

IBM Research Report

MEMSIM User Guide

Karthick Rajamani
IBM Research Division
Austin Research Laboratory
11501 Burnet Road
Austin, TX 78758



Contents

1	Introduction	2
1.1	Organization of Server Main-Memory System	2
1.2	Design of MEMSIM	4
2	Usage	5
2.1	Source Distribution	6
2.2	DRAM and Memory System Parameters	6
2.3	Memory Trace Format	7
2.4	Configuration-File Format	7
2.5	Memsim as a Memory System Service or Component	8
3	Sample Runs	10
3.1	Example 1: Scaled Down System	10
3.2	Example 2: Modeling a small (no-SMI) system	16
3.3	Example 3: Running Memsim as a Memory Service Program	20

Chapter 1

Introduction

This document is a user guide for MEMSIM, a main-memory system simulator. Memsim, developed at the IBM Austin Research Lab, provides parameterized models of state-of-the-art server memory-system components, right from the organization of the memory-controllers to the timing and power of individual SDRAM devices. Models are also provided for design options such as different interleaving schemes, open/closed-page policies, and dynamic power-management. Additionally, new schemes can be constructed using functions in the existing code-base as building blocks (approaches to extend the simulator are not covered in this document).

1.1 Organization of Server Main-Memory System

A typical high-end multi-processor server system would have multiple memory controllers managing access to multiple sets of DRAM memory. In the past, memory controllers have been customized chips designed to manage a specific organization of SDRAM designed for a particular server family. Future server designs appear to bring the memory controller into the processor chip to reduce delays between the controller and processors among other reasons. In Memsim, we model a generic data path (characterized by its width and frequency) between a memory controller and processors independent of the controller location.

Figure 1.1 shows the logical organization of the memory system similar to that modeled in Memsim. The memory controller depicted has command and data interfaces, read/write command queues, and data buffers aspects of which are modeled in Memsim. In addition, it would have additional components such as request-reordering queues and error-handling logic which are not modeled in Memsim.

In high-end server systems, memory controllers are, often, not directly connected to the SDRAM devices. Buffers, that we would refer to as *Synchronous Memory Interface (SMI)* chips, intervene in the connection between the controller and memory. SMIs allow the desired higher-frequency and lower path widths on the controller side and accommodate the slower frequency (and the compensating wider path widths) on the SDRAM side. The figure shows two (independent) ports for the controller with 4 SMIs under each. The SMIs at the same port act in conjunction resulting in transfer of command and data along all of them simultaneously i.e. the path to the SDRAM at one port is the sum of the paths from the 4 SMIs attached to that port. The SDRAM devices themselves are organized on *Dual Inline Memory Modules (DIMMs)*, multiple ones of which are attached to each SMI. Concurrent accesses could be in progress at the multiple DIMMs attached to an SMI.

The DIMMs are composed of multiple *memory-chips* (also called *memory-devices*). x4, x8, and x16 memory-devices (denoting the number of bits accessed on a single request) are typical. A subset of these memory-chips supply the entire bit-width of the DIMM. This subset is known as a *rank*. The DIMMs may be uni-ranked or may have multiple ranks. The memory-chips are typically quad-banked allowing concurrent accesses to proceed even with a single rank of devices on a DIMM. The devices are configured to operate in *burst-mode*, wherein data can be read from or written into a bank on successive edges of the memory clock for the duration of the burst (typically of length 4).

When a cache-line request (fill or writeback) is presented to the memory-controller, it is first mapped to the

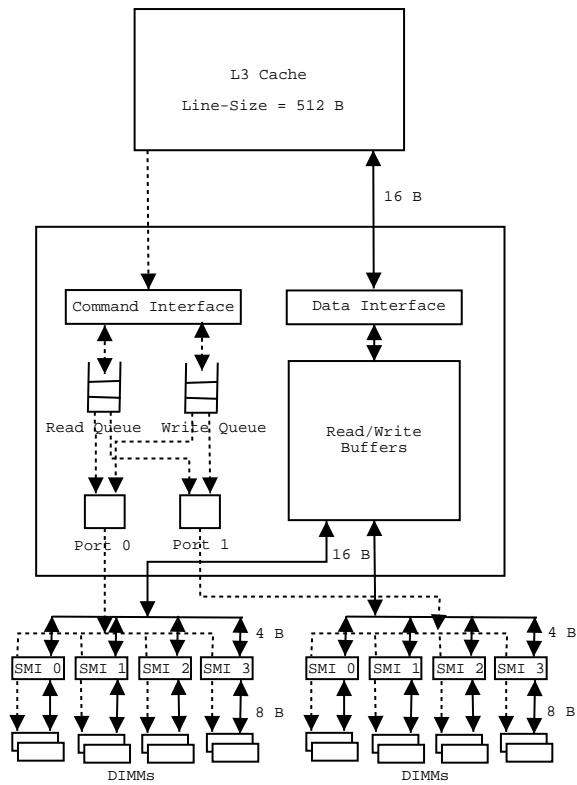


Figure 1.1: Logical Organization of Memory Subsystem of an IBM POWER4 System. The dashed-line depicts the command/address-flow and the solid-lines for the data-flow. The widths of the data-lines are labelled and are given in bytes.

requisite port. The SMIs that are connected to the port provide the entire width of the port. This is achieved by having each SMI provide a portion of the overall width of the port. For instance, as shown in figure 1.1, the SMIs provide 16 bytes of data by each SMI providing 4 bytes. On the other hand, the DIMMs provide 8 bytes worth of data each to the SMIs on a single transfer. For a cacheline worth of data from DIMMs under the 4 SMIs, one would need to employ a burst access (for example a burst length of 4 would provide a cacheline of 128 bytes worth of data for the organization shown in the figure). The width of the data paths and the DIMMs are configurable in Memsim as are the frequencies of the different paths and the SDRAM burst lengths.

