extra_Cycles_for_RdReq_WrReq_RdData_WrData:	{0, 0, 0, 0, 0, 0, 0, 0}
Devices_Attached_to_Slave_by_Port:	Device_2, {"Device_3"}, {"Device_4"}, {"Device_5"}, {"Device_6"}, {"Device_7"}, {"Device_8"} Required. List all Devices through each Slave
Master_First_Word_Flag:	true
Master_Throttle_Enable:	{false,false,false,false,false,false,false,false,false,false,false,false,false,false}
Slave_Throttle_Enable:	{false,false,false,false,false,false,false}
DEBUG:	false
Custom_Arbiter_File:	"none"
Custom_Arbiter_Path:	"none"
Fixed_Priority_Array:	,15,16},{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16},{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}}
Slave_First_Word_Flag:	true /* Not Active in Default Slave */
Custom_Slave_File:	"none"
Ports_to_Plot:	{{1,1}} /* {{n,m},{..}} - nth master, mth slave */
AXI_Sync_Speed_Mhz:	AXI_Speed_Mhz

Required. List all Devices through each Slave

If flow control support for Master/Slave device

PCIe

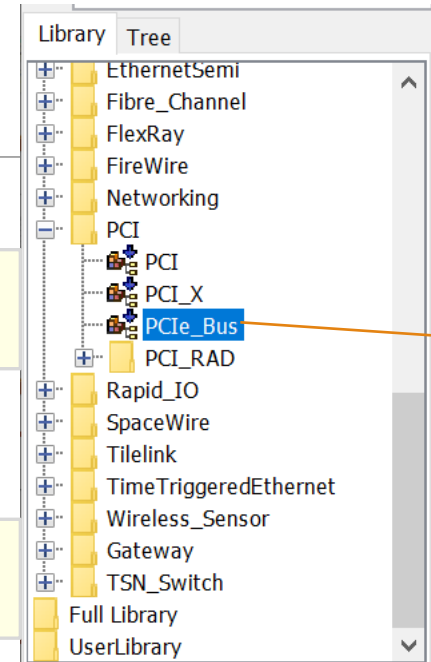
Edit parameters for PCIe_Bus

Architecture_Name:	"Architecture_1"
Bus_Name:	"PCIe_1"
Number_of_Lanes:	16 /* Can be an array */
Slave_Buffer:	512 /* Max Bytes @ Slave */
Master_Buffer:	512 /* Max Bytes @ Master */
_explanation:	Interfaces and Buses->PCI->PCIe_Bus
Header_Bytes:	16 /* 32 Bit Mode, includes CRC Bytes */
Number_of_Ports:	{12, 12} /* Master, Endpoint Ports */
BER:	1E-11
Max_Payload_Size:	64 /* Write, Read Data */
Max_Payload_Req_Size:	128 /* Read Requests */
Read_to_Write_Ratio:	0.5 /* 0.0 to 1.0 */
Devices_Attached_to_Slaves:	{RAM_3},{Dev_4},{Dev_5},{Dev_6},{Dev_7},{Dev_8},{Dev_9},{Dev_10},{De
Root_Complex_Flow_Control:	{false,false,false,false,false,false,false,false,false,false,false,false,false,false,false}
Endpoint_Flow_Control:	{false,false,false,false,false,false,false,false,false,false,false,false,false,false,false}
Enable_Plots:	<input checked="" type="checkbox"/>
Bit_64_Mode:	<input checked="" type="checkbox"/>
NumOfRetry:	4
Timeout:	6E-6
PCIe_MBps:	PCIe_Gen_1

Can be common or array for each port in order starting from top-left and continuing through top-right

Required: List of all the Devices connected to each End-Point

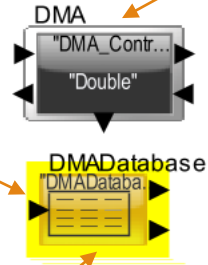
Required: Determines speed



DMA

Library Tree

- Behavior
- Mappers
- Resources
- Power
- Hardware Setup
- ProcessorGenerator
- Cycle_Accurate_Processor
- Memory
- HardwareDevices
 - BusArbiter
 - BusInterface
 - DMA
 - DMADatabase**
 - Bridge
 - Serial_Switch
 - Switch
- Interfaces and Buses
- Full Library
- UserLibrary



Block_Documentation: Enter User Document

Architecture_Name: "Architecture_1"

DMA_Controller_Name: "DMA_Controller_1"

Memory_Database_Reference: "None" **Use Input (none) or Database (Name)**

DMA_to_Device_Cycles: RegEx_or_None

Device_to_DMA_Cycles: RegEx_or_None

Channel_FIFO_Buffers: Integer

Speed_Mhz: Double

DMA_Channels: 8 **Required**

Outstanding_Req_Count: {1, 1, 1, 1, 1, 1, 1, 1} **Per channel, num sent before Ack**

Width_Bytes: 4

Expression_List: /* Template to enter multiple RegEx lines*/

```

input.A_Task_Name = "Activity1"
input.A_DMA_Command = {"Read","write"}
input.A_DMA_Destination = {"RAM","Display"}
input.A_Priority = irand(1,10)
input.A_DMA_Bytes = {1024,1024}
input.A_DMA_Channel = {1,1}
input.A_DMA_Burst_Bytes = {64,64}
    
```

Input to DMA block

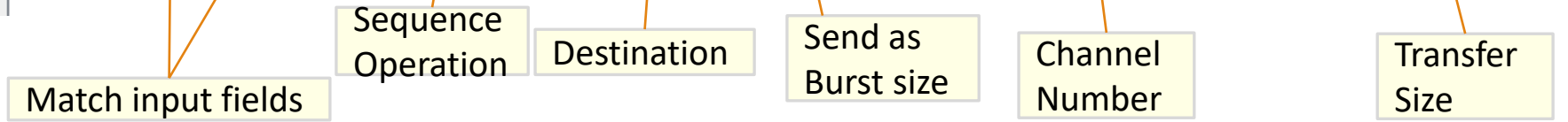
**Sequence of items
Index match
Common Priority**

Linking_Name: "DMADatabase"

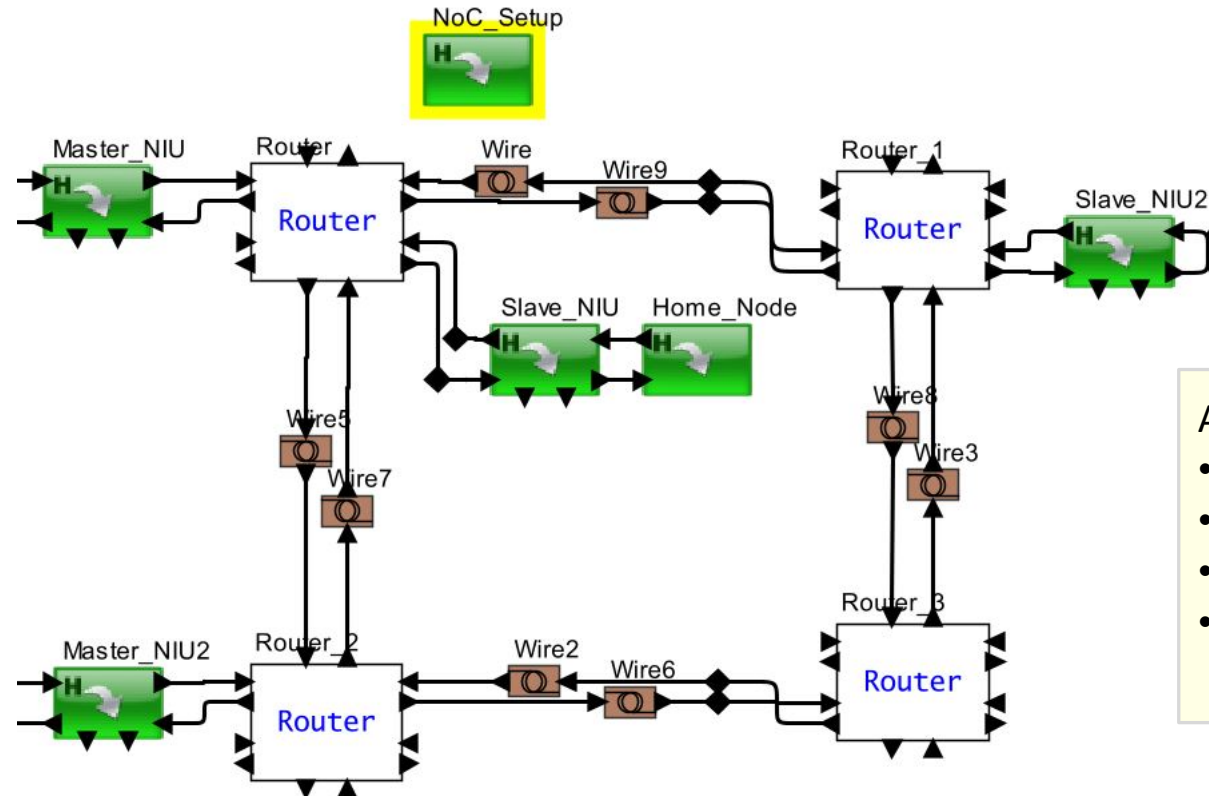
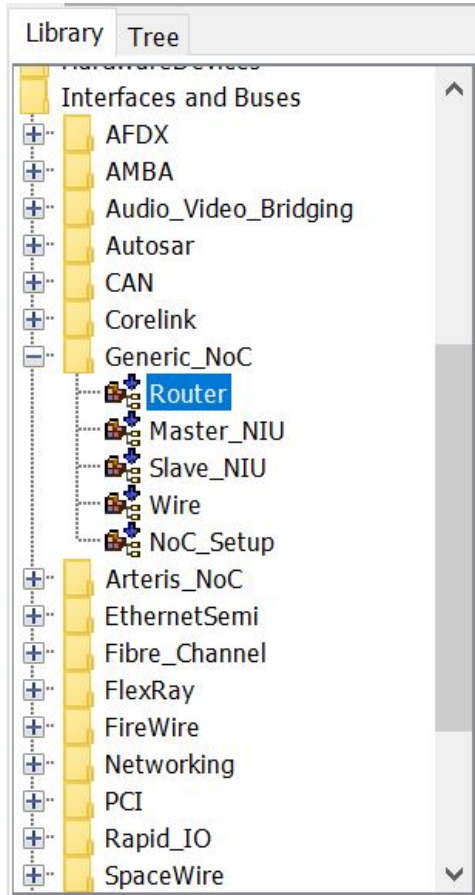
fileOrURL:

Data_Structure_Text:

A_Task_Name	A_Instruction	A_IDX	A_Task_Source	Burst_word_Size	A_Task_Address	A_Command	A_Bytes	A_Priority	A_Destination
Activity1	Load	0	SRAM	32	1	Read	32	0	DMA ;
Activity1	Store	0	DRAM	32	1	Write	32	1	DMA ;
Activity2	Store	0	DRAM	32	2	Write	32	3	DMA ;



Generic NoC (Other NoCs are similar)



- All blocks required
- Use Master NIU when device is to left
 - Use Slave NIU when device is to right
 - Wire is optional for delay and power
 - NoC Setup is just a hierarchical block and inside blocks can be edited

Router, NIU and Wire

Edit parameters for Router

Router_Speed_Mhz:	200.0 /* in Mhz */	Processing Speed
Node_Name:	"R_"+x_val+"_"+y_val /* Do not Modify */	Used for name and Mesh routing
Power_Manager_Name:	"Manager_1"	
x_val:	0	
y_val:	0	
Topology:	Mesh	Used with Loop Topology
No_of_Routers:	4 /* configure only in Loop Topology*/	
Priority_Enable:	true	
Stats_Enable:	<input type="checkbox"/>	
Interconnect_QoS:	Bandwidth_Limiter	Match this with BW
No_Of_VC_Per_Port:	5	
Interface_Buffer_Size:	10	
Bandwidth_Per_VC_MBps:	{200.0,200.0,200.0,200.0,200.0,200.0} /* {"North"	Bandwidth is sum

Commit Add Remove Restore Defa

Edit parameters for Master_NIU

NIU_Name:	"Core_1_NIU"	
Frequency_Mhz:	1200.0	Processing Speed
Flit_Size_Bytes:	4 /* 32 bits */	
QoS_Regulator_Mode:	None	
Stats_Enable:	<input type="checkbox"/>	
Power_Manager_Name:	"Manager_1"	
Output_Buffer_Size:	30	Input is from Device Output is from NoC
Input_Buffer_Size:	30	
Connected_Device:	{ } /* Optional - Device names	Required for Routing

Edit parameters for Wire

Start_Device:	"R_1_1"	
End_Device:	"R_1_2"	
Delay_Name:	Start_Device + "_to_" + End_Device	
Wire_Length:	1e-8	
_flipPortsVertical:	false	
_flipPortsHorizontal:	false	
_rotatePorts:	0	
Clock_Speed_Mhz:	200.0	Processing Speed
Wire_ID:	1	
Wire_Width_Bits:	16	
Repeater_Register:	1	Number of

Architecture Setup and Device Interface

Architecture Setup

All Bus and Hardware block must associate with Architecture Setup

This Block Handles

- Routing
- Plotting
- Statistics
- Debugging for all the Hardware components

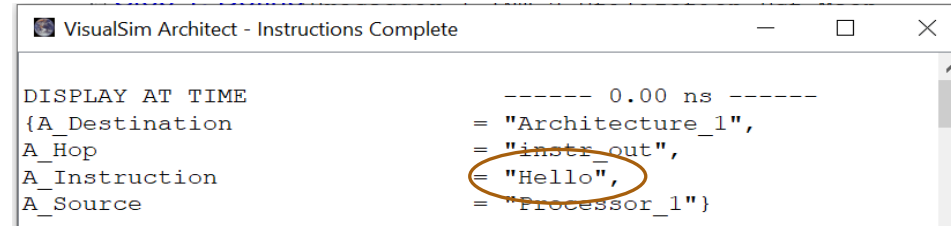


1. There can be multiple architecture setup blocks
2. Each block must have unique name

Important Points for Architecture Setup

1. Routing table

- No need to update this Table
- Use the Device list in the interfaces blocks to define the path to a Slave
- Used to create a connectivity topology between devices connected together
- Sends out “Hello” messages to determine node-to node connectivity



```
VisualSim Architect - Instructions Complete
----- 0.00 ns -----
{A_Destination      = "Architecture_1",
 A_Hop              = "instr_out",
 A_Instruction      = "Hello",
 A_Source           = "Processor_1"}
```

The A_Destination field will be the name of the Architecture_Setup

2. Plotting

- Use the output_plot port to capture statistics in a graphical view
- Need to Configure Parameter for the port to set the number of datasets

3. Statistics

- Statistics accumulated during simulation are sent out multiple times

3. Listener

- View debugging messages for all blocks

Routing Table Construction

- in Architecture Setup

Format Sample:

Source_Node	Destination_Node Hop	Source_Port
Source Processor	Destination DRAM	Next Block Port_1
		Out Port bus_out

Add entries using RegEX


`addToRoutingTable (Architecture_Name, Source_Name, Destination_Name, Hop_Name, Source_Port_Name)`

Delete entries using RegEX


`removeFromRoutingTable (Architecture_Name, Source_Name, Destination_Name, Hop_Name, Source_Port_Name)`

Architecture Setup Configuration

Edit parameters for ArchitectureSetup

Block_Documentation:  This block maintains the Routing information and generates stats|

Architecture_Name: "Architecture_1"

Routing_Table: 

```
/* First row contains Column Names.          */
Source_Node      Destination_Node  Hop      Source_Port ;
Processor_1      L2_Cache          Port_Name_1 bus_out      :
Processor_2      L2_Cache          Port_Name_1 bus_out      :
L2_Cache         DRAM              Port_Name_4 output       :
DRAM             Processor_1       Port_Name_1 output       :
DRAM             Processor_2       Port_Name_3 output       :
Port_Name_1      L2_Cache          Port_Name_2 output2      :
Port_Name_2      DRAM              Port_Name_4 output2      :
Port_Name_3      L2_Cache          Port_Name_2 output2      :
Port_Name_4      Processor_1       Port_Name_1 output1      :
Port_Name_4      Processor_2       Port_Name_3 output1      :
```

Number_of_Samples: 2

Statistics_to_Plot: "Processor_1_PROC_Utilization_Min, Processor_1_PROC_Utilization_Mean, Processor_1_PROC_Utilizatio

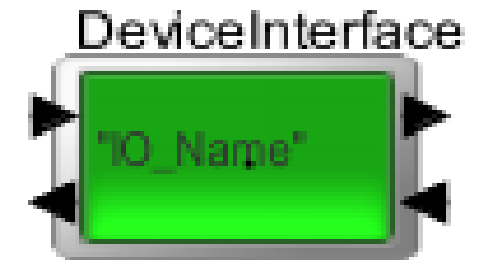
Internal_Plot_Trace_Offset: 2

Listen_to_Architecture_Options: None

Let's learn how the flow works

DeviceInterface Block

- Can be used to define the Source Name, data size, command type and destination for a Master
- Can be used to define the Device name on the Slave side
- Add the Master or Slave block to the Linear Bus, Bridge and AHB Buses automatically
 - ✓ Generate Hello Messages
 - ✓ Eliminates the need for a RegEx functions or manually generate Hello message
- Map fields of other data structure formats to the corresponding fields of the Processor_DS
- Used to connect only in the presence of the Linear Bus, AHB and bridge blocks.
- Not useful with a single AXI or a single PCIe



Hardware Statistics

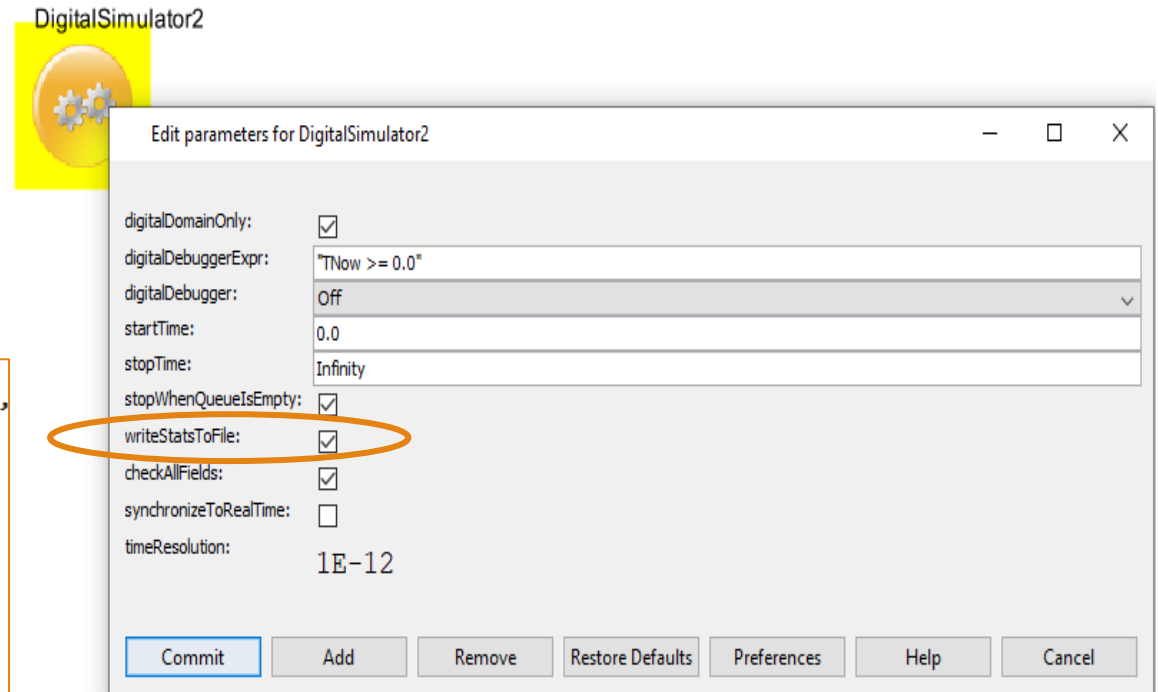
Name	Value	Name	Value	Name	Value
Bus_1_Utilization_Pct_Max	10.1,	Bus_1_Delay_Max	3.600004E-8,	Processor_1_D_1_Hit_Ratio_Max	100.0,
Cache_1_Utilization_Pct_StDev	1.05689435,	Bus_1_IOs_per_sec_StDev	634428.24721,	Processor_1_D_1_KB_per_Thread_StDev	0.0,
Processor_1_D_1_Utilization_Pct_Max	1.45,	Bus_1_Input_Buffer_Occupancy_in_Words_Max	32.0,	Processor_1_I_1_Hit_Ratio_Max	100.0,
Processor_1_INT_1_Utilization_Pct_StDev	0.10969973,	Bus_1_Preempt_Buffer_Occupancy_in_Words_StDev	0.0,	Processor_1_I_1_KB_per_Thread_StDev	0.0,
Processor_1_INT_2_Utilization_Pct_Max	24.55,	Bus_1_Throughput_MBs_Max	176.0,	Processor_1_L_2_Hit_Ratio_Max	100.0,
Processor_1_I_1_Utilization_Pct_StDev	0.01500007,	Cache_1_Delay_Time_StDev	1.499871E-8,	Processor_1_L_2_KB_per_Thread_StDev	0.0,
Processor_1_L_2_Utilization_Pct_Max	3.1,	Cache_1_Hit_Ratio_Max	100.0,	Processor_1_Stall_Time_Pct_Max	1.35,
Processor_1_PROC_Utilization_Pct_StDev	0.03489914,	Cache_1_Memory_Used_By_Processor_1_MB_StDev	8.455181E-5,	Processor_1_Task_Delay_StDev	2.9502E-7,
Processor_1_Pipeline_Utilization_Pct_Max	50.2,	Cache_1_Memory_Used_By_SDRAM_1_MB_Max	2.56E-4,	SDRAM_1_Delay_Time_Max	1.5E-7,
Processor_1_Register_Rd_Utilization_Pct_StDev	0.19330594,	Cache_1_Memory_Used_By_Total_MB_StDev	8.455151E-5,	SDRAM_1_Memory_Used_By_Processor_1_MB_StDev	0.0,
Processor_1_Register_Wr_Utilization_Pct_Max	49.55,	Cache_1_Throughput_MBs_Max	116.0,	SDRAM_1_Memory_Used_By_Total_MB_Max	2.56E-4,
SDRAM_1_Utilization_Pct_StDev	1.91E-07	Processor_1_Context_Switch_Time_Pct_StDev	0.0,	SDRAM_1_Throughput_MBs_StDev	0.0,
				DMA_IO_per_sec_Max	7.45E6,
				DMA_Throughput_MBs_StDev	3.463999999,

writeStats To File

- Generates Statistics for all the blocks in the model at the end of simulation
- Writes into a Text File in the model directory

```

Queue_Statistics      6.000000000000 sec
{BLOCK                = "SR_SrExtend_example.SystemResource_Extend",
DELTA                 = 0.0,
DS_NAME               = "Queue_Common_Stats",
ID                    = 1,
INDEX                 = 0,
Number_Entered        = 7,
Number_Exited         = 1,
Number_Rejected       = 0,
Occupancy_Max         = 6.0,
Occupancy_Mean        = 3.77777777777778,
Occupancy_Min         = 1.0,
Occupancy_StDev       = 1.4740554623802,
Queue_Number         = 1,
TIME                  = 6.0,
Total_Delay_Max       = 4.0,
Total_Delay_Mean      = 4.0,
Total_Delay_Min       = 4.0,
Total_Delay_StDev     = 0.0,
Utilization Mean      = 0.0}
    
```



Bus-Cache-RAM

Bus Arbiter

Communication channel between master and slave devices

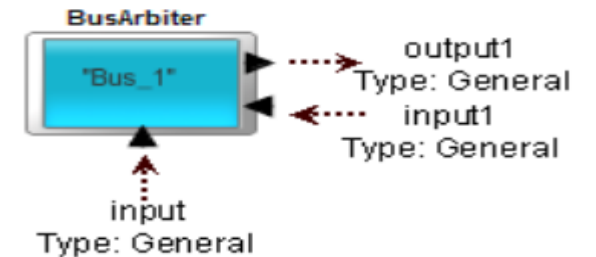
- Buses created using this block include AMBA (AHB and APB) and PCI

Arbiter_Mode supports

- First Come-First Server
- Round-Robin
- Custom

Request on the input Ports

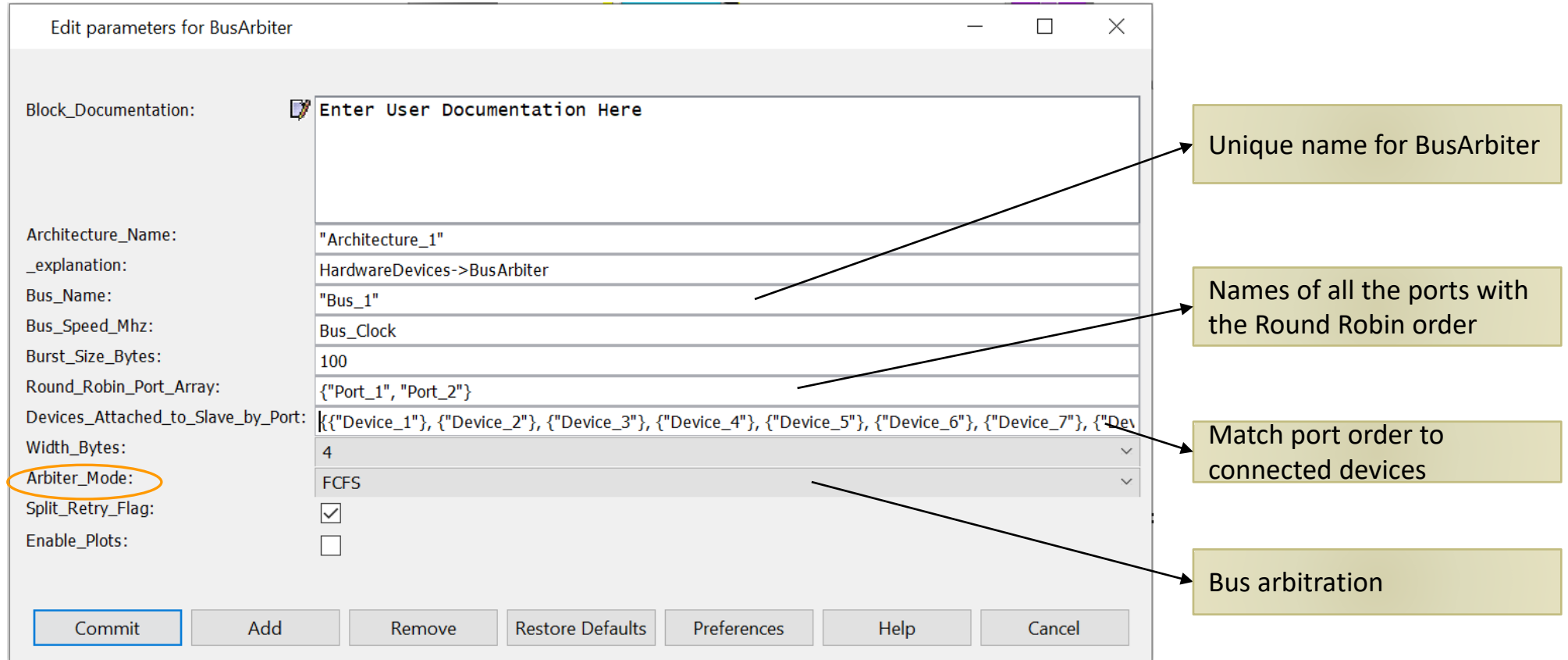
- priority-based
- reordered on the arrival



Read Transaction

- Transaction arrives from master
 - Stored in respective port FIFO buffer.
- Port informs the Bus Arbiter of the transaction
 - When the bus is available, the Controller selects a Port, based on the arbitration.
- When the transaction is accepted for transfer
 - the Data Structure is delayed by one cycle for the Address Control
 - Packet fragmented to match Burst_Length
 - Then the Data Structure is sent out with the delay = $A_Bytes / width / Clock_Speed$
- Return data fragments transferred to the master through the bus
 - Number of cycles depending on the bus width, burst size, bus speed, bytes transferred etc.

Configuration



The screenshot shows the 'Edit parameters for BusArbiter' dialog box. The parameters are as follows:

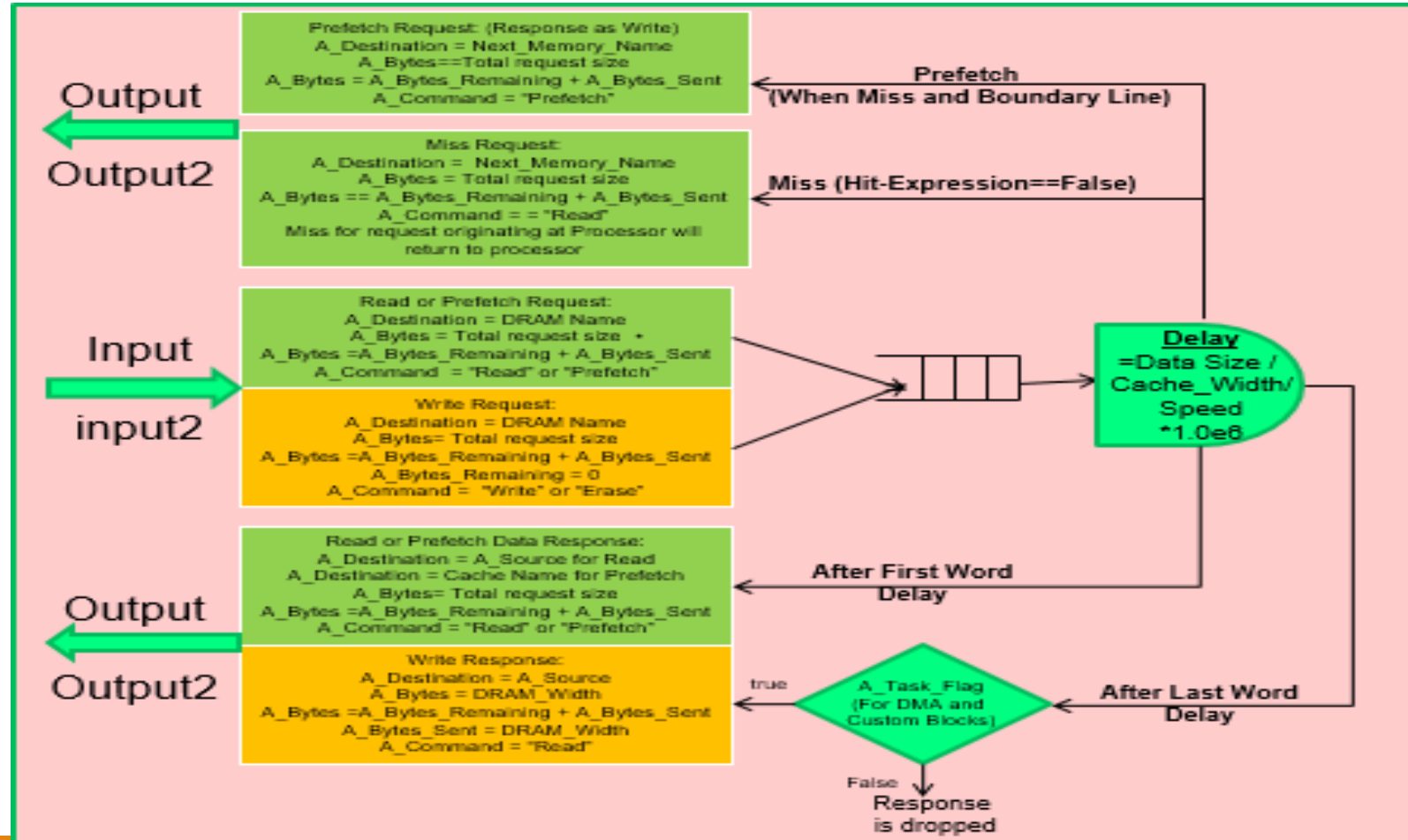
Block_Documentation:	<input type="checkbox"/> Enter User Documentation Here
Architecture_Name:	"Architecture_1"
_explanation:	HardwareDevices->BusArbiter
Bus_Name:	"Bus_1"
Bus_Speed_Mhz:	Bus_Clock
Burst_Size_Bytes:	100
Round_Robin_Port_Array:	{"Port_1", "Port_2"}
Devices_Attached_to_Slave_by_Port:	{{"Device_1"}, {"Device_2"}, {"Device_3"}, {"Device_4"}, {"Device_5"}, {"Device_6"}, {"Device_7"}, {"Dev
Width_Bytes:	4
Arbiter_Mode:	FCFS
Split_Retry_Flag:	<input checked="" type="checkbox"/>
Enable_Plots:	<input type="checkbox"/>

Annotations on the right side of the dialog box:

- Unique name for BusArbiter (points to Architecture_Name)
- Names of all the ports with the Round Robin order (points to Round_Robin_Port_Array)
- Match port order to connected devices (points to Devices_Attached_to_Slave_by_Port)
- Bus arbitration (points to Arbiter_Mode)

Buttons at the bottom: Commit, Add, Remove, Restore Defaults, Preferences, Help, Cancel.

Using Cache Block

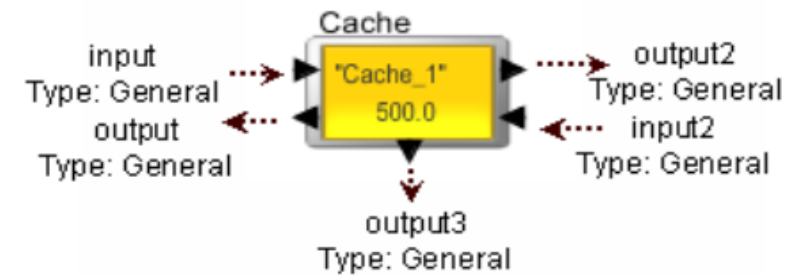


Cache

Emulate a cache in architecture mode

Handles

- Request Queuing
- Cache hit-miss evaluation
- Cache Prefetch, Read/Write
- Cache miss activity to the next level of memory



Operation

1. Queuing

- Receive a request to Read or Write
- Incoming request processed immediately if the Cache is active, else placed in FIFO

2. Catch Hit-miss

- When a new request comes, the cache evaluates the Hit Expression
- A hit occurs
 - If the expression evaluates to a "true"
- If a miss occurs
 - The task sent to the Next_Miss_Memory block

Configuration

Edit parameters for Cache

Block_Documentatio... Enter User Documentation Here

Architecture_Name: "Architecture_1"

Cache_Name: "L2_Cache"

Miss_Memory_Name: "DRAM"

Cache_Speed_Mhz: L2_Cache_Clock

Cache_Size_KBytes: 64.0

Width_Bytes: 4

Words_per_Cache_Line: 16

FIFO_Buffers: 32

Cache_Address: "/* Format: Min_Address,Max_Address. Example:100,200 */"

Cache_Hit_Expression: "rand(0.0,1.0) <= 0.95"

Enable_Hello_Messages:

Commit Add Remove Restore Defaults Preferences Help Cancel

Route the requests to the next level of memory to access when a cache miss occurs or a prefetch is requested

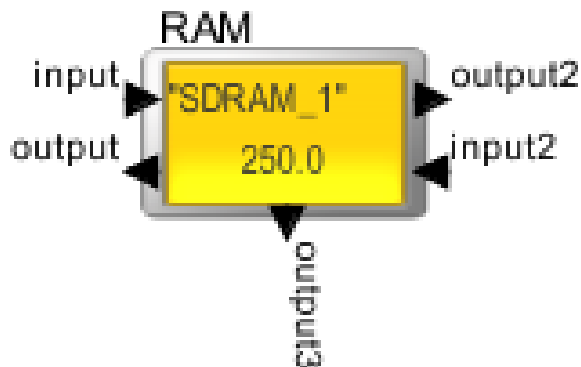
Number of outstanding requests that need to be processed

Expression for the cache hit using RegEx language.

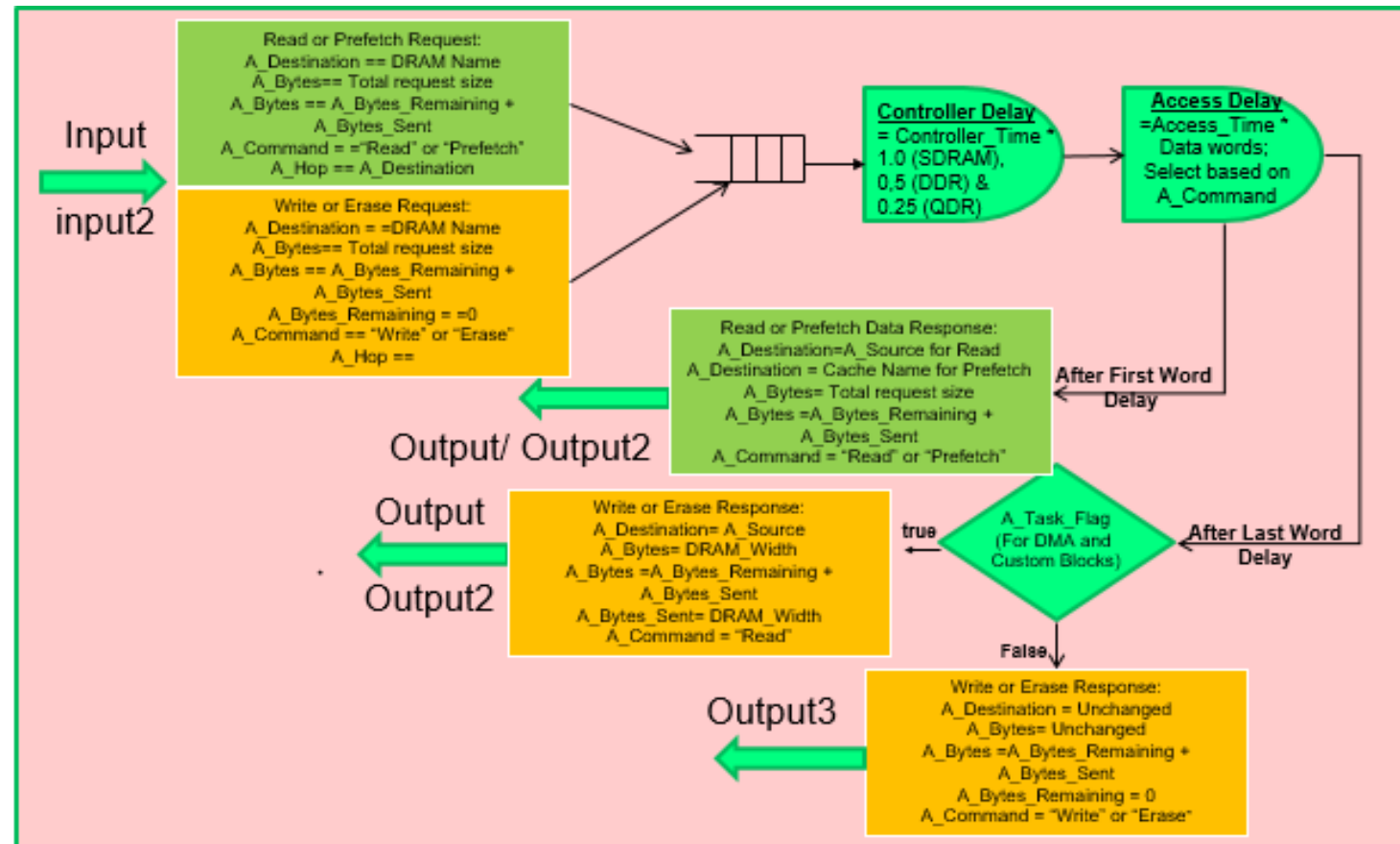
- *true*, then the task had a cache hit
- else a miss occurred.

