Improving Network Response Time and Reducing PCI Bus Traffic Using a Network Processor with Memory

Mark Doughty
Shawn Koch

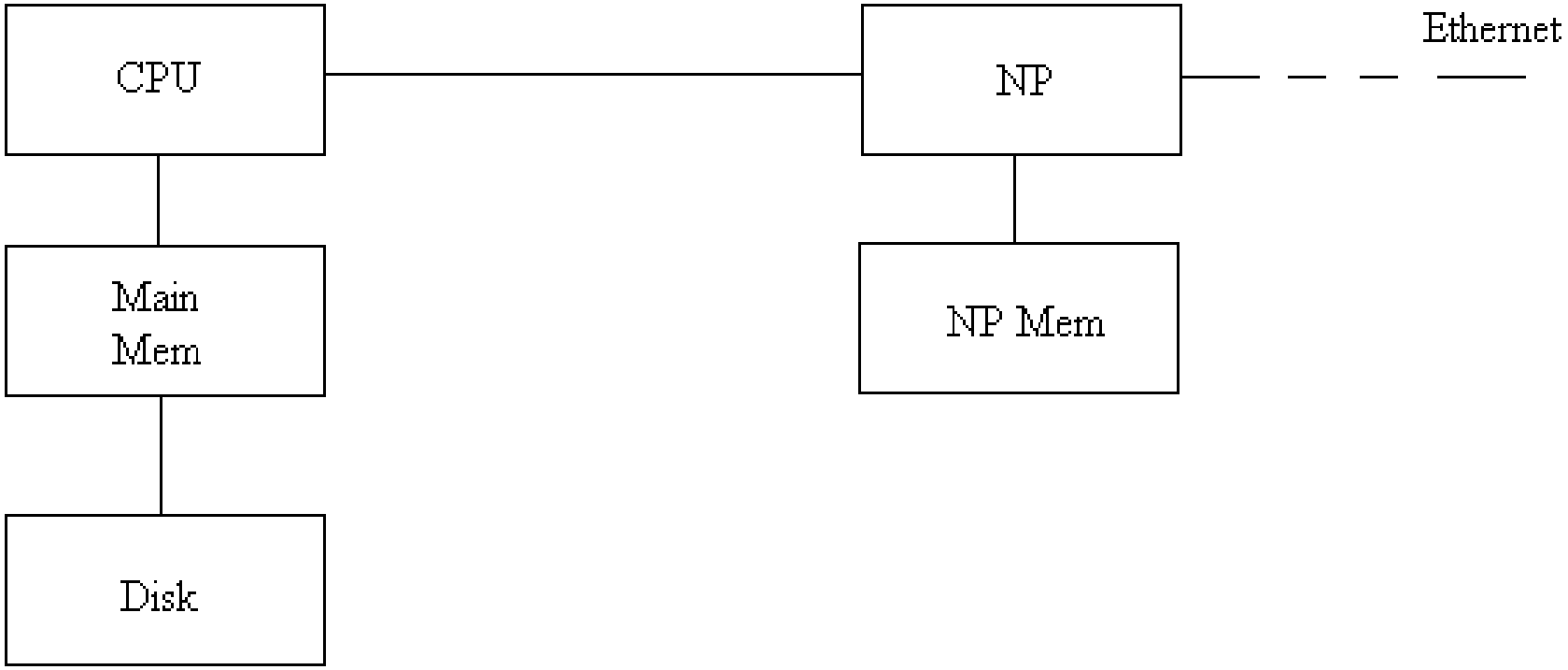
Motivation

- To improve throughput performance for internet web servers
- Number of clients for many web servers increasing significantly every year
- Free up PCI bus bandwidth for host CPU

Concept and Hypothesis

- Cache server requests using a network processor (NP) connected to a small, fast external memory (DRAM)
- We believe that increasing the size of the NP memory will yield both greater throughput and reduced PCI bus traffic for requests

System Block Diagram



Architecture

- System Simulator developed using C++
- Components Modeled
  - CPU, NP
  - CPU main memory and Disk
  - NP external memory
  - Ethernet
  - PCI Bus
  - Requests and Responses
- Each component modeled with variable parameters