For the individual memory-chips, the memory-controller may employ a *close-page-autoprecharge* policy or an *open-page* policy. Reading or writing data from/to a bank involves three distinct operations, each of which has a certain latency and power-cost. The three operations are precharge-delay (t_{RP}), the RAS-to-CAS delay (t_{RCD}), and the column access-time (CL). The close-page policy is the simplest to implement. Here, after a burst-access to a bank is complete, the memory-controller closes the row and issues an auto-precharge on that bank. For this policy, there is a constant access latency of $t_{RCD} + CL$, regardless of whether the next access to the bank is to the same row or not. On the other hand, with an open-page policy, the row is kept open even after the burst, so that if there is good row-locality (eg. some streaming access-pattern is encountered), we need to pay only the CL penalty. However, if the access maps to a different row, the currently open row needs to be closed, precharged, and new row needs to be opened, incurring a penalty of $t_{RP} + t_{RCD} + CL$. Further, to implement an open-page policy, the memory-controller needs to keep track of open pages on the banks. Memsim can be configured to use either the *open-page* or the *close-page* policy and an additional (experimental) hybrid policy. While the experimental hybrid policy is for illustrative purposes, it can be easily extended to support very effective adaptive page policies.

SDRAMs support clock-disabled low power modes: *power-down* and *self-refresh*. Because of its high exit latency the *self-refresh* mode is less useful high-performance server main memory systems. The current version of Memsim provides support for two dynamic SDRAM power management policies that use power-down - one using power-down immediately after a request completes if no pending requests are present for the same set of devices and another that waits a specified duration for new requests before entering power-down. Additional variations can be implemented by extending the functions provided to support these two policies.

1.2 Design of MEMSIM

The MEMSIM simulator uses an event-driven model and is written in C using the CSIM toolkit developed by Mesquite Software Incorporated. The timing and contention behavior of the underlying memory-system is modelled using CSIM *facilities*, which consists of a server and its associated queue, for which a service-discipline is defined (which is first-come first-serve by default). The simulator can be driven by either a memory-trace or by communicating requests to it through a custom shared memory interface. The latter can be used to integrate Memsim with a processor simulator. Each entry of the trace needs to specify:

Memsim is highly customizable. Parameters relating to the organization of the memory system are customizable through a configuration file or on the command-line. Currently, SDRAM timing and power characteristics are specified in a separate header file - Memsim is compiled for a specific SDRAM device type. Policy level parameterization are also configurable through the same input configuration file. MEMSIM also provides a set of interface functions for implementing specific interleaving policies that govern operations like a mapping of a cache-line to a memory-controller or a DIMM segment to a particular bank. Supported interleaving strategies are *spread* or round-robin interleaving vs *fill* at each of the controller, rank and bank levels. In a spread strategy consecutive addresses to a level go to different components at that level. In a fill strategy, consecutive addresses to the same level go to the same component till it is full. Customization of the interleaving to support additional strategies requires source modifications to alter the specific interleaving functions.

Chapter 2

Usage

Running Memsim without any options will provide the usage information similar to the following:

```
Usage: ./memsim <sizeformatGB> config=<Mem-configfile> source=<SHM | path to tracefil  
[options]:  
nrecords=<max records> |  
debugon | norefresh | slowadvance | noprintstats  
pathshort | smipathshort | procpathshort |  
smicmddelay=<delay in pclk> | mccmddelay=<delay in pclk> |  
smisharedrw |  
smireadwidth=<width in bits> | smiwritewidth=<width in bits> |  
bufferedsmireads=0/1 | bufferedsmiwrites=0/1 |  
oldnobuffersoln |  
noscaleactstby |  
reportinterval=<nrecords>  
selfalign
```

The path to the input configuration file (`config=`) and source (`source=`) are compulsory parameters to the command. They need to precede all other parameters, which are optional. Format of the configuration file is discussed in section 2.4. The source file format is discussed in section 2.3. When the source is specified as 'SHM', Memsim works as a memory request service program instead of a trace-driven one. In this mode, Memsim can work in consort with another program/simulator which can model other components and feeds memory requests to Memsim. A brief overview of this mode is provided in section 2.5.

Some other useful command-line parameters:

nrecords=max records Specifies the maximum number of records to process.

debugon Turns on the output of debug messages (to stderr).

norefresh Turns off the modeling of periodic auto-refresh operations.

pathshort Models zero delay for data transfer along the controller-processor path and the SMI-controller path.

smipathshort Zero delay for the SMI-controller path.

procpathshort Zero delay for the controller-processor path.

smicmddelay=delay in pclk and mccmddelay=delay in pclk Command processing times at the SMI and at the memory controller.

smisharedrw Shared (bidirectional) data path between SMI and controller for reads and writes.

bufferedsmireads=0/1 and bufferedsmiwrites=0/1 Whether writes and reads can be buffered at the SMIs or not.

noscaleactstby Turn of the energy-scaling option for number of active banks in a device (use same energy for the presence of active banks on a device independent of the number of active banks) - default is to scale the energy by the number of active banks. This is an 'alternative' power model.

selfalign If the addresses generated in the trace are not aligned to MEMDATA_LENGTH Memsim will signal an error and quit. Using this option lets Memsim continue using the 'aligned' address in place of the problem value. In practice, unaligned addresses would determine the burst-mode to be signaled to the SDRAM determining the order of the bytes in the response.

2.1 Source Distribution

The source distribution consists of the following files:

- 1 Userguide.pdf (this document).
- 2 dram_model.h: the header file that lists the values for the DRAM parameters.
- 3 memsim.c: the main source file containing all the memory simulation code.
- 4 memsim.h: the main header file that includes default parameter values.
- 5 memsim_service.c: source for linking memsim with a another simulator or trace generation program using shared memory.
- 6 memsim_comm.h: header for memsim-other program communication.
- 7 Makefile.
- 8 sample.trace, sample2.trace: sample input trace files.
- 9 sample.config, sample2.config: sample input configuration files.

2.2 DRAM and Memory System Parameters

Key DRAM timing and current parameters are coded into the header file dram_model.h. It is important that all the values here be entered from the data sheet for the DRAM parts that would be used or from actual measurements. It does not make sense to use the current set of hard-coded values as default !