RAM

- Model different dynamic random access memory technologies
- Executes a memory request, read or write (instruction) and returns the request to the source
- **Memory - > RAM**
- FIFO based scheduling



Using RAM Block



RAM

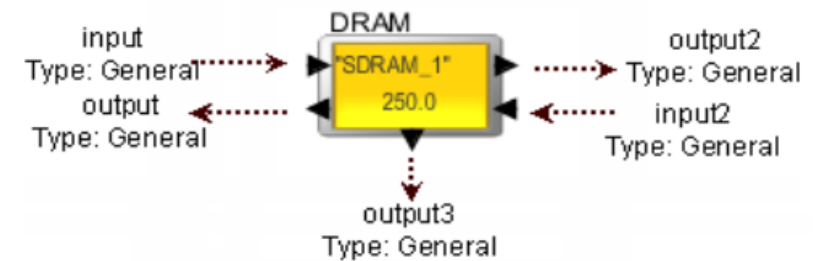
1. Operations of basic memory controller and memory array

2. Handles

- Pre-fetch
- Read
- Write
- Refresh

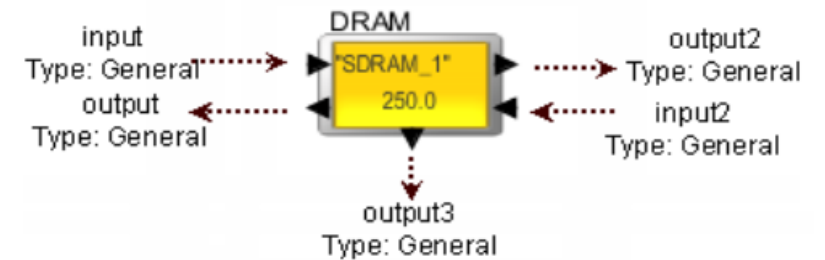
3. Applications

- ROM, RAM, SRAM, DRAM or SDRAM
- DDR, DDR2, DDR3
- SDR, QDR
- VRAM, Direct Rambus, PSRAM, SGRAM
- NAND and NOR flash



Operation

- Receive request for data or instruction
- Requests are queued and processed on Priority
- Operations
 - Read
 - Write
 - Pre-fetch



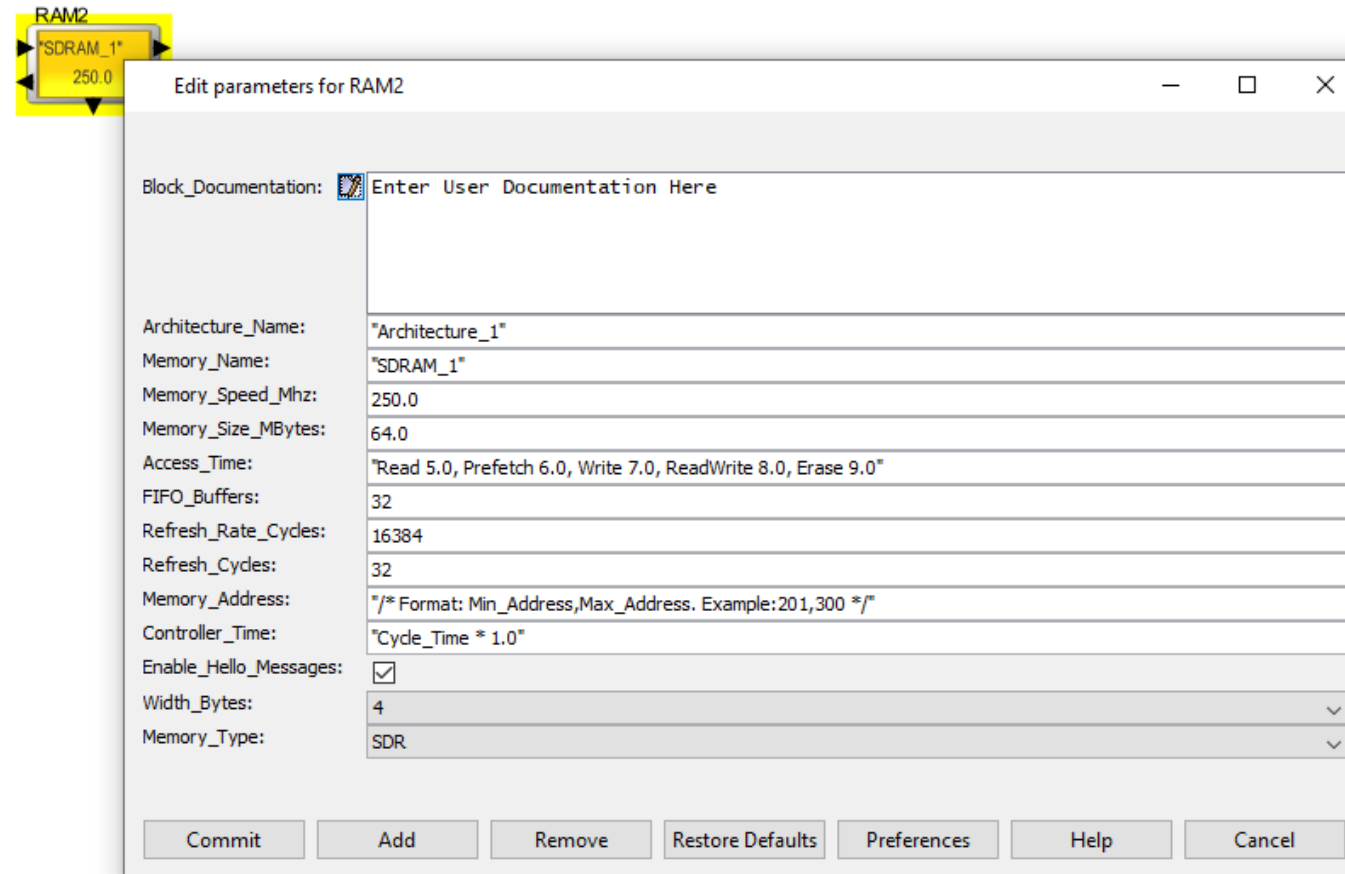
DRAM Features

- Speed
- Size
- Buffer
- Line width
- Memory width
- Access Time - Read, Refresh, Write, Erase, Read/Write,
- Banks
- Refresh
- Controllers - SDR, DDR, DDR-2, DDR3, QDR, RDR, custom

Block Usage

- Synchronous Dynamic (SDRAM)
- Double Data Rate (DDR, DDR-2, DDR-3)
- Quad Data Rate (QDR) SRAM
- Direct Rambus (DRDRAM)
- Video DRAM (VRAM)
- Synchronous Graphics RAM (SGRAM)
- Pseudo Static RAM (PSRAM)
- Disk Drive
- NAND and NOR Flash

Configurations



RAM configurations

Edit parameters for RAM

Block_Documentatio... Enter User Documentation Here

Architecture_Name: "Architecture_1"

Memory_Name: "DRAM"

Memory_Speed_Mhz: DRAM_Clock

Memory_Size_MBytes: 64.0

Access_Time: _Mhz, Write 1000.0/Memory_Speed_Mhz, ReadWrite 1000.0/Memory_Speed_Mhz, Erase 1000.0/Memory_Speed_Mhz"

FIFO_Buffers: 32

Refresh_Rate_Cycles: 16384

Refresh_Cycles: 32

Memory_Address: "/* Format: Min_Address,Max_Address. Example:201,300 */"

Controller_Time: "Cycle_Time * 1.0"

Enable_Hello_Messages:

Width_Bytes: 4

Memory_Type: DDR

Commit Add Remove Restore Defaults Preferences Help Cancel

Number of cycles between refresh.

Duration in cycles of the refresh

How the device know, the RAM is connected?

```

2 OUTPUT AT TIME          ----- 0.0 ps -----
3 {A_Address_Min_Max      = /* Format: Min_Address,Max_Address. Example:2
4 A_Destination           = "Architecture_1",
5 A_Hop                   = "output",
6 A_Instruction            = "Hello",
7 A_Source                 = "DRAM"}

```


Important Concepts

1. Controller Time

- Cycle Time = $1/\text{Memory Speed Mhz}$
- 1.0 for SDR or 1/2 for DDR and quarter for QDR
Example: Cycle_Time * 1.0

2. Access Time

- Access time for Read, Write, Prefetch and Erase is in nanoseconds
Example : Read $1000.0/\text{Memory_Speed_Mhz}$
- Default value
 - Read 5.0
 - Prefetch 6.0
 - Write 7.0
 - ReadWrite 8.0
 - Erase 9.0

Required Fields

- A_Source
- A_Destination
- A_Command
- A_Bytes
- A_Bytes_Remaining
- A_Bytes_Sent
- A_Task_Flag

```
Line          : Incoming Data Structure does not contain one or more of the necessary fields:  
{"A_Destination"}  
Error_Number  : DRAM_001A  
Explanation  : Add the required fields listed.
```

Read Operation

Input

```

DISPLAY AT TIME          ----- 6.00 ns -----
{A_Adr_Ctrl_Flag        = true,
A_Address               = 0L,
A_Address_Max          = 100,
A_Address_Min          = 1,
A_Branch                = false,
A_Bytes                 = 64,
A_Bytes_Remaining      = 60,
A_Bytes_Sent            = 4,
A_Command               = "Read",
A_D_Adr                 = 0L,
A_Data                  = "MyData",
A_Destination           = "DRAM_1",
A_First_Word            = true,
A_Hop                   = "DRAM_1",
A_IDX                   = 0,
A_IDY                   = 0,
A_I_Adr                 = 0L,
A_Instruction           = {"ADD", "ADD"},
A_Instruction_Reorder  = {1, 1, 1, 1, 1, 1, 1, 1},
A_Interrupt             = false,
A_Prefetch              = false,
A_Priority              = 0,
A_Proc_Return          = -1,
A_Protocol_State       = "MyState",
A_Return                = -1,
A_Source                = "Generator",
A_Status                = "Bus_1_Port_2_",
A_Task_Address          = 1,
A_Task_Flag            = true,
A_Task_ID               = 0L,
A_Task_Name             = "Name",
A_Task_Source           = "Src",
A_Time                  = 1.0E-9,
A_Variables             = 16,
BLOCK                   = "Traffic",
DELTA                   = 0.0,
DS_NAME                 = "Processor_DS",
TD                      = 1

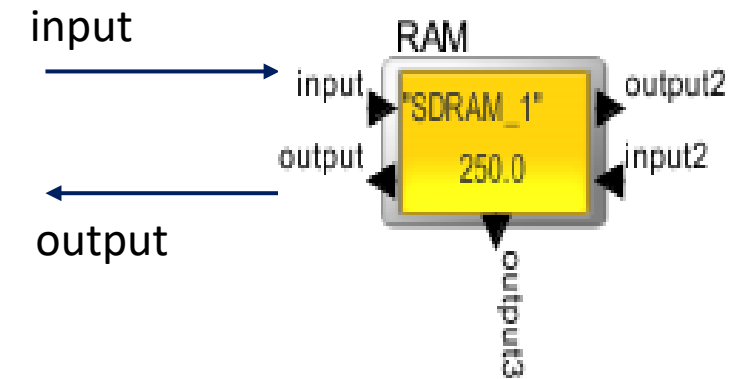
```

Output

```

DISPLAY AT TIME          ----- 26.00 ns -----
{A_Adr_Ctrl_Flag        = true,
A_Address               = 0L,
A_Address_Max          = 100,
A_Address_Min          = 1,
A_Branch                = false,
A_Bytes                 = 64,
A_Bytes_Remaining      = 60,
A_Bytes_Sent            = 4,
A_Command               = "Write",
A_D_Adr                 = 0L,
A_Data                  = "MyData",
A_Destination           = "Generator",
A_First_Word            = true,
A_Hop                   = "Bus_1_Port_2_",
A_IDX                   = 0,
A_IDY                   = 0,
A_I_Adr                 = 0L,
A_Instruction           = {"ADD", "ADD"},
A_Instruction_Reorder  = {1, 1, 1, 1, 1, 1, 1, 1},
A_Interrupt             = false,
A_Prefetch              = false,
A_Priority              = 0,
A_Proc_Return          = -1,
A_Protocol_State       = "MyState",
A_Return                = -1,
A_Source                = "DRAM_1",
A_Status                = "Bus_1_Port_2_",
A_Task_Address          = 1,
A_Task_Flag            = true,
A_Task_ID               = 0L,
A_Task_Name             = "Name",
A_Task_Source           = "Src",
A_Time                  = 1.0E-9,
A_Variables             = 16,
BLOCK                   = "Traffic",
DELTA                   = 0.0,
DS_NAME                 = "Processor_DS",
ID                      = 1,

```



Write Operation

i) Input

```

DISPLAY AT TIME          ----- 1.00 ns ---
{A_Adr_Ctrl_Flag        = true,
A_Address               = 0L,
A_Address_Max           = 100,
A_Address_Min           = 1,
A_Branch                = false,
A_Bytes                 = 256,
A_Bytes_Remaining       = 0,
A_Bytes_Sent            = 256,
A_Command               = "Write",
A_D_Addr                = 0L,
A_Data                  = "MyData",
A_Destination           = "DRAM_1",
A_First_Word            = true,
A_Hop                   = "DRAM_1",
A_IDX                   = 0,
A_IDY                   = 0,
A_I_Addr                = 0L,
A_Instruction           = {"ADD", "ADD"},
A_Interrupt             = false,
A_Prefetch              = false,
A_Priority               = 1,
A_Proc_Return           = -1,
A_Protocol_State        = "MyState",
A_Return                = -1,
A_Source                 = "Generator",
A_Status                = "Status",
A_Task_Address          = 1,
A_Task_Flag             = true,
A_Task_ID               = 0L,
A_Task_Name             = "Name",
A_Task_Source           = "Src",
A_Time                  = 0.0,
A_Variables             = 16,
BLOCK                   = "Traffic",
DELTA                   = 0.0,
DS_NAME                 = "Processor_DS",
ID                       = 1,

```

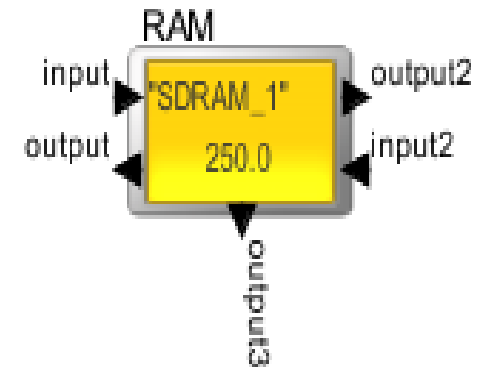
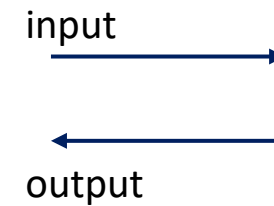
Output

```

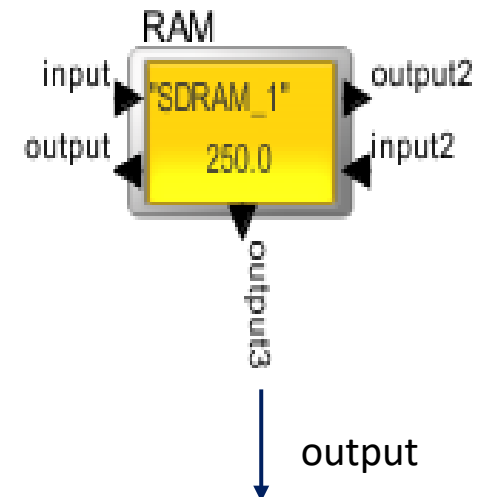
DISPLAY AT TIME          ----- 651.00 ns ---
{A_Adr_Ctrl_Flag        = true,
A_Address               = 0L,
A_Address_Max           = 100,
A_Address_Min           = 1,
A_Branch                = false,
A_Bytes                 = 256,
A_Bytes_Remaining       = 0,
A_Bytes_Sent            = 256,
A_Command               = "Read",
A_D_Addr                = 0L,
A_Data                  = "MyData",
A_Destination           = "Generator",
A_First_Word            = true,
A_Hop                   = "Generator",
A_IDX                   = 0,
A_IDY                   = 0,
A_I_Addr                = 0L,
A_Instruction           = {"ADD", "ADD"},
A_Interrupt             = false,
A_Prefetch              = false,
A_Priority               = 1,
A_Proc_Return           = -1,
A_Protocol_State        = "MyState",
A_Return                = -1,
A_Source                 = "DRAM_1",
A_Status                = "Status",
A_Task_Address          = 1,
A_Task_Flag             = true,
A_Task_ID               = 0L,
A_Task_Name             = "Name",
A_Task_Source           = "Src",
A_Time                  = 0.0,
A_Variables             = 16,
BLOCK                   = "Traffic",
DELTA                   = 0.0,
DS_NAME                 = "Processor_DS",
ID                       = 1,

```

i) A_Task_Flag= True



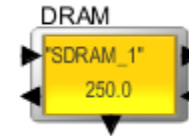
ii) A_Task_Flag= False



Notes on RAM Block

Read Request

- ✓ Accepts one request
- ✓ Returns the first word based on RAM Word width
- ✓ Delays internally for the remaining words
- ✓ Generates Controller time for the first word only
- ✓ Access_Time is based on A_Bytes



Write Data

- ✓ Accepts each word/burst
- ✓ Controller time for the first word/burst only
- ✓ Access delay is based on the A_Bytes_Sent field
- ✓ Standard output is to output3 at the end of all words
- ✓ If A_Task_Flag == true, then sends to the Bus after the access time for the last word

Latency Computation & Statistics

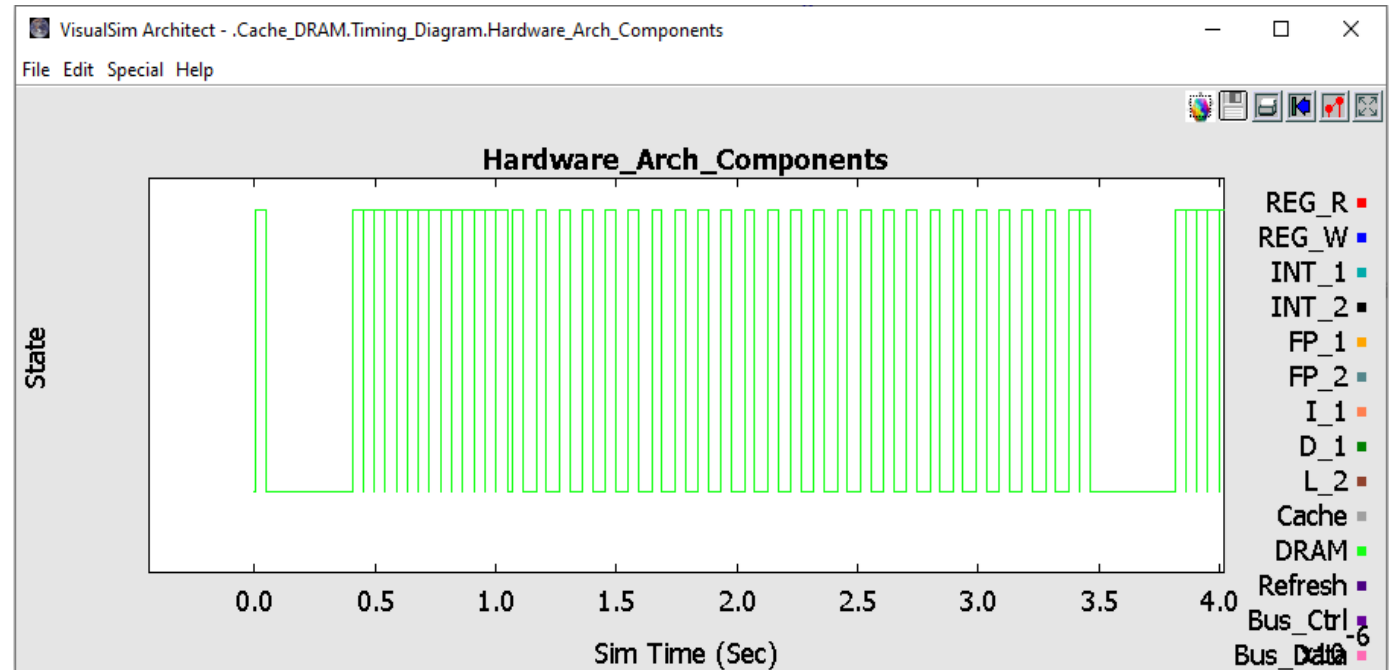
Generate using Architecture Setup block

- **Latency = (Number of words + Access cycles – 1) * Memory cycle time** where,
 - ✓ Number of Words = Bytes sent/ Width Bytes
 - ✓ Memory Cycle time = $1.0e-6$ / Memory speed in Mhz

```
DRAM_Delay_Time_Max      = 3.8E-8,  
DRAM_Delay_Time_Mean     = 2.3E-8,  
DRAM_Delay_Time_Min      = 8.0E-9,  
DRAM_Delay_Time_StDev    = 1.5E-8,  
DRAM_Memory_Used_By_Processor_1_MB_Max = 0.003328,  
DRAM_Memory_Used_By_Processor_1_MB_Mean = 0.003008,  
DRAM_Memory_Used_By_Processor_1_MB_Min = 0.00288,  
DRAM_Memory_Used_By_Processor_1_MB_StDev = 1.3424753256579E-4,  
DRAM_Memory_Used_By_Total_MB_Max = 0.003328,  
DRAM_Memory_Used_By_Total_MB_Mean = 0.003008,  
DRAM_Memory_Used_By_Total_MB_Min = 0.00288,  
DRAM_Memory_Used_By_Total_MB_StDev = 1.3424753256579E-4,  
DRAM_Throughput_MBs_Max = 832.0,  
DRAM_Throughput_MBs_Mean = 752.0,  
DRAM_Throughput_MBs_Min = 720.0,  
DRAM_Throughput_MBs_StDev = 33.5618831414437,
```

Timing Diagrams

- Use Timing Diagram Block
Hardware setup -> Timing Diagram



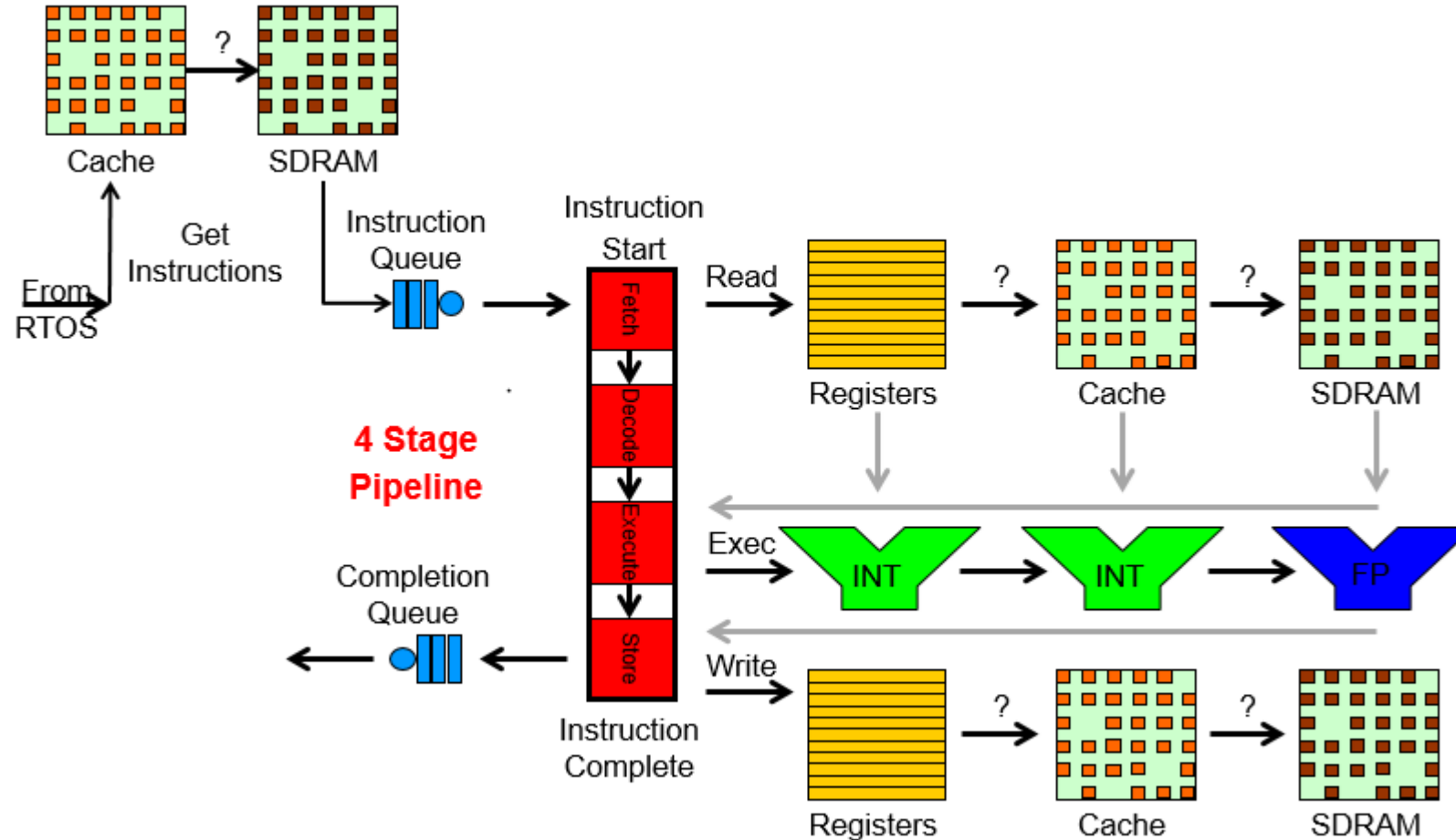
Learn More by Reviewing Training Recordings

Watch Tutorials

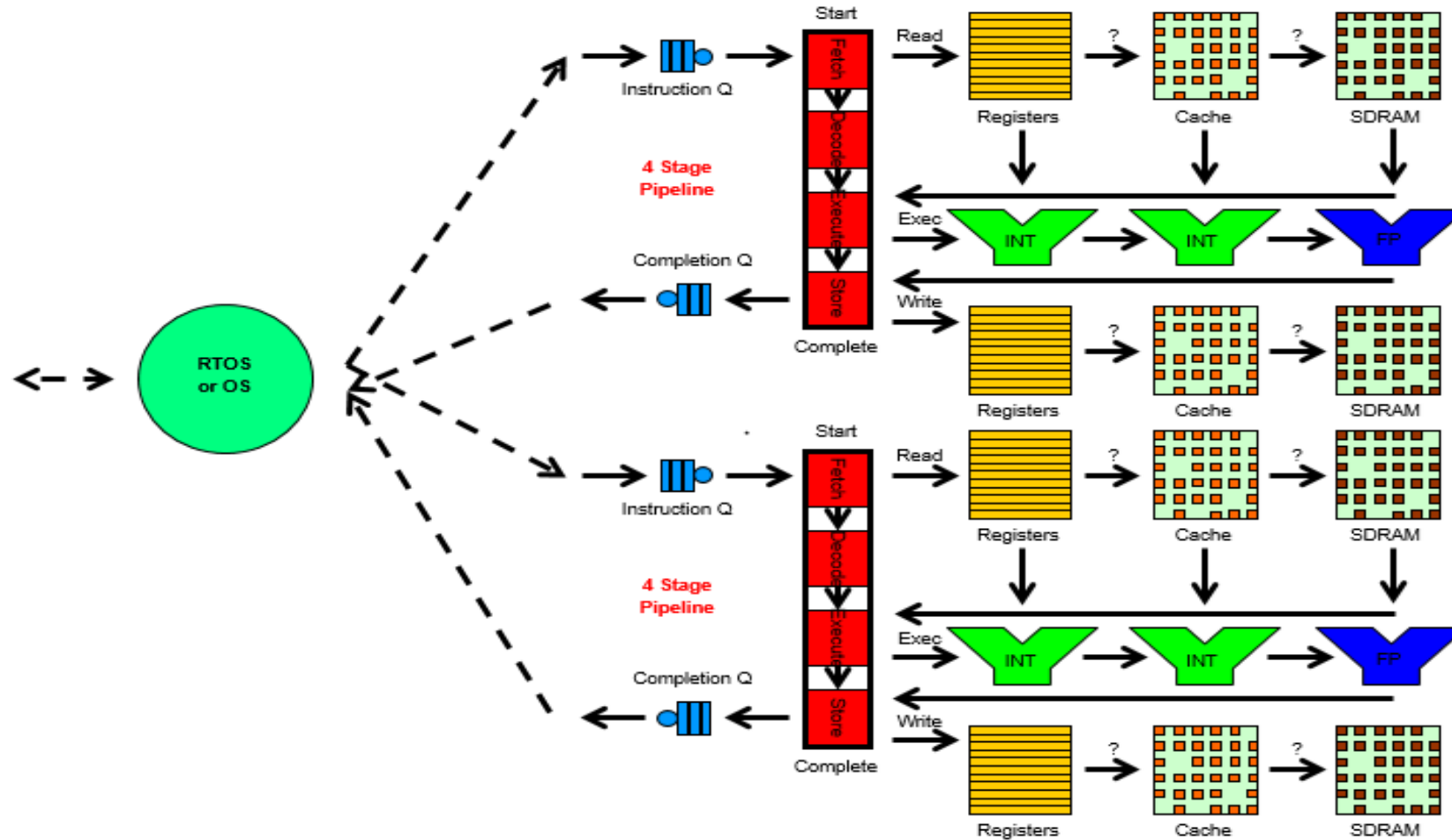
- Training Part 1 (54 minutes):
<https://www.youtube.com/watch?v=9JHcLm0w2-4>
- Training Part 2 (65 minutes):
<https://www.youtube.com/watch?v=LY-imqaSBwc>
- Training Part 3 (42 minutes):
<https://www.youtube.com/watch?v=3H7YaZ0wrwg>

Processor Modeling

Processor Model



Multi-Processor Model



Instruction_Set

- Divided according to their Execution Unit
- Names must be INT_x (Integer) and FP_x (Floating) where x is a number
- All required instructions must be in the Instruction_Set, distributed across the Execution Unit list
- Execution Units are either Integer or Floating. Branch is Integer while Vector and Graphics are Floating-Point
- Each line must have the Instruction Name, Minimum cycles, Maximum Cycles (Optional) and ends with ;
- Minimum and Maximum cycles are used for those instruction with variable cycles

Processor

Block_Documentation: Enter User Documentation Here

Architecture_Name: Architecture_Setup_Name

Processor_Name: Processor_Name **Unique Name**

Processor_Setup: /* First row contains Column Names. */

Parameter_Name	Parameter_Value
Processor_Instruction_Set:	ARM_INSTR
Number_of_Registers:	32
Processor_Speed_Mhz:	ClockRate
Context_Switch_Cycles:	4 /* switch between threads */
Instruction_Queue_Length:	128
Instructions_per_Cycle:	2
Number_of_Pipeline_Stages:	10
Number_of_INT_Execution_Units:	5
Number_of_FP_Execution_Units:	1
ROB_Size:	40 /* Re order Buffer size */
Number_of_Cache_Execution_Units:	2 /* I-cache and D-cache */
External_I_Cache:	{}
External_D_Cache:	{Outstanding_Req_Count=2}

Pipeline_Stages: /* First row contains Column Names. */

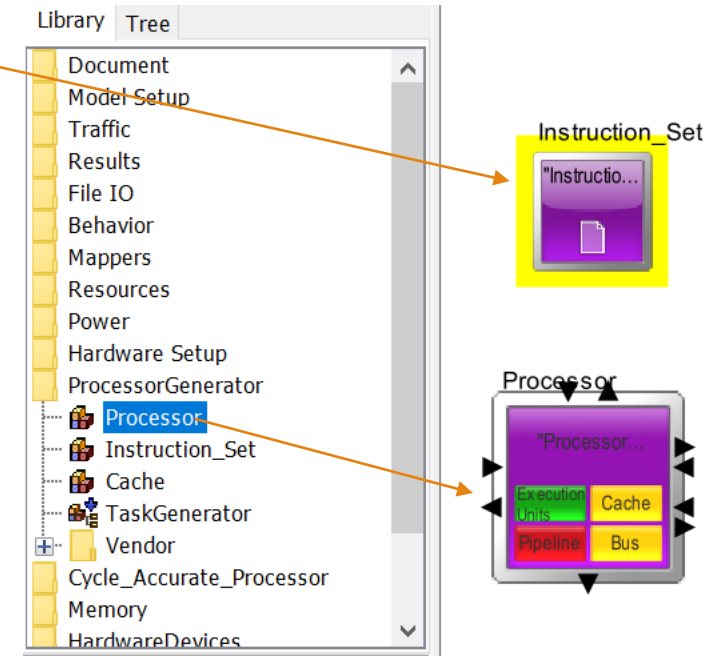
Stage_Name	Execution_Location	Action	Condition
1_INSTRFETCH	I_Cache	instr	none
2_INSTRFETCH	I_Cache	wait	none
3_INSTRFETCH	D_Cache	read	none
4_DECODE	none	exec	none
4_RENAME	D_Cache	wait	none
5_DISPATCH	none	issue	3
6_ROBL	none	exec	none
6_IW	none	exec	none
7_II	none	exec	none
8_EXECUTE	ARM	exec	none
9_EXECUTE	ARM	wait	none
10_STORE	D_Cache	write	none

Enable_Hello_Messages:

Processor_Bits: 64

Required

Key Parameters to modify



Pipeline

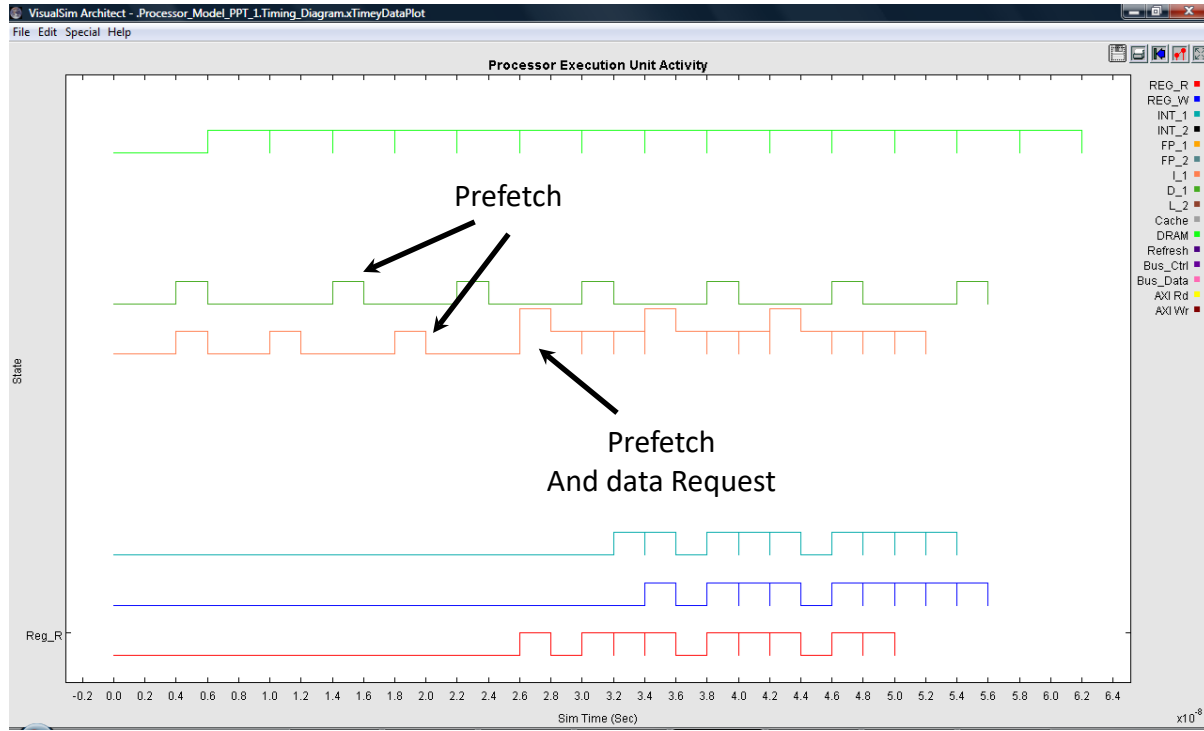
- Match the pipeline specified in the datasheet
- Can handle address generation (one cycle delay), instruction fetch, instruction decode (wait for the instruction return and one cycle delay), data fetch, execute, write back and unused stages (for a delay or adding empty pipeline stages and
- Four columns
- Name is *_AnyName- The numbers must be sequential and the last number must match the parameter- Number_of_Pipeline_Stages
- If using Actions- read, write and wait, Execution_Location is required
- For Action- exec, it can be none (if a cycle delay is intended), Execution Unit of this processor, another processor or a SystemResource_Extend/ SystemResource
- Condition is currently not used

Examples of Pipeline usage

```

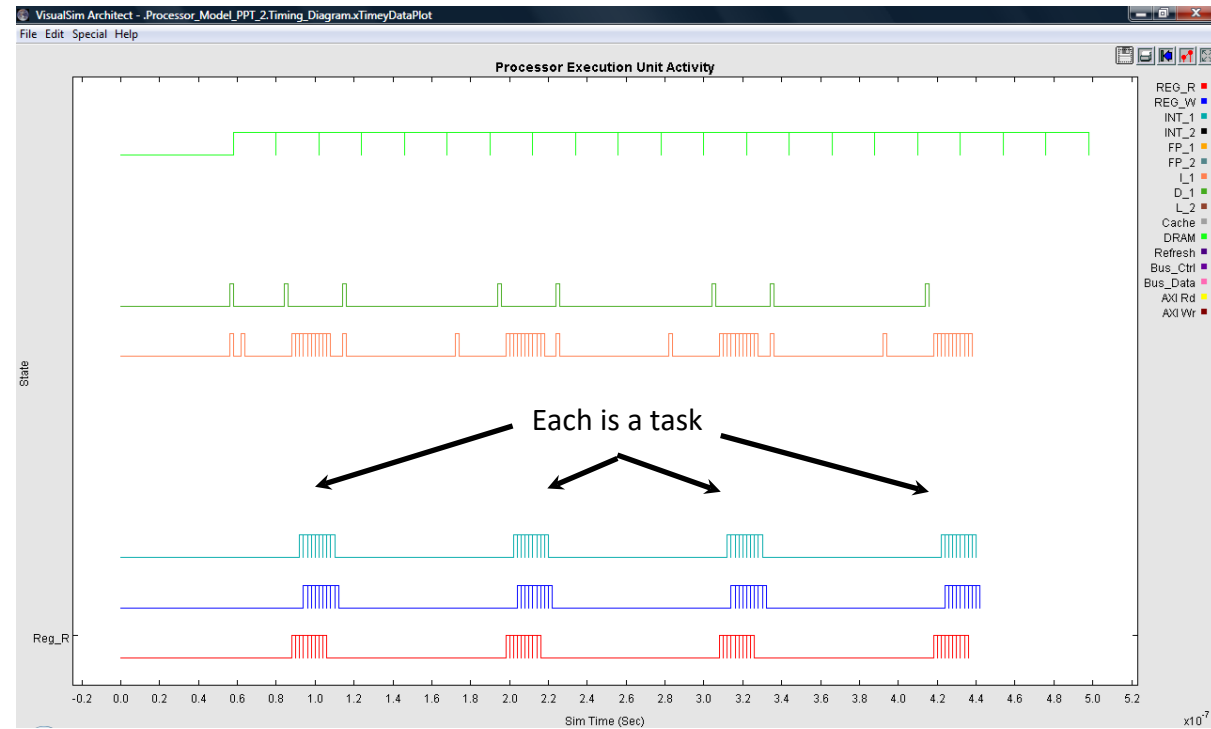
/ First row contains Column Names. /
Stage_Name      Execution_Location  Action      Condition ;
1_INSTRFETCH    I_Cache_Name        instr       none ;
2_INSTRFETCH    I_Cache_Name        wait        none ;
3_INSTRFETCH    D_Cache_Name        read        none ;
4_DECODE        none                 exec        none ;
4_RENAME        D_Cache_Name        wait        none ;
5_DISPATCH      none                 issue       3 ;
6_ROBL          none                 exec        none ;
// Re Order Buffer Loading
6_IW            none                 exec        none ;
// Instruction Window
7_II            none                 exec        none ;
// Instruction Issue
8_EXECUTE       ARM                  exec        none ;
9_EXECUTE       ARM                  wait        none ;
10_STORE        D_Cache_Name        write       none ;

```



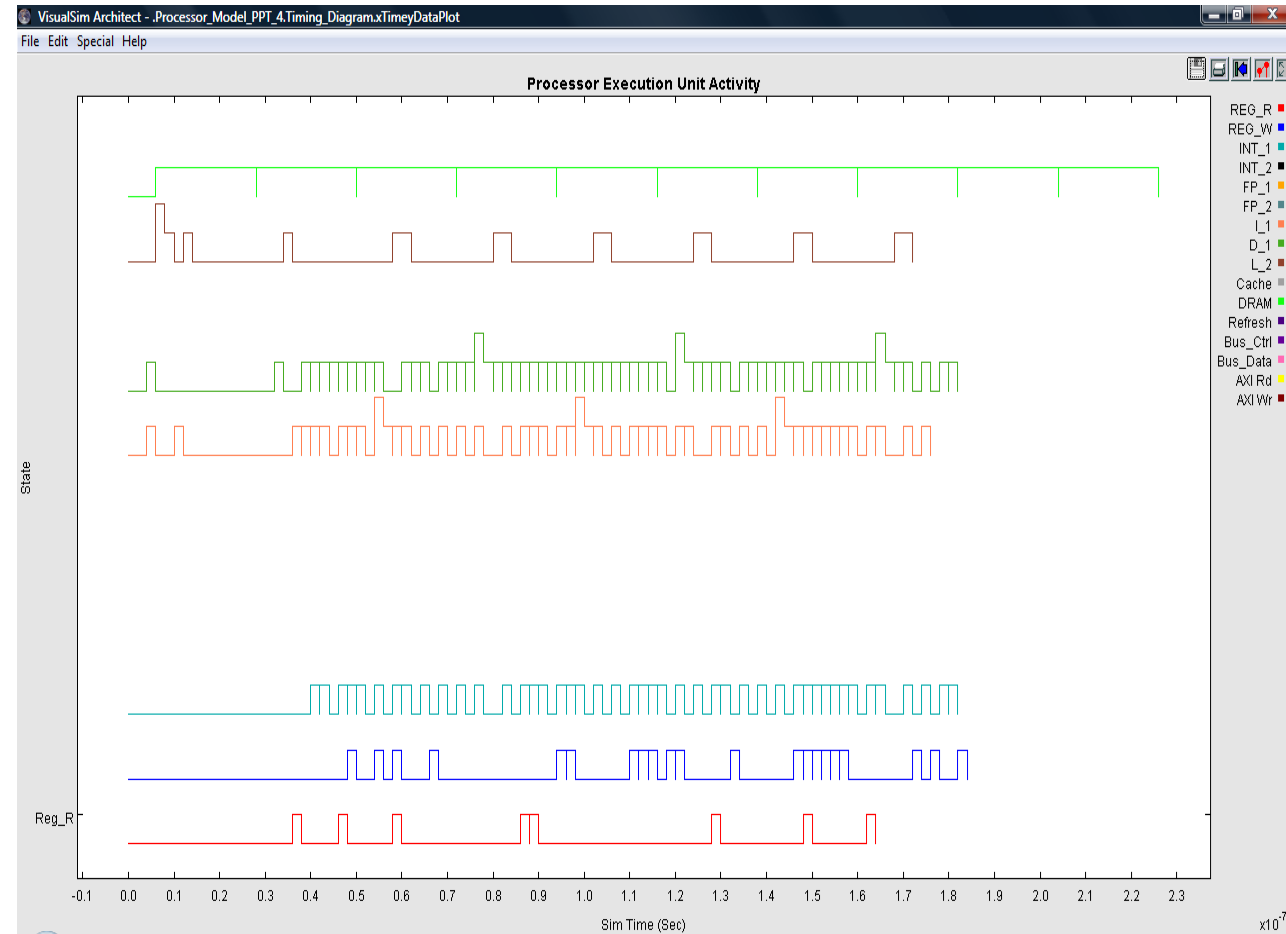
Direct DRAM
I_1, D_1 to DRAM
One INT_1
No Bus