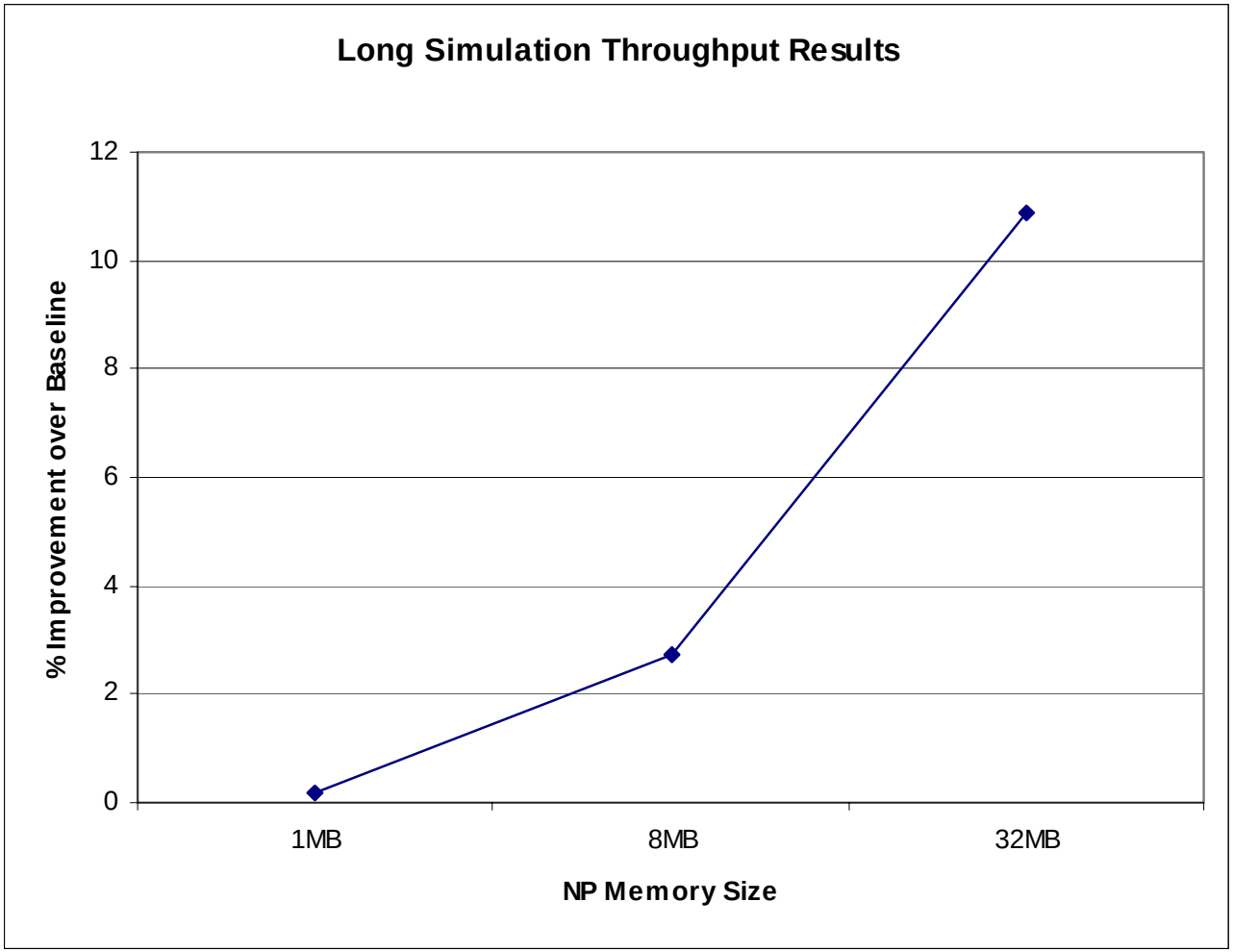
Configuration Files

- Consists of all parameters and debug flags
- 26 different configuration files were run on the simulator
- 2 baseline configuration files, one for “long” simulations and one for “short” sims