Notes:

- 1 PCLK_FREQ is the processor clock frequency in hz. The smallest time-step for memsim is 1 cycle of this frequency. PCLK_BY_DRAMCLK is the ratio of the processor clock frequency to the DRAM clock frequency (for DDR SDRAM, this is still the ratio of processor clock to memory clock e.g. for DDR400 one would use 200MHz as the value for the memory clock to derive this ratio). TRANSFER_PER_CLOCK is 1 for SDR SDRAM and 2 for DDR SDRAM.
- 2 All the DRAM timing constraint paramemters are given in units typically used in datasheets. tRP, tRCD, tRAS, tRC, tRRD, tRFC, and tREFI are in nanoseconds. CL, tDQSS, tWR, and tWTR are in DRAM clock cycles.
- 3 The maxVdd and nomVdd values are in volts, the IDDX values are in amperes, and WattperDQ (max DC power per data and data-strobe pins) is in watts. The calculation of WattperDQ is dependent on the SDRAM data-sheet parameter IOL and actual on-dimm and termination resistances. The hard-coded resistance values are arbitrary ones, though considered 'typical'. Use values of parts from your actual memory design.

2.3 Memory Trace Format

Memsim adopts a very simple ascii trace format. Each record has to be present in its own line and has four fields separated by white-spaces:

- 1 Integer timestamp in processor clocks (64-bit value).
- 2 Integer address (64-bit value).
- 3 Size of the request: Currently the value for this parameter is ignored. The size of all requests is assumed to be the value for the parameter MEMDATA.LENGTH defined in file memsim.h.
- 4 Single bit for request type: A 0 denoting read request or a 1 denoting write request.

2.4 Configuration-File Format

The inputs that define the memory system organization are specified in a configuration file identified to Memsim on the command-line with the 'config=' option. Each line in the configuration file gives the description of one parameter and its value. The parameters (i.e. their description) has to exactly match the list below. Skipping a parameter is allowed (hard-coded default would be used), but not changing the order in which they are given in the configuration file. File `sample.config` in the distribution is one such configuration file.

The descriptions of the parameters are:

- **Physical Memory Size (MB):** This is size of the physical memory to be simulated and is specified in Megabytes. It is divided up equally between all the memory controllers. Default is 4096MB.
- **Memory Controllers:** Total number of memory-controllers in the system. Default is 1.
- **Number of Ports Per Memory Controller:** Number of populated ports on each memory controller in the system. Default is 1.
- **Enable MC Interleaving:** "1" if *Spread* interleaving is to be used; "0" if cache-line *Fill* scheme is desired. Default is 0.
- **MC Queue Length:** This is the length of the combined read/write request-queue in each memory-controller. Default is 16.
- **SMIs Per Port:** Number of Synchronous Memory Interfaces attached to a memory-controller port. This is also the number of channels per memory controller port. Default is 4.
- **DIMMs Per SMI:** Number of Dual In-Line Memory Modules attached to each SMI. Default is 4.
- **Banks Per Chip:** Number of banks within each memory-device/chip. Default is 4.
- **DIMM Width:** Bit-width of a DIMM for data (excluding parity/ECC). Default is 64.
- **Device Width:** Bit-width of a DRAM memory-device. DIMM Width divided by Device Width establishes the number of devices per rank. Default is 8.
- **Device Capacity:** Capacity of a memory-device in kilobytes. Default is 32KB (256Mb).
- **Devices Per DIMM:** The number of memory-devices resident on a DIMM. This divided by the number of devices per rank establishes the number of ranks on a DIMM. Default is 8.
- **DRAM Burst Length:** Burst-length of the DRAM devices. Default is 4.

- **Rows Per Bank:** Number of rows in each DRAM bank. Default is 8K.
- **Columns Per Bank:** Number of columns in each DRAM bank. Default is 1K.
- **Enable Interleaved Bank-Fill:** "1" if the *Bank-Interleave* scheme is to be used; "0" if *Bank-Fill* is the choice. Default is 1.
- **DRAM Page Policy:** The page policy to be adopted at beginning of the run for all DRAM rows/pages. "1" for open-page; "0" for close-page-autoprecharge policy. Default is 0.
- **Enable Hybrid Page-Policy:** "0" if only open- or close-page alone;"1" for the *last-access* policy. The *last-access* policy decides on the new policy for a row based on what would have been more beneficial for the last access. The policy for a row can potentially change with every access. Default is 0.
- **Enable Power Management:** "1" if power-downs are to be used; "0" otherwise. Default is 0.
- **Power Management Policy:** "0" for the *Immediate Powerdown* policy; "1" for *Delayed Powerdown*. Default is 0.
- **Power Down Check Interval:** Number of processor-clocks to use for the *Delayed Powerdown* power-management policy. Default is 100.
- **Enable Interleaved Rank-Fill:** "1" to enable *Rank-Interleave* scheme;"0" for *Rank-Fill*. Default is 1..
- **Buffered SMI Reads:** "1" to enable reads to be buffered at the SMI. If buffered, the read from the DRAM and the transfer to the memory controller proceed in sequence but independently. If "0" i.e. not buffered the transfer to the memory controller proceeds in conjunction with the read from the DRAM. No buffer size limits are placed for buffered SMI reads. Default is "0".
- **Buffered SMI Writes:** "1" to enable writes to be buffered at the SMI. If buffered, the transfer of data on the write from the memory controller and to the DRAM proceed in sequence but independently. If "0" i.e. not buffered the transfer to the memory controller proceeds in conjunction with the write to the DRAM. No buffer size limits are placed for buffered SMI writes. Default is "1".

2.5 Memsim as a Memory System Service or Component

In addition to being used as a trace-driven simulator, Memsim can serve as a component simulating the main memory system in a larger simulation environment. Memsim can be considered as a memory system service in that situation interacting with a client program. Memory requests are passed to Memsim through shared memory with synchronization between the client program and Memsim carried out using semaphores. Memsim is written to interact with a cycle-accurate client program synchronizing every cycle.

The files `memsim_service.c` and `memsim_comm.h` provide the facilities for communication and synchronization between Memsim and the client program. The file `memsim_service.c` also contains the code for a sample client program (that reads a trace file and feeds the requests to Memsim) that can be built with the provided Makefile.