Single Task with 9 ADD instructions
1 Word per cache line
1.0 hit-ratio for Register



Direct DRAM
I_1, D_1 to DRAM
One INT_1
No Bus

Four Tasks with 9 ADD each
10 Words per cache line
A_Variables =
Number_of_Registers



Direct DRAM
I_1, D_1 to DRAM
One INT_1
No Bus

Single Task with 9 ADD instructions
10 Words per cache line
0.25 hit-ratio for registers

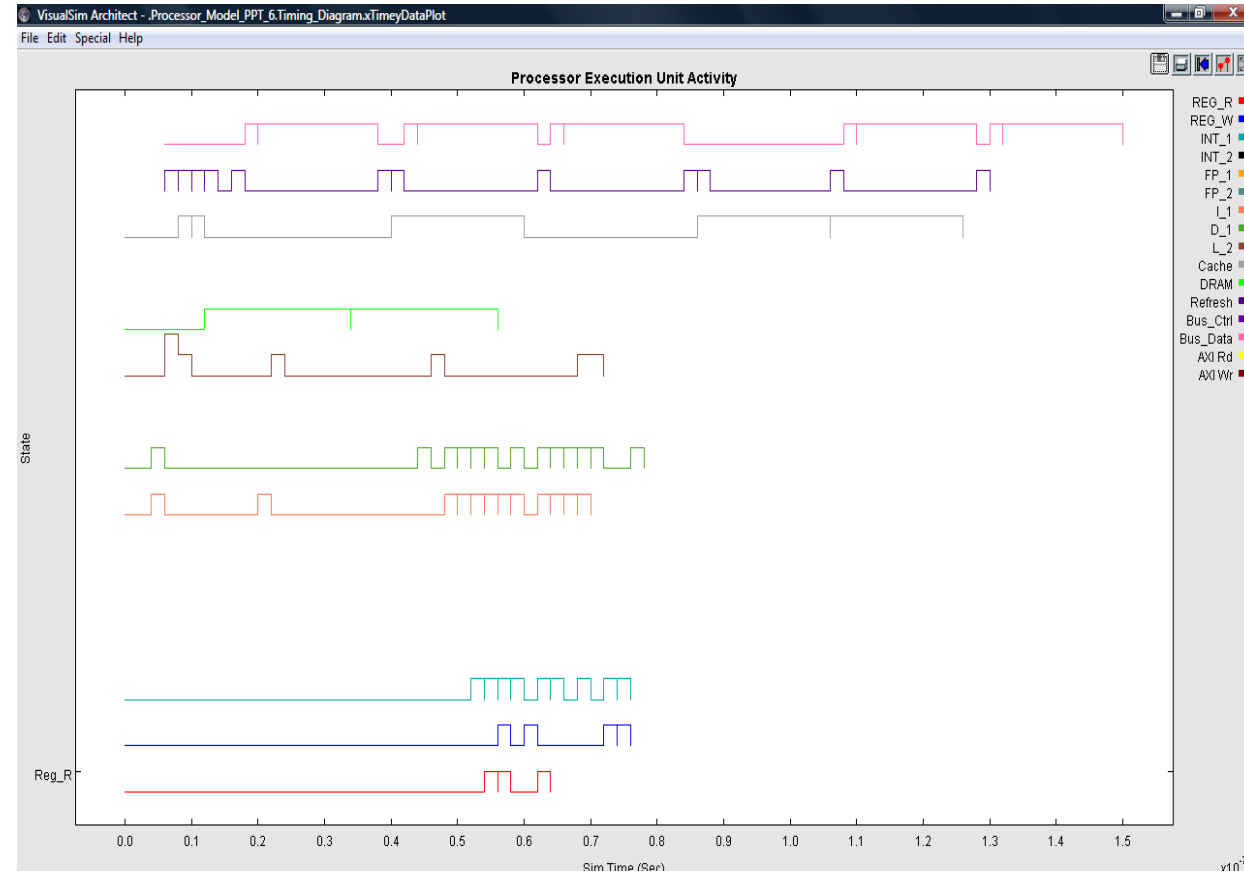
Direct DRAM
I_1 and D_1 to L2; L2 to
DRAM
One INT_1
No Bus

Single Task with 45 ADD
instructions
10 Words per cache line
0.25 Hit-Ratio for Registers

VS_AR/doc/Doc_Support/Processor_Model_PPT_5.xml



VS_AR/doc/Doc_Support/Processor_Model_PPT_6.xml

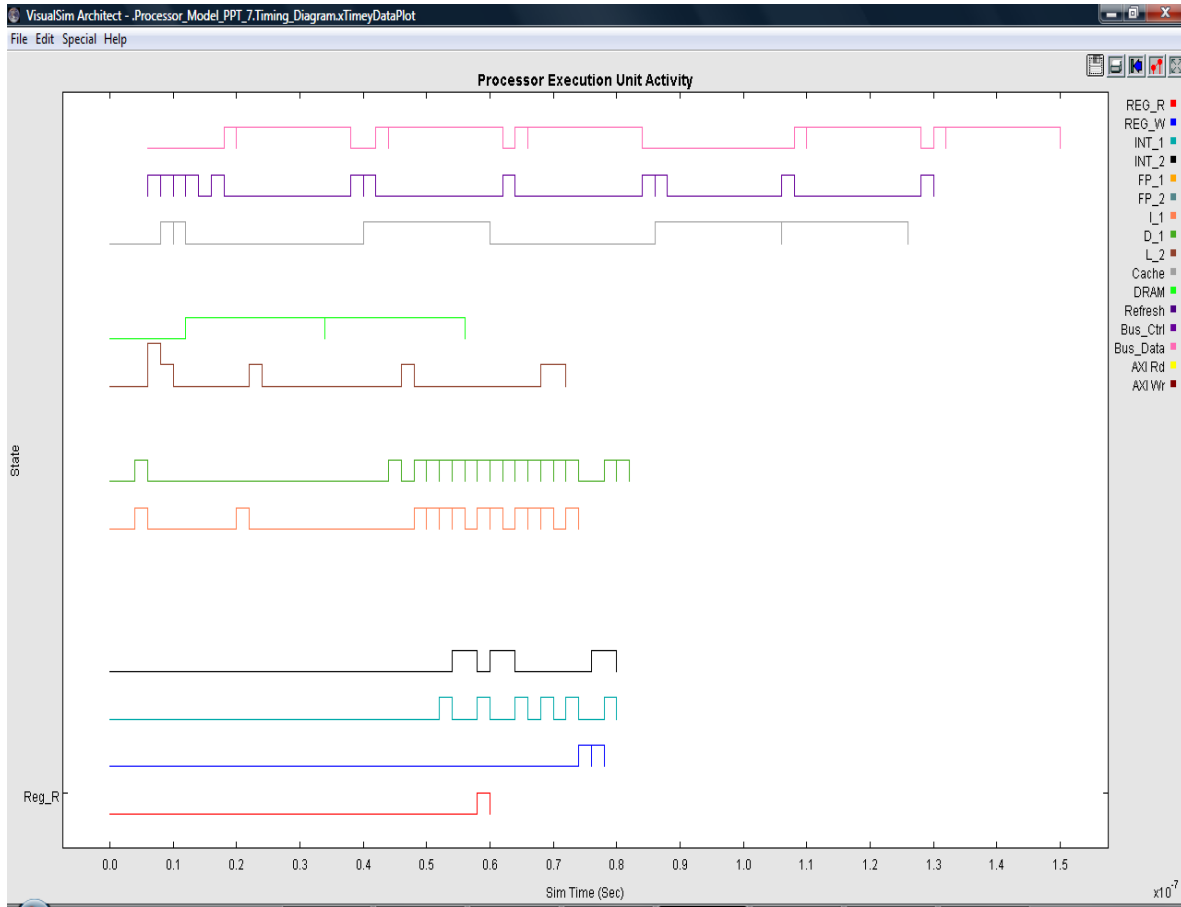


Direct DRAM
 I_1 and D_1 to L2; L2 to DRAM
 One INT_1
 Bus to DRAM

Single Task with 9 ADD Instructions
 I_1 and D_1 to L2; L2 to Cache
 10 Words per cache line
 0.25 hit-ratio for registers
 Cache to DRAM
 One INT_1
 Bus to DRAM

Single Task with 9 ADD instruction
 10 Words per cache line
 0.25 hit-ratio for registers

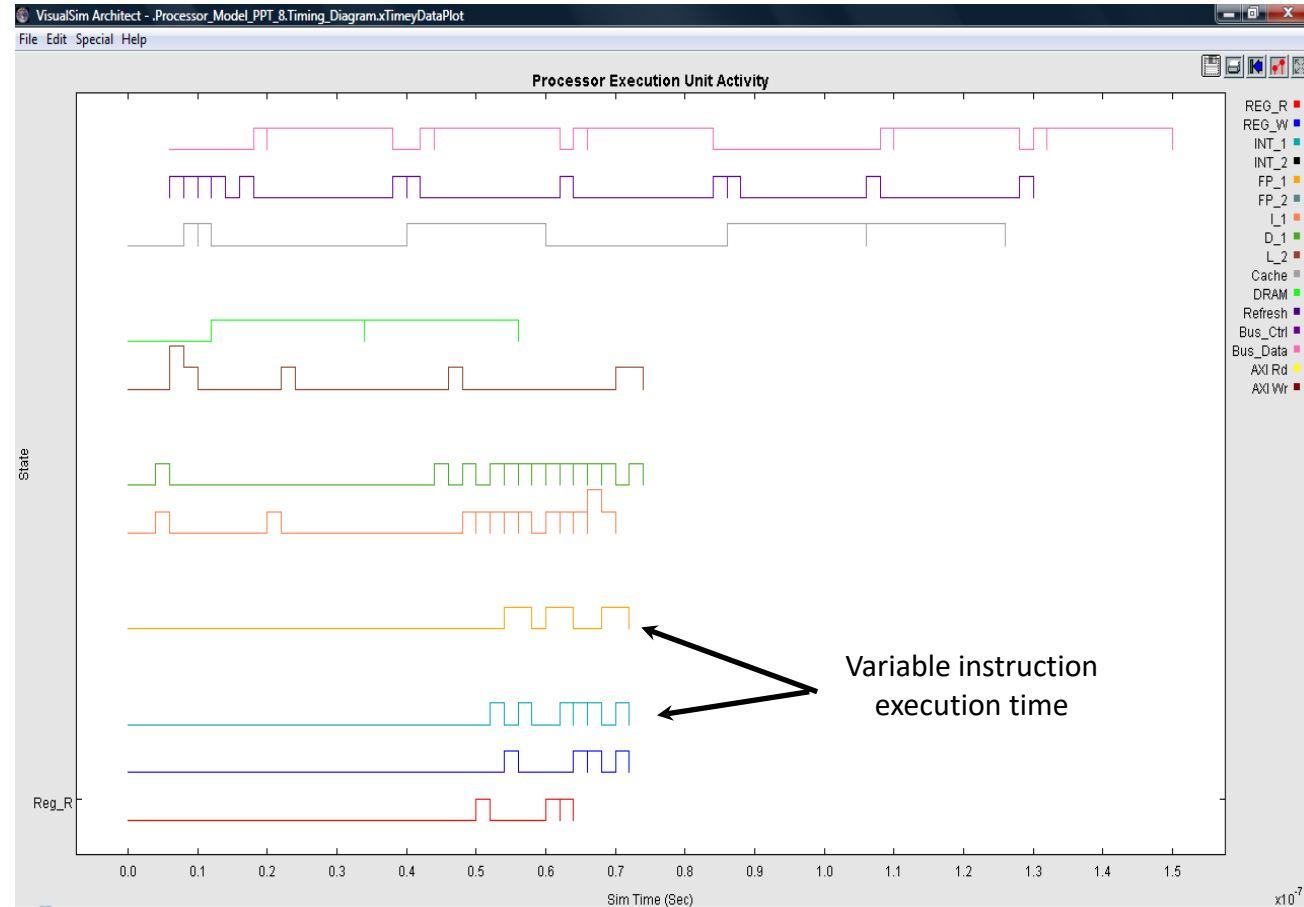
VS_AR/doc/Doc_Support/Processor_Model_PPT_7.xml



I_1 and D_1 to L2;
L2 to Cache; Cache to DRAM
One INT_1, One INT_2
Bus to DRAM

Single Task with alternate ADD and ADDN
10 Words per cache line
 $A_Variables = \text{Number_of_Registers} * 4$

VS_AR/doc/Doc_Support/Processor_Model_PPT_8.xml

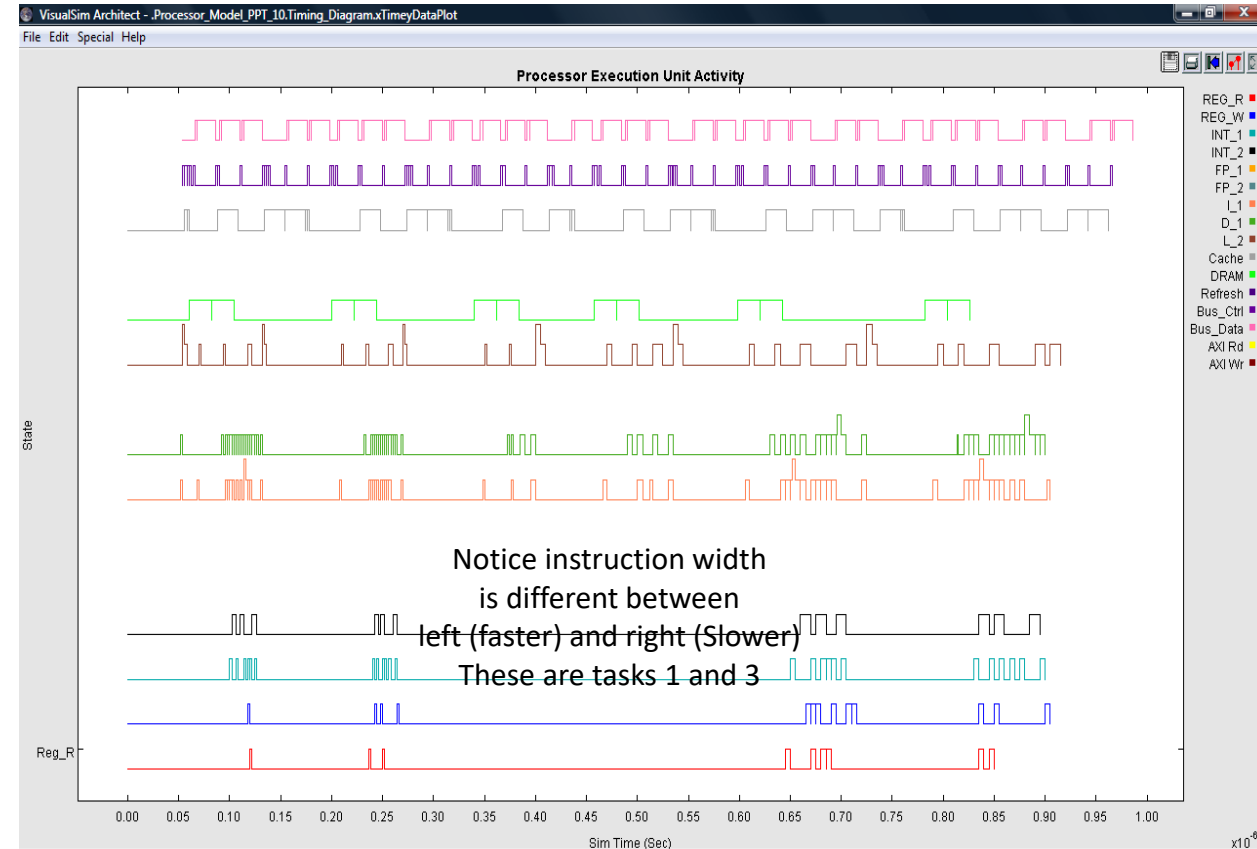
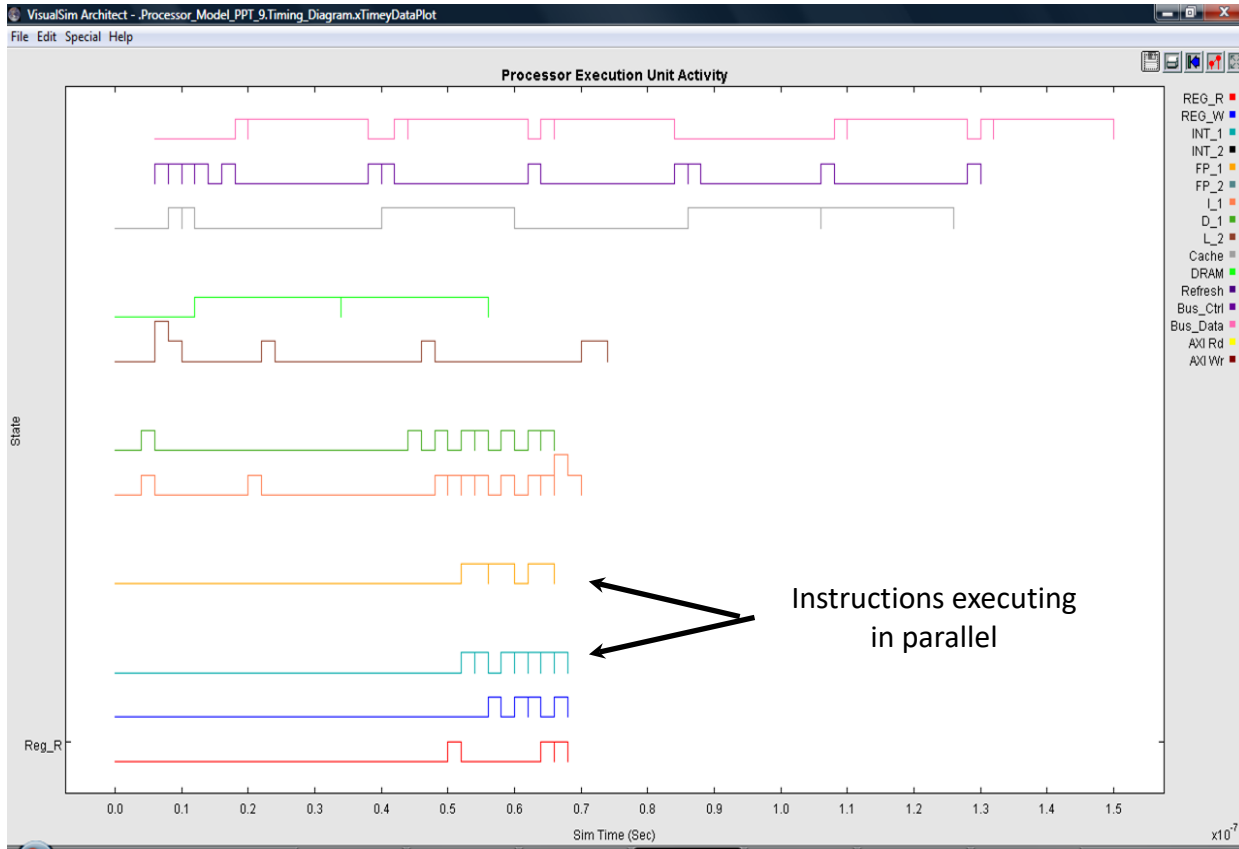


I_1 and D_1 L2; L2 to Cache; Cache
to DRAM
One INT_1, One FP_1
Bus to DRAM

Single Task with 9 ADD and ADDN
Instr
10 Words per cache line
0.25 hit-ratio for Register

VS_AR/doc/Doc_Support/Processor_Model_PPT_9.xml

VS_AR/doc/Doc_Support/Processor_Model_PPT_10.xml



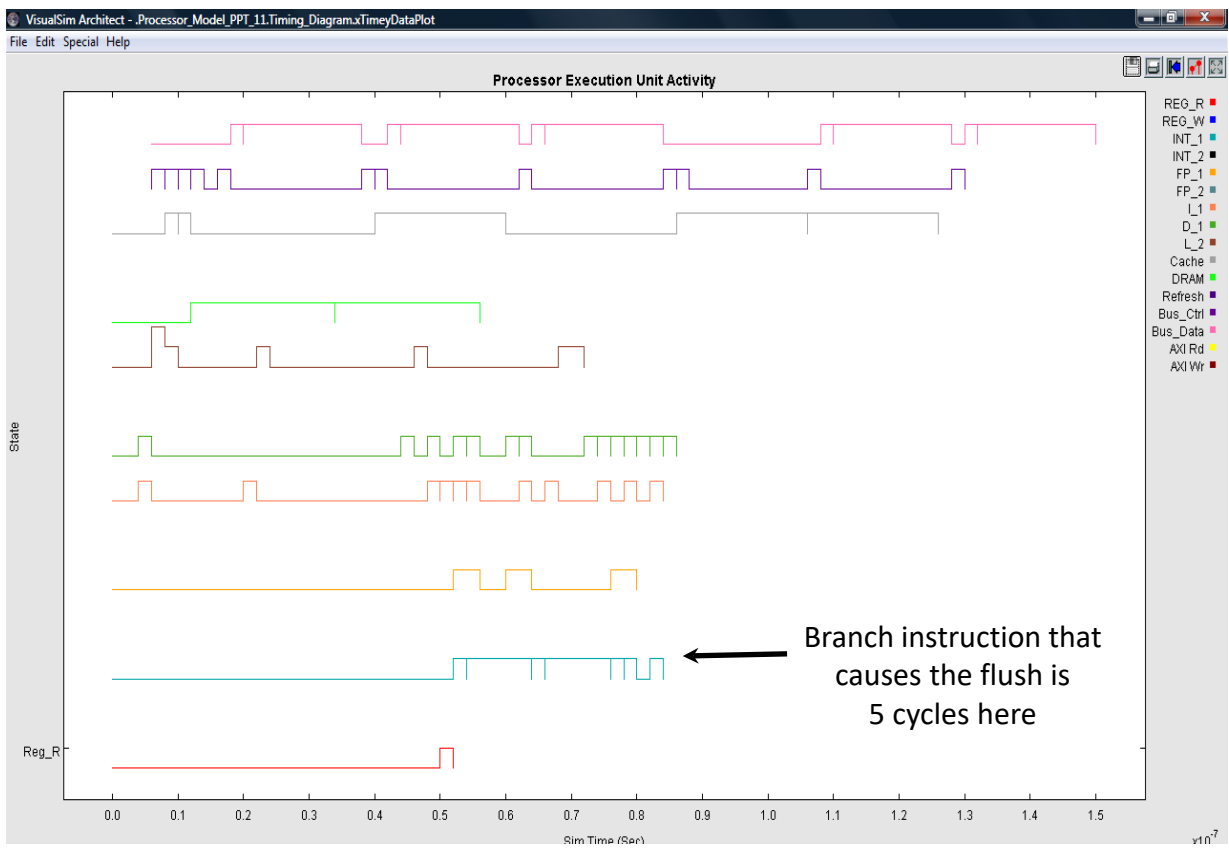
I_1 and D_1 L2; L2 to Cache; Cache to DRAM
 One INT_1, One INT_2
 Bus to DRAM
 10 Words per cache line

Single Task with 9 ADD and ADDN
 0.25 Hit-ratio for registers
 Multiple instructions per cycle

I_1 and D_1 L2; L2 to Cache;
 Cache to DRAM
 One INT_1, One INT_1
 Bus to DRAM
 10 Words per cache line

Three tasks
 Clock change in task 2 from 500 to 200 Mhz
 0.25 Hit-ratio for Registers

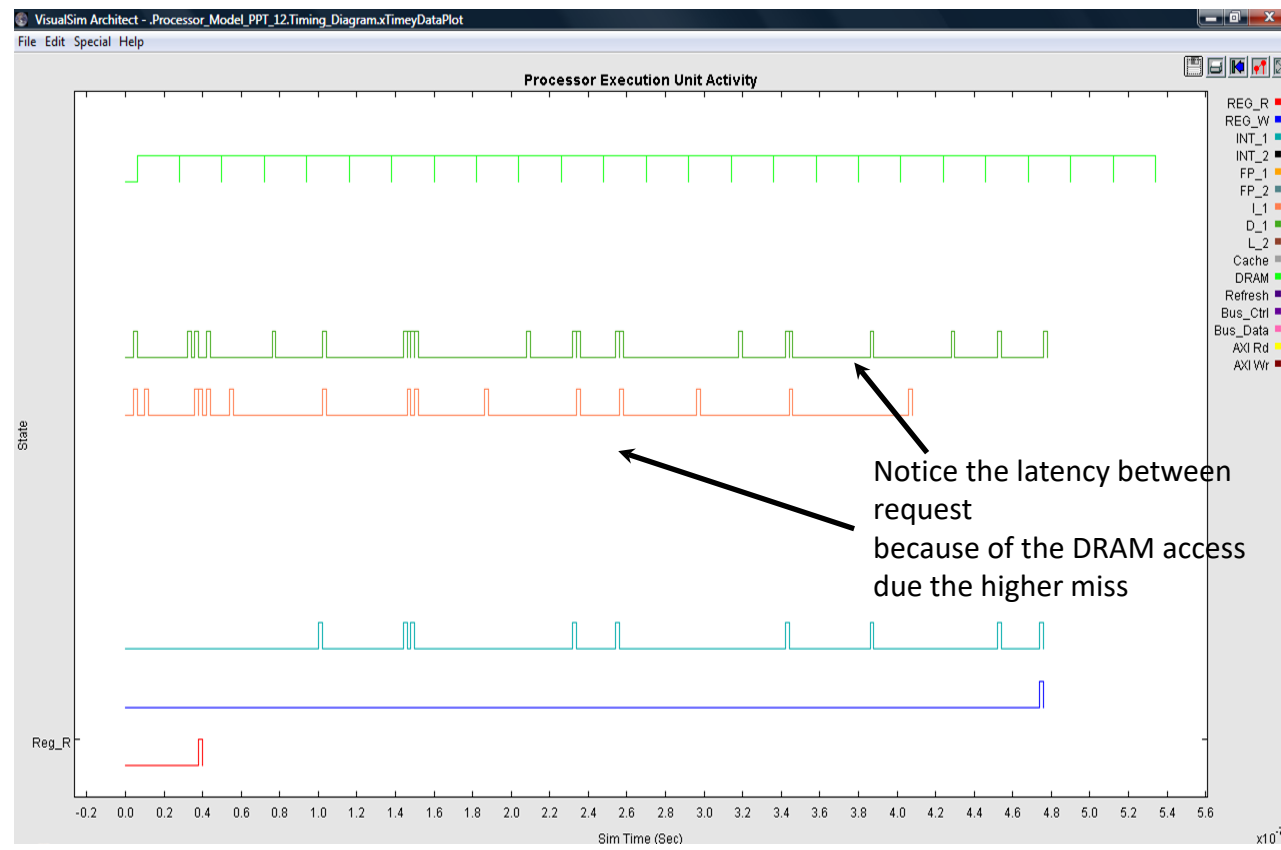
VS_AR/doc/Doc_Support/Processor_Model_PPT_11.xml



I_1 and D_1 L2; L2 to Cache; Cache to DRAM
 One INT_1, One FP_1
 Bus to DRAM
 Single Task; Pipeline flush with a ADD, ADDN and *b

10 Words per cache line
 A_Variables =
 Number_of_Registers * 4
 Multi-instruction per cycle

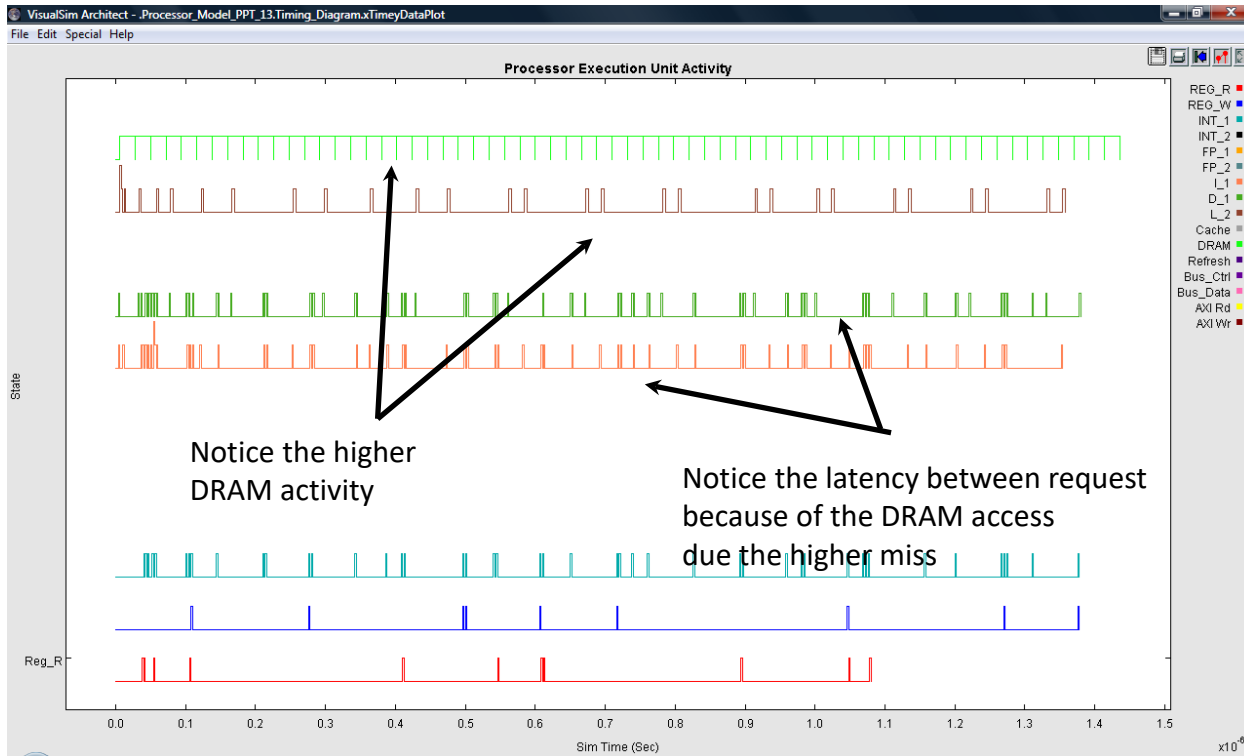
VS_AR/doc/Doc_Support/Processor_Model_PPT_12.xml



Direct DRAM
 I_1, D_1 to DRAM
 One INT_1

Single Task with 9 ADD instructions
 10 Words per cache line
 0.25 Hit-ratio for Registers
 Added Hit-Ratio parameter to both I_1 and D_1

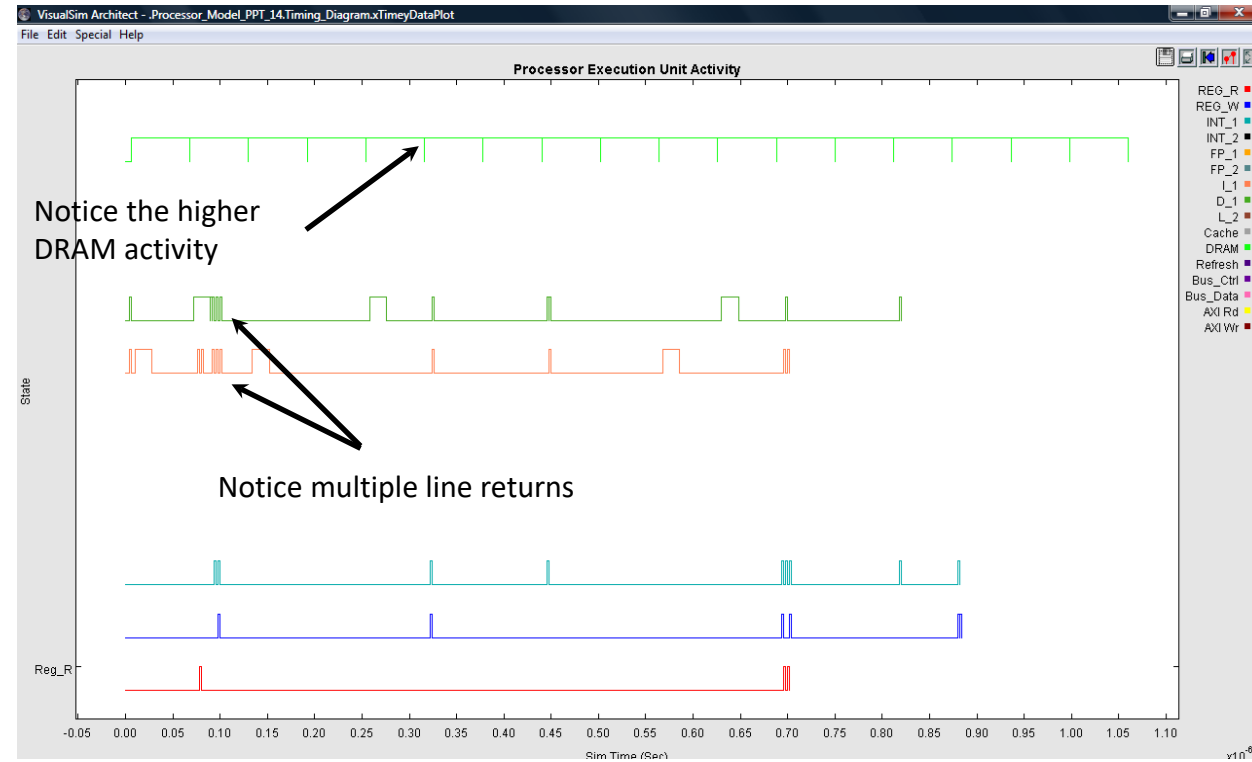
VS_AR/doc/Doc_Support/Processor_Model_PPT_13.xml



Direct DRAM
I_1, D_1 to DRAM
One INT_1

Single Task with 9 ADD instructions
10 Words per cache line
0.25 Hit-ratio for Registers
Added Hit-Ratio parameter to I_1, D_1 and L_2

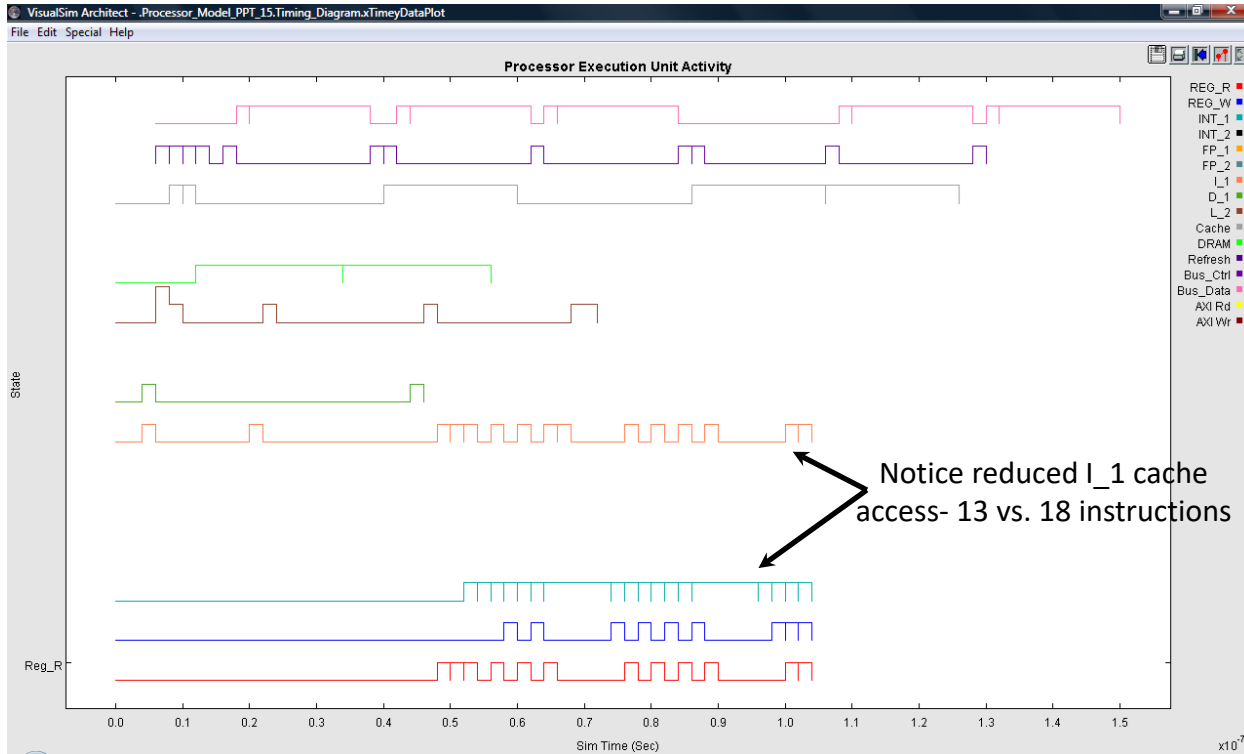
VS_AR/doc/Doc_Support/Processor_Model_PPT_14.xml



I_1, D_1 to DRAM
One INT_1
Cache Prefetch Line
parameter added and set to 3

Single Task with 9 ADD instructions
10 Words per cache line
0.25 Hit-ratio for Registers
Added Hit-Ratio parameter to I_1 and D_1

VS_AR/doc/Doc_Support/Processor_Model_PPT_15.xml



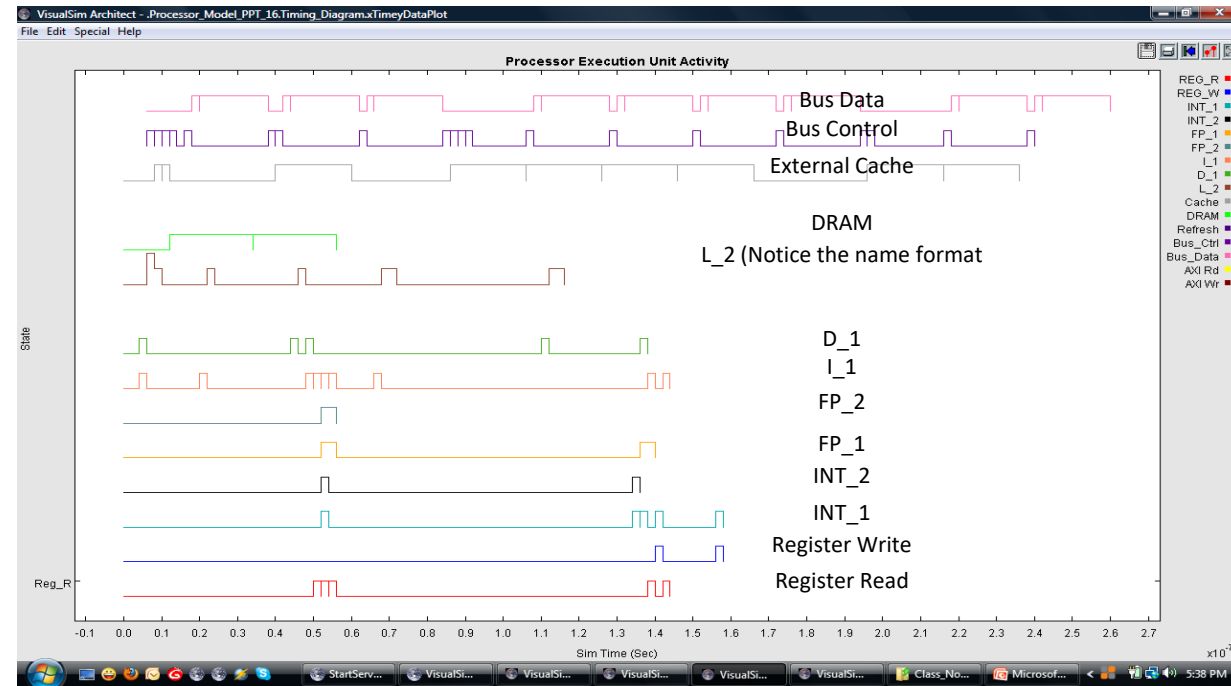
I₁ and D₁ L2; L2 to Cache
 Cache to DRAM
 One INT₁, One FP₁
 Bus to DRAM

Single Task; Pipeline flush with a *b,
 ADD and ADDN
 10 Words per cache line
 Multi-instruction per cycle
 Added Cache_Loop_Words to define
 loop length