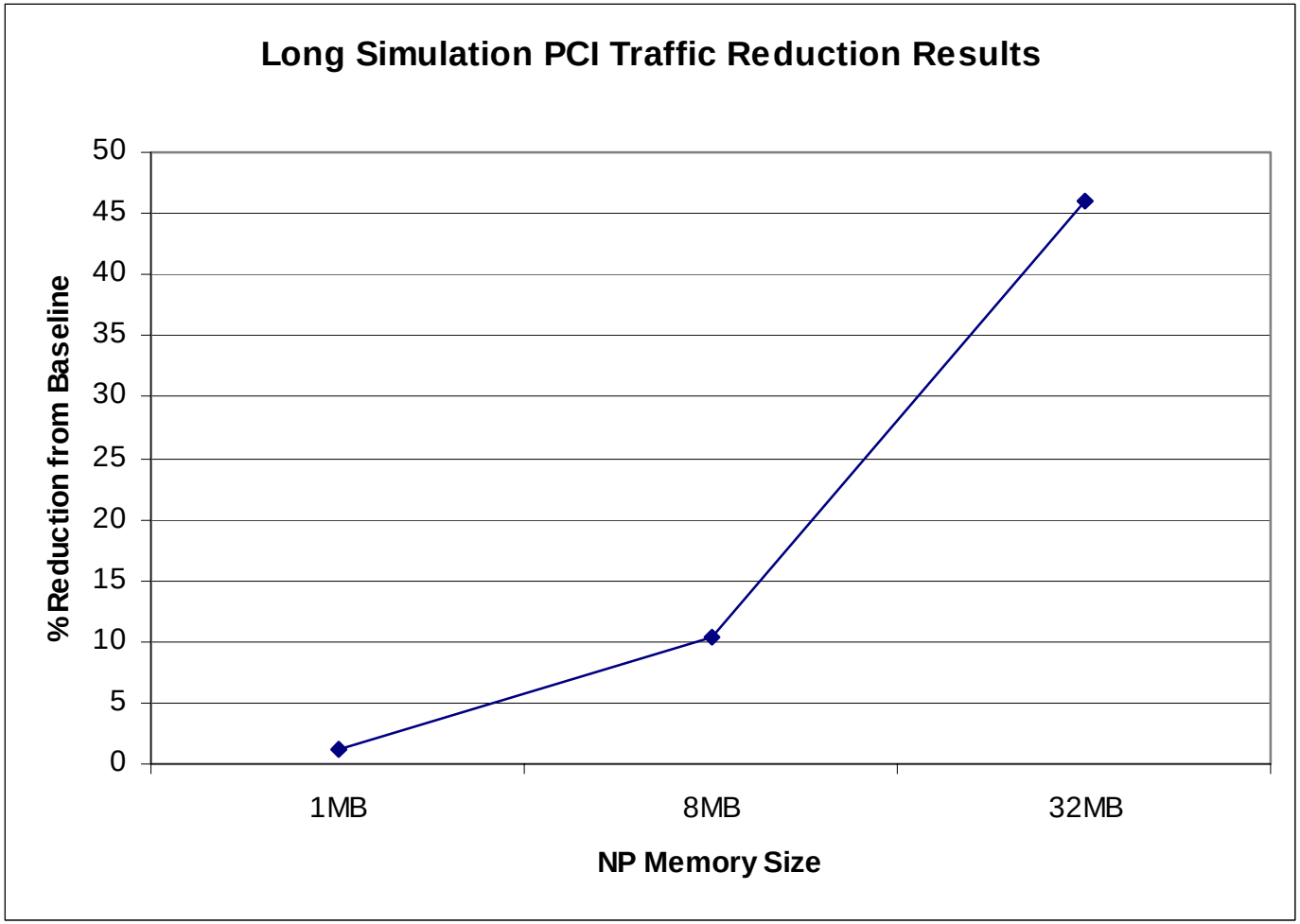
Simulations

- Did not include disk due to unreasonable simulation time duration
- Scaled parameters to increase simulation speed
- Parameters based on current technology
- Measured average response time and total PCI bus traffic
- Request IDs and response sizes randomly generated