The API calls that a client program would use include:

commClientInit() Initializes the shared variable pointers on the client side, among other things.

commClientSynchTime() Synchronizes with memsim for the current clock tick in the client program to have both programs move in lock-step.

sendRecord() Sends a record from the client program to Memsim (actually the shared memory from where Memsim will pick it up).

commCheckForResponse() Look for responses from Memsim for memory requests submitted.

The `main()` function in `memsim_service.c` that is compiled in for `memsim_client` (with the flag `COMPILE_AS_CLIENT`) illustrates the usage of these calls.

Chapter 3

Sample Runs

In this chapter, we illustrate the use of MEMSIM by walking through a couple of examples.

3.1 Example 1: Scaled Down System

In the first one, the input workload is a memory-trace whose filename is `sample.trace`. The configuration-file name is `sample.config`. The contents of `sample.trace` is like:

```
0 536871168 128 0
0 536871040 128 0
0 536870784 128 0
1 536870656 128 0
2 536871296 128 0
3 536871424 128 0
4 536870528 128 0
4 536870912 128 0
7 640 128 0
11 512 128 0
13 256 128 0
17 128 128 0
18 768 128 0
19 896 128 0
26 0 128 0
27 384 128 0
...
```

In the above trace snippet, the first field gives the timestamp of the request in processor-cycles, the second field the memory-address. The third and fourth fields specify the size of the request (which is same as the cache line-size) and the request-type respectively (which in the above snippet is composed of only reads).

The contents of the `sample.config` file are:

```
Physical Memory Size (MB): 1024
Memory Controllers: 8
Enable MC Interleaving: 1
MC Queue Length: 64
SMIs Per Port: 4
DIMMs Per SMI: 4
Banks Per Chip: 4
```

```
DIMM Width: 64
Device Width: 8
Device Capacity: 512
Devices Per DIMM: 16
DRAM Burst Length: 4
Rows Per Bank: 128
Columns Per Bank: 1024
Enable Interleaved Bank-Fill: 1
DRAM Page Policy: 0
Enable Hybrid Page Policy: 0
Enable Power Management: 1
Power Management Policy: 0
Power Down Check Interval: 0
Enable Interleaved Rank-Fill: 1
Buffered SMI Reads: 0
Buffered SMI Writes: 1
```

The above configuration represents a 1GB system composed of eight memory-controllers, each having four SMIs per port. Each SMI has four x64 DIMMs underneath it and each DIMM is dual-ranked with x8 devices of 512KB each. In this example, the size of the system has been scaled down by a factor of 64 from an actual system. Essentially the system being modeled is a 64GB system with same number of devices as the simulated system. A close-page policy is used with immediate power-down power management policy. Interleaving is used at all the levels - controller, rank and bank. In order to run the simulation, at the command-prompt we need to type:

```
memsim config=sample.config source=sample.trace
```

The output of MEMSIM for this run is:

```
Command: ./memsim config=sample.config source=sample.trace
Physical Memory Size (MB): 1024
Memory Controllers: 8
Enable MC Interleaving: 1
MC Queue Length: 64
Bus Retry Cycles: 32
SMIs Per Port: 4
DIMMs Per SMI: 4
Banks Per Chip: 4
DIMM Width: 64
Device Width: 8
Device Capacity: 512
Devices Per DIMM: 16
DRAM Burst Length: 4
Rows Per Bank: 128
Columns Per Bank: 1024
Enable Interleaved Bank-Fill: 1
DRAM Page Policy: 0
Enable Hybrid Page Policy: 0
Enable Power Management: 1
Power Management Policy: 0
Power Down Check Interval: 0
Enable Interleaved Rank-Fill: 1
Number of records to be processed: 0
Buffered SMI Reads: 0
Buffered SMI Writes: 1
```

***** OVERALL EXECUTION STATISTICS *****

Total Simulation Time: 46980.000000 Processor Clocks (0.000023 Seconds)
Number of records read 10000, requests processed 10000
REFRESH Tried = 192, Delayed 192, Maximum Delay 200.00, Average Delay 57.64, Average Access D

***** ENERGY STATISTICS *****

Total EACT: 0.00264815 J Average PACT: 112.735128 W
Total EWR: 0 J Average PWR: 0.000000 W
Total ERD: 0.000962963 J Average PRD: 40.994592 W
Total EDQ: 0.000198374 J Average PDQ: 8.445057 W
Total EREF: 0.000258844 J Average PREF: 11.019346 W
Total EACT_STBY: 0.00215361 J Average PACT_STBY: 91.681977 W
Total EPRE_STBY: 0.000372536 J Average PPRE_STBY: 15.859325 W
Total EPRE_PDN: 0.000293559 J Average PPRE_PDN: 12.497176 W
Total EACT_PDN: 0 J Average PACT_PDN: 0.000000 W
Total Energy: 0.00688803 J Average Power: 293.232601 W

Components of Active-Standby Energies:

Total EACT_STBY - Wakeup: 0 J Average EACT_STBY - Wakeup: 0.000000 W

Components of Precharge-Standby Energies:

Total EPRE_STBY - Wakeup: 0.000127956 J Average EPRE_STBY - Wakeup: 5.447235 W
Total EPRE_STBY - Device-Idle: 0.00024458 J Average EPRE_STBY - Device-Idle: 10.412090 W
Total EPRE_STBY - tRP: 0 J Average EPRE_STBY - tRP: 0.000000 W

***** PER-MC ENERGY STATISTICS *****

	ACT	WR	RD	DQ	REF	ACT_STBY	PRE_STBY	ACT_PDN	PRE_PDN		
MC 0	0	0.000331	0.000000	0.000120	0.000025	0.000032	0.000269	0.000049	0.000000	0.000037	
MC 1	1	0.000331	0.000000	0.000120	0.000025	0.000032	0.000270	0.000049	0.000000	0.000036	
MC 2	2	0.000331	0.000000	0.000120	0.000025	0.000032	0.000271	0.000049	0.000000	0.000036	
MC 3	3	0.000331	0.000000	0.000120	0.000025	0.000032	0.000271	0.000049	0.000000	0.000036	
MC 4	4	0.000331	0.000000	0.000120	0.000025	0.000032	0.000270	0.000049	0.000000	0.000036	
MC 5	5	0.000331	0.000000	0.000120	0.000025	0.000032	0.000262	0.000030	0.000000	0.000038	
MC 6	6	0.000331	0.000000	0.000120	0.000025	0.000032	0.000270	0.000049	0.000000	0.000036	
MC 7	7	0.000331	0.000000	0.000120	0.000025	0.000032	0.000270	0.000049	0.000000	0.000036	