All Processor features

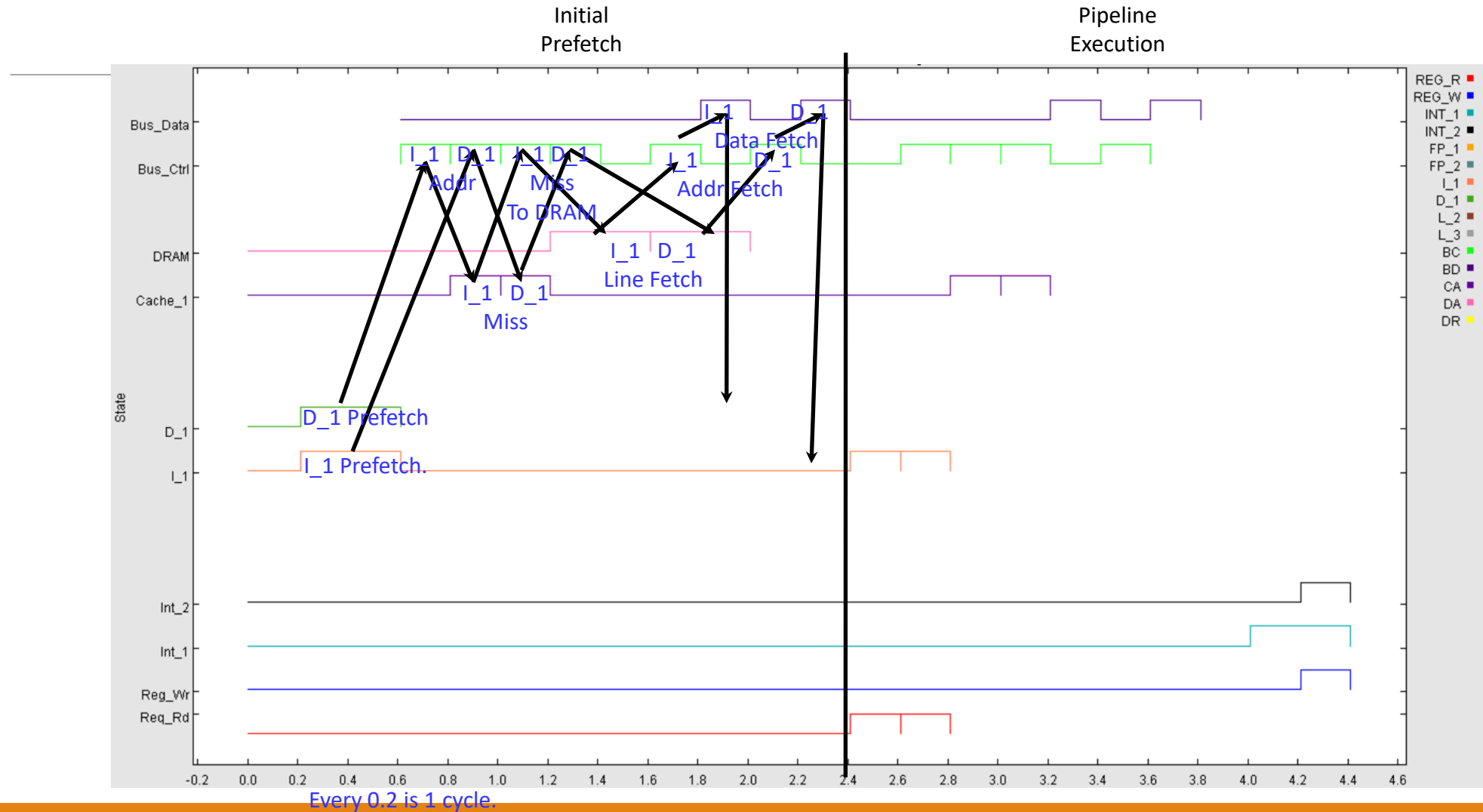
Refer to model

VS_AR/doc/Doc_Support/Processor_Model_PPT_16.xml



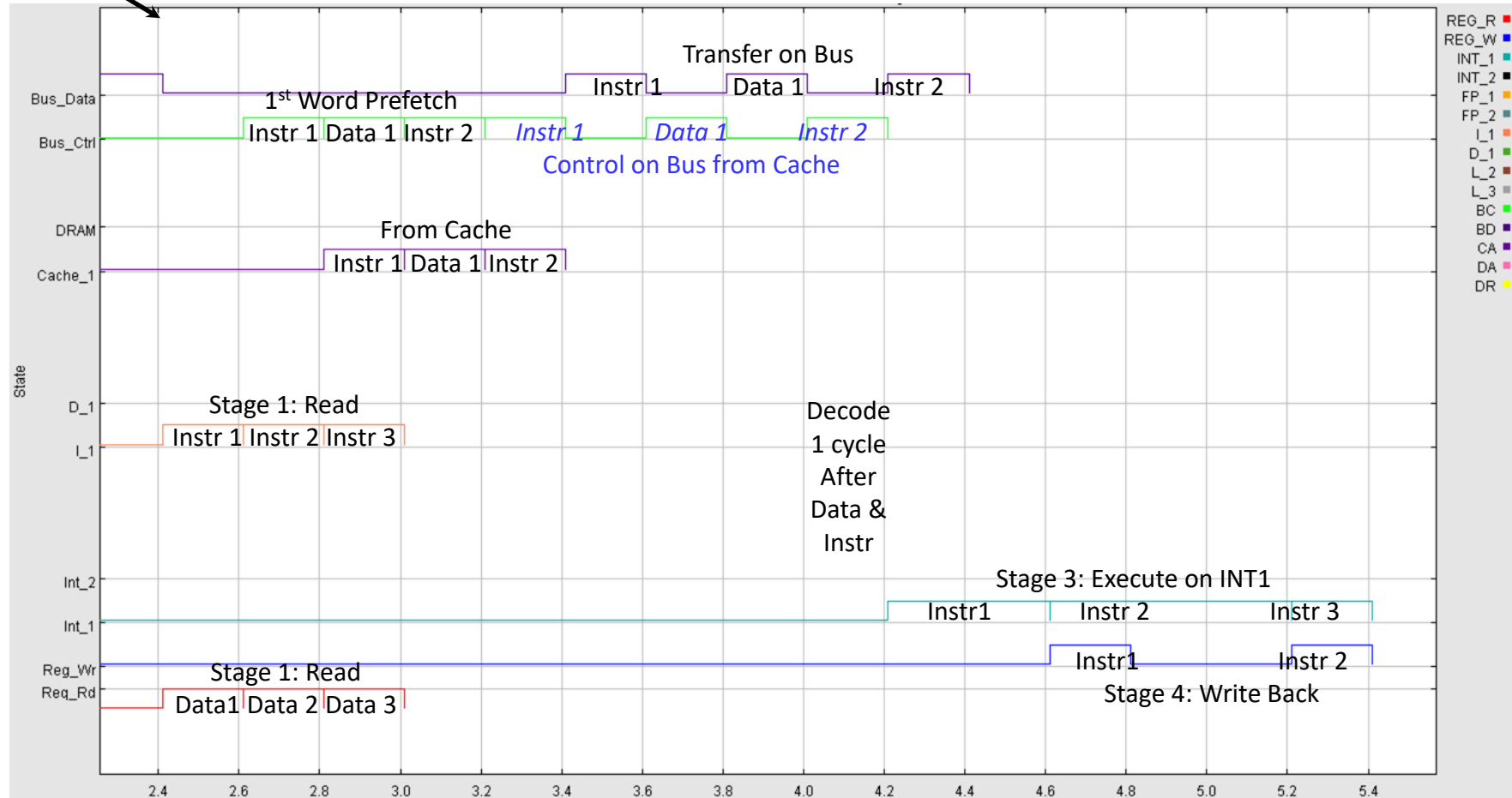
x10⁻⁷

Processor Timing Diagram- Prefetch



Processor Timing Diagram- Pipeline

3 instructions
ADD, SUB, MUL
Cycles: 2,3,4



Grid line is 1 cycle.

```

arm_isa_a65_ae_gem5.txt - Notepad
File Edit Format View Help
/* Instruction Set or File Path. */

Mnew Ra Rb Rc Rd Re Rf Rg Rh ; /* Label */
ARM INTG ALU FPNEOSIMD LDSTR ;

INTG      INT_1 INT_2 ; /* Integer_add */
ALU       INT_3 INT_4 ; /*ALU Branch*/
FPNEOSIMD FP_1      ; /*Floating point/NEON/ASIMD instructions*/
LDSTR     INT_5      ; /*Load/Store instructions*/

begin size_config
  Read   6      64      INT_5[1:163] ;
  Write  6      128     INT_5[164:500] ;
end size_config

begin execUnit_config
  Queue_Size INT_1 16 ;
  Queue_Size INT_2 16 ;
  Queue_Size INT_3 16 ;
  Queue_Size INT_4 16 ;
  Queue_Size FP_1 16 ;
  Queue_Size FP_2 16 ;
  Queue_Size INT_5 12 ;
end execUnit_config

```

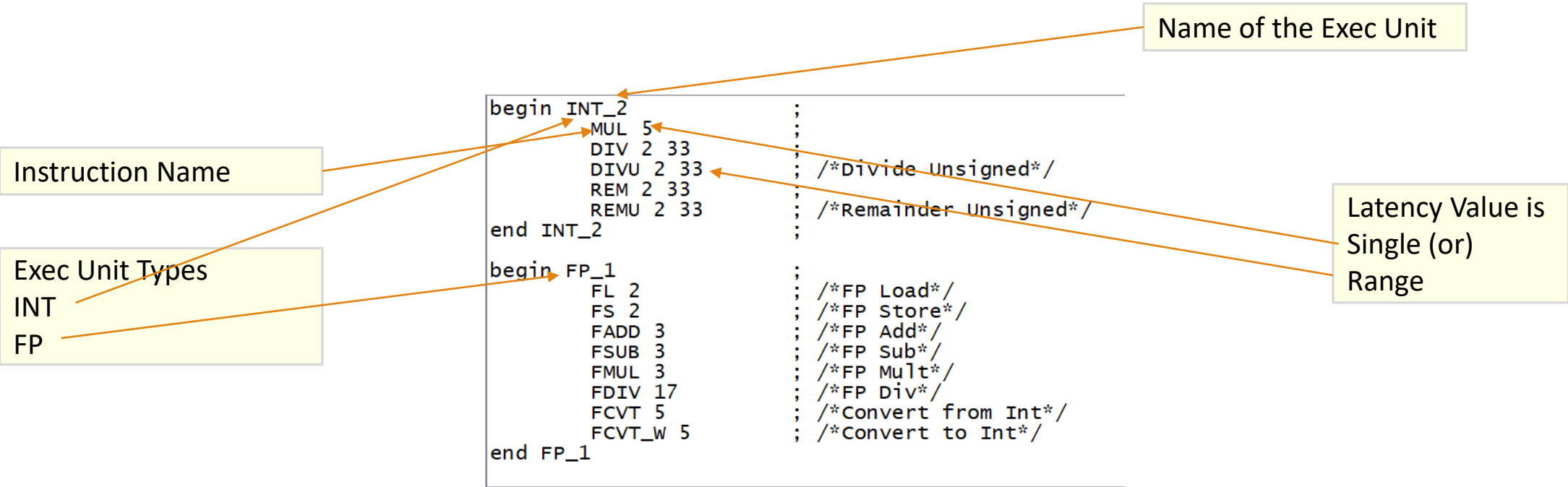
Instruction Set

Execution Units

Datapath to D-cache

Issue Queue

Instruction Set



Traffic Profile

- Using traces from ARM fast Models
- Using traces from GEM5 Model
- Generating instruction traces using a Task Generator in VisualSim

Traces from ARM fast model

```
cpu0.INST: DEBUG_STATE=N PC=0x0000000088000260 OPCODE=0xb86768a7 SIZE=0x04 MODE=EL2h ISET=AArch64 PADDR=0x0000000088000260 NSDESC=0x01
PADDR2=0x0000000088000260 NSDESC2=0x01 NS=0x01 ITSTATE=0x00 INST_COUNT=0x0000000007e1d3b LOCAL_TIME=0x00000000022c7cf0
CURRENT_TIME=0x000000133e561ed0 CORE_NUM=0x00 DISASS="LDR      w7,[x5,x7]"
```

Trace from ARM Fast Model

Parsed and Read into VisualSim

```
{"LDR","EOR","UBFIZ","LDR","EOR","UBFIZ","LDR",
",","EOR","B","CMP","B_NE","LDR","EOR","UBFIZ",
"LDR","EOR","UBFIZ","LDR","EOR","UBFIZ","LDR",
",","EOR","UBFIZ","LDR","EOR","B","CMP","B_NE",
"LDR","EOR","UBFIZ","LDR","EOR","UBFIZ","LDR",
",","EOR","UBFIZ","LDR","EOR","UBFIZ","LDR","EOR",
",","B","CMP","B_NE","LDR","EOR","UBFIZ","LDR",
",","EOR","UBFIZ","LDR","EOR","UBFIZ","LDR","EOR",
```

1	Instr	Min_latency_ps	Max_Latency_ps
2	LDR	10000	10000
3	EOR	10000	10000
4	UBFIZ	10000	10000
5	B	10000	10000
6	CMP	10000	10000
7	B.NE	10000	10000

```
VisualSim Architect - .fast_Model_Trace_Demo.TextDisplay

DISPLAY AT TIME           ----- 0.0 ps -----
{CORE_NUM                  = "0x00",
CPU_Action                 = "INST",
CPU_Name                   = "cpu0",
CURRENT_TIME               = "0x000000133e561ed0",
DEBUG_STATE                = "N",
DISASS                     = "LDRw7,[x5,x7]",
INST_COUNT                 = "0x0000000007e1d3b",
ISET                       = "AArch64",
ITSTATE                    = "0x00",
LOCAL_TIME                 = "0x00000000022c7cf0",
MODE                       = "EL2h",
NS                         = "0x01",
NSDESC                     = "0x01",
NSDESC2                    = "0x01",
OPCODE                     = "0xb86768a7",
PADDR                      = "0x0000000088000260",
PADDR2                     = "0x0000000088000260",
PC                         = "0x0000000088000260",
SIZE                       = "0x04"}
```

Branch Mis-Prediction from Traces

```

INST: PC=0x000000008000062c OPCODE=0x54000168 SIZE=0x04 MODE=EL1h ISET=AArch64
PADDR=0x000000008000062c NSDESC=0x01 PADDR2=0x000000008000062c NSDESC2=0x01 NS=0x01
ITSTATE=0x00 INST_COUNT=0x000000000001080c LOCAL_TIME=0x0000000000041eb0
CURRENT_TIME=0x000000002eab7ab0 CORE_NUM=0x00 DISASS="B.HI      {pc}+0x2c ; 0x80000658"

WAYPOINT: PC=0x000000008000062c ISET=AArch64 TARGET=0x0000000080000658
TARGET_ISET=AArch64 TAKEN=N IS_COND=Y CORE_NUM=0x00

INST: PC=0x0000000080000630 OPCODE=0x7100151f SIZE=0x04 MODE=EL1h ISET=AArch64
PADDR=0x0000000080000630 NSDESC=0x01 PADDR2=0x0000000080000630 NSDESC2=0x01 NS=0x01
ITSTATE=0x00 INST_COUNT=0x000000000001080d LOCAL_TIME=0x000000000005f370
CURRENT_TIME=0x000000002ead4f70 CORE_NUM=0x00 DISASS="CMP      w8,#5"
  
```

Branch Misprediction penalty is calculated from the timestamp values:

Here,

$$\begin{aligned}
 \text{Total time taken for Mis predicted instr} &= \text{int}(0x000000002ead4f70) - \text{int}(0x000000002eab7ab0) \\
 &= 783110000 - 782990000 = 120000 \text{ picoseconds} = 12 \text{ ticks} \\
 &= 1 \text{ tick}(\text{instruction latency}) + 11 \text{ ticks}(\text{branch penalty})
 \end{aligned}$$

Traces generated from GEM5

```

info: Entering event queue @ 0. Starting simulation...
2000: system.cpu.icache: getBusPacket created ReadReq addr 0x440 size 64
2000: system.cpu.dcache: recvTimingSnoopReq for ReadReq addr 0x440 size 64
2000: system.cpu.dcache: handleSnoop for ReadReq addr 0x440 size 64
2000: system.cpu.dcache: handleSnoop snoop miss for ReadReq addr 0x440 size 64
2000: system.mem_ctrls: recvTimingReq: request ReadReq addr 1088 size 64
2000: system.mem_ctrls: Read queue limit 32, current size 0, entries needed 1
2000: system.mem_ctrls: Address: 1088 Rank 0 Bank 0 Row 0
2000: system.mem_ctrls: Read queue limit 32, current size 0, entries needed 1
2000: system.mem_ctrls: Adding to read queue
2000: system.mem_ctrls: Request scheduled immediately
2000: system.mem_ctrls: Single request, going to a free rank
2000: system.mem_ctrls: Timing access to addr 1088, rank/bank/row 0 0 0
2000: system.mem_ctrls: 2000,ACT2
2000: system.mem_ctrls: VISUALSIM_LOG: Rank: 0 Bank: 0 SIZE: 64 ACT: 0 READ: 13750 Address: 1088 Row: 0
2000: system.mem_ctrls: Activate at tick 2000
2000: system.mem_ctrls: Activate bank 0, rank 0 at tick 2000, now got 1 active
2000: system.mem_ctrls: Access to 1088, ready at 46250 bus busy until 46250.
46250: system.mem_ctrls: processRespondEvent(): Some req has reached its readyTime
46250: system.mem_ctrls: Responding to Address 1088.. 46250: system.mem_ctrls: Done
73250: system.cpu.icache: Handling response ReadResp for addr 0x440 size 64 (ns)
73250: system.cpu.icache: Block for addr 0x440 being updated in Cache
73250: system.cpu.icache: Block addr 0x440 (ns) moving from state 0 to state: 7 (E) valid: 1 writable: 1 readable: 1 dirty: 0 tag: 0
73250: system.cpu.icache: Leaving recvTimingResp with ReadResp for addr 0x440
79000: system.cpu T0 : @_start : mov fp, #0 : IntAlu : D=0x0000000000000000
79000: system.cpu.icache: access for ReadReq addr 0x450 size 4
79000: system.cpu.icache: ReadReq (ifetch) addr 0x450 size 4 (ns) hit state: 7 (E) valid: 1 writable: 1 readable: 1 dirty: 0 tag: 0
81000: system.cpu T0 : @_start+4 : mov lr, #0 : IntAlu : D=0x0000000000000000
81000: system.cpu.icache: access for ReadReq addr 0x454 size 4
81000: system.cpu.icache: ReadReq (ifetch) addr 0x454 size 4 (ns) hit state: 7 (E) valid: 1 writable: 1 readable: 1 dirty: 0 tag: 0
83000: system.cpu.dcache: access for ReadReq addr 0x8de50 size 4
83000: system.cpu.dcache: ReadReq addr 0x8de50 size 4 (ns) miss
85000: system.cpu.dcache: getBusPacket created ReadReq addr 0x8de40 size 64
85000: system.cpu.icache: recvTimingSnoopReq for ReadReq addr 0x8de40 size 64
85000: system.cpu.icache: handleSnoop for ReadReq addr 0x8de40 size 64
85000: system.cpu.icache: handleSnoop snoop miss for ReadReq addr 0x8de40 size 64

```


Trace file Converted to VisualSim Format

TimeStamp	Source_Device_Name	Dest_Device_Name	Comment	Command	Address	size
2000	cpu	icache	getBusPacket	ReadReq	0x440	64
2000	cpu	dcache	recvTimingSnoopReq	ReadReq	0x440	64
2000	cpu	dcache	handleSnoop-snoop miss	ReadReq	0x440	64
2000		mem_ctrls	recvTimingReq	ReadReq	1088	64
2001		mem_ctrls	Responding to Address		1088	64
73250	cpu	icache	Handling response	ReadResp	0x440	64
154000	cpu	dcache	access	WriteReq	0x8de50	4

Cache and Memory stats

Demo output csv from gem5 traces.

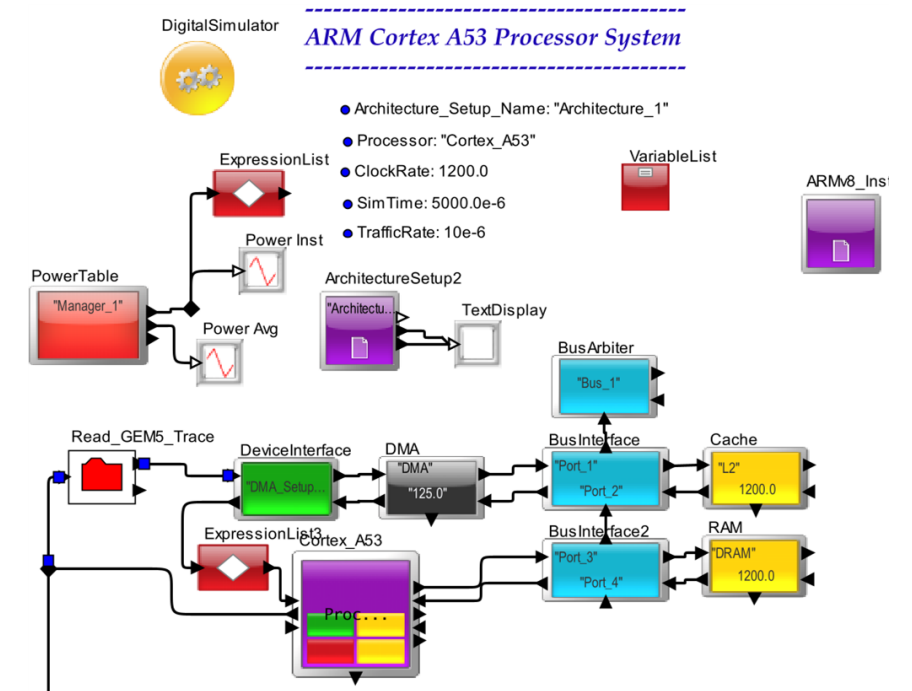
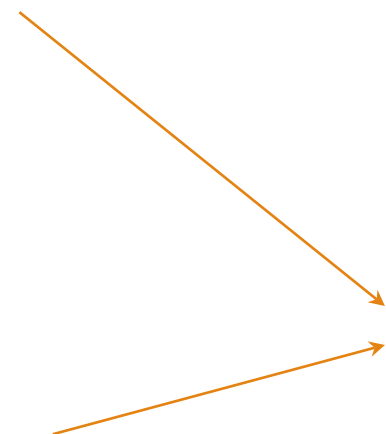
Time_Stamp	CPU_Core	Instructions	Execution_unit	
79000	T0	mov	IntAlu	D=0x0000000000000000
81000	T0	mov	IntAlu	D=0x0000000000000000
83000	T0	ldr	MemRead	D=0x0000000000000008
154000	T0	str	MemWrite	D=0x00000000beffff54 A=0xbeffff50
156000	T0	subi_uop	IntAlu	D=0x00000000beffff50
158000	T0	str	MemWrite	D=0x0000000000000000 A=0xbeffff4c
160000	T0	subi_uop	IntAlu	D=0x00000000beffff4c

Different cpu cores stats

Using Trace in VisualSim

TimeStamp	Source_Device_Name	Dest_Device_Name	Comment	Command	Address	size
2000	cpu	icache	getBusPacket	ReadReq	0x440	64
2000	cpu	dcache	recvTimingSnoopReq	ReadReq	0x440	64
2000	cpu	dcache	handleSnoop-snoop miss	ReadReq	0x440	64
2000		mem_ctrls	recvTimingReq	ReadReq	1088	64
2001		mem_ctrls	Responding to Address		1088	64
73250	cpu	icache	Handling response	ReadResp	0x440	64
154000	cpu	dcache	access	WriteReq	0x8de50	4

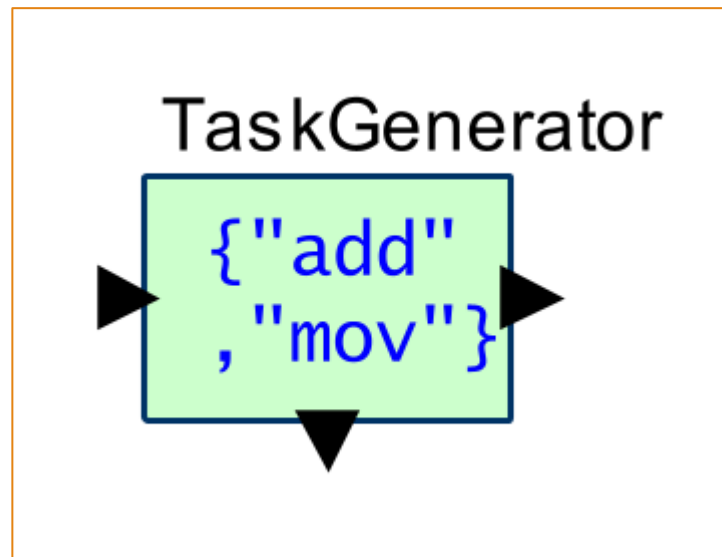
Time_Stamp	CPU_Core	Instructions	Execution_unit	
79000	T0	mov	IntAlu	D=0x0000000000000000
81000	T0	mov	IntAlu	D=0x0000000000000000
83000	T0	ldr	MemRead	D=0x0000000000000008
154000	T0	str	MemWrite	D=0x00000000beffff54 A=0xbeffff50
156000	T0	subi_uop	IntAlu	D=0x00000000beffff50
158000	T0	str	MemWrite	D=0x0000000000000000 A=0xbeffff4c
160000	T0	subi_uop	IntAlu	D=0x00000000beffff4c



These instructions read via TrafficReader as input to the Processor block

Using Task Generator Module

- More dynamic and distributed traffic profile can be generated
- “n” number of Software tasks can be defined



Task Generator – config file

Task_Name	Relative_Time(double)_or Number_Instructions(int)	Type	Pct	Type	Pct	Type	Pct	Type	Pct	Type	Pct	Type	Pct	Type	Pct
High Mapper	702	ALU	13	MDU	7	FPU	5	BU	10	SU	10	LU	15	PU	5
Low Mapper	568	ALU	17	MDU	11	FPU	10	BU	20	SU	4	LU	13	PU	7
Sensor	4021	ALU	19	MDU	13	FPU	6	BU	16	SU	8	LU	0	PU	5
Read_Frame	3500	ALU	22	MDU	11	FPU	9	BU	8	SU	13	LU	18	PU	6
Decode_Frame	428	ALU	31	MDU	12	FPU	8	BU	27	SU	0	LU	0	PU	8
Video_Post_Proc	964	ALU	13	MDU	7	FPU	4	BU	29	SU	17	LU	0	PU	6
Render_Frame	355	ALU	17	MDU	10	FPU	6	BU	11	SU	3	LU	10	PU	7
Format_Conv	604	ALU	17	MDU	9	FPU	5	BU	21	SU	9	LU	8	PU	13
Rotate_Frame	557	ALU	18	MDU	9	FPU	5	BU	21	SU	9	LU	8	PU	11
Display	800	ALU	7	MDU	3	FPU	2	BU	5	SU	17	LU	26	PU	2

Software tasks

Number of instructions per task

The Total Number of instructions are made up of instructions of different types. The percentages of each type of instruction is specified here.

Task Generator - Config File

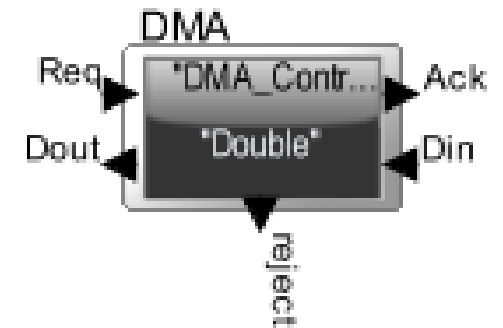
Description	Type (used below)	Mnemonic List	*/
Arith_1	ALU	ADD SUB	;
Arith_2	MDU	SMULH SDIV	;
Float	FPU	FADD FSUB FMUL FDIV FSQRT	FCVTL ;
Branch_Unit	BU	JMP JMP.6 JMP.10 JMP.14	;
Shift_Unit	SU	LSLV LSRV	;
Logic_Unit	LU	AND ORR XOR	;
Mispredict_Unit	PU	*JMP	;
LoadStore_Unit	LSU	LDRSW STRB	;
Move_Unit	MU	MOV	;

This type descriptor is used in the previous slide. User can specify the percentage of each type of instruction for each software operation

Technology-specific Hardware Modeling

DMA

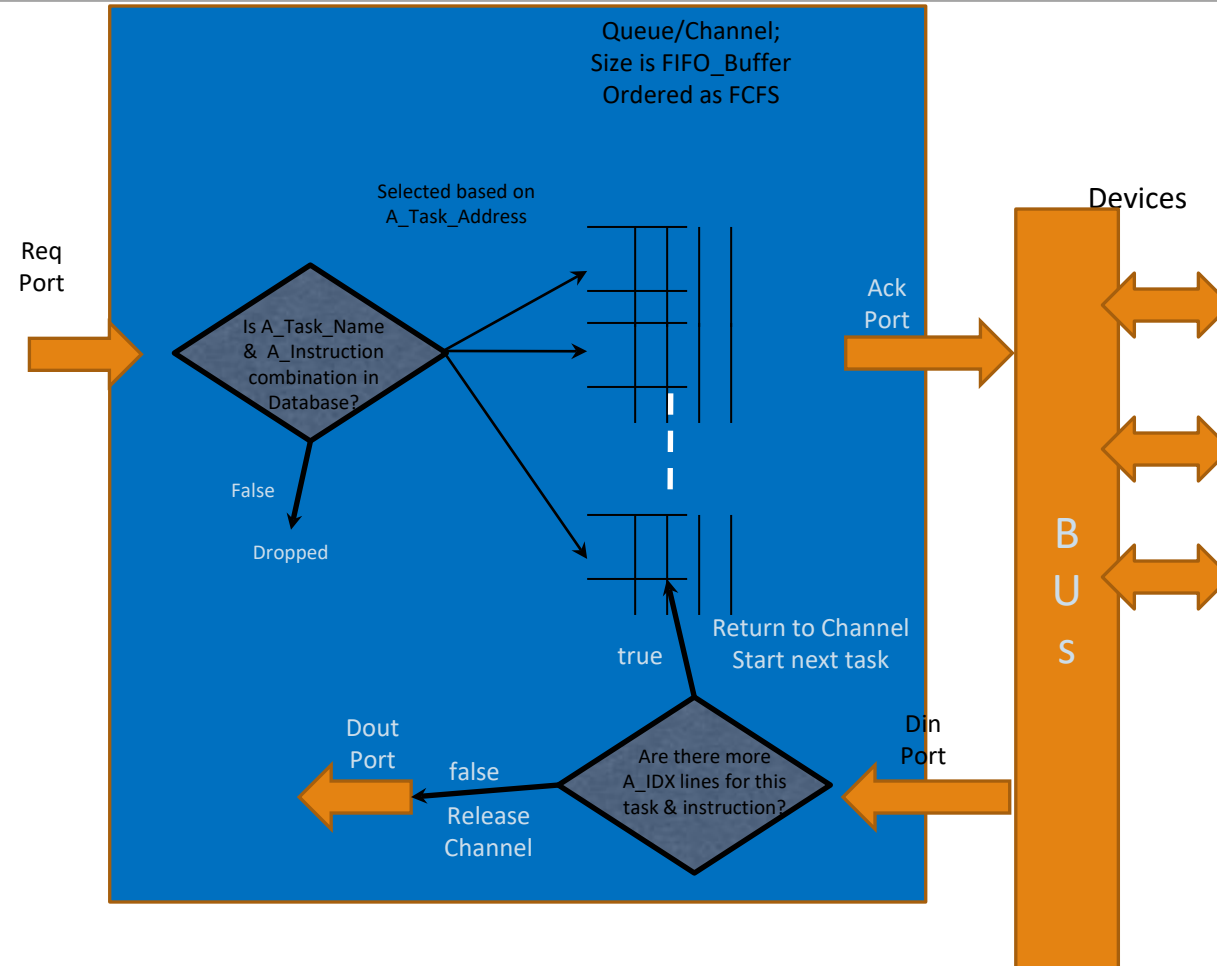
- DMA block that represents a memory controller that sits between the Processor or bus or DeviceInterface Block and the Memory bank.
- Receive requests directly from the processor block or from the Req port.
- Performs Sequential operation
- Channels operate concurrently
- Request fragments into Burst Size & each one is sent out as one task
- Delays are based on block parameters & no additional delay
- All fragments of Write must arrive to release channel
- **Hardware Devices - > DMA**
- Two ways to characterize the task operation of the DMA Block:
 - ✓ Using Database
 - ✓ Using Data Structure Fields



DMA Block Notes

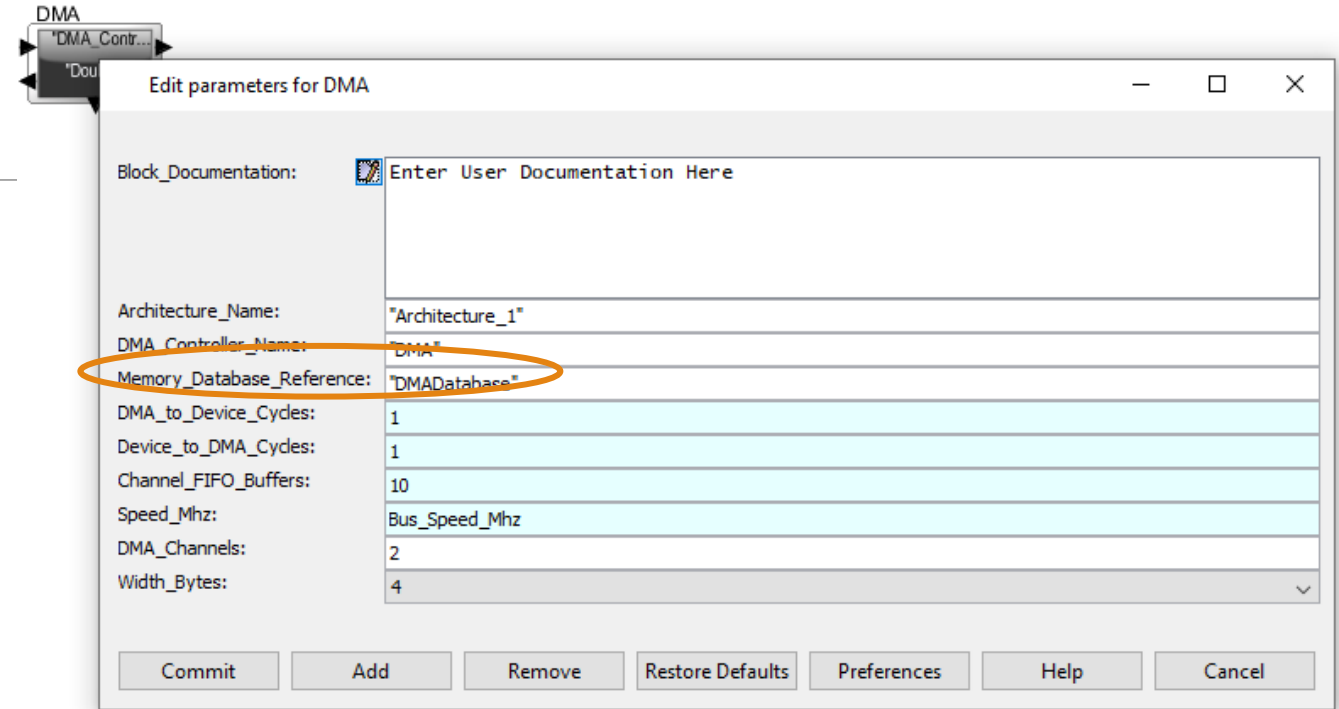
- **A_Bytes_Sent** = Bus_Width
- **A_Bytes** = Burst_Size or remaining Bytes
- Output Request has **A_Task_Flag** = true
- **A_Command**: Read returns as a Write
- **A_Command**: Write returns as a Read
- If Req port connected to non-Architecture block, then entry in Routing Table is required.
- A_Instruction must not have a # prefix

DMA Block Diagram



Using Database

- Database Fields are:
 - ✓ A_Task_Name
 - ✓ A_Instruction
 - ✓ A_IDX
 - ✓ A_Task_Source
 - ✓ Burst_Word_Size
 - ✓ A_Task_Address
 - ✓ A_Command
 - ✓ A_Bytes
 - ✓ A_Priority
 - ✓ A_Destination



A_Task_Name	A_Instruction	A_IDX	A_Task_Source	Burst_Word_Size	A_Task_Address	A_Command	A_Bytes	A_Priority	A_Destination
My_Task_1	LD_LDR	0	DRAM	128	1	Read	128	0	DMA
My_Task_1	LD_LDR	1	DRAM	128	1	Read	128	1	DMA
My_Task_2	LD_LDR	0	DRAM	128	1	Write	128	1	DMA
My_Task_3	LD_LDR	0	Display	128	2	Write	128	3	DMA

DMA Database Block Entry Example



A_Task_Name	A_Instruction	A_IDX	A_Task_Source	Burst_Word_Size	A_Task_Address	A_Command	A_Bytes	A_Priority	A_Destination	;
Ext_Int	Load	0	Ext_DRAM	32	1	Read	128	0	DMA	;
Ext_Int	Load	1	Int_DRAM	32	1	Write	128	1	DMA	;
Serial	Store	0	Int_DRAM	32	2	Write	1024	3	DMA	;
Identifier : Match Request A_Task_Name	Instruction: Match Request A_Instruction	Handle many tasks per Req	Target device	Burst size is per task	Channel Number	Operation	Data transfer size is unique per task	Priority for queuing on Bus and others with priority	Target DMA block	

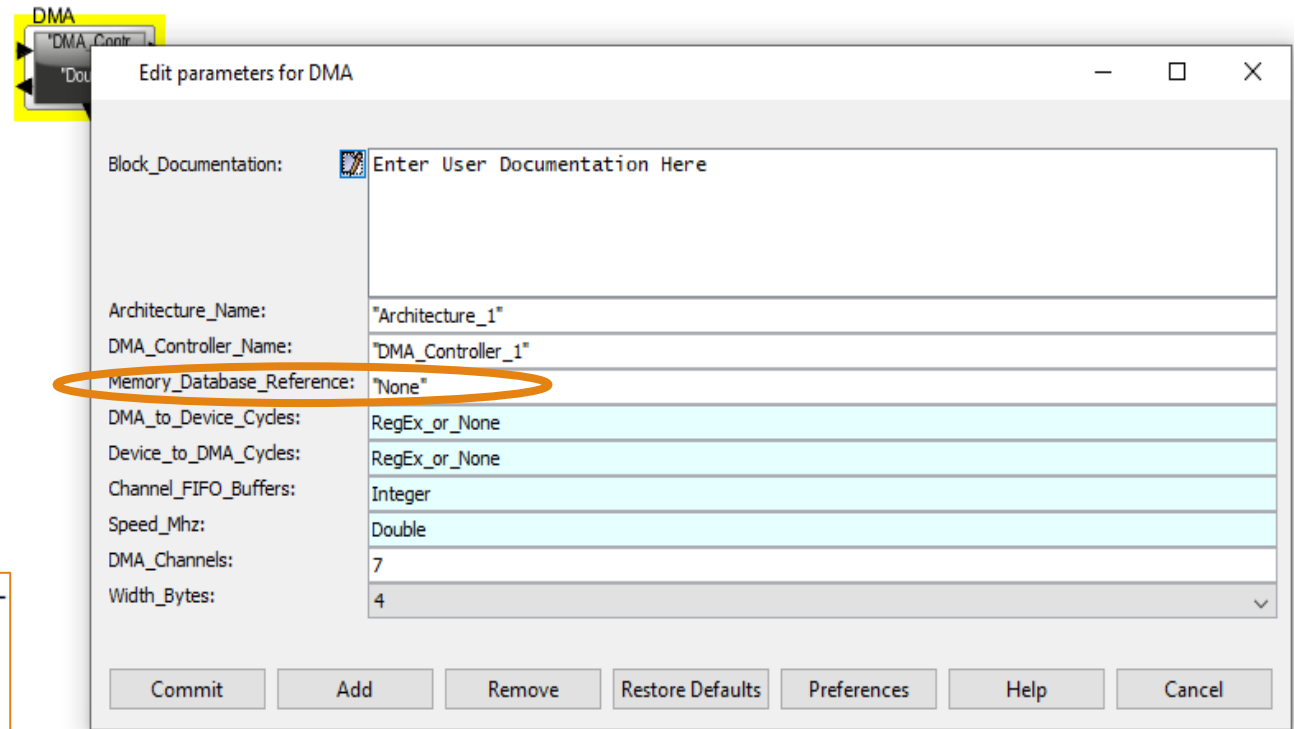
Using Data Structure

- Fields necessary in Data Structure

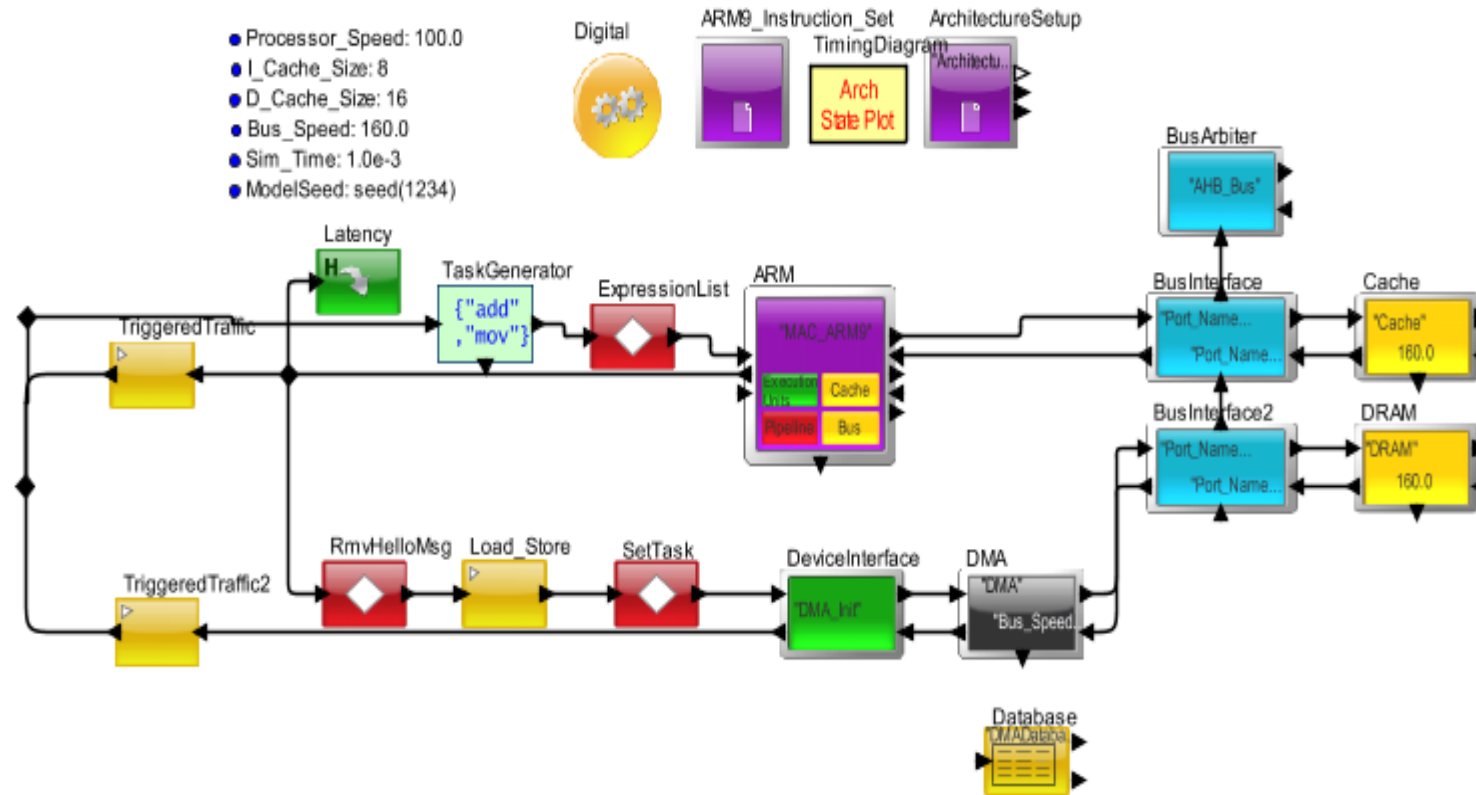
- ✓ A_Task_Name
- ✓ A_DMA_Command
- ✓ A_DMA_Destination
- ✓ A_Priority
- ✓ A_DMA_Bytes
- ✓ A_Task_Address
- ✓ A_DMA_Burst_Byte

```

DISPLAY AT TIME          ----- 0.0 ps -----
{A_DMA_Burst_Bytes      = 64,
A_DMA_Bytes              = {1024, 1024},
A_DMA_Command           = {"Read", "Write"},
A_DMA_Destination       = {"RAM", "Display"},
A_Priority               = 5,
A_Task_Address          = 2,
A_Task_Name             = "Activity1",
    
```