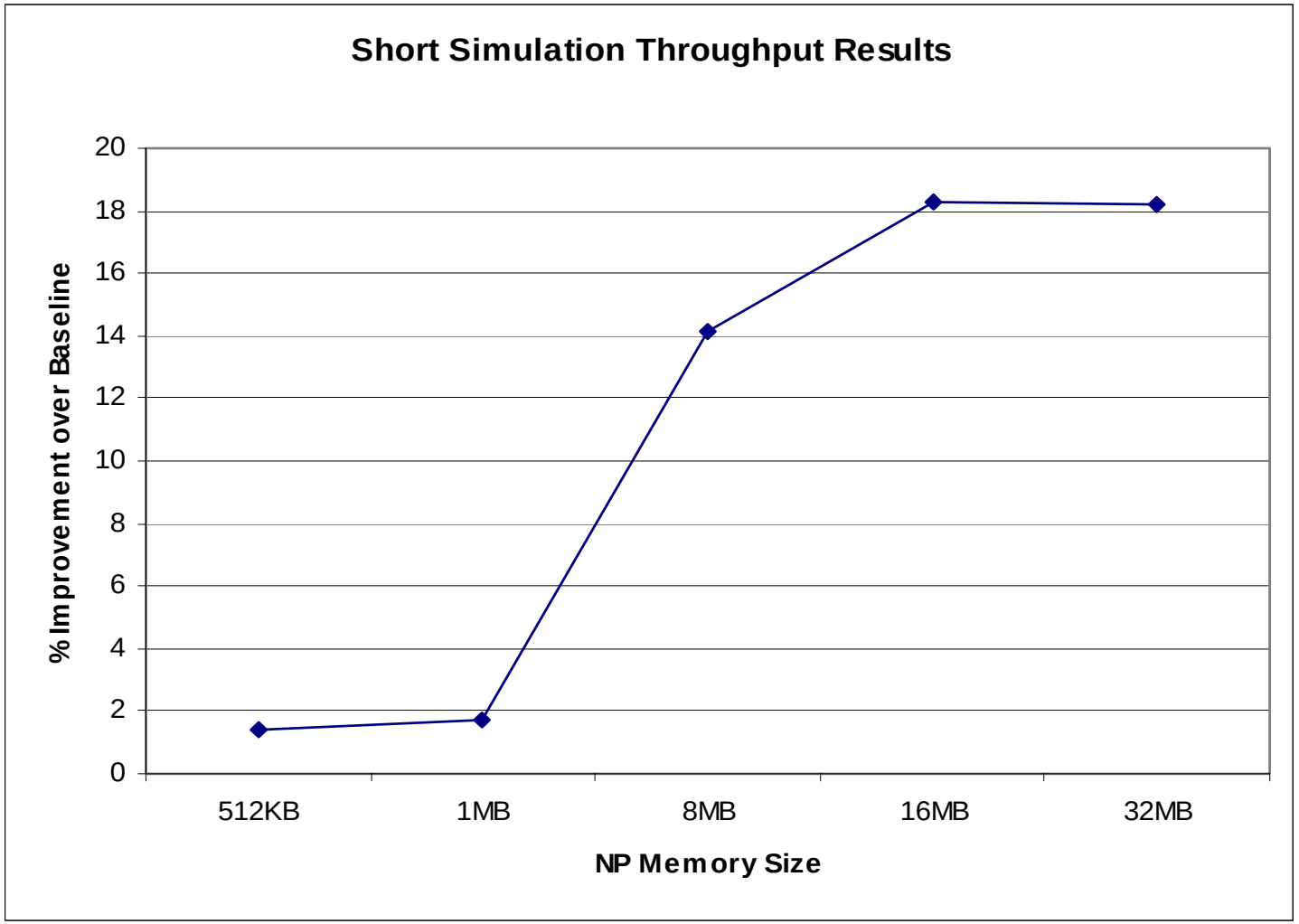
Results


Long Simulation Throughput Results
% Improvement over Baseline
12
10
8
6
4
2
0
1MB
8MB
32MB
NP Memory Size