***** PER-MC POWER STATISTICS *****

	ACT	WR	RD	DQ	REF	ACT_STBY	PRE_STBY	ACT_PDN	PRE_PDN		
MC 0	0	14.081	0.000	5.120	1.055	1.377	11.471	2.087	0.000	1.555	
MC 1	1	14.092	0.000	5.124	1.056	1.377	11.487	2.084	0.000	1.554	
MC 2	2	14.103	0.000	5.128	1.056	1.377	11.540	2.081	0.000	1.551	
MC 3	3	14.081	0.000	5.120	1.055	1.377	11.520	2.076	0.000	1.552	
MC 4	4	14.092	0.000	5.124	1.056	1.377	11.506	2.089	0.000	1.552	
MC 5	5	14.103	0.000	5.128	1.056	1.377	11.166	1.274	0.000	1.626	
MC 6	6	14.092	0.000	5.124	1.056	1.377	11.511	2.080	0.000	1.553	

MC 7 14.092 0.000 5.124 1.056 1.377 11.483 2.088 0.000 1.554

***** POWER-DOWN STATISTICS (In Processor Clocks)*****

Number of rank power downs: 5618
Number of Delayed Requests Due to Power-Down: 5618

***** ROW-HIT STATISTICS *****

Total Row Hits: 2
Address-Reuse Hits: 0

Total Row Misses: 9998
Cold-Bank Misses: 256
Row-Switch Misses: 9742

***** UTILIZATION STATISTICS *****

MEMORY-BUS
0 1 2 3 4 5 6 7

0.425 0.426 0.426 0.425 0.426 0.426 0.426 0.426

***** MC-SMI Bus Usage (In Processor Clocks)*****

Port Bus 0: Read Delay 7150 (average 11, num 628), Write Delay 0 (average 0, num 0)
Port Bus 1: Read Delay 7222 (average 11, num 630), Write Delay 0 (average 0, num 0)
Port Bus 2: Read Delay 7653 (average 12, num 636), Write Delay 0 (average 0, num 0)
Port Bus 3: Read Delay 7622 (average 12, num 634), Write Delay 0 (average 0, num 0)
Port Bus 4: Read Delay 7396 (average 12, num 630), Write Delay 0 (average 0, num 0)
Port Bus 5: Read Delay 1288 (average 31, num 41), Write Delay 0 (average 0, num 0)
Port Bus 6: Read Delay 7456 (average 12, num 634), Write Delay 0 (average 0, num 0)
Port Bus 7: Read Delay 7167 (average 11, num 627), Write Delay 0 (average 0, num 0)