Example of DMA in model



Statistics

- The Architecture setup block generates statistics for DMA
 - ✓ IO_per_sec : Input and output transactions per second

- The data structure coming out from **dout** port will be having two added fields :

- ✓ Task_Latency (which gives the latency for the task completed)
- ✓ Task_Throughput (which gives the throughput for the task)

- DMA controller produces Traces

```

Task_Latency           = 1.04716E-4,
Task_Throughput       = 3.738083216568E7,
Time_Array            = {2.30001E-4, 3.05343E-4, 3.10456E-4, 3.10461E-4, 3.15598E-4, 3.15603E-4, 3.1987E-4, 3.19875E-4, 3.25012E-4, 3.25017E-4, 3.25017E-4, 3.25017E-4, 3.25017E-4},
Trace_Array           = {"DMA_Task_in", "DMA_Read_From_DRAM", "DMA_Read_Completed_From_DRAM", "DMA_Read_From_DRAM", "DMA_Read_Completed_From_DRAM"}
    
```

```

DMA_IO_per_sec_Max    = 9.8E6,
DMA_IO_per_sec_Mean   = 9.77E6,
DMA_IO_per_sec_Min    = 9.74E6,
DMA_IO_per_sec_StDev  = 30000.0,
DMA_Throughput_MBs_Max = 156.8,
DMA_Throughput_MBs_Mean = 156.48,
DMA_Throughput_MBs_Min = 156.16,
DMA_Throughput_MBs_StDev = 0.320000000000053,
    
```

Bridges and Switches

Bridge

- Blocking
- Delay is variable based on data size and speed
- Single source<->Destination

Switch (Blocking)

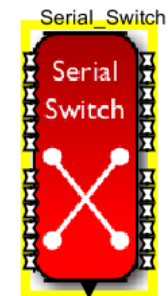
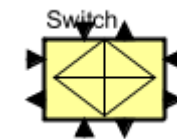
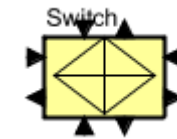
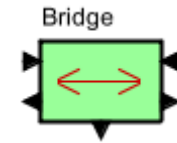
- Single interconnect
- 4 connected Devices and can add more
- Single-cycle delay for Read and Multi-cycle delay for Write

Switch (Non blocking)

- Point-to-point mesh with multiple channels per wire
- Basic block contains for 4 device connections; Can be expanded
- Single-cycle delay for Read and Multi-cycle delay for Write

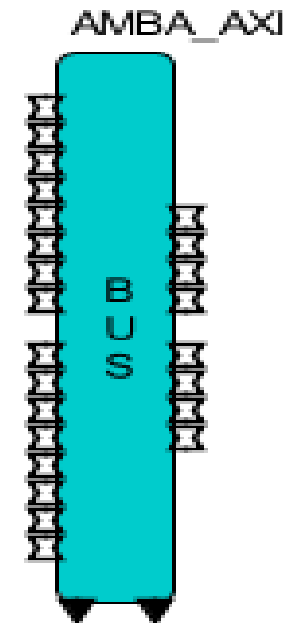
Serial_Switch (Non blocking)

- Point-to-point mesh with multiple channels per wire and multiple PHY per wire
- Basic block contains for 16 device connections. Can be expanded
- Single-cycle delay for Read and Multi-cycle delay for Write

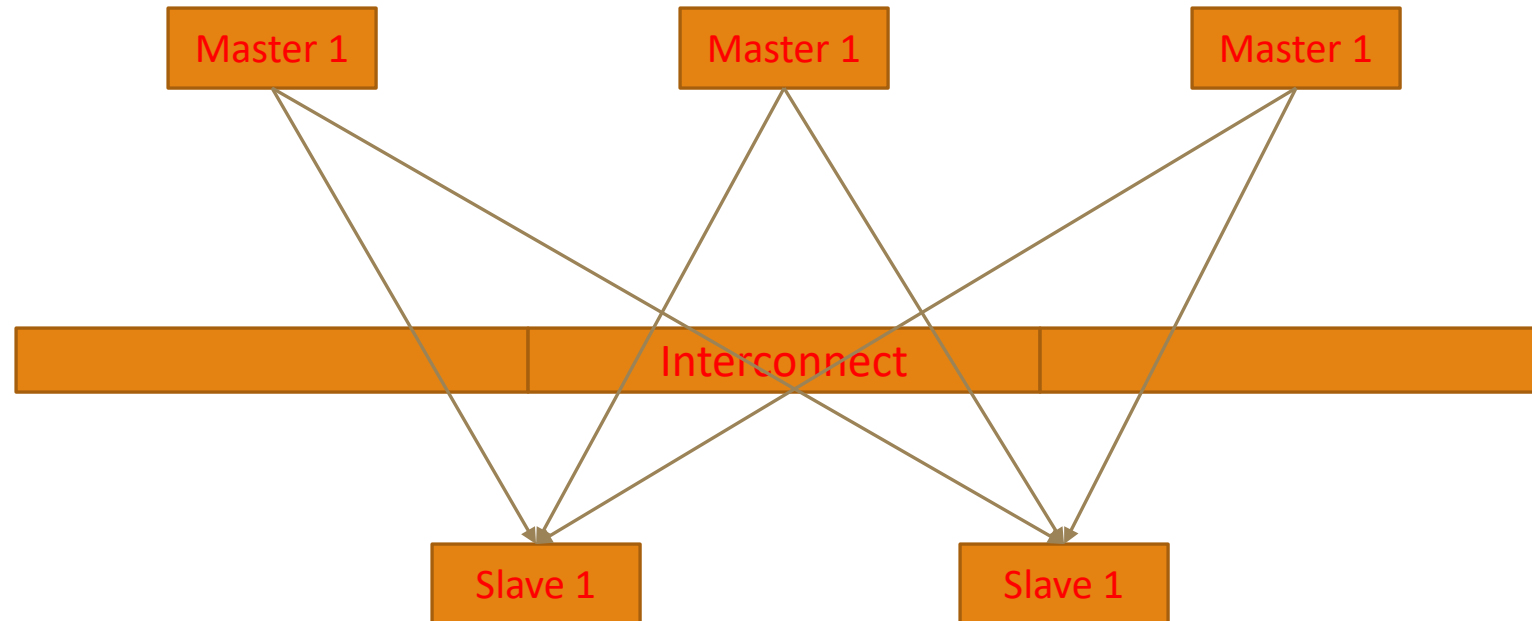


AXI BUS

- Suitable for high bandwidth and low latency Designs
- Enable high frequency operation without using bridges
- Supports 16 Masters and 8 Slave Ports
- Provides Statistics
- Arbitration Algorithms
 - ✓ Fixed Time Slot
 - ✓ Round Robin
 - ✓ User Algorithm
- Loops every cycle
 - ✓ Test for available requests
 - ✓ Test of Read Threshold and Wait Flags



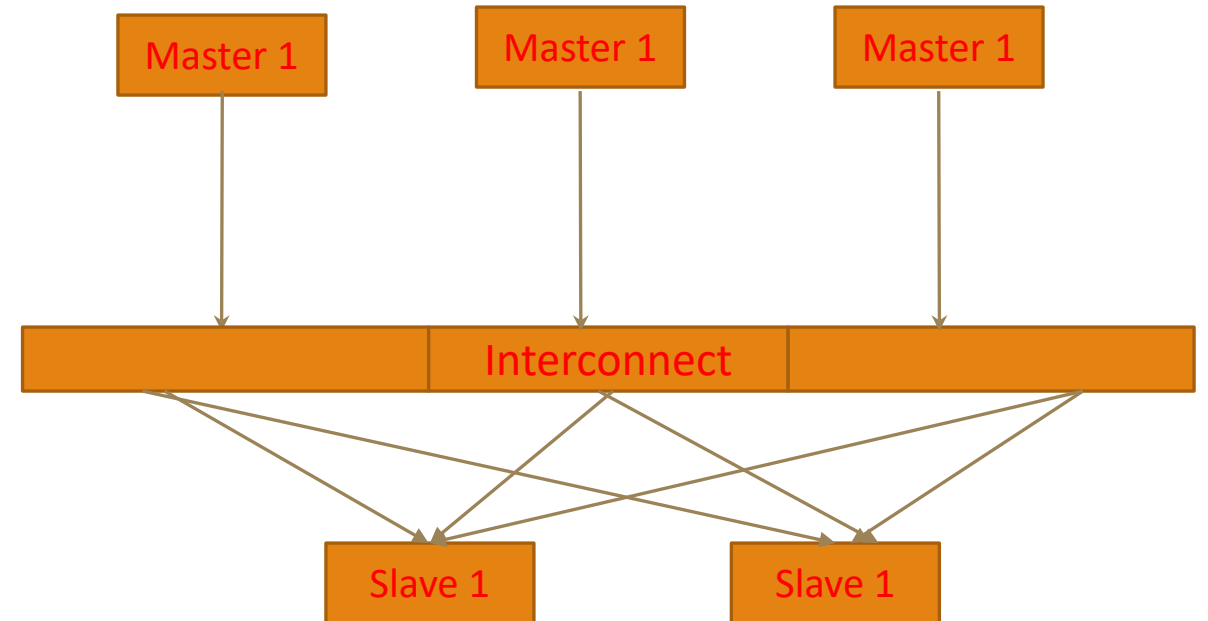
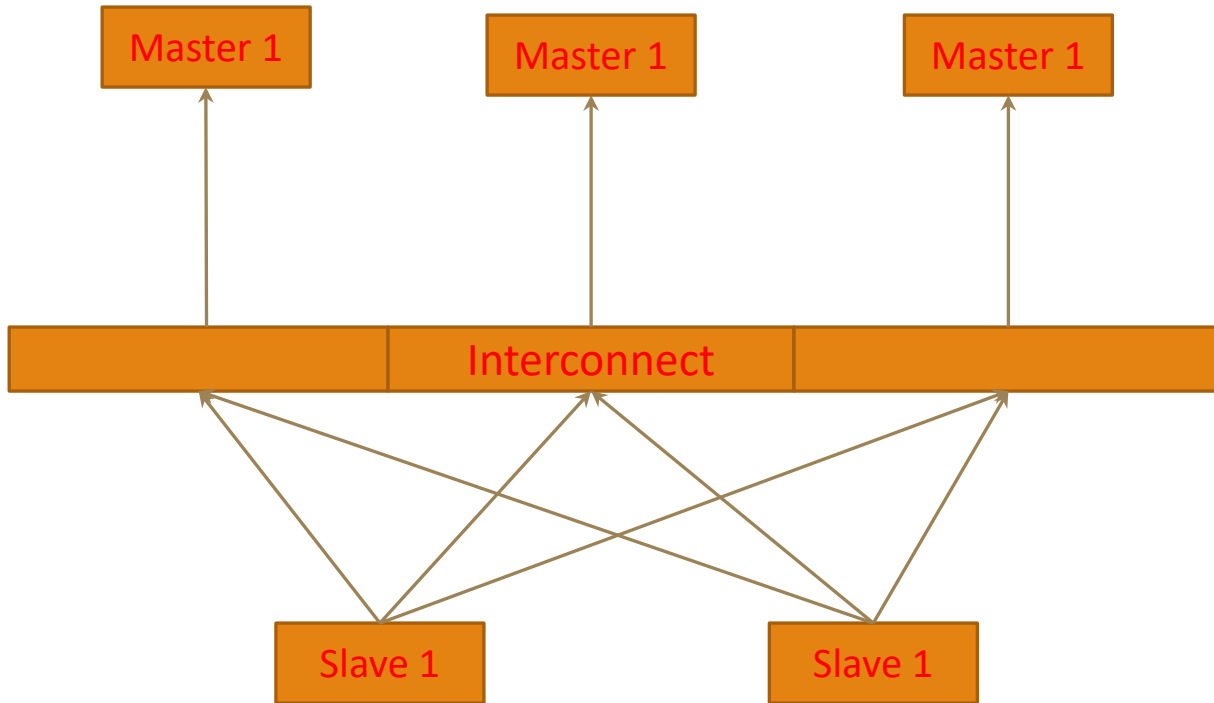
AXI Bus Block Diagram- Read/Write Request Channel



There is a separate set of Read Request
and a separate set of Write Request.

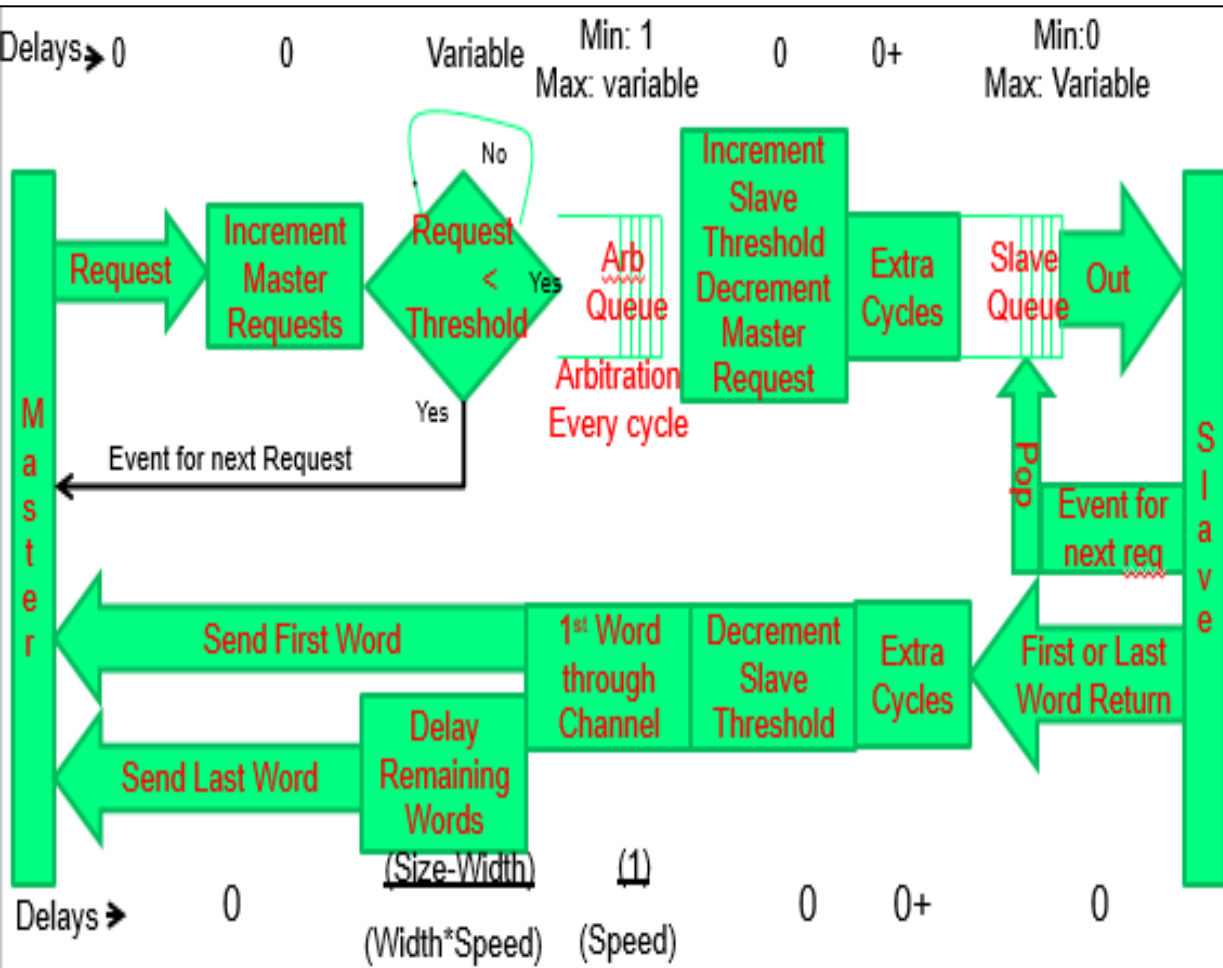
AXI Bus Block Diagram- Read Data Channel

AXI Bus Block Diagram- Write Data Channel

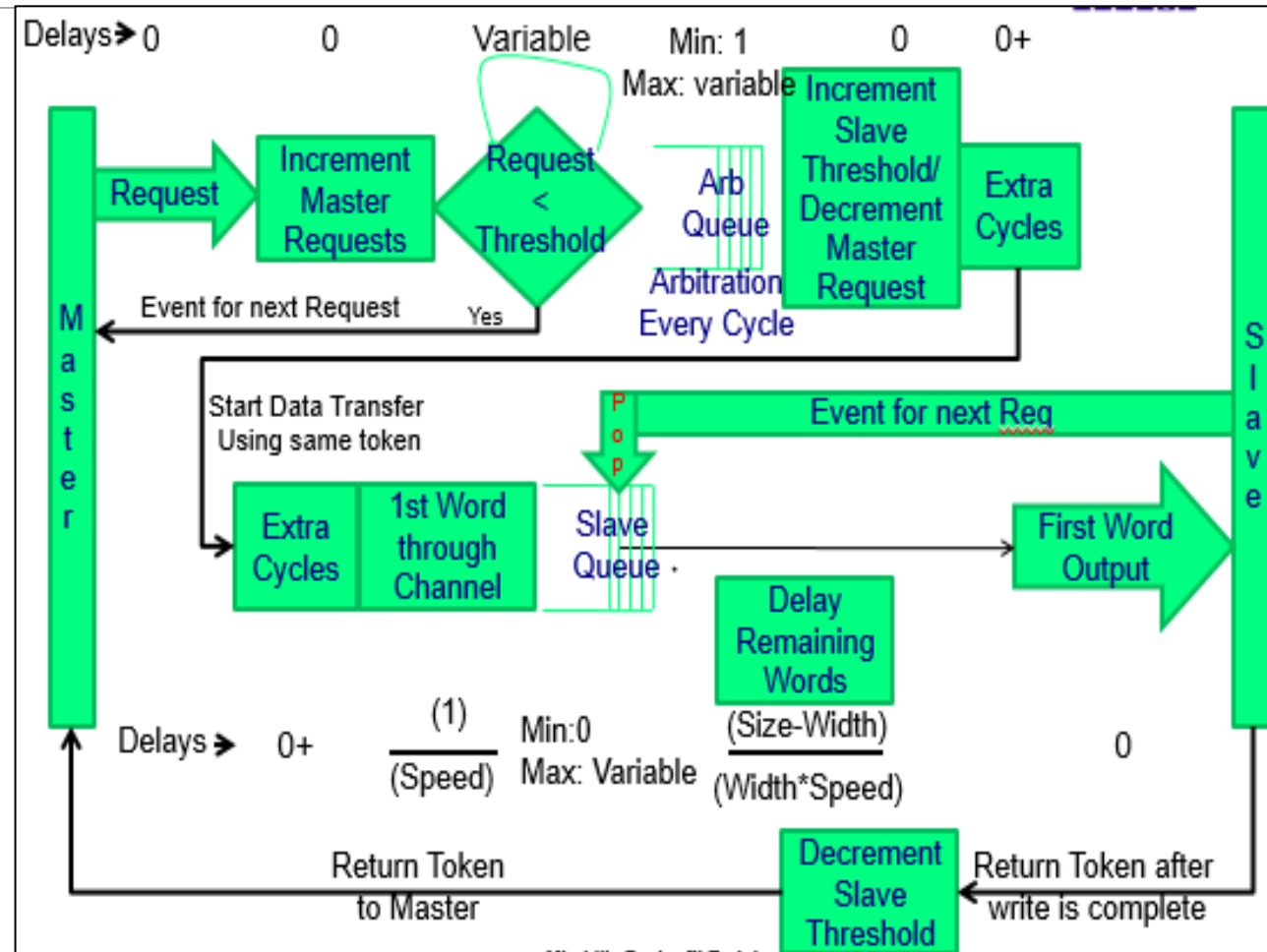


AXI Bus Block Diagram:

Read Flow



Write Flow



AXI Bus Block Diagram: Arrays

Master Outstanding- Read

- Array starting from master # 0

Master Outstanding- Write

- Array starting from master # 0

Slave Threshold- Read

- Array starting from Slave # 0

Slave Threshold- Write

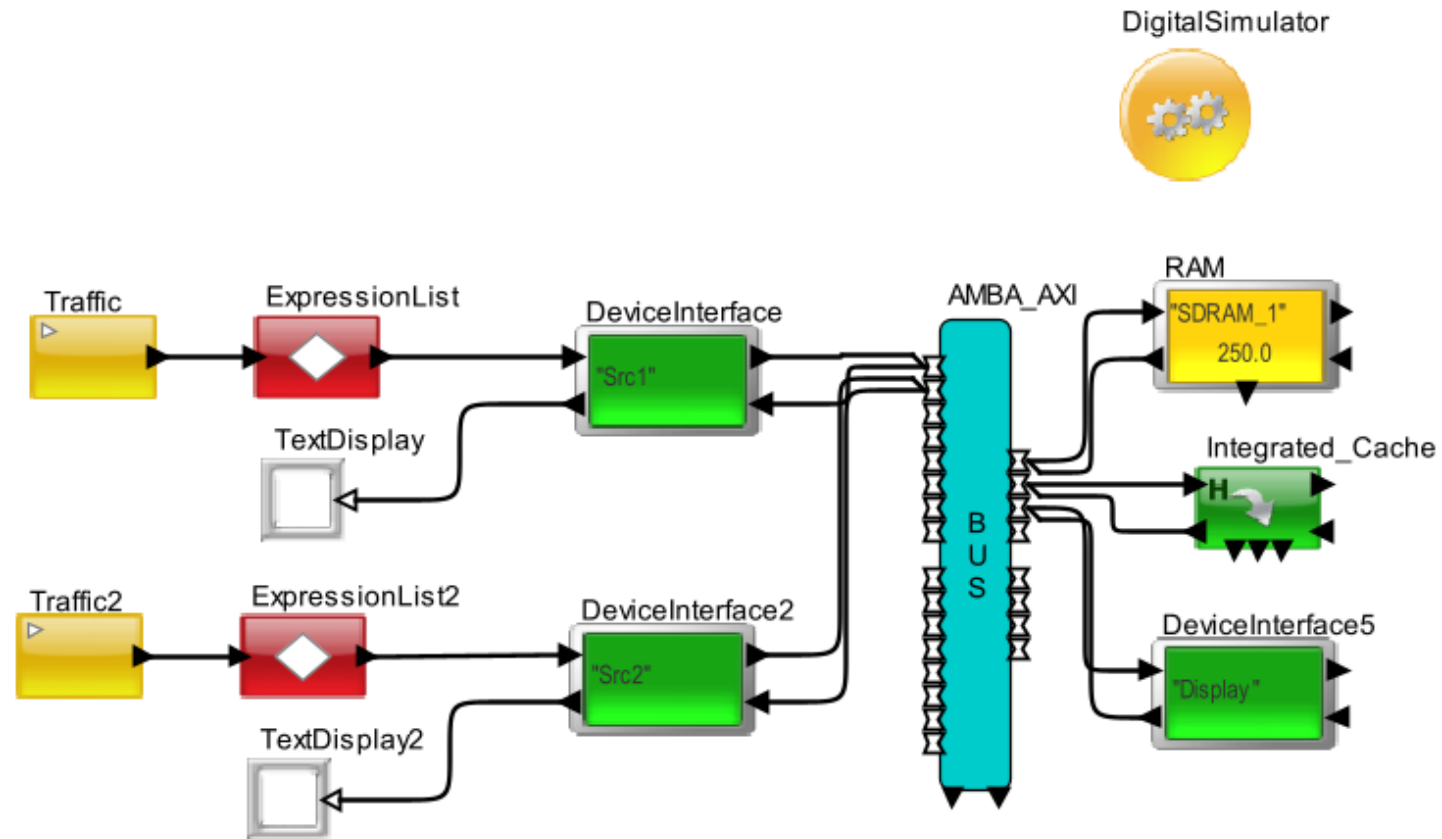
- Array starting from Slave # 0

Configuration

Interfaces and Buses ->AMBA -> AMBA_AXI

Edit parameters for AMBA_AXI4	
Architecture_Name:	"Architecture_1"
Bus_Name:	"AXI_Level_2"
AXI_Speed_Mhz:	Bus_Speed
AXI_Cycle_Time:	1.0E-06 / AXI_Speed_Mhz
_explanation:	Interfaces and Buses->AHB->AXI_Bus
Bus_Width:	8
Read_Threshold:	8
Write_Threshold:	8
Master_Request_Threshold:	{4,2,2,4,2,2,2}
Number_Masters:	2
Number_Slaves:	3
Threshold_Trans_T_Bytes_F:	true
Arbiter_FIX_1_RR_2_CUSTOM_3:	1
Slave_Speeds_Mhz:	Mhz, AXI_Speed_Mhz,AXI_Speed_Mhz, AXI_Speed_Mhz, AXI_Speed_Mhz, AXI_Speed_Mhz}
Extra_Cycles_for_RdReq_WrReq_RdData_WrData:	{0, 0, 0, 0, 0, 0, 0}
Devices_Attached_to_Slave_by_Port:	{{"DDRDRAM"}, {"DDRDRAM2"}, {"DDRDRAM3"}, {"Device_4"}}
Master_First_Word_Flag:	true
Master_Throttle_Enable:	{false, false, false, false, false, false, false, false, false, false, false, false, false, false}
Slave_Throttle_Enable:	{false, false, false, false}
DEBUG:	false
Custom_Arbiter_File:	"none"
Custom_Arbiter_Path:	"none"
Fixed_Priority_Array:	16}, {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}, {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}}
Slave_First_Word_Flag:	true /* Not Active in Default Slave */
Custom_Slave_File:	"none"
Ports_to_Plot:	{0,0} /* master n, slave m, 0 disables */

How to Connect?



Statistics

- Master/Slave combination
 - ✓ Queue number 1 for Master 1 to Slave 1
 - ✓ Queue Number 2 for Master 1 to Slave 2
 - ✓ Read Request and Write Request
- For each master
 - ✓ Read Data Channel and Write Data Channel
 - ✓ Queue number is the port number
- List of overall statistics
 - ✓ Number_Entered
 - ✓ Number_Exited
 - ✓ Occupancy_Max/Min/Std_Deviation/Mean
 - ✓ Total_Delay_Max/Min/Std_Deviation/Mean
 - ✓ Utilization_Mean

```

DISPLAY AT TIME          ----- 200.00000 us -----
{AXI_1_Master_1_N_OverFlows = 90,
DS_NAME                    = "AXI_1_Detected_OverFlows"}

DISPLAY AT TIME          ----- 200.00000 us -----
{AXI_1_Master_1_Read_Data_Bytes = 21856,
AXI_1_Master_1_Read_Data_MBps  = 109.28,
AXI_1_Slave_1_Read_Data_Bytes  = 20832,
AXI_1_Slave_1_Read_Data_MBps  = 104.16,
AXI_1_Slave_2_Read_Data_Bytes  = 1024,
AXI_1_Slave_2_Read_Data_MBps  = 5.12,
DS_NAME                      = "AXI_1_Rd_Wr_MBps"}

DISPLAY AT TIME          ----- 200.00000 us -----
{AXI_1_Slave_1_Rd_Threshold_Usage = 2.0,
AXI_1_Slave_Transactions          = 684,
DS_NAME                            = "AXI_1_Slave_1_Rd_Threshold"}

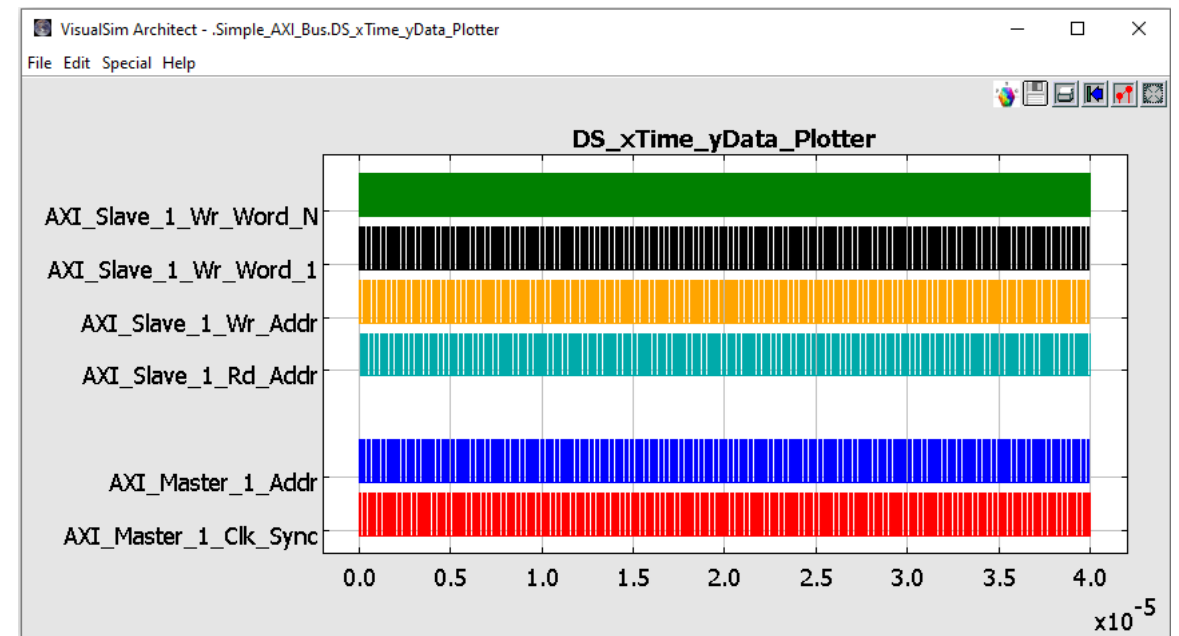
DISPLAY AT TIME          ----- 200.00000 us -----
{AXI_1_Slave_2_Rd_Threshold_Usage = 1.0,
AXI_1_Slave_Transactions          = 32,
DS_NAME                            = "AXI_1_Slave_2_Rd_Threshold"}

DISPLAY AT TIME          ----- 200.00000 us -----
{BLOCK                       = "AXI_Bus.AXI_Bus2.Master_1.Master",
DS_NAME                        = "Master_Transactions_1",
Number_Entered                 = 719,
Number_Exited                  = 716,
Occupancy_Max                  = 3.0,
Occupancy_Mean                 = 0.5519163763066,
Occupancy_Min                  = 0.0,
Occupancy_StDev                = 0.5492359152205,
Total_Delay_Max                = 3.285E-7,
Total_Delay_Mean               = 3.6423988842399E-8,

```

Enable Plot

- Ports_to_Plot parameter- { master number, slave number }
- Connect DS_xTime_yData_Plotter to *plot_out* port
- Configure The plotter to –
 - ✓ Field Trace Name - Plot_Name
 - ✓ Field_Y_Value - Plot_Value
 - ✓ Field_Color - Plot_Color
 - ✓ Field_Offset - Plot_Offset
- Plots the transaction between the Selected master and slave



Integrated Cache

Integrated Cache can be used as L1(Instruction and/or Data), L2 and L3 cache in both stochastic and cycle accurate mode.

Stochastic Mode:

- Hit or miss of input request will be determined by the instruction hit ratio/data hit ratio.
- If it is a hit, request will be processed and response will be returned to the source. If it is a miss, request will be sent to next level cache or memory to fetch whole block of data, while the request is waiting in the buffer,.

Address_Based Mode

- Hit or miss of the input request will be determined by the availability of the requested address in the cache.
- If it is a hit, request will be processed and response to the requested address will be returned. If it is a miss, whole block of address range will be fetched from next block of memory and the request waiting in the buffer will be processed.

Flow control:

- User can run the model either with flow control or without flow control. By default block will be used in without flow control.
- Input flow control can be achieved by including a field named "Event_Name" in the input data structure and a TIMEQ to trigger the next request. The next request will be triggered only when the data is processed by the cache.
- Output flow control can be achieved by setting the "Output_Flow_Control" parameter as true.



Integrated Cache Configuration

Key Configuration parameters:

- ✓ **Cache_Speed_Mhz:** Speed of the cache in Mega hertz, which is used to calculate the cycle time. Eg: 1000.0 (1GHz)
- ✓ **Cache_Width_Bytes:** Word size in cache, cache will transfer this bytes of data in single clock cycle. Eg: 8 (8 bytes or 64 bit)
- ✓ **Cache_Size_KB:** Overall cache size in Kilo Bytes Eg: 64
- ✓ **Block_Size_KB:** Cache will be organized as blocks of memory based on this size. Eg: 1
- ✓ **Loop_Ratio:** Loop ratio used in stochastic mode to represent repetition of same instruction fetching. Eg: 0.2 (between 0 to 1)
- ✓ **Overhead_Cycles:** Overhead delay of the cache block Eg: 1
- ✓ **First_Word:** If it is true the response will be returned, when the first word of the requested size is fetched. Otherwise, the response will be returned at the end of requested words

Integrated Cache Operation

- Command : Read_Instr
 - ✓ Read_instr is used only for instruction cache.
 - ✓ Check hit or miss, if hit, number of cycles will be delayed based on A_Bytes and send it out.
 - ✓ If it is a miss, it will put the request in the buffer, single cycle for request generation will be performed and send it to next level memory.
- Command: Read_Req / Write_Req
 - ✓ These commands are used only for data cache.
 - ✓ Check hit or miss, if hit number of cycles will be performed for read and write
 - If the cache is in write_through it will sends the data to the next level cache at the same time.
 - If the cache is in write back, it will just update the current cache block. If read request comes to the same address, whole block of data will be updated to the next level cache.
- Command: Read / Write
 - ✓ Cache will consider the request for the data cache and process the cycles of delay based on the A_Bytes.

Integrated Cache connection

Input Ports

- **to_cache**: Request going in to the cache, it can be connected to a processor or device interface through a bus.
 - **Expected fields**: A_Command, A_Source, A_Destination, A_Priority, A_Bytes, A_Task_Flag, A_I_Addr, A_D_Addr
- **fm_cache**: Response from the cache after the hit occurs in the data.

Output Ports

- **to_next_cache**: Request going to next level cache or memory in the case of a miss or an update(Write through or write back). It can be connected to next level memory through a bus.
 - **Expected Fields**: Cache_Event, Block_No, A_Command, A_Source, A_Destination, A_Priority, A_Bytes, A_Task_Flag, A_I_Addr, A_D_Addr
- **fm_next_cache**: Return Data from the next level memory.

Integrated Cache Statistics

Number of Statistics samples can be generated during the simulation time using the parameter “No_of_Statistics’

```

VisualSim Architect - .Cache_and_mem.L2_Cache_Statistics
----- 500.2795000 us -----
{A_Hit_Ratio           = 93.2203389830508,
A_Miss_Ratio          = 6.7796610169492,
A_Prefetch_Ratio      = 0.0,
A_Read_MBs            = 0.181248,
A_Read_MBs_per_Second = 362.4960072499201,
A_Total_MBs           = 0.193536,
A_Total_MBs_per_Second = 387.0720077414401,
A_Write_MBs           = 0.012288,
A_Write_MBs_per_Second = 24.57600049152,
BLOCK                 = "L2_Cache",
Buffer_Occupancy      = 0,
DS_NAME               = "Header_Only",
ID                    = 1,
Number_Entered         = 177,
Number_Returned       = 177,
TIME                  = 5.002795E-4, |
Utilization            = 12.1845002436899}
  
```

Integrated Cache Debug messages

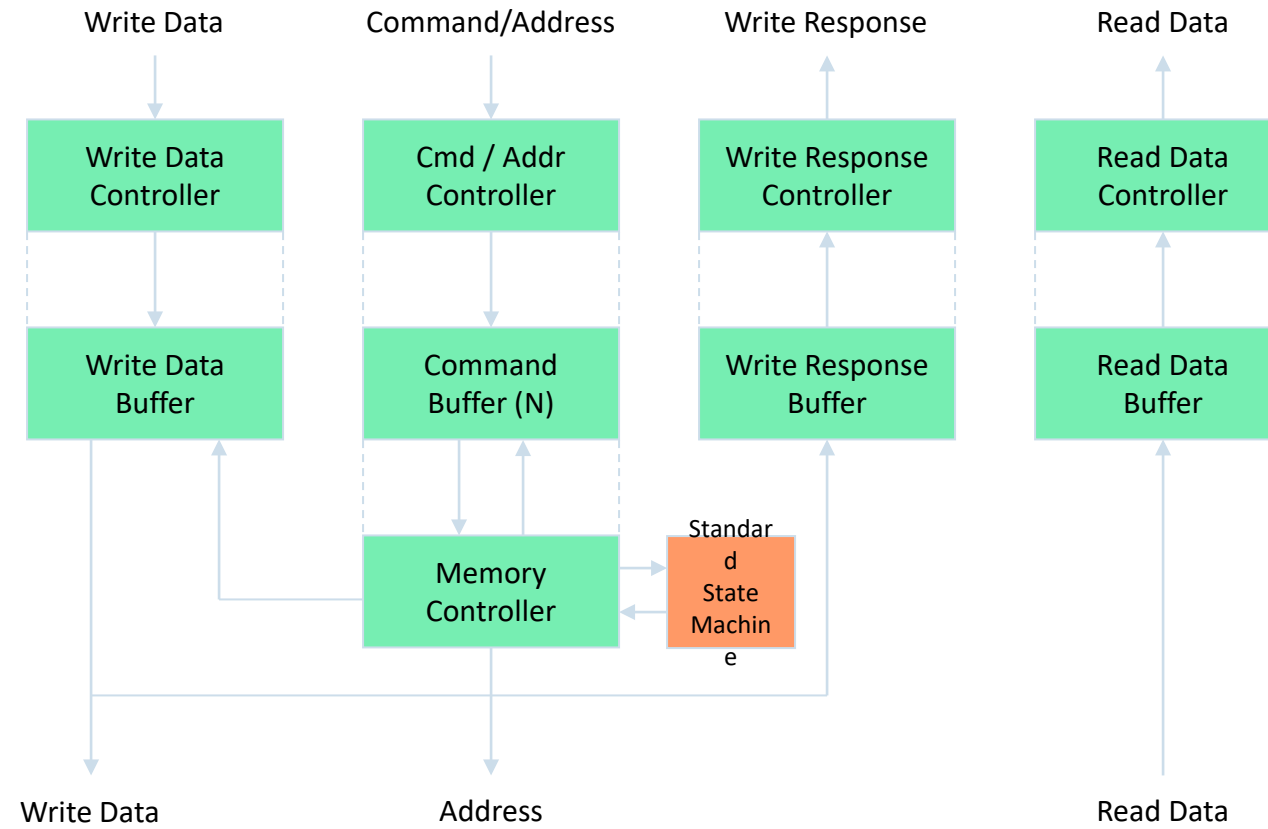
User RegEx Exception:

Block allocation in I_1 is incorrect, check Block_Configuration parameter

Cache size configuration should match with the number of blocks allocated.