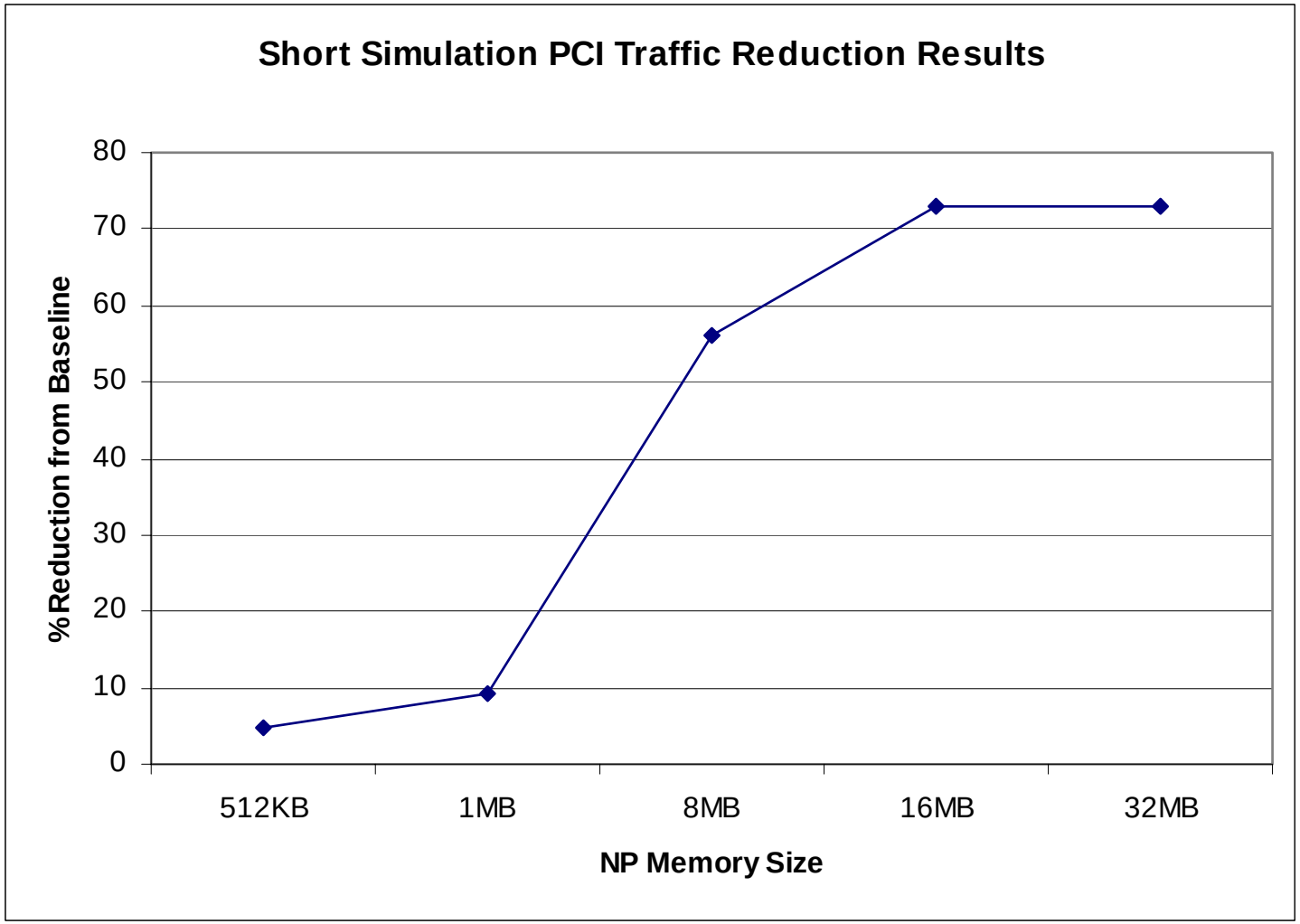
Results


Long Simulation PCI Traffic Reduction Results
% Reduction from Baseline
50
45
40
35
30
25
20
15
10
5
0
1MB
8MB
32MB
NP Memory Size

Results


Short Simulation Throughput Results
% Improvement over Baseline
20
18
16
14
12
10
8
6
4
2
0
512KB
1MB
8MB
16MB
32MB
NP Memory Size

Results


Short Simulation PCI Traffic Reduction Results
% Reduction from Baseline
80
70
60
50
40
30
20
10
0
512KB
1MB
8MB
16MB
32MB
NP Memory Size

Simulations

Simulation Name	Parameter Varied From Baseline	% Throughput Improvement	% PCI Traffic Reduction
LongBaseline	Default Basline Settings Used, no NP mem	N/A	N/A
Long1MB	NP memory size set to 1MB	0.18	1.18
Long8MB	NP memory size set to 8MB	2.71	10.45
Long32MB	NP memory size set to 32MB	10.89	45.93
ShortBaseline	Default Basline Settings Used, no NP mem	N/A	N/A
Short512KB	NP memory size set to 512MB	1.41	4.67
Short1MB	NP memory size set to 1MB	1.73	9.10
Short8MB	NP memory size set to 8MB	14.12	55.99
Short16MB	NP memory size set to 16MB	18.26	72.97
Short32MB	NP memory size set to 32MB	18.19	72.97
ShortLowLatBase	Baseline with lower main memory latency	2.31	0.981
Short500MHzNP	8MB NP memory, NP speed = 500MHz	14.6	54.97
Short5000DiffReqs	8MB NP memory, 5000 diff. Request IDs	3.06	12.68
Short10000DiffReqs	8MB NP memory, 10000 diff. Request IDs	1.99	5.91
ShortLongNPLat	8MB NP memory, NP mem. lat. increase	-3.09	55.90
ShortSmallResp	8MB NP memory, Response size shorter	19.67	73.79
ShortBigResp	8MB NP memory, Response size longer	-17.46	-62.12
ShortBigLineSize	Baseline with increased line size for mem.	2.82	2.07
ShortBest	32MB NP memory, line size bigger NP mem	19.45	72.97
ShortFastBus	Baseline with fast bus (2x)	11.08	N/A
ShortSlowBus	Baseline with slow bus (2x)	-21.87	N/A
Short8MBFastBus	8MB NP memory, fast bus (2x)	16.72	55.99
Short8MBSlowBus	8MB NP memory, slow bus (2x)	5.77	55.99

Conclusion

- Experimental results show that this is an effective method for servers dealing primarily with static requests
- Further simulations required for analysis of dynamic content
- Best results seen when all static requests fit in NP memory
- Good method for increasing throughput in current server systems

Discussion