***** STALL STATISTICS *****

Total Number of Stalls: 0

Total Number of Real Stalls: 0, Time Stalled = 0

PER-MC STALLS

0 1 2 3 4 5 6 7

0 0 0 0 0 0 0 0

***** DRAM MODES (Percentage of Overall Time)*****

	nReqs	nActs	Time(pclks)	ACT_STBY	ACT_STBY_WAKEUP	PRE_STBY_WAKEUP	PRE_STBY_DEV_IDL						
Dev-00-00	157	157	46980	27.79%	0.00%	1.72%	4.18%	0.00%	6.68%	0.00%	0.89%	0.00%	65.42%
Dev-00-01	156	156	46980	27.70%	0.00%	1.70%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.56%
Dev-00-02	156	156	46980	27.64%	0.00%	1.70%	4.17%	0.00%	6.64%	0.00%	0.89%	0.00%	65.60%
Dev-00-03	156	156	46980	27.59%	0.00%	1.70%	4.14%	0.00%	6.64%	0.00%	0.89%	0.00%	65.67%
Dev-00-04	156	156	46980	27.54%	0.00%	1.68%	4.20%	0.00%	6.64%	0.00%	0.89%	0.00%	65.68%
Dev-00-05	156	156	46980	27.60%	0.00%	1.72%	4.20%	0.00%	6.64%	0.00%	0.89%	0.00%	65.58%
Dev-00-06	156	156	46980	27.65%	0.00%	1.68%	4.09%	0.00%	6.64%	0.00%	0.89%	0.00%	65.68%
Dev-00-07	156	156	46980	27.72%	0.00%	1.72%	4.17%	0.00%	6.64%	0.00%	0.89%	0.00%	65.49%
Dev-01-00	158	158	46980	27.91%	0.00%	1.75%	4.28%	0.00%	6.73%	0.00%	0.89%	0.00%	65.17%
Dev-01-01	156	156	46980	27.65%	0.00%	1.68%	4.09%	0.00%	6.64%	0.00%	0.89%	0.00%	65.68%
Dev-01-02	156	156	46980	27.67%	0.00%	1.70%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.58%
Dev-01-03	156	156	46980	27.70%	0.00%	1.70%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.56%
Dev-01-04	156	156	46980	27.66%	0.00%	1.70%	4.17%	0.00%	6.64%	0.00%	0.89%	0.00%	65.57%
Dev-01-05	156	156	46980	27.61%	0.00%	1.70%	4.11%	0.00%	6.64%	0.00%	0.89%	0.00%	65.67%
Dev-01-06	156	156	46980	27.62%	0.00%	1.68%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.65%
Dev-01-07	156	156	46980	27.69%	0.00%	1.72%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.54%
Dev-02-00	158	158	46980	28.08%	0.00%	1.75%	4.22%	0.00%	6.73%	0.00%	0.89%	0.00%	65.06%
Dev-02-01	156	156	46980	27.75%	0.00%	1.68%	4.10%	0.00%	6.64%	0.00%	0.89%	0.00%	65.58%
Dev-02-02	157	157	46980	27.91%	0.00%	1.75%	4.17%	0.00%	6.68%	0.00%	0.89%	0.00%	65.28%
Dev-02-03	156	156	46980	27.75%	0.00%	1.70%	4.13%	0.00%	6.64%	0.00%	0.89%	0.00%	65.53%
Dev-02-04	156	156	46980	27.89%	0.00%	1.70%	4.09%	0.00%	6.64%	0.00%	0.89%	0.00%	65.43%
Dev-02-05	156	156	46980	27.74%	0.00%	1.70%	4.10%	0.00%	6.64%	0.00%	0.89%	0.00%	65.57%
Dev-02-06	156	156	46980	27.75%	0.00%	1.70%	4.14%	0.00%	6.64%	0.00%	0.89%	0.00%	65.52%
Dev-02-07	156	156	46980	27.69%	0.00%	1.72%	4.17%	0.00%	6.64%	0.00%	0.89%	0.00%	65.52%
Dev-03-00	157	157	46980	27.83%	0.00%	1.75%	4.17%	0.00%	6.68%	0.00%	0.89%	0.00%	65.36%
Dev-03-01	156	156	46980	27.75%	0.00%	1.70%	4.12%	0.00%	6.64%	0.00%	0.89%	0.00%	65.53%
Dev-03-02	156	156	46980	27.86%	0.00%	1.70%	4.12%	0.00%	6.64%	0.00%	0.89%	0.00%	65.43%
Dev-03-03	156	156	46980	27.74%	0.00%	1.70%	4.10%	0.00%	6.64%	0.00%	0.89%	0.00%	65.56%
Dev-03-04	156	156	46980	27.75%	0.00%	1.70%	4.12%	0.00%	6.64%	0.00%	0.89%	0.00%	65.54%
Dev-03-05	156	156	46980	27.61%	0.00%	1.72%	4.19%	0.00%	6.64%	0.00%	0.89%	0.00%	65.59%
Dev-03-06	156	156	46980	27.76%	0.00%	1.68%	4.10%	0.00%	6.64%	0.00%	0.89%	0.00%	65.56%
Dev-03-07	156	156	46980	27.88%	0.00%	1.72%	4.10%	0.00%	6.64%	0.00%	0.89%	0.00%	65.40%
Dev-04-00	158	158	46980	27.94%	0.00%	1.75%	4.26%	0.00%	6.73%	0.00%	0.89%	0.00%	65.16%
Dev-04-01	156	156	46980	27.65%	0.00%	1.70%	4.16%	0.00%	6.64%	0.00%	0.89%	0.00%	65.59%
Dev-04-02	156	156	46980	27.82%	0.00%	1.70%	4.13%	0.00%	6.64%	0.00%	0.89%	0.00%	65.46%
Dev-04-03	156	156	46980	27.64%	0.00%	1.70%	4.17%	0.00%	6.64%	0.00%	0.89%	0.00%	65.60%
Dev-04-04	156	156	46980	27.74%	0.00%	1.70%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.52%
Dev-04-05	156	156	46980	27.58%	0.00%	1.72%	4.19%	0.00%	6.64%	0.00%	0.89%	0.00%	65.62%
Dev-04-06	156	156	46980	27.68%	0.00%	1.68%	4.15%	0.00%	6.64%	0.00%	0.89%	0.00%	65.59%
Dev-04-07	156	156	46980	27.84%	0.00%	1.72%	4.11%	0.00%	6.64%	0.00%	0.89%	0.00%	65.43%
Dev-05-00	157	157	46980	27.01%	0.00%	3.36%	0.29%	0.00%	6.68%	0.00%	0.89%	0.00%	68.44%
Dev-05-01	156	156	46980	26.86%	0.00%	3.30%	0.21%	0.00%	6.64%	0.00%	0.89%	0.00%	68.74%
Dev-05-02	157	157	46980	26.99%	0.00%	3.34%	0.29%	0.00%	6.68%	0.00%	0.89%	0.00%	68.49%
Dev-05-03	156	156	46980	26.77%	0.00%	3.36%	0.26%	0.00%	6.64%	0.00%	0.89%	0.00%	68.72%
Dev-05-04	156	156	46980	26.95%	0.00%	3.32%	0.25%	0.00%	6.64%	0.00%	0.89%	0.00%	68.59%
Dev-05-05	156	156	46980	26.83%	0.00%	3.34%	0.20%	0.00%	6.64%	0.00%	0.89%	0.00%	68.73%
Dev-05-06	156	156	46980	26.87%	0.00%	3.36%	0.19%	0.00%	6.64%	0.00%	0.89%	0.00%	68.68%
Dev-05-07	157	157	46980	27.07%	0.00%	3.38%	0.20%	0.00%	6.68%	0.00%	0.89%	0.00%	68.45%
Dev-06-00	157	157	46980	27.78%	0.00%	1.75%	4.20%	0.00%	6.68%	0.00%	0.89%	0.00%	65.39%
Dev-06-01	156	156	46980	27.79%	0.00%	1.70%	4.09%	0.00%	6.64%	0.00%	0.89%	0.00%	65.53%
Dev-06-02	156	156	46980	27.81%	0.00%	1.70%	4.11%	0.00%	6.64%	0.00%	0.89%	0.00%	65.47%
Dev-06-03	156	156	46980	27.68%	0.00%	1.70%	4.13%	0.00%	6.64%	0.00%	0.89%	0.00%	65.59%
Dev-06-04	156	156	46980	27.66%	0.00%	1.70%	4.14%	0.00%	6.64%	0.00%	0.89%	0.00%	65.60%

```

Dev-06-05 156 156 46980 27.60% 0.00% 1.72% 4.20% 0.00% 6.64% 0.00% 0.89% 0.00% 65.59%
Dev-06-06 156 156 46980 27.69% 0.00% 1.68% 4.11% 0.00% 6.64% 0.00% 0.89% 0.00% 65.62%
Dev-06-07 157 157 46980 27.98% 0.00% 1.75% 4.10% 0.00% 6.68% 0.00% 0.89% 0.00% 65.28%
Dev-07-00 157 157 46980 27.72% 0.00% 1.75% 4.22% 0.00% 6.68% 0.00% 0.89% 0.00% 65.42%
Dev-07-01 156 156 46980 27.63% 0.00% 1.70% 4.13% 0.00% 6.64% 0.00% 0.89% 0.00% 65.64%
Dev-07-02 156 156 46980 27.74% 0.00% 1.70% 4.11% 0.00% 6.64% 0.00% 0.89% 0.00% 65.54%
Dev-07-03 156 156 46980 27.65% 0.00% 1.70% 4.16% 0.00% 6.64% 0.00% 0.89% 0.00% 65.60%
Dev-07-04 156 156 46980 27.58% 0.00% 1.70% 4.18% 0.00% 6.64% 0.00% 0.89% 0.00% 65.64%
Dev-07-05 156 156 46980 27.58% 0.00% 1.72% 4.18% 0.00% 6.64% 0.00% 0.89% 0.00% 65.62%
Dev-07-06 156 156 46980 27.61% 0.00% 1.68% 4.18% 0.00% 6.64% 0.00% 0.89% 0.00% 65.64%
Dev-07-07 157 157 46980 27.93% 0.00% 1.75% 4.10% 0.00% 6.68% 0.00% 0.89% 0.00% 65.33%