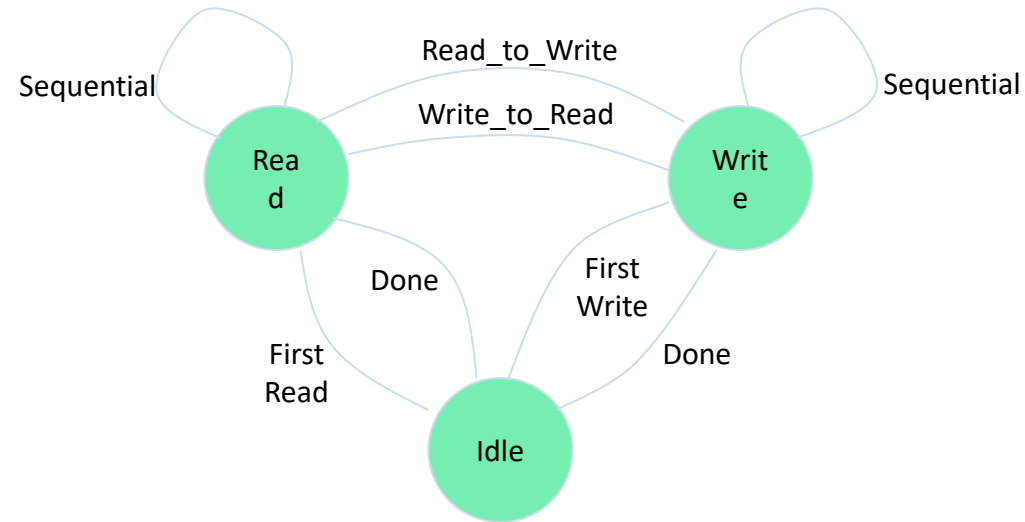
```
Block_Configuration:  /* Cache Configuration */
Cache_Allocation      No_of_Blocks  No_of_I_Blocks  No_of_D_Blocks  ;
Proc_1                16           {2,20}         {0,0}           ;"
```

Memory_Controller



Memory_Controller

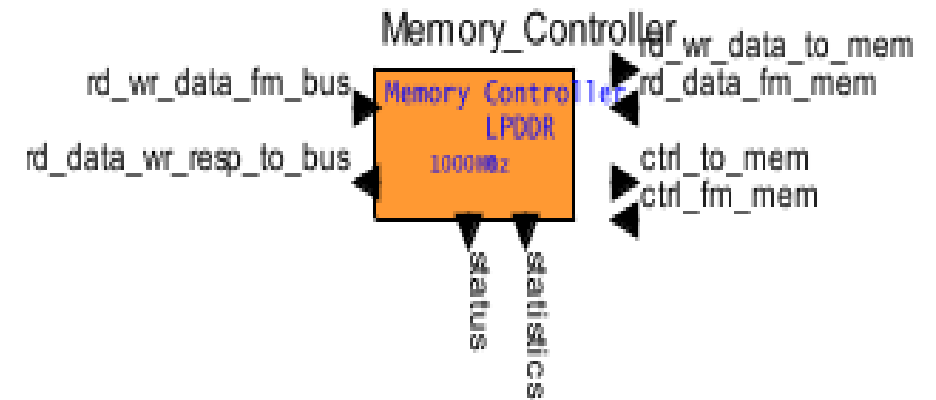
Algorithm State Machine



Consistent with JESD 209 LPDDR, Aug 2007

Memory Controller

- Oversees the operation and perform handshaking with the HWD RAM
- Oversees the sequential Read/Write operation and Holding the request when the DRAM is busy
- Data Structure Fields Used:
 - ✓ A_Address_Min
 - ✓ A_Address_Max
 - ✓ A_Command
 - ✓ A_Bytes
 - ✓ A_Bytes_Remaining
 - ✓ A_Bytes_Sent



Memory_Controller

- Constructed as 7 individual blocks for easy understanding
- Data Structure Fields used
 - ✓ A_Address_Min, A_Address_Max, A_Command
 - ✓ A_Bytes, A_Bytes_Remaining, A_Bytes_Sent
- Arbitration Algorithm
 - ✓ First Come-First Serve provided as standard algorithm
 - ✓ Custom algorithm add as a separate script file
- Processing Arrays (Accessible Externally)
 - ✓ Command Type, Bank and Page Address, Data Structures
- A_Command is suffixed with Memory transition
 - ✓ Read_Sequential, Write_Sequential, Read_Precharge, etc.

Statistics of Memory Controller

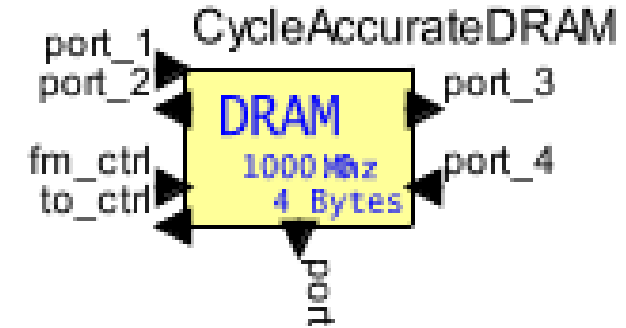
- Read_MBs_per_Second:
Total Read command processed by memory controller as number of Mega Bytes per second
- Read_IOs_per_Second:
Total IOs that the block can handle for one second
- Wr_IOs_Per_Second, Wr_MBs_per_Second

```

DISPLAY AT TIME          ----- 100.00 ns -----
{BLOCK                   = "LPDDR_Mem_Ctr1",
Command_Queue_Count     = 8,
Command_Queue_Max       = 1.0,
Command_Queue_Mean      = 0.5,
Command_Queue_Min       = 0.0,
Command_Queue_StDev     = 0.5,
DELTA                   = 0.0,
DS_NAME                  = "VM_Memory_Controller_Stats",
ID                       = 1,
INDEX                   = 0,
Read_Bytes              = 128,
Read_IO_Count           = 8,
Read_IOs_per_Second     = 2.9090909090909E8,
Read_MBs                = 1.28E-4,
Read_MBs_per_Second     = 4654.545454545454,
Read_Reoval_Position    = {8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
TIME                    = 1.0E-7,
Total_Bytes             = 128,
Total_Delay_Max         = 3.5E-9,
Total_Delay_Mean        = 1.5E-9,
Total_Delay_Min         = 5.0E-10,
Total_Delay_StDev       = 1.2247448713916E-9,
Total_Entered           = 8,
Total_Exited            = 8,
Total_IO_Count          = 8,
Total_IOs_per_Second    = 2.9090909090909E8,
Total_MBs               = 1.28E-4,
Total_MBs_per_Second    = 4654.545454545454,
Total_Refresh_Pct       = 0.0,
Total_Reoval_Position   = {8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
Write_Bytes              = 0,
Write_IO_Count          = 0,
Write_IOs_per_Second    = 0.0,
Write_MBs               = 0.0,
Write_MBs_per_Second    = 0.0,
Write_Reoval_Position   = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
  
```

Cycle Accurate_DRAM – Key Features

- Bank Address Decoder
 - ✓ Bank, Row, Column
- Page Address Decoder
- Timing accuracy
 - ✓ Extensive list of access types for read and Write
- Types of Read and Write
 - ✓ Sequential, non-sequential, Random Burst, Multi-row
- Refresh per Bank
 - ✓ Maximum time between refresh
 - ✓ Types: Sequential, After N Idle Cycles, Deadline
- Power using RegEx
- Statistics
 - ✓ Command Profile, Power Profile, Memory State Profile



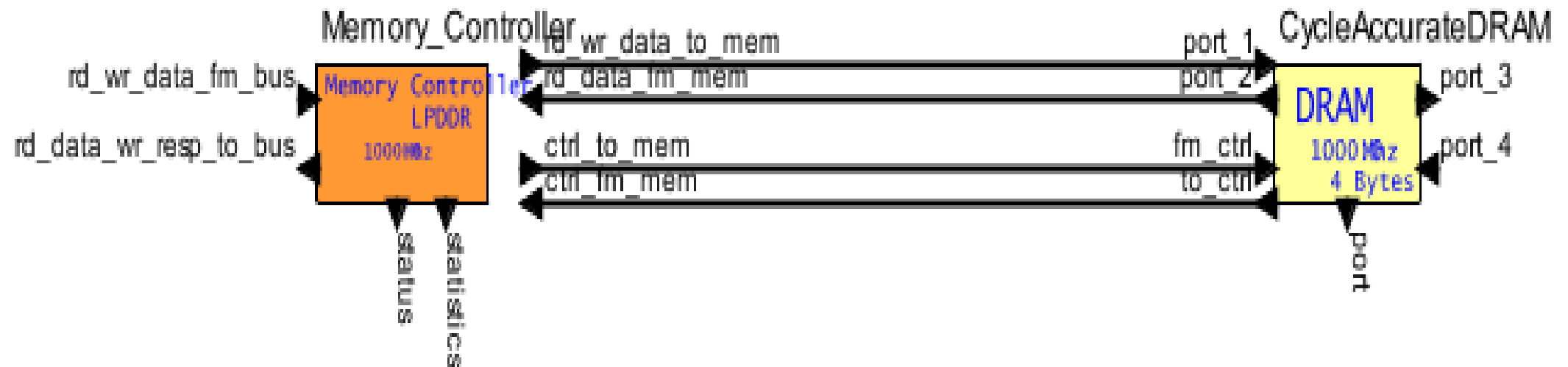
HW-DRAM Statistics

- Provides the number of cycles taken by each process in the memory block
- The port named “port” gives the statistics

```

DISPLAY AT TIME                ----- 1.0 ps -----
DRAM      Setup:
tRRD      Cycles: 9             Bank Select
tCL       Cycles: 18           Col Address Strobe
tRAS      Cycles: 32           Row Address Strobe
tRCD      Cycles: 18           Row Col Delay
tRP       Cycles: 18           Row Precharge
tRL       Cycles: 11           Read Latency
tWL       Cycles: 14           Write Latency
tWtR      Cycles: 4            Write to Read
DQSS Max  Cycles: 1            DQSS Max Cycles -- Bidirectional Data Strobe
DQSS Min  Cycles: 0            DQSS Min Cycles -- Bidirectional Data Strobe
tRTP      Cycles: 4
  
```

How to connect ?



Configuration

Memory Controller

Edit parameters for Memory_Controller

Architecture_Name:	"Architecture_1"
Controller_Name:	"LPDDR"
Controller_Speed_Mhz:	1000.0
Memory_Width_Bytes:	4
Bus_Width_Bytes:	4
Command_Buffer_Length:	8
Commands_in_a_Row:	8
Memory_Column:	{0,9}
Memory_Row:	{10,24}
Memory_Bank:	{25,28}
DRAM_Return_Cycles:	0
DEBUG:	true
_explanation:	Hardware_Modeling->Memory->Memory_Controller
Memory_Rank:	{29}
Burst_Length:	4 /* 2, 4, 8 */
First_Word_Flag:	true
HW_DRAM_Name:	"SDRAM"
Power_Manager_Name:	"none" /* Default */
Mfg_Suggest_Timing:	{16,16,16,32} /* tCL, tRCD, tRP, tRAS */
Extra_Timing:	{0,2,1,1,3,1,0,1,0,30} /* DQSS, tWTR, tRRD, tWR, tRL, tWL, tDQSK, tRTP, tHWpre, tFAW */
DDR4_Timing:	{6.0,4.0,5.0,6.0,4.0,5.0,30.0} /* CCD_L, CCD_S, CCD, RRD_L, RRD_S, RRD, FAW; units dks, except FAW ns */
Number_of_Samples:	10
DRAM_Type:	DDR4
Number_of_Ranks:	2
Number_of_Bank_Groups:	2
writeStats_to_File:	false

Commit Add Remove Restore Defaults Preferences Help Cancel

HW-DRAM

Edit parameters for CycleAccurateDRAM

Architecture_Name:	"Architecture_1"
HW_DRAM_Name:	"SDRAM"
HW_DRAM_Speed_Mhz:	1000.0
Sim_Time:	1.0E-05
_explanation:	Hardware_Modeling->Memory->HW_DRAM
Memory_Width_Bytes:	4
Burst_Length:	4 /* 2, 4, 8 */
Mfg_Suggest_Timing:	{16,16,16,32} /* tCL, tRCD, tRP, tRAS */
Extra_Timing:	{0,2,1,1,3,1,0,1,0,16} /* DQSS, tWTR, tRRD, tWR, tRL, tWL, tDQSK, tRTP, tHWpre, tFAW */
Retention_Time:	64.0E-03 /* 64.0 msec */
Enable_External_Data:	false
Address_Bit_Map:	{{0,9},{10,24},{25,27},{30}} /* col, row, bank, rank (min, max) Bit Position */
Standard_Name:	"none" /*reads DDR_Memory_Standards.txt */
Standard_File:	<input type="text"/> Browse
Power_Manager_Name:	"none" /* Default */
Memory_Controller:	"LPDDR" /* Default */
DEBUG:	false
State_Plot_Enable:	false
DRAM_Type:	DDR4
Fine_Granularity_Refresh:	FGR_1x
Fine_Granularity_Refresh_Time:	166.0E-09
REFpb_T_REFab_F:	true
Refresh_Statistical:	true

Commit Add Remove Restore Defaults Preferences Help Cancel

HW_DRAM- Timing Parameters

Name	Description
tCL	CAS Latency time
tRAS	Active to Precharge delay
tRP	DRAM RAS# Precharge
tRCD	DRAM RAS# to CAS# Delay
tWTR	Minimum time interval between end of WRITE and READ command
tWR	Minimum time interval between end of WRITE and PRECHARGE command
tRRD	Minimum time interval between successive ACTIVE commands to different banks
x Cycles	Time between Read to Read or Read to Write for non-consecutive or non-sequential access to the same row in a bank
tDQSS	Deviation from the clock for a Write operation
tRL	Read latency between the request and the actual output of the first bit
tWL	tWL=Write Latency between the request and the actual output of the first bit
tDQSCK	1
tRTP	Read to precharge
tHWPre	Time between the request and the actual precharge. This is a hardware limitation.

CycleAccurate_DRAM: Same Row Read Timing

CycleAccurate_DRAM: Random Read Timing

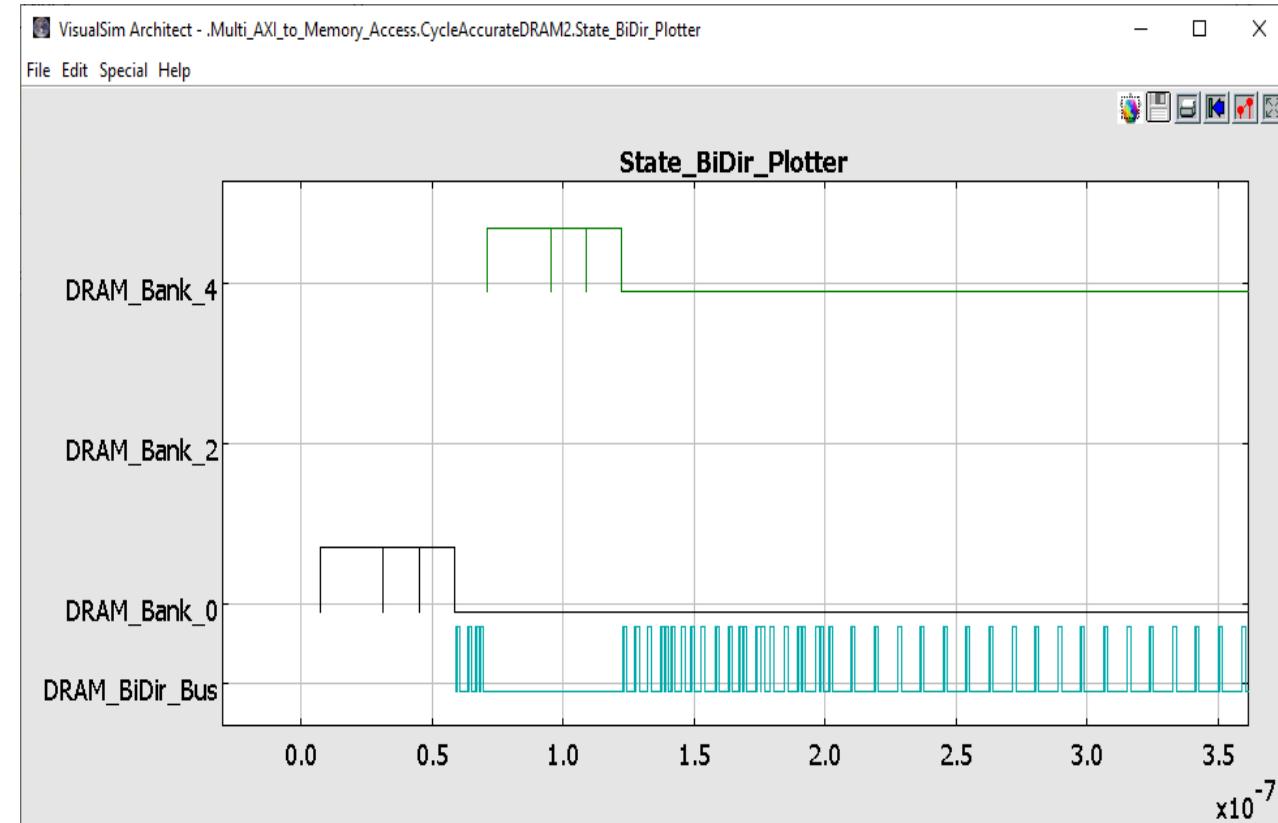
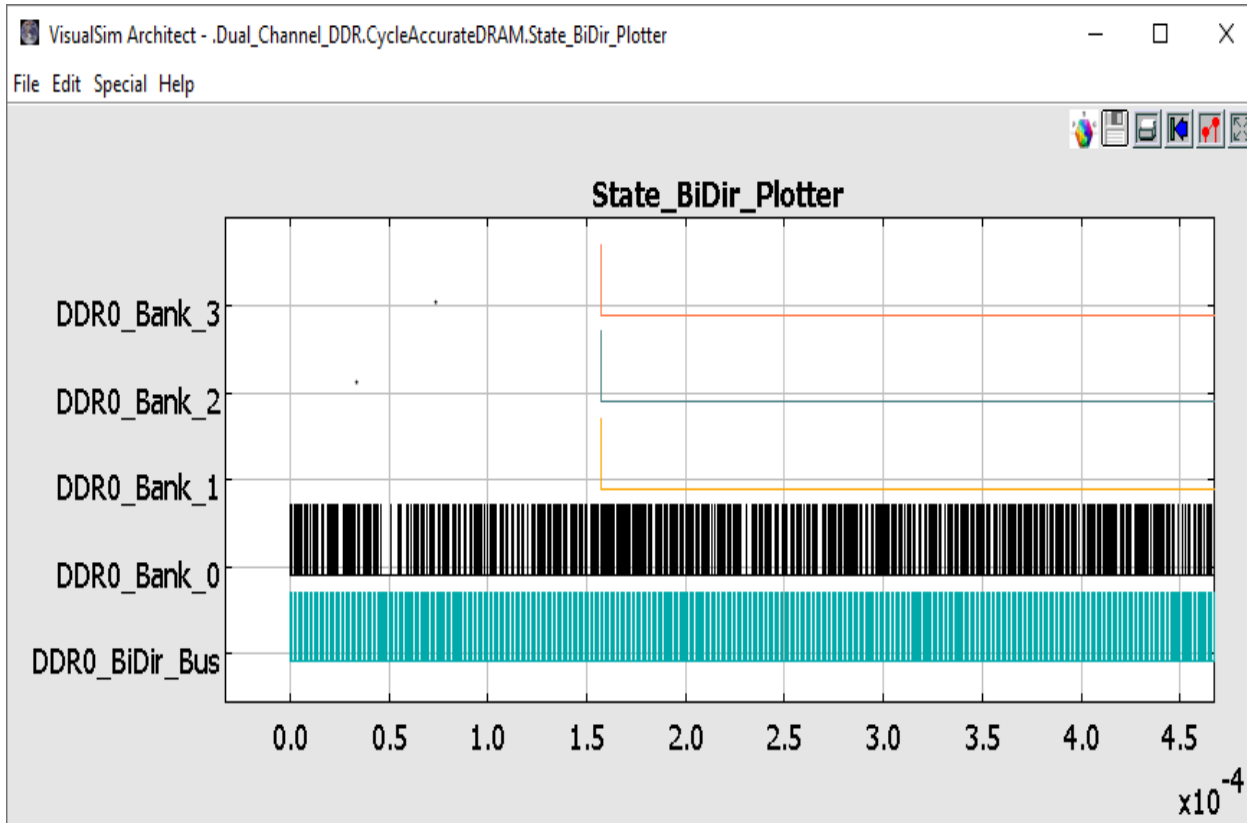
Type	Timing
1st Request at starting	Bus cycle + tRRD (if switched) + tRAS + M_Cycle * 1st word + M_Cycle * Remaining Word
Sequential or Consecutive Burst (same row)	Bus cycle + tRRD (if switched) + M_Cycle * 1st word + M_Cycle * Remaining Word
Non-Sequential or Non-Consecutive Burst (Same row but not immediate word)	Bus cycle + tRRD (if switched) + xCycles + tCL + M_Cycle * 1st word + M_Cycle * Remaining Word

Type	Timing
New Row Burst	Bus cycle + tRRD (if switched) + tRP (Active to Active) + tRAS + M_Cycle * 1st word + M_Cycle * Remaining Word
Multi-row or row crossing Burst	Bus cycle + tRRD (if switched) + xCycles + tCL (if non-sequential, else ignored) + M_Cycle * 1st word + M_Cycle * Remaining Word on current row + tRP (Active to Active) + tRAS + M_Cycle * Remaining Word
Read following a Write	Add tWTR

CycleAccurate_DRAM: Write Timing

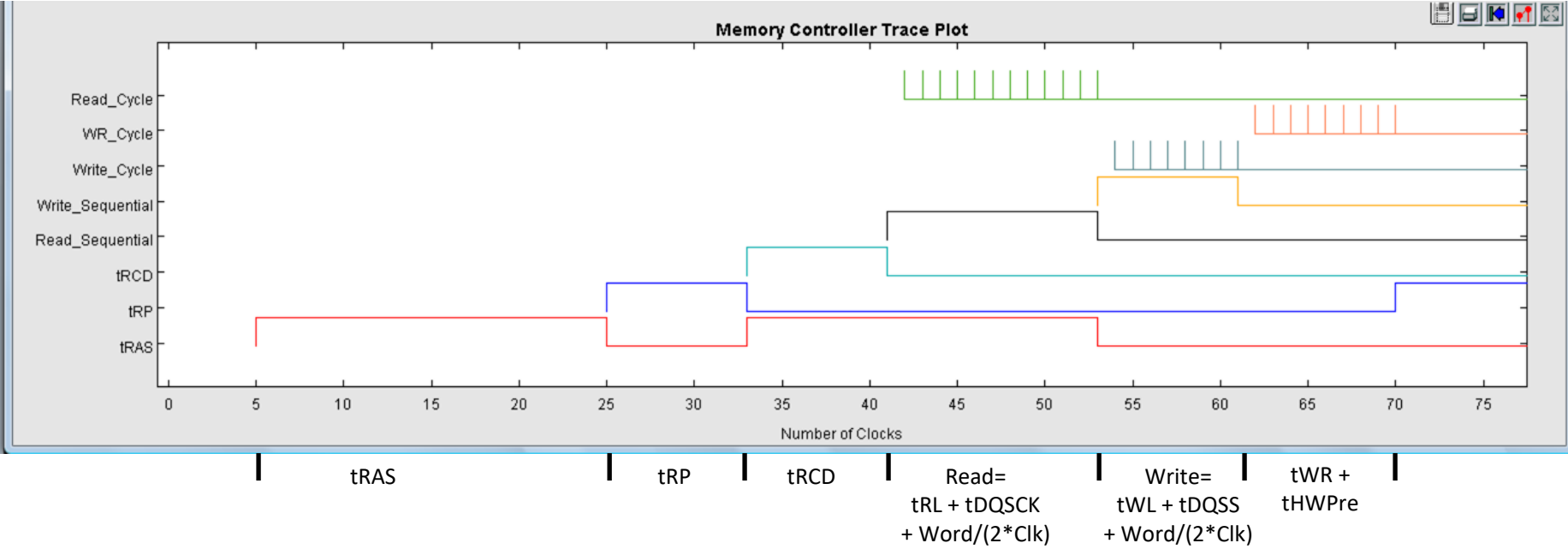
Type	Timing
Sequential or Consecutive Burst (Following another Write)	Bus cycle + tRRD (if switched) +DQSS (75% to 125% of 1 cycle) + M_Cycle * Number of Words
Non-Sequential or Non-Consecutive Burst (same row following Read or Write)	Bus cycle + tRRD (if switched) + xCycles + tCL +DQSS (75% to 125% of 1 cycle) + M_Cycle * Number of Words
Random Burst (Follow Write)	Bus cycle + tRRD (if switched) + tRAS + tRP (Active to precharge) + tWR (Write to Active) +DQSS (75% to 125% of 1 cycle) + M_Cycle * Number of Words
Random Burst (Follow Read)	Bus cycle + tRRD (if switched) + tRAS + tRP (Active to precharge) + DQSS (75% to 125% of 1 cycle) + M_Cycle * Number of Words

Timing Diagram



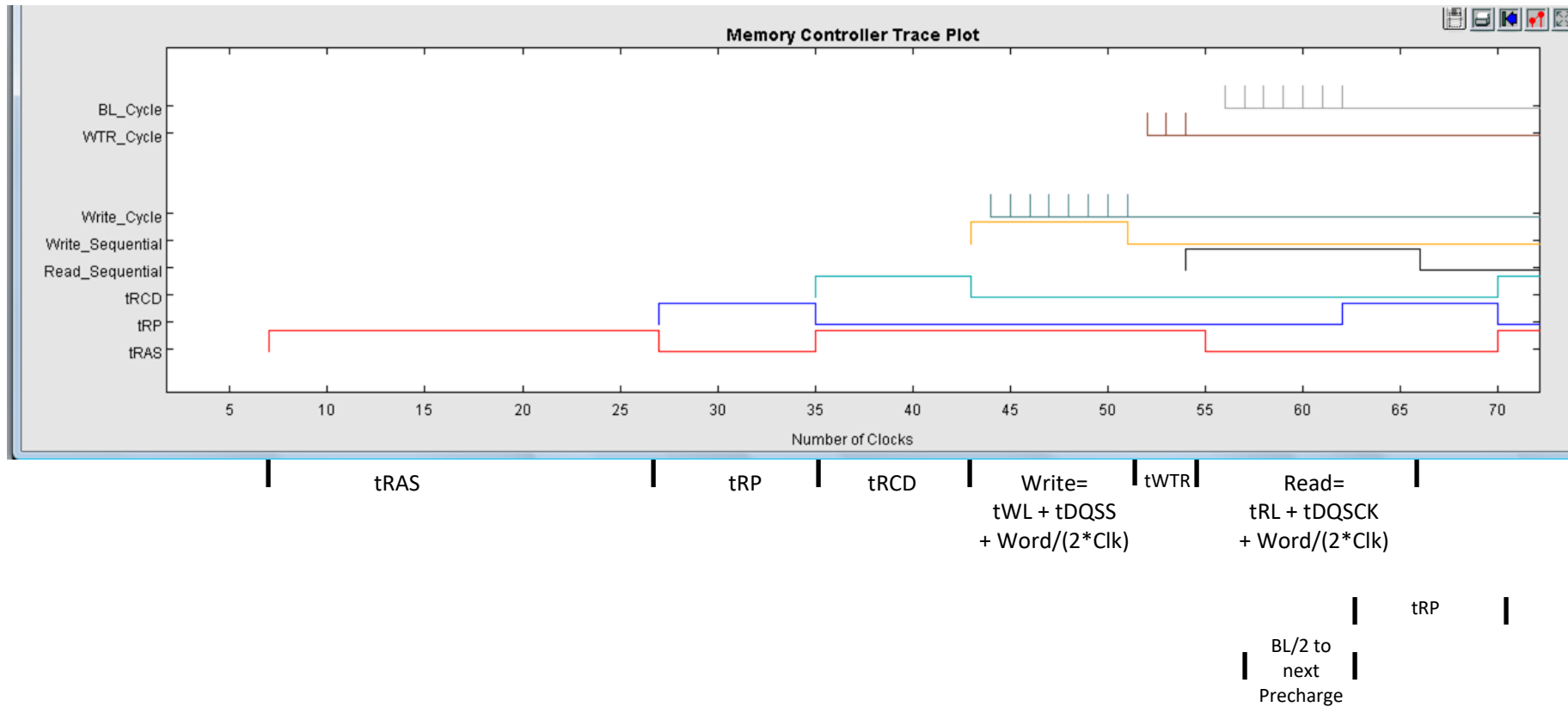
Memory_Controller Signals and Delays

Read following by Write

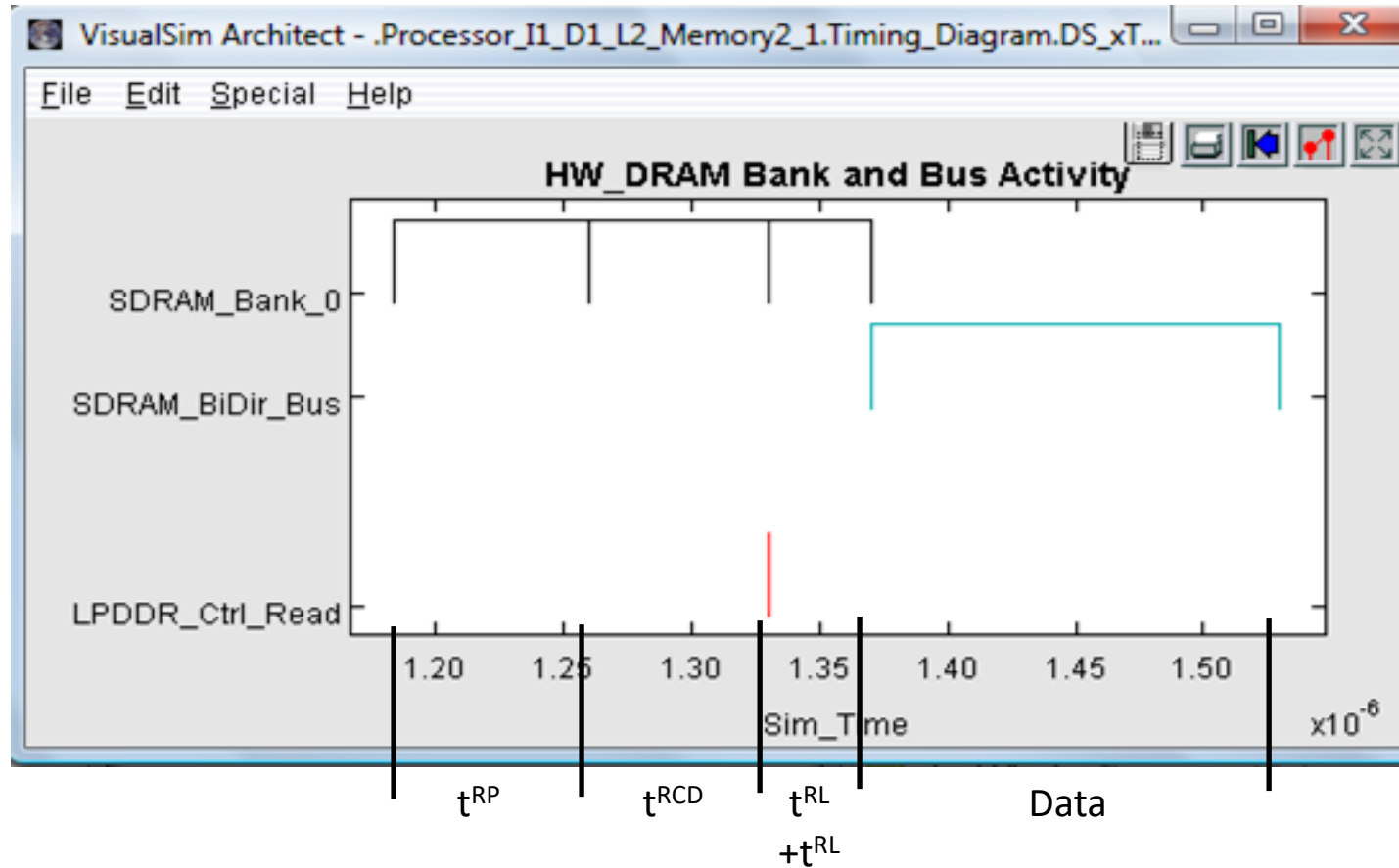


Memory_Controller Signals and Delays

Write following by Read



Memory_Controller, CycleAccurate_DRAM Banks and Bi-Directional Bus- Read Operation



Power Modeling

Purpose

Design with energy as one of the key decision factors

Get early feedback on the power requirements and system limitations

Set the requirements and golden references for the system implementation

Explore the power from end-to-end

Determine the system scalability with power as key consideration

VisualSim Power Technology

Comprehensive power solution

- Generation, storage, consumption and management

Graphical entry for all definitions

- States, transition and state change logic
- Complex expression that can incorporate switches, area, LDO efficiency, Frequency and Voltage
- Battery charging mode

Database of pre-defined battery data

Application at the system, sub-system and semiconductor levels

Variety of applications- automotive, multimedia, avionics and space, industrial and semiconductor

Reason for Early System Level Power Exploration

Power-based problems are one of the primary causes of costly re-spins

- Heat dissipation
- Low battery life
- Lackluster power-performance trade-off

Peak power consumption

- Determines the cooling, heat sink and other mechanical enclosure designs
- Energy usage, heat dissipation

Map the battery life and power consumed for variety of use cases and workloads

Role of overclocking and power consumption in failures

Understand where the power is being consumed in the system

Concept of VisualSim Power Technology

Based on system model activity and state change logic

Covers task-based power, transitions, management logic

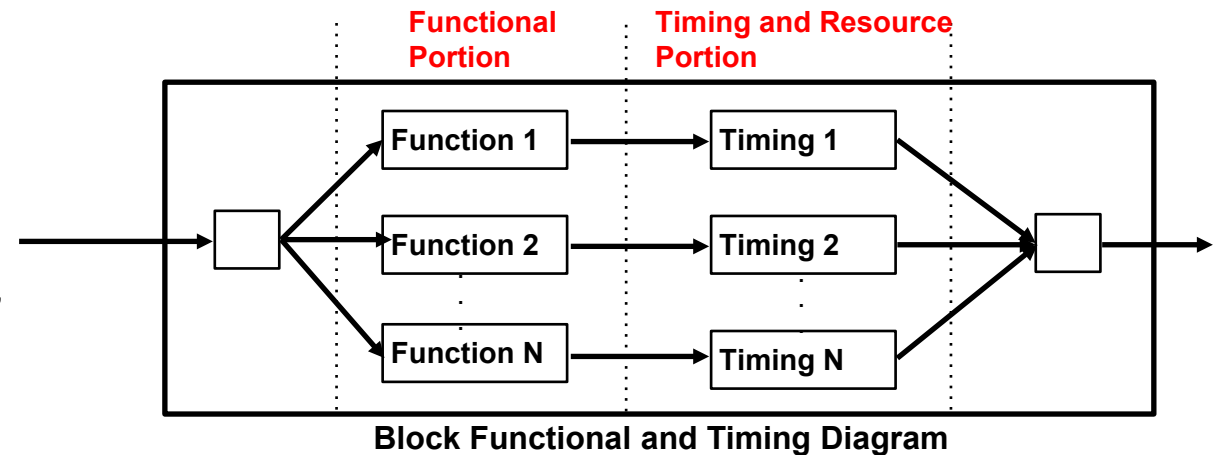
Incorporate the hardware, software and network

Looks at each of the entities in detail

- Generation from multiple sources- wind, solar, motor, steady, custom
- Storage- looks at various types of batteries
- Consumption at various rates by multiple devices and different clock speeds
- Management based on time and custom logic

Generate power profile for downstream test

Reports are average, instant, battery life, usage, comparison between input, available and consumed



Block Power Mode Diagram

Power Table

- Used to analyze the power consumption, battery discharge, dynamic system changes, power state changes of the devices which impacts the system timing.
- Enables to design application-based power schedulers and make trade-offs between performance and power consumption including battery drain.
- Outputs the instantaneous, average and State_Change information of top level and the other powerTable located in hierarchical blocks.



Library Blocks Supported

Library components

- System_Resource , System_Resouce_Extend
- Server
- Channel
- Custom Device- Single
- Custom Device-Multiple instances of same block

System-level libraries

- Processor
- Cache
- Bus Controller
- HW-DRAM, RAM
- AXI Bus , AHB/APB

Operation

- When the operation state of a device changes (idle, standby, wait, and busy), the power level goes to the new state
- There is a delay to go to the new state called State Transition delay. The transition delay is denoted by t_{OnOff}
- The power level of a state can be changed dynamically using asynchronous state change parameter or RegEX function
- Any number of devices can be associated with a single powerTable block

Configurations

Power Table configurations

- **PowerTable block-> Configure**

- Unique name for this Power Manager. Can be name + instance number

- **Manager_Setup** parameter :

- ✓ **Architecture block** : Lists the devices supported by power table

- ✓ **Power States:**

- ❖ **Active:** Power consumption when device is processing a task

- ❖ **Standby:** Power consumption when device is idle

- ❖ **Wait:** Power Consumption when device is waiting for a response

- ❖ **Idle:** Power consumption during Off state of the device

- ✓ **Operating State:**

- ❖ **Existing:** Initial state of the device

- ❖ **OffState:** Off state of the device

- ❖ **OnState:** Active/ON state of the device

- ✓ **State Transitions**

- ❖ **t_OnOff:** transition time delay from Active state to other

- ✓ **Parameters**

The screenshot shows the configuration window for a PowerTable block. It includes several sections:

- Block_Documentation:** A text area containing the note: "This is the Excel spreadsheet import. The power information is maintained here."
- Manager_Name:** A text field containing "Manager_1".
- FileURL:** A text field with a "Browse" button.
- Manager_Setup:** A large text area containing a table definition for "Power_Table". The table has columns for Device Name, Power States (Standby, Active, Wait, Idle), Operating States (Existing, OffState, OnState), State Transitions (t_OnOff, Mhz, Volts), and Speed/Existence flags. A sample row is provided for "Scheduler_SW".
- Table Definition:**

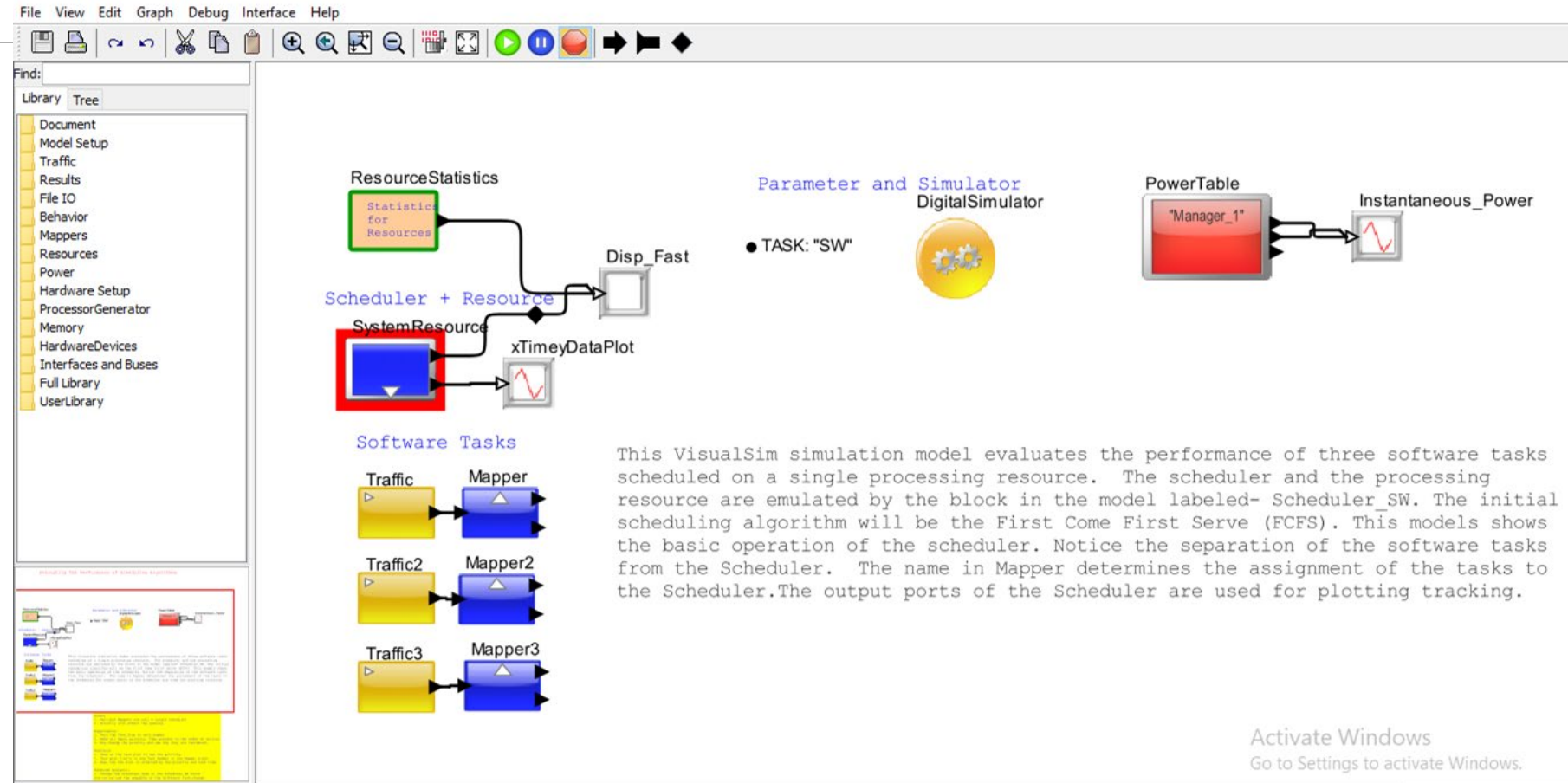
```
/* Power_Table. First row contains Column Names, expressions valid for entries except Device Name.
-----Device Name-----Power States-----Operating States-----State Transitions----- --Speed-- --Exist-- */
Architecture_Block  Standby Active Wait Idle Existing OffState OnState t_OnOff Mhz Volts ;
Scheduler_SW        10.0  4.0  1.0  0.0 Standby Standby Active 1.0E-6 1000.0 1.0 ;
```
- Table Definition:**

```
/* Async_State_Change. First row contains Column Names, expressions valid for entries except Device Name.
-----Device Name-----Time State----- */
Architecture_Block  State Time_or_Express Next ;
```
- Table Definition:**

```
/* First row contains Column Names.
-----Reference-----Expression----- */
Name Value ;
```
- Battery_Units:** A dropdown menu set to "Mill_Watts".
- State Plot Enable:** A checkbox that is currently unchecked.
- Buttons:** Commit, Add, Remove, Restore Defaults, Preferences, Help, Cancel.