```

***** RESPONSE TIME STATISTICS (In Processor Clocks)*****

BOX 1: Read Request-Time

statistics on elapsed times

minimum	105.000000	mean	145.382600
maximum	404.000000	variance	1547.488166
range	299.000000	standard deviation	39.338126
observations	10000	coefficient of var	0.270583

lower limit	frequency	proportion	cumulative proportion	
100.00000	5510	0.551000	0.551000	*****
125.00000	43	0.004300	0.555300	.
150.00000	1845	0.184500	0.739800	*****
175.00000	1260	0.126000	0.865800	*****
200.00000	1251	0.125100	0.990900	*****
225.00000	24	0.002400	0.993300	.
250.00000	14	0.001400	0.994700	.
275.00000	16	0.001600	0.996300	.
300.00000	10	0.001000	0.997300	.
325.00000	3	0.000300	0.997600	.
350.00000	18	0.001800	0.999400	.
375.00000	4	0.000400	0.999800	.
400.00000	2	0.000200	1.000000	.

statistics on population

initial	0	minimum	0	mean	30.945636
final	0	maximum	76	variance	260.114456
entries	10000	range	76	standard deviation	16.128064
exits	10000			coeff of variation	0.521174

BOX 2: Write Request-Time

statistics on elapsed times

> no data recorded since creation or reset


```

statistics on population

initial      0      minimum      0      mean      0.000000
final       0      maximum      0      variance   0.000000
entries     0      range        0      standard deviation 0.000000
exits       0      coeff of variation 0.000000

```

All of the output is sent to stderr. The output has the following sections:

- Command line and configuration options.
- Overall execution and refresh statistics (the latter can be ignored except when developing extensions or changing the refresh code).
- Total SDRAM Energy and Power Statistics (with breakdown of consumption in different activities/states).
- Energy totals per memory controller.
- If power management was used, then the number of times power down was used and its effect on subsequent requests.
- SDRAM row/page hit statistics - can be used as a measure of how effective a particular page policy would be (stats are independent of actual policy used). A row hit indicates a new request to a bank was to the same row (open or closed) as the previous request to that bank and a miss indicates that new request is to a different row.
- Memory controller (MC) bus utilization (fraction of time used), delay on the MC-SMI bus due to contention between the multiple ranks and banks under the SMI.
- Statistics on the stalls the simulator encountered due to the controller queues being full (whenever any of the memory controller's queue is full, the entire simulation stalls).
- Rank-wise access/state statistics on the SDRAM devices.
- Read and write request response time statistics - note the stall time does not count towards the response time statistics (response time begins only after the request has been added to a memory controller queue).

3.2 Example 2: Modeling a small (no-SMI) system

For example 2, consider a smaller system with a single memory controller, no SMIs, and 4 uni-ranked DIMMs with width 64, with the memory controller accessing 2 DIMMs at a time (i.e. a 128-bit path from the memory controller to the DIMMs). This organization would be similar to a 'dual-bank' PC memory system with 2 slots. For a cache-line size of 128 bytes, a burst length of 8 would suffice to access a complete cache line in one request. To model the absence of the SMIs, but yet model the contention on the SDRAM to memory controller buses we can adopt the following approach in the current version of Memsim. On the command line we can use the options `bufferedsdmireads=0` and `bufferedsdmiwrites=0` (or equivalent lines in the configuration file), `smisharedrw` and `smicmddelay=0`. This will force the code to model a zero delay SMI, with the only latency seen between the memory controller and SDRAM being the SDRAM access time. The contention for the controller-DIMM bus between the multiple ranks or slots (and banks on the same rank) will still be modeled for this zero latency SMI.

The output for such a run is given below:

Modeling zero processing delay at SMI

Command: ./memsim config=sample2.config source=sample2.trace smisharedrw smicmddelay=0 smire

Physical Memory Size (MB): 1024

Memory Controllers: 1

Enable MC Interleaving: 0

MC Queue Length: 16

Bus Retry Cycles: 32

SMIs Per Port: 2

DIMMs Per SMI: 2

Banks Per Chip: 4

DIMM Width: 64

Device Width: 8

Device Capacity: 32768

Devices Per DIMM: 8

DRAM Burst Length: 8

Rows Per Bank: 8192

Columns Per Bank: 1024

Enable Interleaved Bank-Fill: 1

DRAM Page Policy: 1

Enable Hybrid Page Policy: 0

Enable Power Management: 1

Power Management Policy: 0

Power Down Check Interval: 0

Enable Interleaved Rank-Fill: 1

Number of records to be processed: 0

Buffered SMI Reads: 0

Buffered SMI Writes: 0

***** OVERALL EXECUTION STATISTICS *****

Total Simulation Time: 1622660.000000 Processor Clocks (0.000811 Seconds)

Number of records read 10000, requests processed 10000

REFRESH Tried = 208, Delayed 208, Maximum Delay 151.00, Average Delay 111.71, Average Access

***** ENERGY STATISTICS *****

Total EACT: 0.000548034 J Average PACT: 0.675476 W

Total EWR: 0.000167644 J Average PWR: 0.206629 W

Total ERD: 0.000773452 J Average PRD: 0.953314 W

Total EDQ: 0.000159334 J Average PDQ: 0.196387 W

Total EREF: 0.000140207 J Average PREF: 0.172812 W

Total EACT_STBY: 0.00103451 J Average PACT_STBY: 1.275077 W

Total EPRE_STBY: 0.000154144 J Average PPRE_STBY: 0.189990 W

Total EPRE_PDN: 3.59778e-07 J Average PPRE_PDN: 0.000443 W

Total EACT_PDN: 0.00168486 J Average PACT_PDN: 2.076667 W

Total Energy: 0.00466255 J Average Power: 5.746794 W

Components of Active-Standby Energies:

Total EACT_STBY - Wakeup: 0.000103613 J Average EACT_STBY - Wakeup: 0.127708 W

Components of Precharge-Standby Energies:

Total EPRE_STBY - Wakeup: 2.14444e-06 J Average EPRE_STBY - Wakeup: 0.002643 W

Total EPRE_STBY - Device-Idle: 0.000152 J Average EPRE_STBY - Device-Idle: 0.187347 W
Total EPRE_STBY - tRP: 0 J Average EPRE_STBY - tRP: 0.000000 W

***** PER-MC ENERGY STATISTICS *****

ACT WR RD DQ REF ACT_STBY PRE_STBY ACT_PDN PRE_PDN
MC 0 0.000548 0.000168 0.000773 0.000159 0.000140 0.001035 0.000154 0.001685 0.000000

***** PER-MC POWER STATISTICS *****

ACT WR RD DQ REF ACT_STBY PRE_STBY ACT_PDN PRE_PDN
MC 0 0.675 0.207 0.953 0.196 0.173 1.275 0.190 2.077 0.000

***** POWER-DOWN STATISTICS (In Processor Clocks)*****

Number of rank power downs: 8033
Number of Delayed Requests Due to Power-Down: 8033

***** ROW-HIT STATISTICS *****

Total Row Hits: 6015
Address-Reuse Hits: 0

Total Row Misses: 3985
Cold-Bank Misses: 2
Row-Switch Misses: 3983

***** UTILIZATION STATISTICS *****

MEMORY-BUS
0

0.099

***** MC-SMI Bus Usage (In Processor Clocks)*****

Port Bus 0: Delay 2495 (average 30, num 83)

***** STALL STATISTICS *****

Total Number of Stalls: 0

Total Number of Real Stalls: 0, Time Stalled = 0

PER-MC STALLS

0

0

***** DRAM MODES (Percentage of Overall Time)*****

	nReqs	nActs	Time(pclks)	ACT_STBY	ACT_STBY_WAKEUP	PRE_STBY_WAKEUP	PRE_STBY_DEV_IDL
Dev-00-00	5984	4004	1622660	30.31%	2.46%	0.05%	7.74%
Dev-00-01	4016	135	1622660	18.87%	2.46%	0.06%	0.69%

***** RESPONSE TIME STATISTICS (In Processor Clocks)*****

BOX 1: Read Request-Time

statistics on elapsed times

minimum	83.000000	mean	111.021663
maximum	422.000000	variance	1202.367106
range	339.000000	standard deviation	34.675166
observations	8032	coefficient of var	0.312328

lower limit	frequency	proportion	cumulative proportion	
75.00000	5842	0.727341	0.727341	*****
100.00000	66	0.008217	0.735558	.
125.00000	28	0.003486	0.739044	.
150.00000	1985	0.247136	0.986180	*****
175.00000	6	0.000747	0.986927	.
200.00000	2	0.000249	0.987176	.
225.00000	18	0.002241	0.989417	.
250.00000	19	0.002366	0.991783	.
275.00000	19	0.002366	0.994148	.
300.00000	12	0.001494	0.995642	.
325.00000	11	0.001370	0.997012	.
350.00000	9	0.001121	0.998132	.
375.00000	7	0.000872	0.999004	.
400.00000	8	0.000996	1.000000	.

statistics on population

initial	0	minimum	0	mean	0.549546
final	0	maximum	2	variance	0.341874
entries	8032	range	2	standard deviation	0.584700
exits	8032			coeff of variation	1.063970

BOX 2: Write Request-Time

statistics on elapsed times

minimum	128.000000	mean	140.816565
---------	------------	------	------------

maximum	379.000000	variance	535.857031
range	251.000000	standard deviation	23.148586
observations	1968	coefficient of var	0.164388

lower limit	frequency	proportion	cumulative proportion	
125.00000	1923	0.977134	0.977134	*****
150.00000	1	0.000508	0.977642	.
175.00000	1	0.000508	0.978150	.
200.00000	7	0.003557	0.981707	.
225.00000	3	0.001524	0.983232	.
250.00000	3	0.001524	0.984756	.
275.00000	20	0.010163	0.994919	.
300.00000	3	0.001524	0.996443	.
325.00000	1	0.000508	0.996951	.
350.00000	0	0.000000	0.996951	.
375.00000	6	0.003049	1.000000	.

statistics on population

initial	0	minimum	0	mean	0.170786
final	0	maximum	1	variance	0.141618
entries	1968	range	1	standard deviation	0.376322
exits	1968			coeff of variation	2.203473

3.3 Example 3: Running Memsim as a Memory Service Program

In addition to using Memsim as a stand-alone, trace-driven memory system simulator it can also be used as a component of a larger simulation environment or as a memory system service to a client program. The API supporting this is contained in the files `memsim_comm.h` and `memsim_service.c`. The attached Makefile also generates a sample client program `memsim_client`. `memsim_client` reads in memory requests from a trace file and feeds it to `memsim` just like requests could be fed to `memsim` from a different simulation environment. Both programs synchronize with the other for each of the (processor) clock cycles they simulate. To use `memsim` in this fashion, one would first start `memsim` as follows:

```
memsim config=sample.config source=SHM <any other options>
```

And then the program feeding the requests to Memsim would be started. If that program is `memsim_client` it takes in two arguments - the first one is the trace file and the second one is the number of records, for e.g. :

```
memsim_client sample.trace 9999
```

For another program to synchronize and use Memsim it would have to use the Memsim communication API as it is used in the `main()` function in `memsim_service` corresponding to `memsim_client`.