Example model

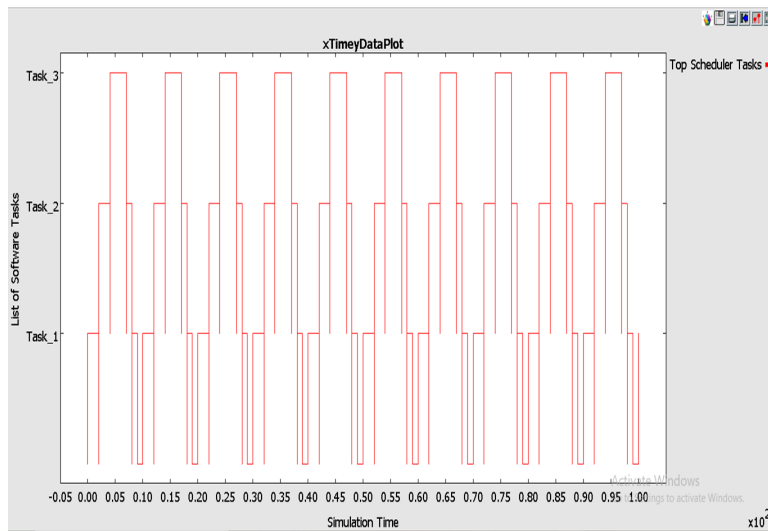
Three software task scheduled on single processing resource



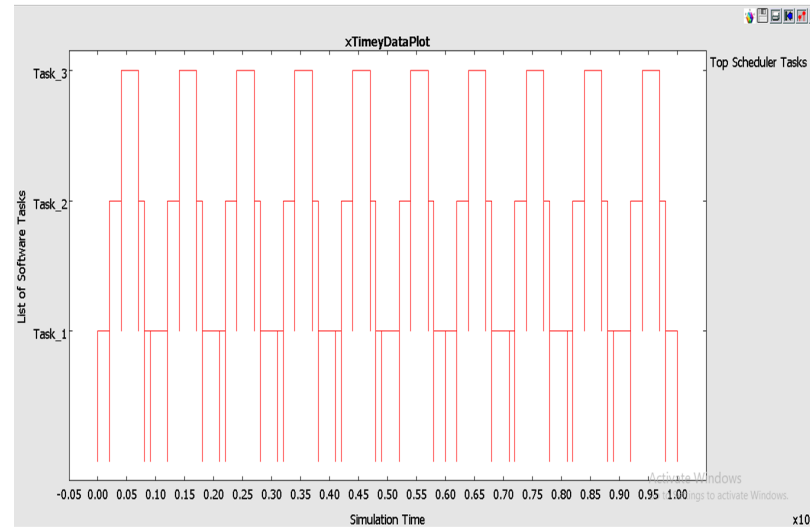
VisualSim plot

Top Scheduler tasks

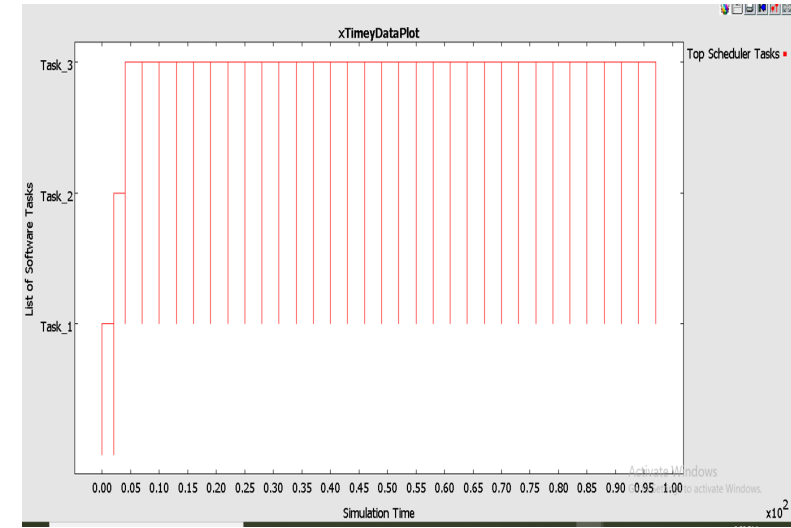
Traffic Rate= 10 sec



Traffic Rate= 5 sec

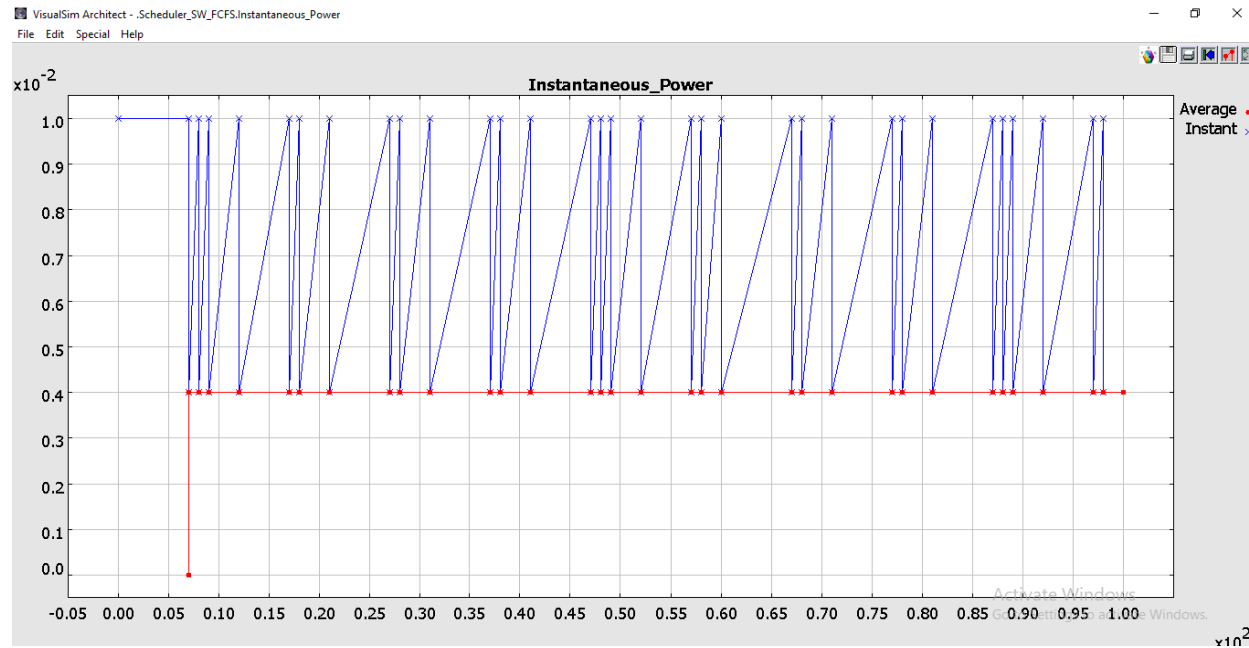


Traffic Rate= 3 sec

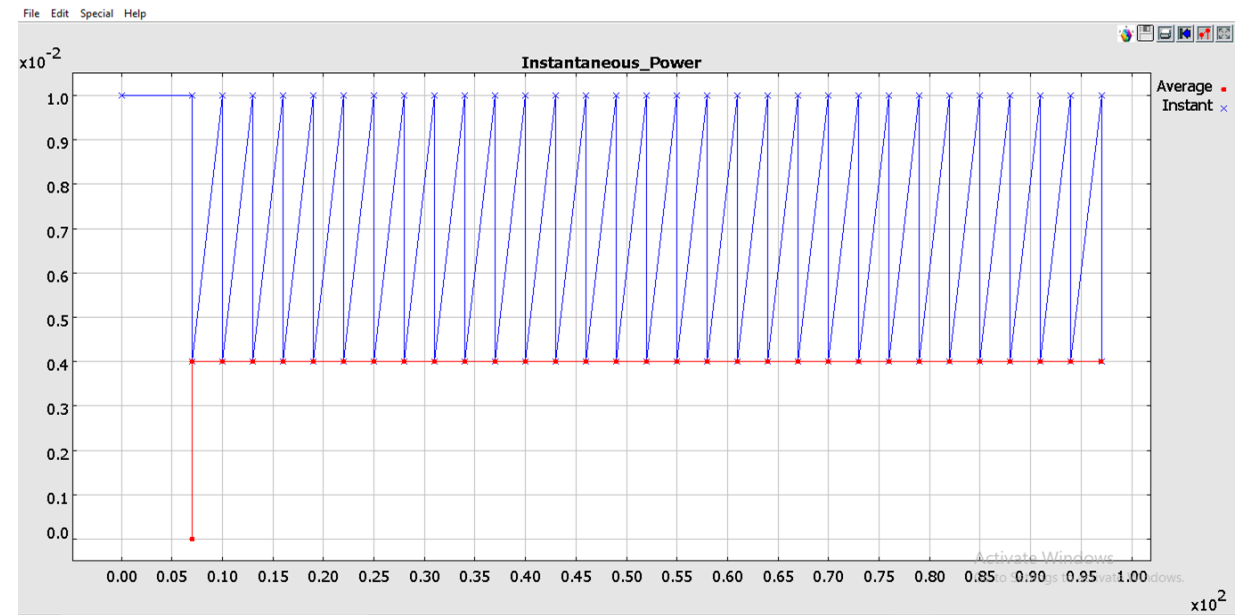


Power plots

Power plot with traffic rate= 10 sec



Power plot with traffic rate =3 sec



Power Table configurations continued

- Default states are Active, Standby and Wait. User can add any number of additional states. For adding new states: Power Table-> Manager_Setup-> Power states column-> add the additional state and its corresponding time. List of States must be prior to Existing.
- Async_State_Change window is used if the block changes its state asynchronously. The format is "<Device_Name> <State> <Time_or_Expression> <Next_State>".
- Expression_List can be used to define the logic and declare the values that can be used elsewhere in the PowerTable. The format for using Expression_List is "<Name> <value or expression> ;
- Similarly the user can set the configuration of other blocks by:
Right click on the block-> Customize-> Configure
- Doubt on the working on any block? Right click on the block-> Documentation-> Get Documentation

Power Modeling Usage

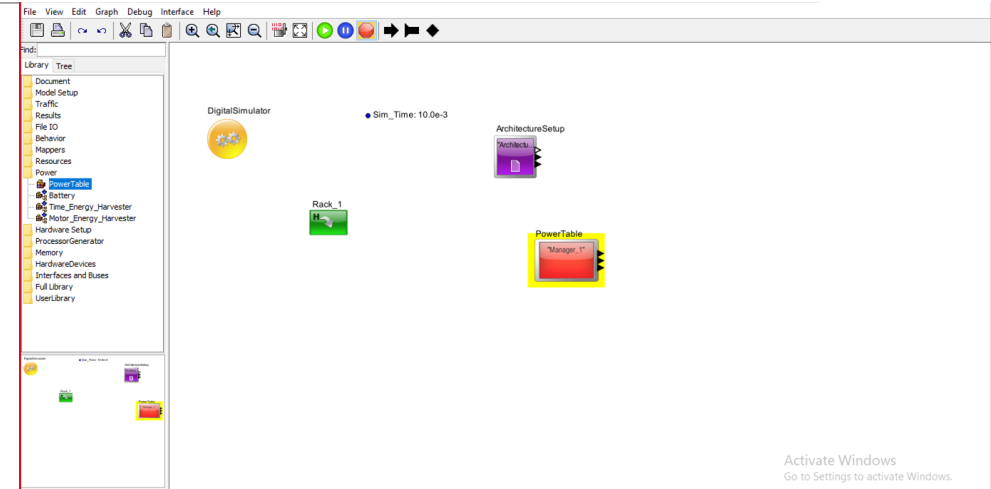
- Hardware library and schedulers
 - ✓ No model construction required
- Custom blocks including accelerators
 - ✓ Use the expression and variable to change the power values
 - ✓ Define custom power states
 - ✓ Use the Time_State to define movement from one state to another, especially for inactive states
 - ✓ Use the Power RegEx functions to change state or modify power level
- For model operations
 - ✓ Cycle-by-cycle table of instant power state of all devices (powerManager)

Example model for hierarchical power table

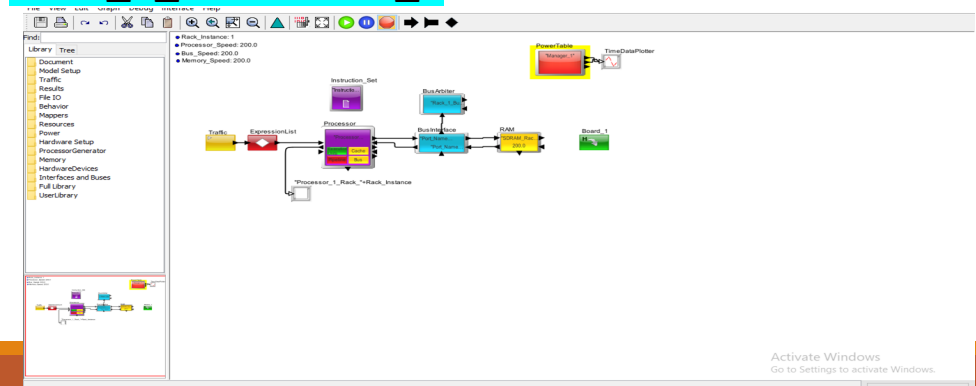
- Power Table can be built hierarchically
- Top Level Power table is connected to the Low level Power Tables.
For example, this top tower table will contain all the power tables in the below hierarchies.

These 3 levels can be viewed by:
 Rack 1-> open block
 board1-> open block
 section1-> open block

Top level of the model




Architecture_1_Processor_1_Rack_1_Board_1_Section_1_SubSection_1



Power table for hierarchical


- The processor, memory and bus are present in architectural block column since the model contains three devices. So each power table of each subsection will contain these three architectural blocks.
- By enabling the “state plot enable” parameter and the setting in digital simulator, it will create a GNU plot image, which contains much larger amount of information.

Edit parameters for PowerTable


Block_Documentation:  This is the Excel spreadsheet import. The power information is maintained here.

Manager_Name: *Manager_+Rack_Instance


fileOrURL:

Manager_Setup:  /* Power_Table. First row contains Column Names, expressions valid for entries except Device Name.

Device Name	Power States	Operating States	State Transitions	Speed	Exist
Architecture_Block	Standby	Active Wait Idle Existing	OffState OnState t_OnOff	Mhz Volts	;
Architecture_1_Processor_1_Rack_+Rack_Instance	sdv	300.0 0.0 0.0	Standby Standby Active 0.0	1000.0 1.0	;
Architecture_1_Rack_1_Bus	70.0	300.0 0.0 0.0	Standby Standby Active 0.0	1000.0 1.0	;
Architecture_1_SDRAM_Rack_1	70.0	300.0 0.0 0.0	Standby Standby Active 0.0	1000.0 1.0	;

Delay_to_Change_State:  /* Delay_to_Change_State. First row contains Column Names, expressions valid for entries except Device Name. where State to same State can extend a Power State

Device Name	Start	Expression	Next
Architecture_Block	State	Time	State
Architecture_1_Rack_1_Bus	Standby	1.0e-3	Idle

Expression_List:  /* First row contains Column Names.

Reference	Expression
Name	Value
Cycle_t	1.0E-6 / Mhz
sdv	70.0*1.5/2.5

Battery_Units: Milli_Watts

State Plot Enable:

Commit Add Remove Restore Defaults Preferences Help Cancel

Activate Windows
Go to Settings to activate Windows.

Power table for hierarchical continued

- The expression list row is useful for defining variables. For example, when the processor has to go to standby state, a variable name is found in the standby column. The value of this variable “ sty” is read from the expression list.
- The “delay to change state” row is for power management. For example the Bus is in standby state for more than 1 ms, then it is put to Idle state. The “state” column gives the state in which the block is present. The “Time” column tells how long the bus can stay in standby state, if the time is exceeded, then the “Next” column gives to which state the bus will be going to. For example here it is Idle state.

Statistics

- Instantaneous Power (port)
 - This outputs the instantaneous power of all the devices listed in the Manager_Setup field.
- Average Power consumed (port)
 - This outputs the average power consumed so far during the simulation.
- Power Dissipated (port)
- Instant (powerCurrent) and total power consumed (powerCumulative) by device
 - Cumulative- This is the accumulation of all the energy from all the devices listed in the PowerTable's Manager_Setup table.

Listen To Block

```
VisualSim Architect - .Multi-Core_DVFS_Design.PowerTable2
File Help
devNames (4) = [Scheduler_0, Scheduler_1, Scheduler_2, Scheduler_3]
Column (Architecture_Block) = "Scheduler_0"
Column (Standby) = "Standby_Power"
Column (Active) = "Active_Power"
Column (wait) = 0.0
Column (Idle) = 0.0
Column (Existing) = "Standby"
Column (OffState) = "Standby"
Column (OnState) = "Active"
Column (t_OnOff) = 2.0E-9
Column (Mhz) = 1000.0
Column (Volts) = 1.0
colNames (Scheduler_0) = [Standby, Active, Wait, Idle, Existing, OffState, OnState, t_OnOff, Mhz, Volts, c_Standb
+Express (Scheduler_0) = [Standby_Power, Active_Power, null, null, Standby, Standby, Active, null, null, null, nu
Column (Architecture_Block) = "Scheduler_1"
Column (Standby) = "Standby_Power"
Column (Active) = "Active_Power"
Column (wait) = 0.0
Column (Idle) = 0.0
Column (Existing) = "Standby"
Column (OffState) = "Standby"
Column (OnState) = "Active"
Column (t_OnOff) = 0.004
Column (Mhz) = 1000.0
Column (Volts) = 1.0
colNames (Scheduler_1) = [Standby, Active, Wait, Idle, Existing, OffState, OnState, t_OnOff, Mhz, Volts, c_Standb
+Express (Scheduler_1) = [Standby_Power, Active_Power, null, null, Standby, Standby, Active, null, null, null, nu
Column (Architecture_Block) = "Scheduler_2"
Column (Standby) = "Standby_Power"
Column (Active) = "Active_Power"
Column (wait) = 0.0
Column (Idle) = 0.0
Column (Existing) = "Standby"
Column (OffState) = "Standby"
Column (OnState) = "Active"
Column (t_OnOff) = 6.0E-9
Column (Mhz) = 1000.0
Column (Volts) = 1.0
colNames (Scheduler_2) = [Standby, Active, Wait, Idle, Existing, OffState, OnState, t_OnOff, Mhz, Volts, c_Standb
+Express (Scheduler_2) = [Standby_Power, Active_Power, null, null, Standby, Standby, Active, null, null, null, nu
Column (Architecture_Block) = "Scheduler_3"
Column (Standby) = "Standby_Power"
Column (Active) = "Active_Power"
Column (wait) = 0.0
Column (Idle) = 0.0
Column (Existing) = "Standby"
Column (OffState) = "Standby"
Column (OnState) = "Active"
Column (t_OnOff) = 2.0E-8
Column (Mhz) = 1000.0
Column (Volts) = 1.0
colNames (Scheduler_3) = [Standby, Active, Wait, Idle, Existing, OffState, OnState, t_OnOff, Mhz, Volts, c_Standb
+Express (Scheduler_3) = [Standby_Power, Active_Power, null, null, Standby, Standby, Active, null, null, null, nu
Scheduler_0 = {Active="Active_Power", Architecture_Block="Scheduler_0", Average=0.0, Cumulative=0.0, C
Scheduler_1 = {Active="Active_Power", Architecture_Block="Scheduler_1", Average=0.0, Cumulative=0.0, C
Scheduler_2 = {Active="Active_Power", Architecture_Block="Scheduler_2", Average=0.0, Cumulative=0.0, C
Scheduler_3 = {Active="Active_Power", Architecture_Block="Scheduler_3", Average=0.0, Cumulative=0.0, C
wrapTime = 0.0319999999
Existing Scheduler_0 (Standby)
On State Scheduler_0 (Active)
Off State Scheduler_0 (Standby)
Set Power Scheduler_0 (Standby) = 1.0E-5
Existing Scheduler_1 (Standby)
On State Scheduler_1 (Active)
Off State Scheduler_1 (Standby)
Set Power Scheduler_1 (Standby) = 1.0E-5
Existing Scheduler_2 (Standby)
On State Scheduler_2 (Active)
Off State Scheduler_2 (Standby)
Set Power Scheduler_2 (Standby) = 1.0E-5
Existing Scheduler_3 (Standby)
On State Scheduler_3 (Active)
Off State Scheduler_3 (Standby)
Set Power Scheduler_3 (Standby) = 1.0E-5
Total Initial Power = 4.0E-5
updatePower (Scheduler_0, Active) @ 4.0E-8
New Async Scheduler_0 (Standby=>Active) = 0.0 @ 4.0E-8
Set State Scheduler_0 (Active) = 0.0 @ 4.0E-8
```

PowerTable RegEx Function

Function & Argument Type(s)	Description	Example
powerCumulative (String power_manager_name, String block_name)	Gets the cumulative power consumed for a device.	powerCumulative ("ARM_Power_Manager", "Architecture_1_Bus_1")
powerCurrent (String power_manager_name, String block_name)	Gets the instantaneous power as a double value for the device.	powerCurrent ("ARM_Power_Manager ", "Architecture_1_Bus_1")
powerManager (String power_manager_name)	Gets the complete power table.	powerManager ("ARM_Power_Manager ")
powerUpdate (String power_manager_name, String block_name, String power_state)	Updates the current power state of the block. LHS value is the new power state of the block.	powerUpdate ("ARM_Power_Manager ", "Architecture_1_Bus_1", "Standby")
powerUpdateN (String power_manager_name, String block_name, String power_state, integer Queue_Number)	Updates the current power state of the Smart_Timed_Resource block. LHS value is the new power state of the block.	powerUpdateN ("ARM_Power_Manager ", "STR_Queue", "Standby",2)

- **powerManager()**

- ✓ Gives the power statistics for all the details associated with the Power Table as an array

```
DISPLAY AT TIME          ----- 10.10 ns -----
{Architecture_1_AHB_Bus   = {c_Wait = 0.0, Cumulative = 2.5E-10, Average = 0.025, Time = 1.0E-8, Standby = 25.0, OffState = "Standby", Architecture_Block =
"Architecture_1_AHB_Bus", Active = 100.0, Idle = 0.0, NewState = "Active", c_Standby = 2.5E-10, c_Active = 0.0, OnState = "Active", ID = 1, Wait = 0.0, Volts = 1.0,
c_Idle = 0.0, Current = 0.1, t_OnOff = 0.0, Mhz = 1000.0, Existing = "Active"},
Architecture_1_DMA       = {c_Wait = 0.0, Cumulative = 0.0, Average = 0.0, Time = 0.0, Standby = 50.0, OffState = "Standby", Architecture_Block =
"Architecture_1_DMA", Active = 150.0, Idle = 0.0, NewState = "Standby", c_Standby = 0.0, c_Active = 0.0, OnState = "Active", ID = 0, Wait = 0.0, Volts = 1.0, c_Idle =
0.0, Current = 0.05, t_OnOff = 0.0, Mhz = 1000.0, Existing = "Standby"},
Architecture_1_DRAM      = {c_Wait = 0.0, Cumulative = 0.0, Average = 0.0, Time = 0.0, Standby = 50.0, OffState = "Standby", Architecture_Block =
"Architecture_1_DRAM", Active = 150.0, Idle = 0.0, NewState = "Standby", c_Standby = 0.0, c_Active = 0.0, OnState = "Active", ID = 0, Wait = 0.0, Volts = 1.0, c_Idle =
0.0, Current = 0.05, t_OnOff = 0.0, Mhz = 1000.0, Existing = "Standby"},
Architecture_1_MAC_ARM9  = {c_Wait = 0.0, Cumulative = 1.0E-9, Average = 0.1, Time = 1.0E-8, Standby = 75.0, OffState = "Standby", Architecture_Block =
"Architecture_1_MAC_ARM9", Active = 200.0, Idle = 100.0, NewState = "Active", c_Standby = 0.0, c_Active = 0.0, OnState = "Active", ID = 3, Wait = 100.0, Volts = 1.0,
c_Idle = 1.0E-9, Current = 0.2, t_OnOff = 0.0, Mhz = 1000.0, Existing = "Active"},
total                    = {Average = 0.225, Cumulative = 2.25E-9, Time = 1.0E-8, Current = 0.4}}
```

- **stateChange()**

- ✓ *stateChange(power_manager_name, Device name, Operating State, New State Name)*
- ✓ Dynamically changes the state of a device to a new state
- ✓ It is similar to the function of Delay_to_Change_State parameter

Battery

Used to capture

- Rate of consumption
- Impact of continuous charging
- Lifecycle loss due to power spikes and thermal shock
- (Experimental) Heat and Temperature

Types of batteries support

- Battery database support NiCd, Li-Ion, NiMh, LdAcid

Activities modeled

- Charging- SOC threshold, Turbo charge, all input charge
- Discharge- From the PowerTable
- Lifecycle, discharge

Battery Block

Battery_Name:	"Battery_1"																
BatteryProfileFile:	Battery_Database_Battery_1.txt																
Battery_Selection:	Li-ion																
customCharging:	<input type="checkbox"/>																
SOC:	80.0/*in percentage*/																
TurboCharge:	<input type="checkbox"/>																
Turbo_Charger_Table:	<table border="1"><thead><tr><th>ID</th><th>percentage</th><th>ChargeHour</th><th>;</th></tr></thead><tbody><tr><td>"1"</td><td>25</td><td>15</td><td>;</td></tr><tr><td>"2"</td><td>50</td><td>30</td><td>;</td></tr><tr><td>"3"</td><td>100</td><td>120</td><td>;</td></tr></tbody></table>	ID	percentage	ChargeHour	;	"1"	25	15	;	"2"	50	30	;	"3"	100	120	;
ID	percentage	ChargeHour	;														
"1"	25	15	;														
"2"	50	30	;														
"3"	100	120	;														
plot:	<input type="checkbox"/>																
SimTime:	100.0e-3																
resistance:	1.2e-3 /* in Ohm */																
Current:	1.35 /* in A */																
Temp_Coeff:	0.000038																
Temperature:	25																
Heat_Capacity:	5700																

Requires Unique name

Default database provided

Select battery type

Default charging is the value arrive. Click to Custom State of Charge for charging to resume.

Enable Turbo charging

Time taken to get to each level of charge. Similar to phone battery chargers

Experimental; For Temperature and heat computation

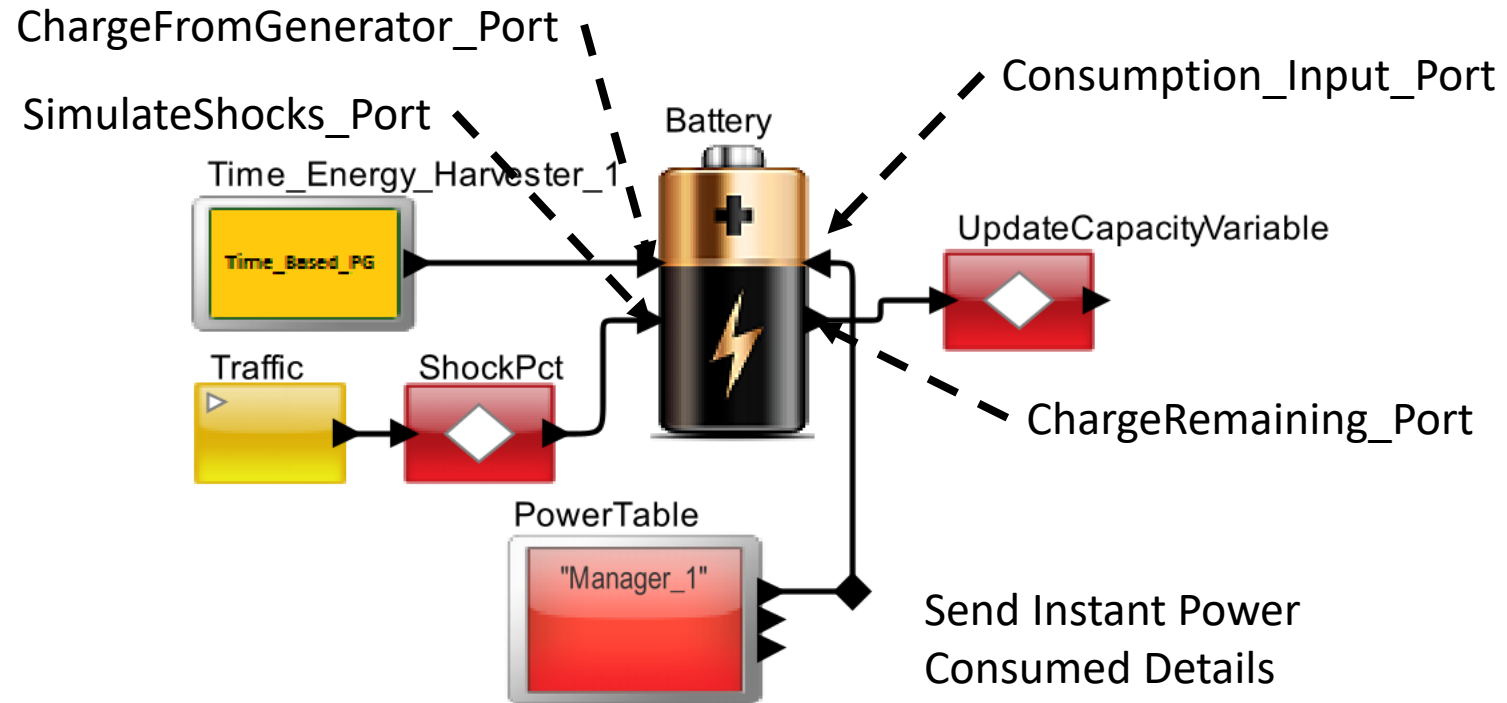
Note:

Power generated when not charging is wasted.

User can add items to the database

Edit the instance to add the new battery types

Block Ports



Reports

Heat Display- experimental

Input charge- Total charge arriving vs charge consumed

Peak Energy Available- Design capacity on total capacity; Total Charge Capacity is the available charge

Power Consumed- Ignore

Battery Life Remaining Percentage- Reduction in battery life overtime as a result of spikes, thermal shock

Available Battery Charge- Ignore

Power Generator

Time-Energy

- Constant Power Source
- File Based
- Time Based

Motor-Energy

- Motor Based Power Generation
- Wind Based Power Generation


Time-Energy Settings

UseTraceFile: false /* boolean enables mode */

Trace_File_Name: /* file name */

UseTimeBased: false /* boolean enables mode */

Time_Based_Duration: 10.0E-03 /* time seconds */

Time_Based_Charge_Setup: 

/* Time-Based Charge Profile */					
ID	StartWHR	EndWHR	Efficiency	PercentTime	
1	0.0	20.0	100.0	15.0	;
2	20.0	20.0	100.0	50.0	;
3	20.0	0.0	100.0	15.0	;
4	0.0	0.0	100.0	20.0	;

UseConstant: true /* boolean enables mode */

ConstantChargeCapacity: 500.0 /* charge rate in Watt-Sec */

SimTime: 1.0

Using Trace file from existing system

Using Time-based

1. Duration is the period
2. Setup has % of time in the Period of each time
3. Charge can increase or reduce during the period
4. Efficiency is amount of charge converted

Constant output at the set rate

Motor-Energy Settings

Use_Wind_Turbine:	false
Wind_Turbine_Setup:	<pre>/* wind_Turbine Charge Profile */ ID Duration Speed Efficiency ; 1 4.0 3.728 95.0 ; 2 4.0 4.970 97.0 ; 3 4.0 5.592 100.0 ; 4 4.0 6.213 100.0 ; 5 4.0 4.970 97.0 ; 6 4.0 4.319 95.0 ;</pre>
r:	1.0/*Rotor Radius in meter*/
row:	1.225/*air Density*/
k:	0.000133/*Constant*/
Cp:	0.40/*Maximum Power Coefficient*/
Use_Motor_Generator:	true
Motor_Charge_Setup:	<pre>/* Motor-Generator Charge Profile */ ID RPM Duration ChargeCapacityWHR Efficiency ; 1 0 10.0 0.0 100.0 ; 2 2500 10.0 55.0 100.0 ; 3 2500 10.0 55.0 100.0 ; 4 4000 10.0 105.0 100.0 ; 5 4500 10.0 110.0 100.0 ; 6 4000 10.0 105.0 100.0 ; 7 2500 10.0 55.0 100.0 ; 8 2500 10.0 55.0 100.0 ; 9 0 10.0 0.0 100.0 ;</pre>
SimTime:	1.0

Compute the Wind Power based on air speed and efficiency

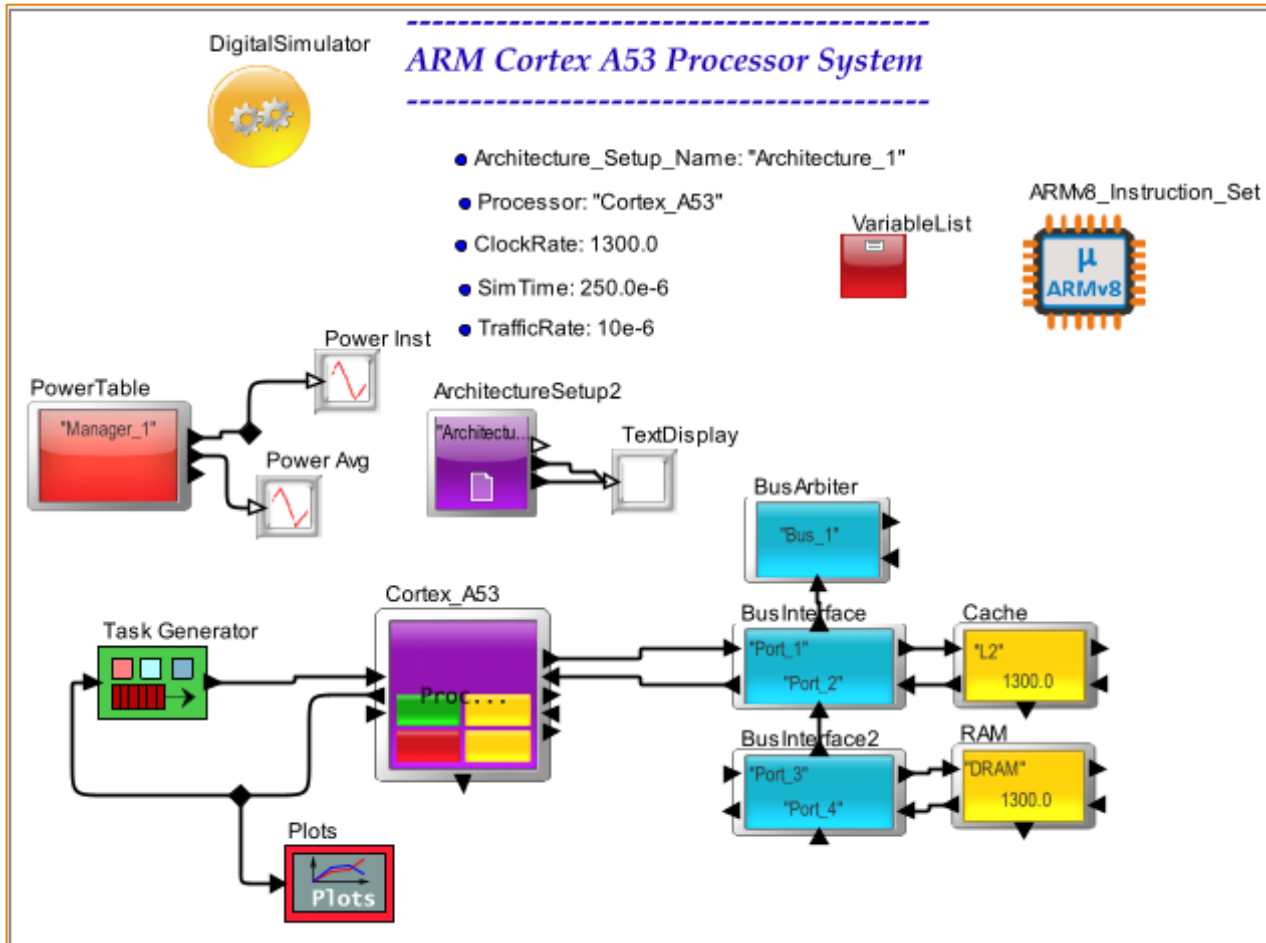
Motor based on RPM

POWER ANALYSIS AND MANAGEMENT



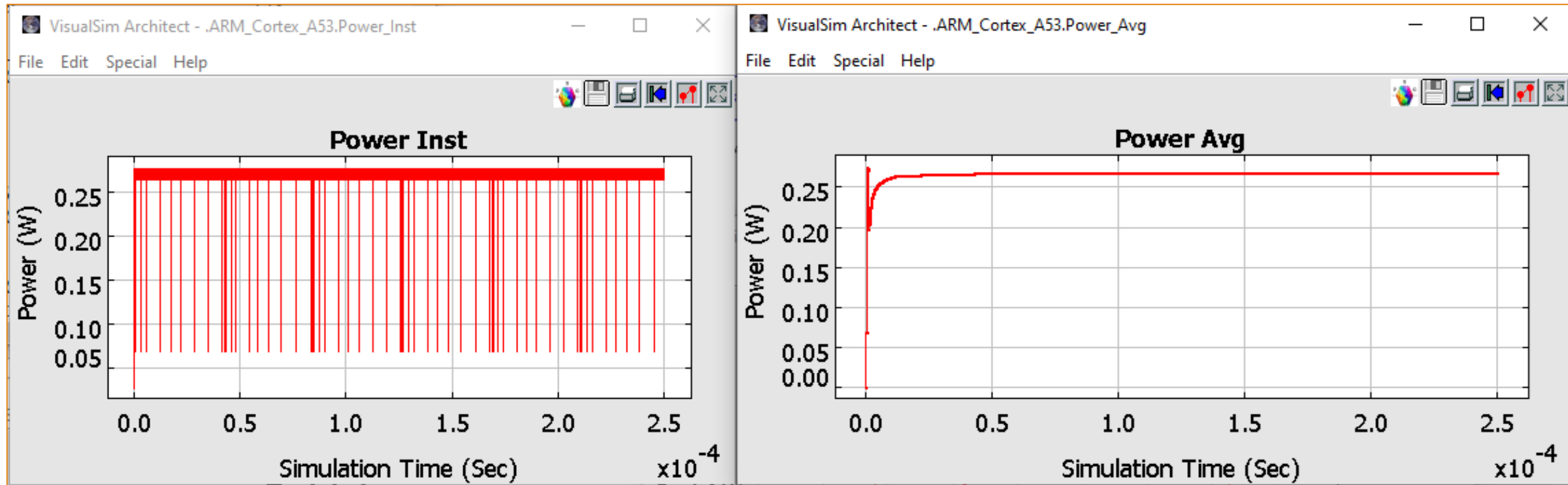
COMPARISON

VisualSim Model of a Single Core A53



- Clock Rate = 1300 Mhz
- Pipeline type = In-Order
- Pipeline Stages = 7
- Cache = 64 KBytes of I-Cache and
- D-Cache = 256 KBytes of L2 Cache

VisualSim Power Plots



Results comparison

Frequency	Max Power observed	Real System 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%

2 cases of Power calculation within VisualSim

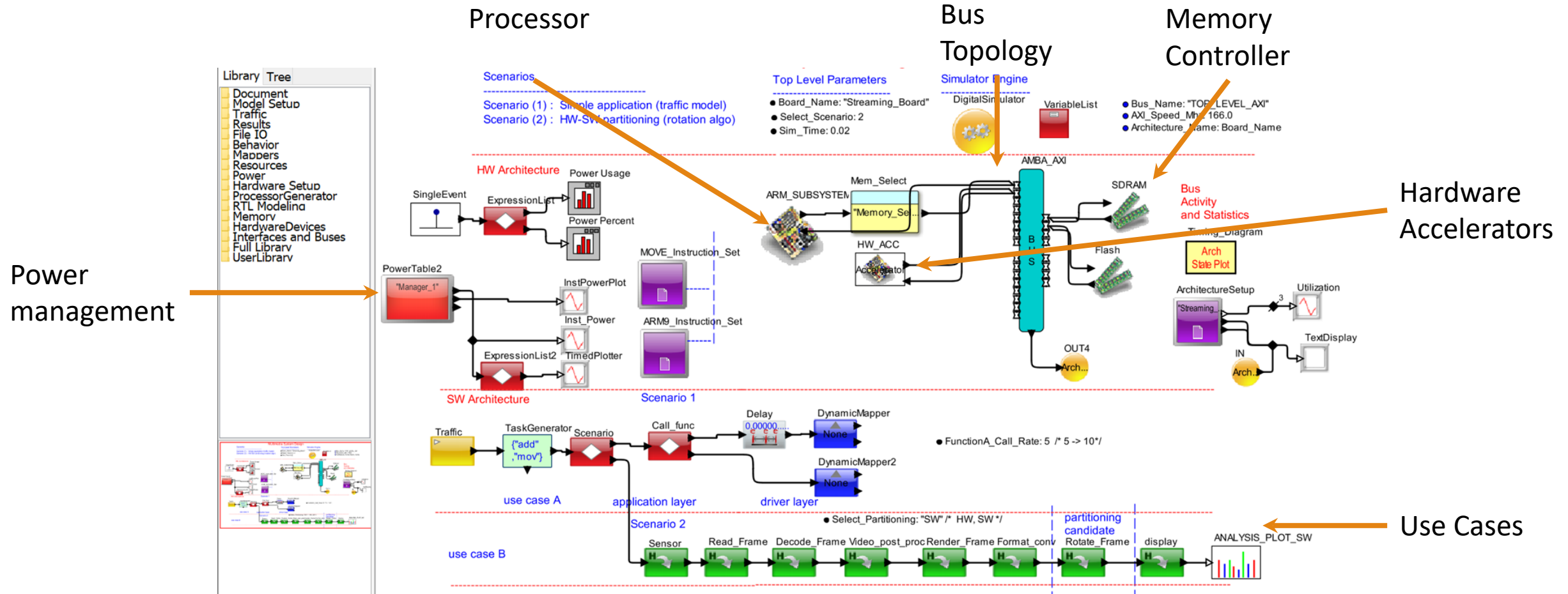
- **CASE 1** : Modules which support Power calculation with preconfigured power states:
 - Processor - Preconfigured architecture
 - Server, Channels etc – Architecture is defined using these building blocks

- **CASE 2** : Custom Modules
 - Custom Power states
 - Custom Architecture
 - RegEx functions are used to define the power profile of a module

Designs using CASE 1 modules

PREDEFINED POWER STATES

Model SoC Architecture and Map the MPEG Application



Specification, demonstration and exploration using a single model

Evaluations and Decisions



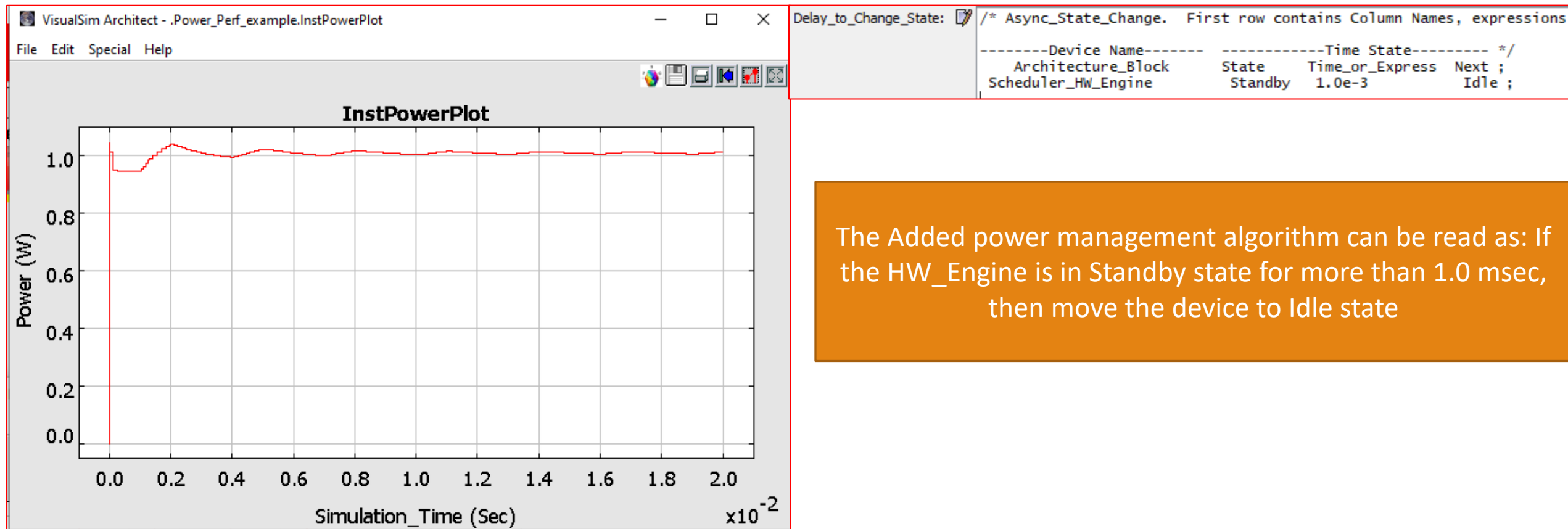
SW->All functions implemented in Software on ARM A53
HW->Rotate Frame implemented as an accelerator. All other functions are on ARM A53

SW->meets power but not performance
HW->Meets performance but power is too high.

Decision
 Add the accelerator but implement power management



Results after adding power management

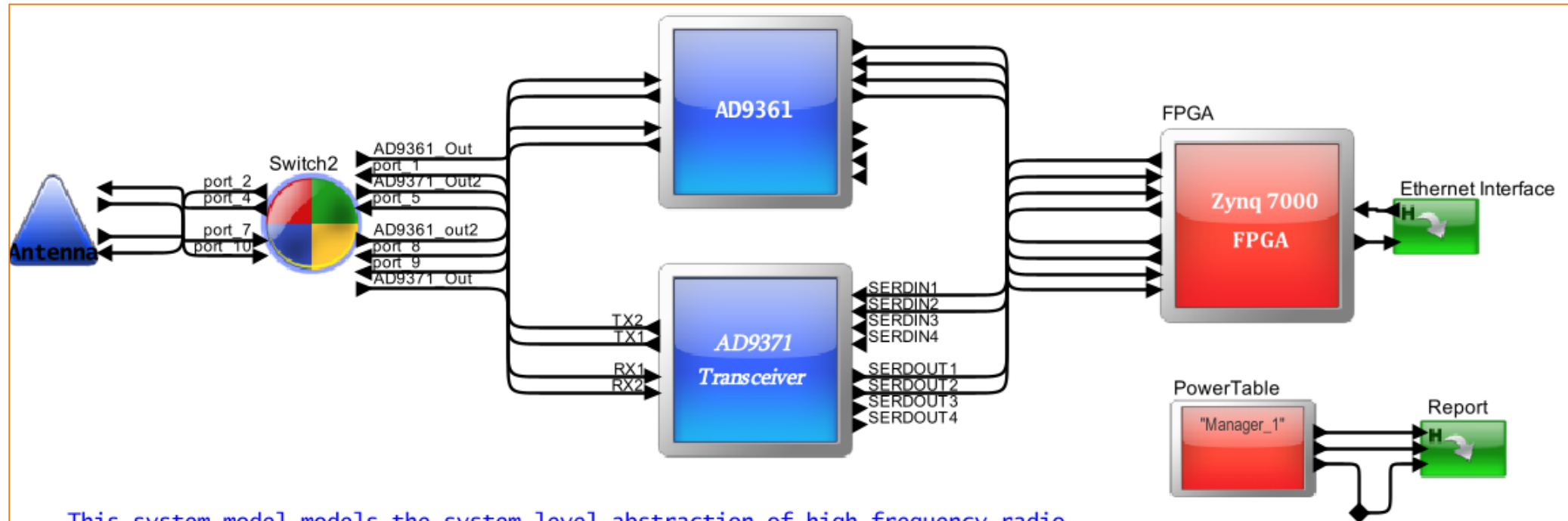


The Added power management algorithm can be read as: If the HW_Engine is in Standby state for more than 1.0 msec, then move the device to Idle state

Designs using CASE 2 modules

CUSTOM POWER STATES

Analog + Digital system designs



This system model models the system level abstraction of high frequency radio. The system is broadly classified in to three subsystems, namely; Ethernet Interface, Zync 7000 FPGA for modulation and de-modulation, AD9371 or AD9361 as transceiver and RF module.

Ethernet Interface can send information either with a fixed data rate or variable data rate. Data rate can be configured using the parameter Data_Rate with values {10,10}, where first value has the Minimum Data Rate and the second value has Maximum Data Rate.

Power RegEx functions used in ExpressionLists



Setting to Active state

Setting to Standby state

Power Table Definition RegEx functions

```

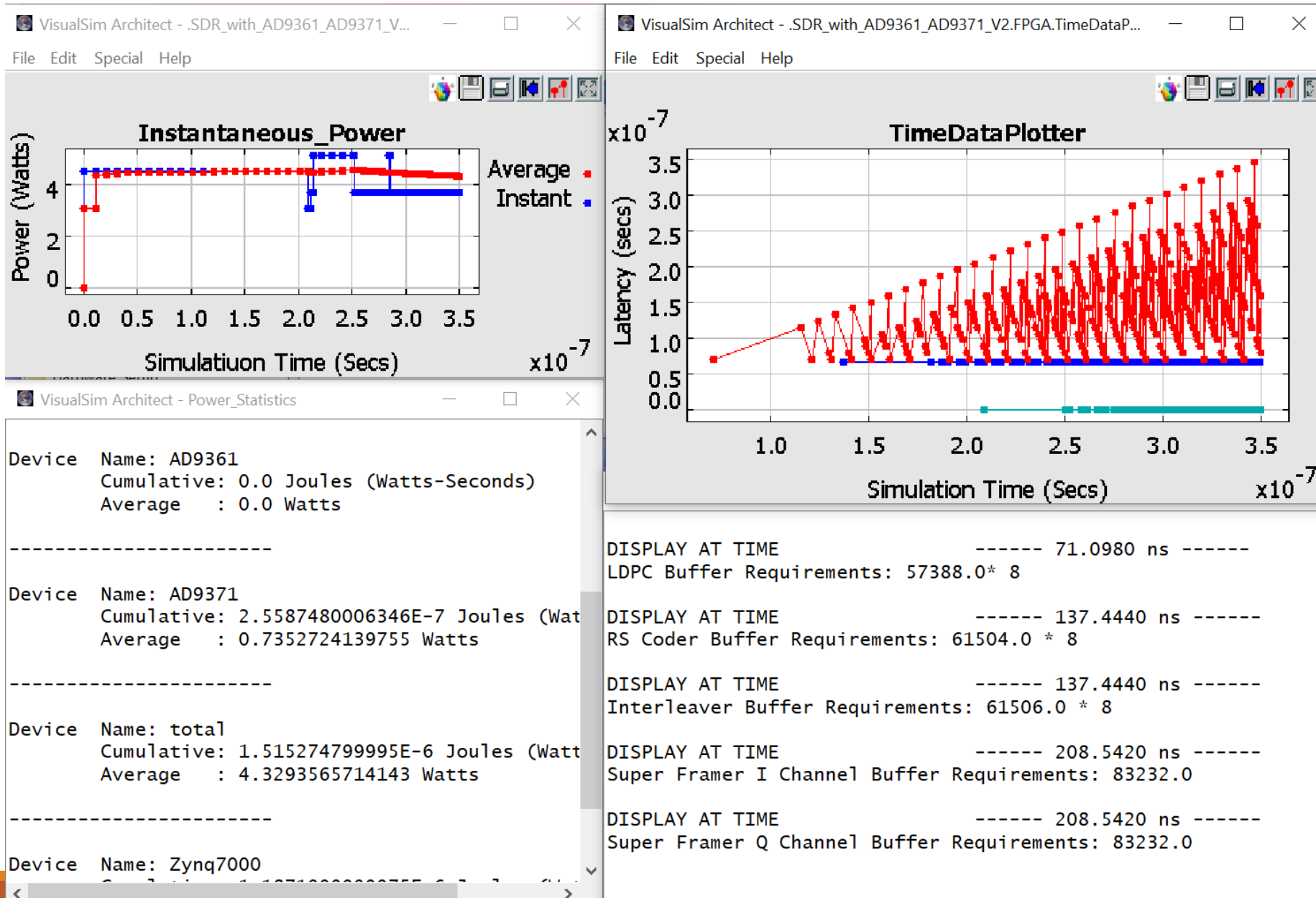
Manager_Setup: /* Power by Battery_Units, File (*.txt, *.csv, *.xml) or text below.

-----Name-----      -----Power by State-----      --Transition-- */
      Architecture_Block      Active  Standby  Wait  Idle      Existing  OffState  OnState  t_OnOff      Mhz      Volts  ;
AD9371      1.1      0.5      0.2  0.1      Standby  Standby  Active  Cycle_t      1000.0  1.0  ;
Zynq7000      3.8      2.4      1.2  0.1      Standby  Standby  Active  Cycle_t      1000.0  1.0  ;
AD9361      2.0      0.2      0.5  0.01      Standby  Standby  Active  Cycle_t      1000.0  1.0

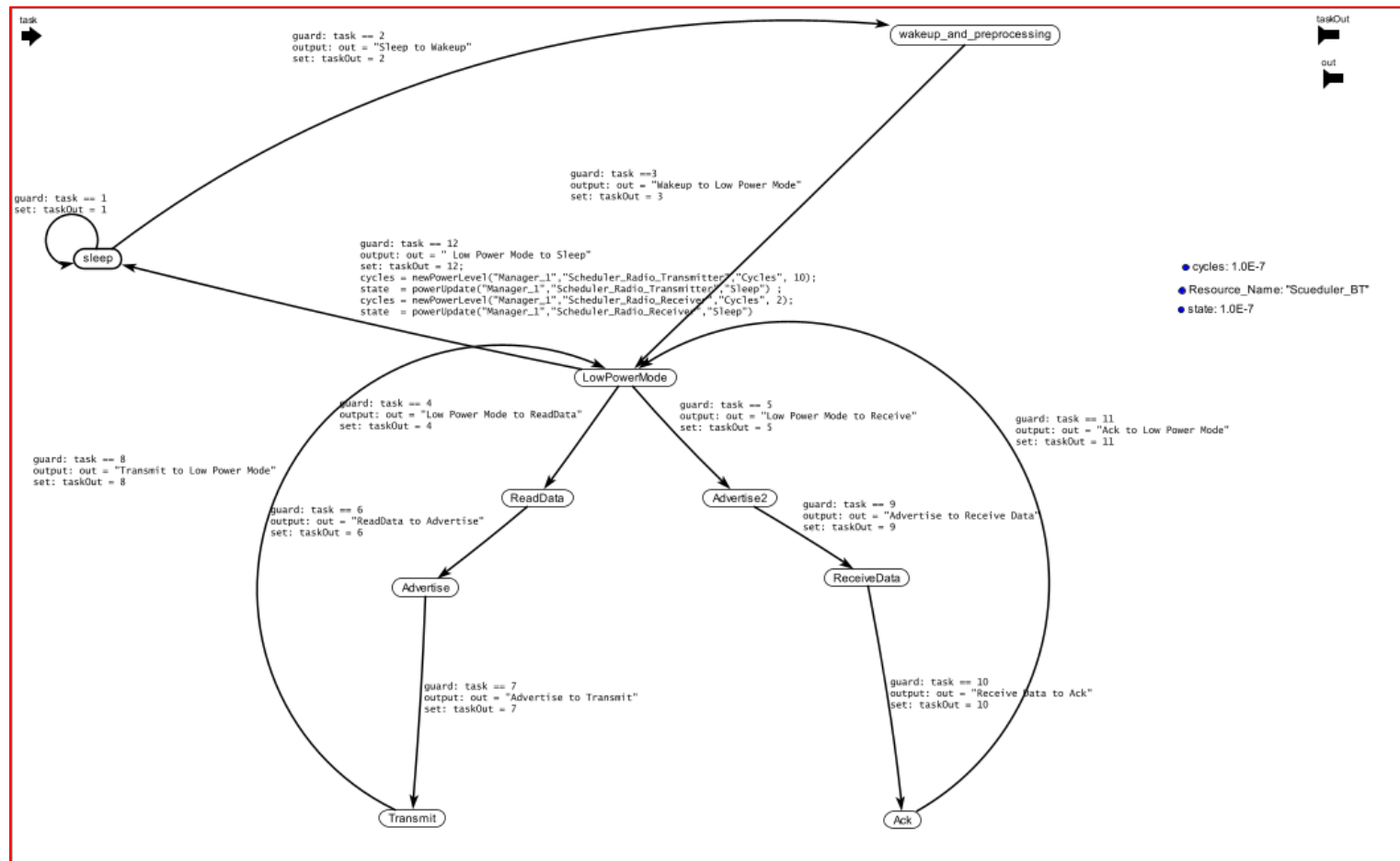
```

RegEx functions used	Description
powerUpdate("Manager_1","AD9361","Active")	The module defined by the name of "AD9361" in powerTable "Manager_1" is moved to "Active" State
powerUpdate("Manager_1","AD9361","Standby")	The module defined by the name of "AD9361" in powerTable "Manager_1" is moved to "Standby" State

Statistics



Control System Design – state machines



Bluetooth module in IoT Demo model

Power Table Definition

Manager_Setup:	Architecture_Block	Standby	Active	Wait	Idle	Existing	OffState	OnState	t_OnOff	Mhz	Volts	;
	Architecture_1_Device_1_ARM CortexM0	6.0e-3	ProcPwr	3.0e-3	0.0	Standby	Standby	Active	0.0 100.0	1.0		;
	Architecture_1_Device_1_Bus_1	100.0e-3	BusPwr	10.0e-3	0.0	Standby	Standby	Active	0.0 100.0	1.0		;
	Architecture_1_Device_1_SDRAM_1	10.0e-3	MemPwr	10.0e-3	0.0	Standby	Standby	Active	0.0 100.0	1.0		;
	Device_1_LP_Bluetooth	(0.9*0.9e-6)	(0.9*36.445)	0.0001	0.1	Standby	Idle	Active	0.0 100.0	1.0		;

Expressions and equations are used to define the power state values

```

Expression_List:
/* First row contains Column Names.
-----Reference----- Expression----- */
Name      Value
Cycle_t   1.0E-6 / Mhz
ProcPwr   Processor_Speed*12.5e-2
BusPwr    Processor_Speed * 3.5e-3
MemPwr    Memory_Speed * 1.0e-3
    
```


Usage of RegEx functions

```
Active_Count += 1
```

```
if(Power_Flag){  
    powerUpdate (Power_Manager_Name, Master_Act_Power, "Active")  
}
```

Enable/Disable Power

Change state

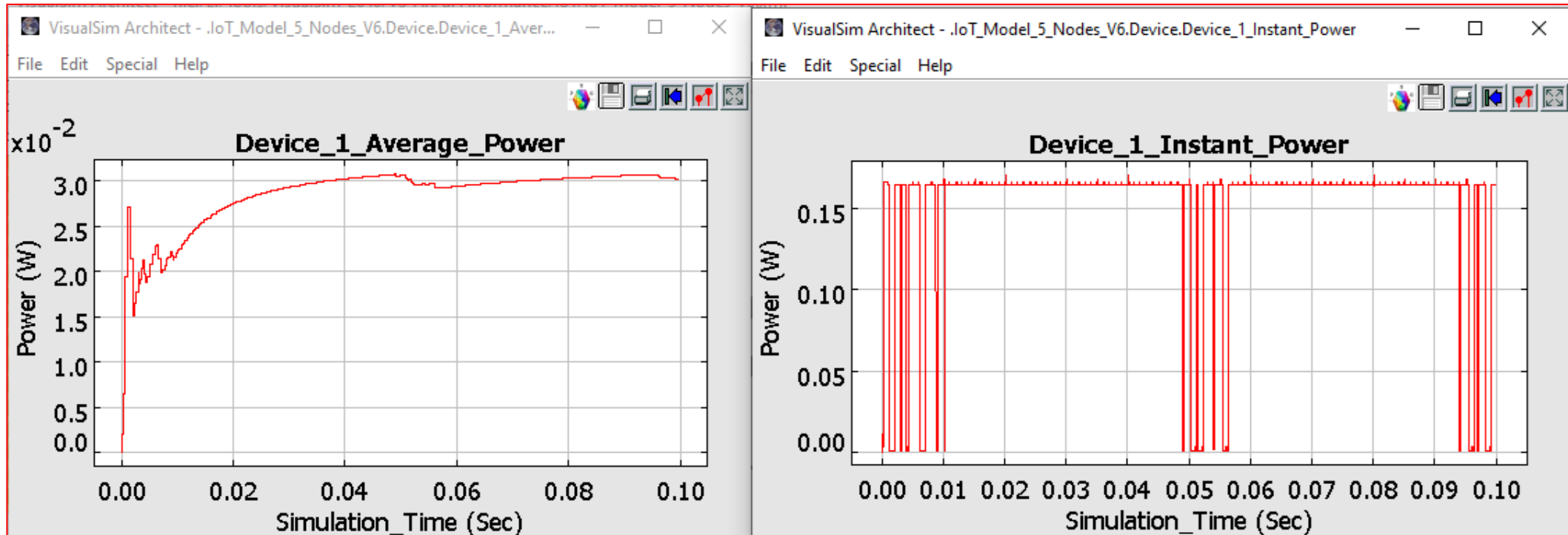
```
TIMEQ ("Master_Processing", port_token, 0, AXI_Cycle_Time)
```

Delay

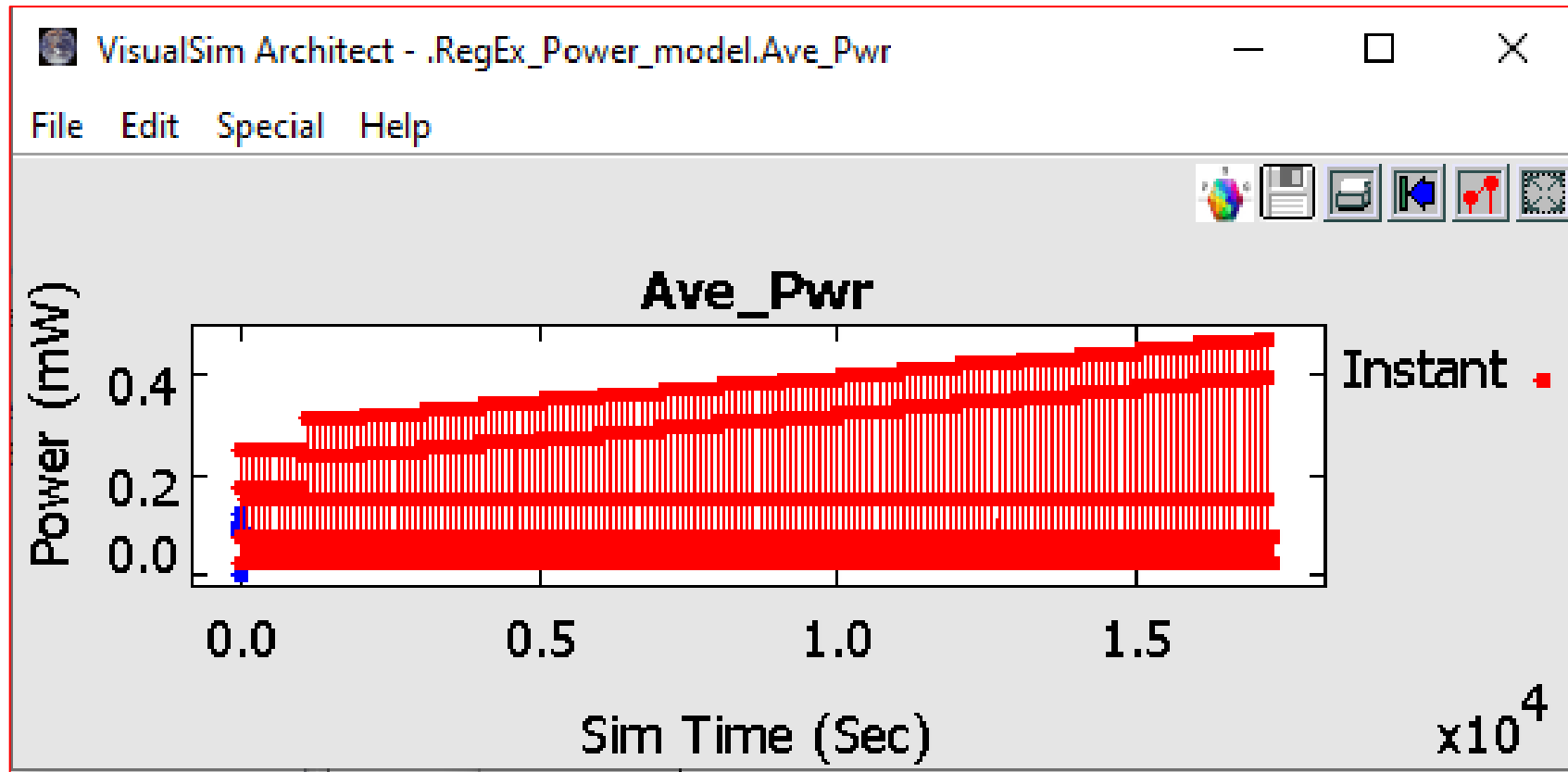
```
Active_Count = Active_Count -1
```

```
if(Power_Flag && Active_Count == 0){  
    powerUpdate (Power_Manager_Name, Master_Act_Power, "Standby")  
}
```

Generated Stats

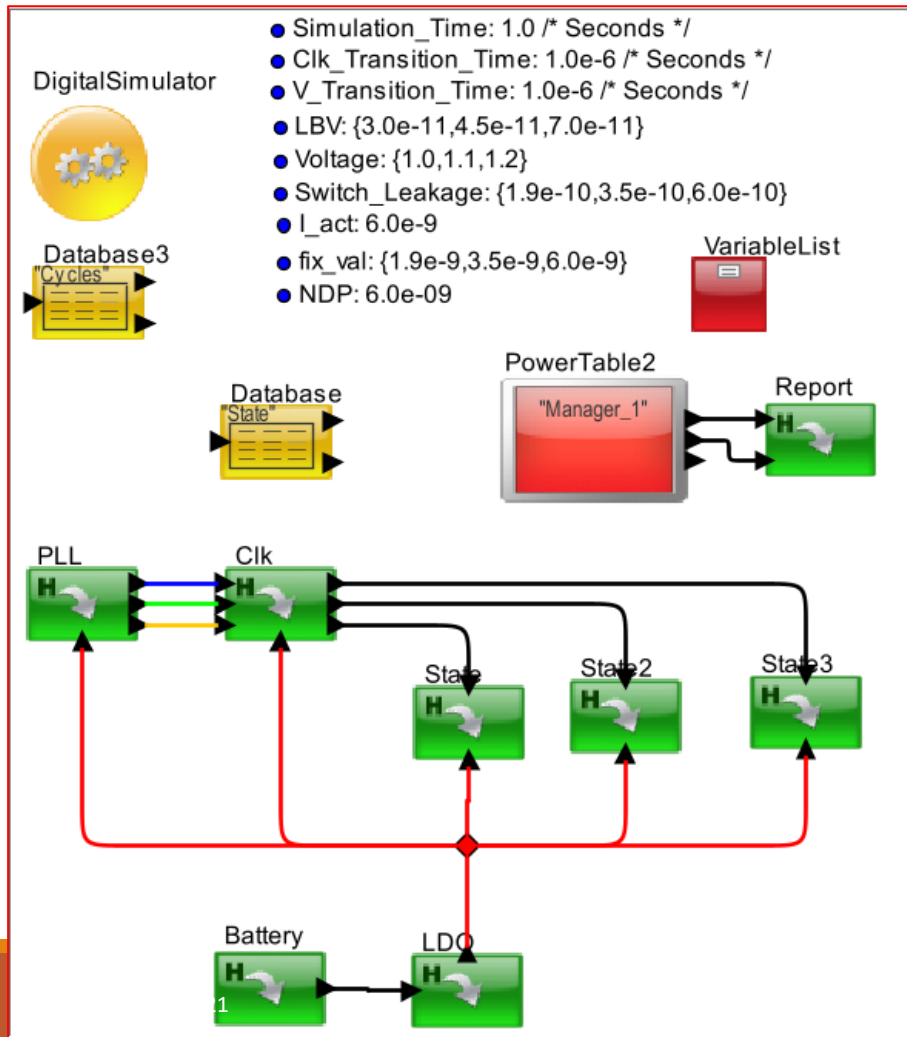


Aging of devices




As time passes, the power efficiency goes down making more power consumed by devices for the same process

More Power Levels with Dynamic State values



RegEx functions used	Description
powerManager("Manager_1")	Reads the current power Table configuration for all devices defined

Power Table Configuration

Manager_Setup:  /* Power_Table. First row contains Column Names, expressions valid for entries except Device Name.

-----Device Name-----	-----Power States-----	-----Operating States-----	-----State Transitions-----	-----Speed-----	-----Exist-----	*/
Architecture_Block	OFF PLL_OFF CLK_OFF PROC_OF G2_OFF G3_OFF	Sleep Active Standby	G2 G3 IDLE ON	Wait Existing	OffState OnState	t_OnOff Mhz Vd
PLL	0.0	1.0e-10 0.0	0.0 0.0 0.0	PLL_ON 0.0	OFF	Active 5.0e-9 100.0 1.0
CLKMUX	0.0	1.0e-10 0.0	CLK_G2 CLK_G3 0.0	0.0	OFF	Active 5.0e-9 100.0 1.0
PROC	0.0	1.0e-10 0.0	PROC_G2 PROC_G3 PROC_Idle 0.0	0.0	OFF	Active 5.0e-9 100.0 1.0
G2	0.0	1.0e-10 0.0	0.0 0.0 0.0	0.0	OFF	Active 5.0e-9 100.0 1.0
G3	0.0	1.0e-10 0.0	0.0 0.0 0.0	0.0	OFF	Active 5.0e-9 100.0 1.0

New power states added to satisfy the requirement

State Values are dynamically calculated

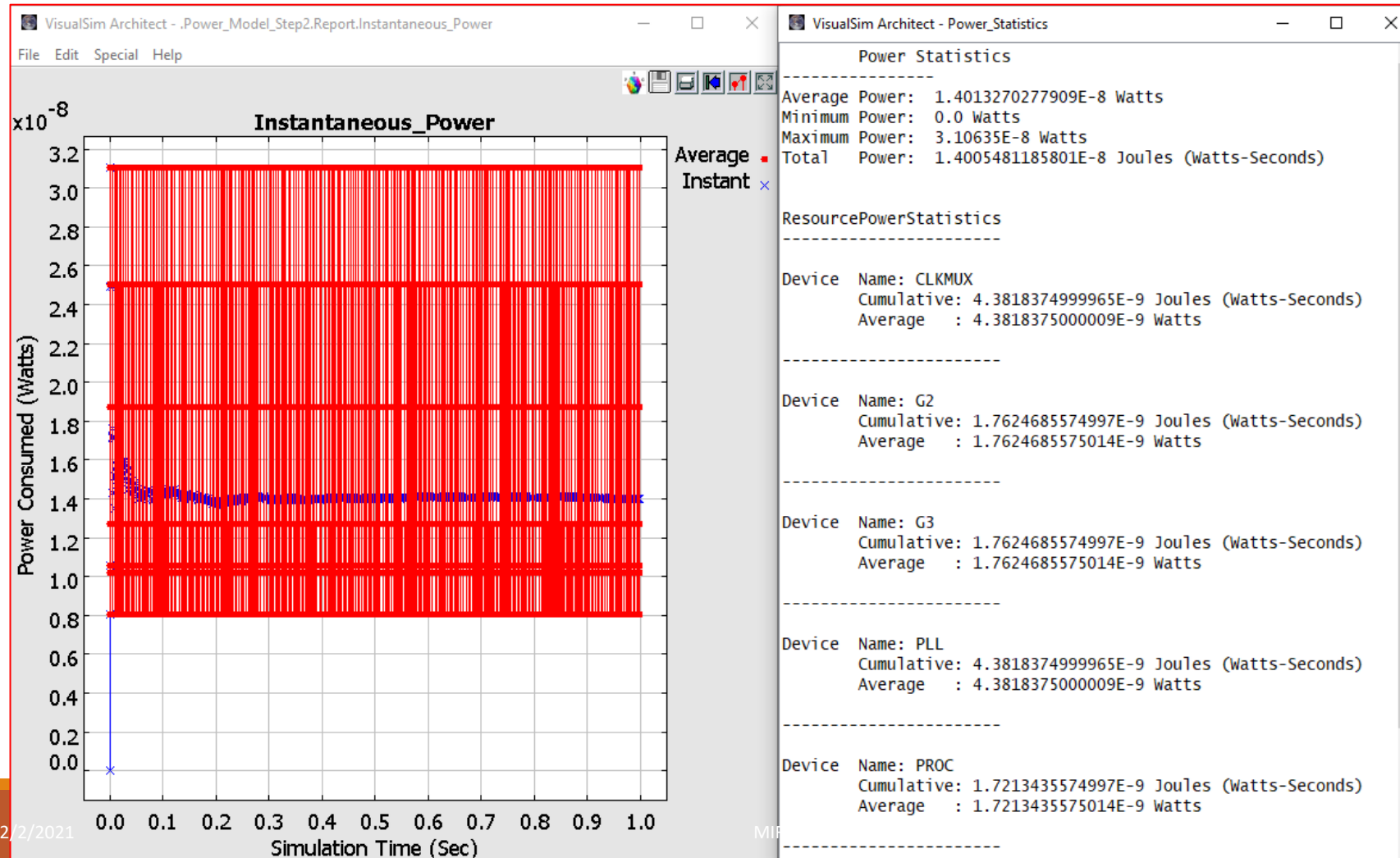
```

Expression_List:
/* First row contains Column Names.
-----Reference----- Expression-----
                Name                               Value
G2_OFF          G2_S*Switch_Leakage(G2_V)*G2_Vin ;
G2_Sleep        clk*G2_Vin*NDP*G2_Vin+G2_A*LBV(G2_V)*G2_Vin+G2_S*Switch_Leakage(G2_V)*G2_Vin ;
G2_Act          clk*G2_Vin*NDP*G2_Vin+G2_A*LBV(G2_V)*G2_Vin+G2_S*Switch_Leakage(G2_V)*G2_Vin ;
G3_OFF          G3_S*Switch_Leakage(G3_V)*G3_Vin ;
G3_Sleep        clk*G3_Vin*NDP*G3_Vin+G3_A*LBV(G3_V)*G3_Vin+G3_S*Switch_Leakage(G3_V)*G3_Vin ;
G3_Act          clk*G3_Vin*NDP*G3_Vin+G3_A*LBV(G3_V)*G3_Vin+G3_S*Switch_Leakage(G3_V)*G3_Vin ;
PROC_OF         PROC_S*Switch_Leakage(PROC_V)*PROC_Vin ;
PROC_G2         clk*PROC_Vin*NDP*PROC_Vin+PROC_A*LBV(PROC_V)*PROC_Vin+PROC_S*Switch_Leakage(PROC_V)*PROC_Vin ;
PROC_G3         clk*PROC_Vin*NDP*PROC_Vin+PROC_A*LBV(PROC_V)*PROC_Vin+PROC_S*Switch_Leakage(PROC_V)*PROC_Vin ;
PROC_Idle       clk*PROC_Vin*NDP*PROC_Vin+PROC_A*LBV(PROC_V)*PROC_Vin+PROC_S*Switch_Leakage(PROC_V)*PROC_Vin ;
PLL_ON          I_act*1*PLL_Vin ;
PLL_OFF         fix_val(PLL_V)*PLL_Vin ;
CLK_Sleep       I_act*1*CLK_Vin ;
CLK_G2          I_act*1*CLK_Vin ;
CLK_G3          I_act*1*CLK_Vin ;
CLK_OFF         fix_val(CLK_V)*CLK_Vin ;

```

Each expression uses parameter values and global variable values which are modified during the simulation which gives us a dynamic power profile

Generated stats



Impact of capacitance

- PowerTable can also show the impact of capacitance (delay to reach a certain power level)
- PowerTable has a column called “t_OnOff”, which can be used to specify the time it will take to reach the peak power level once the change of state has been made.

```

Manager_Setup:
state Transitions----- --Speed--  --Exist-- */
ON      Wait Existing OffState OnState t_OnOff      Mhz      Volts  ;
PLL_ON  0.0  OFF      OFF      Active  5.0e-9      100.0    1.0    ;
0.0     0.0  OFF      OFF      Active  5.0e-9      100.0    1.0    ;
: 0.0   0.0  OFF      OFF      Active  5.0e-9      100.0    1.0    ;
0.0     0.0  OFF      OFF      Active  5.0e-9      100.0    1.0    ;
0.0     0.0  OFF      OFF      Active  5.0e-9      100.0    1.0    ;

```


Power unit – W, mW, uW

Battery_Units:	Milli_Watts
State_Plot_Enable:	Micro_Watts
	Milli_Watts
	Watts

Pulldown option for the user – in power Table
If Micro_Watts is selected, then all state values entered
will be in Micro_Watts

Creating States for all Devices including Wires

```

/* Power_Table. First row contains Column Names, expressions valid for entries except Device Name.
   where "Scheduler_" or "STR_" + BlockName; Processor, Bus, DRAM = Architecture_Name + "_" + BlockName
-----Device Name----- -----Power States----- -----Operating States----- --toActive-- --Speed-- --Exist-- */
Architecture_Block      Standby Active      Wait Idle Retention      Existing OffState OnState t_OnOff      Mhz      Volts ;
Architecture_1_LPDDR    70.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
RAS_Power_SDRAM        70.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
RRD_Power_SDRAM        70.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
Read_Power_SDRAM       70.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
Write_Power_SDRAM      70.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
ACT_Active_SDRAM       70.0  100.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
ACT_Standby_SDRAM      30.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
WTR_Power_SDRAM        30.0  300.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
Architecture_1_L3_Cache 70.0  Partial_Power_Val 0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
RNF                    70.0  Partial_Power_Val 0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
RNI                    70.0  Partial_Power_Val 0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
HNF                    70.0  Partial_Power_Val 0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
XP                    70.0  175.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;
Wire_Power             70.0  175.0          0.0 0.0 10.0          Standby Standby Active  Cycle_t    1000.0  1.0 ;

```

Power Management for all Devices

/* Delay_to_Change_State. First row contains Column Names, expressions valid for entries except Device Name.

where State to same State can extend a Power State

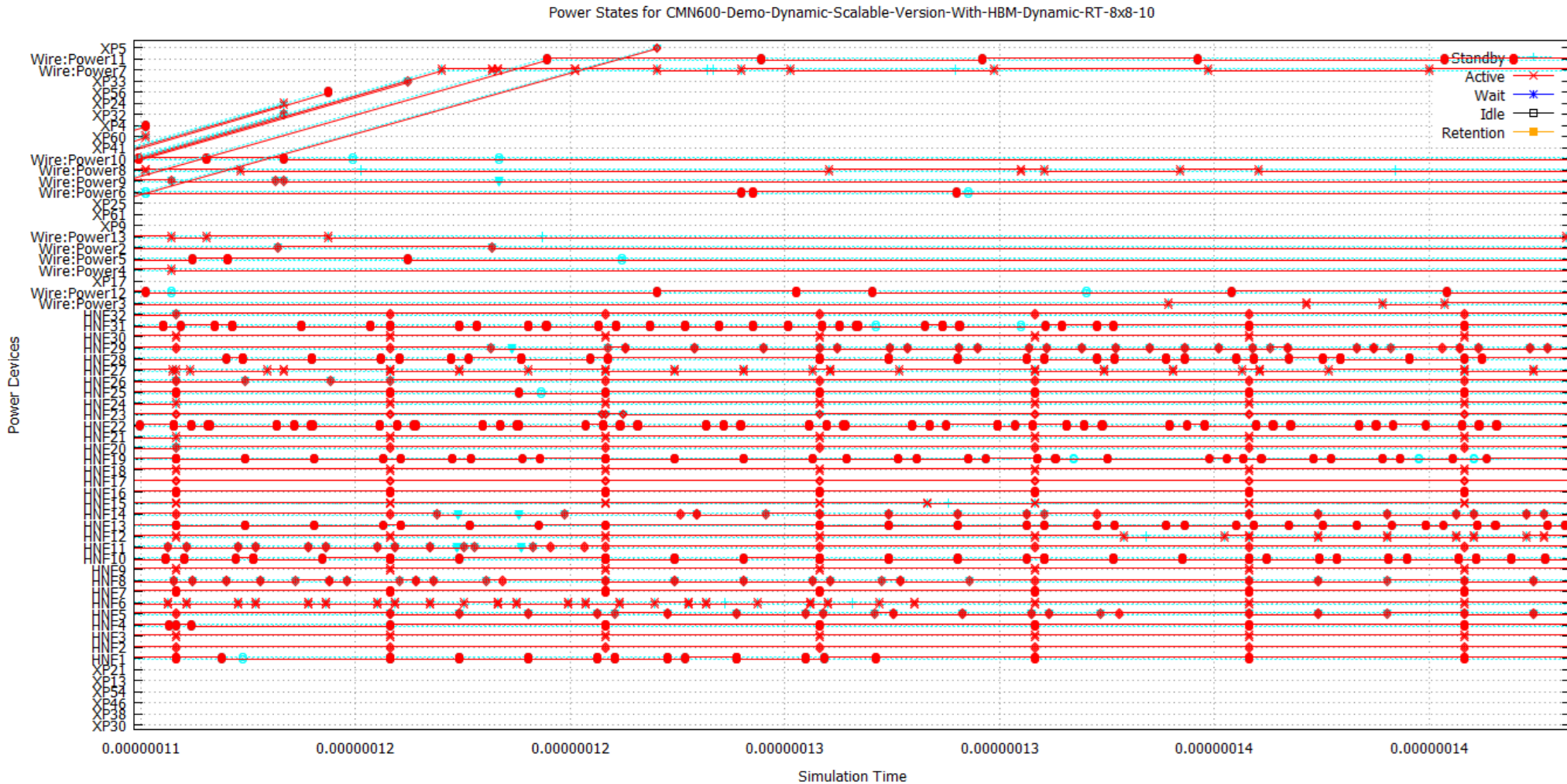
```

-----Device Name----- --Start-----Expression-----Next--- */
Architecture_Block      State      Time      State ;
Architecture_1_L3_Cache Standby   100.0e-9  Retention ; /*
Functional Retention mode */
RNF                      Standby   5.0e-6    Idle      ;
HNF                      Standby   5.0e-6    Retention ;
Wire_Power               Standby   5.0e-6    Idle      ;

```

State_Plot_Enable (GNUPlot)

State_Plot_Enable:



power plot
in gnu plt
format is
created



VISUALSIM TRAINING
