

White Paper

**Intel Information
Technology**

Streaming, VHD, Computing
Models, Client Virtualization

Streaming and Virtual Hosted Desktop Study

Benchmarking Results

As part of an ongoing evaluation of emerging computing models for enterprise operations, Intel Information Technology studied the impacts of streamed client and Virtual Hosted Desktop (VHD) computing models on server and network utilization. Utilization varied depending on the model studied, illustrating that each model is more efficient for specific workloads. VHD is better suited for standardized applications with relatively static screens, such as data entry. Streaming more efficiently supports multimedia and other dynamic applications, including unified communications.

Contents

Executive Summary.....	2
Background.....	2
Benchmark Study.....	3
Technical Architecture.....	3
Server Utilization Results.....	5
Network Utilization Results.....	6
Conclusion.....	8

Executive Summary

Which computing model makes more efficient use of servers and the network: streaming software to clients that run applications locally, or Virtual Hosted Desktops (VHDs), which rely on remote processing on a virtualized server? Intel Information Technology compared the impact on servers and the network for both models.

We designed and executed an application script to simulate a realistic workload based on standard office applications. We utilized identical server, network, and client configurations for both computing models to minimize any performance differences due to hardware. In addition, for network utilization testing, we included 'traditional' clients with the Operating System (OS) installed locally and evaluated the impact on the network when these clients ran locally embedded applications and when applications were streamed to them.

Benchmark findings were as follows:

- **Server** - Server utilization was significantly lower for streaming than VHD. Compute-intensive and multimedia applications caused high server utilization in VHD. Streaming the applications to the VHD virtual machine improved performance over embedded applications running on the VHD. For 20 clients, server utilization was about 44 percent lower for streaming than VHD. Streaming improved VHD server utilization by about 15 percent.
- **Network** - Network utilization was consistently low for VHD. Streaming over time became more network efficient, as the OS and applications were cached locally. At boot time, streaming briefly consumed the network. For 20 clients, the VHD network sent metric was less than 5 Mbps as compared to 15 Mbps for streaming.

- **Efficiency** - Comparing computing model efficiencies, VHD was more efficient for workloads with low screen refresh. Streaming was more efficient for a wider variety of applications including graphics, multimedia, flash animation, and real-time collaboration.

Background

New computing models are emerging in the enterprise. As networks become more robust, the common theme is to enable centralized, network-based services that are provided on-demand.

One of the computing models that meets these criteria is streamed client computing. With streaming, the server delivers the OS and/or applications over the network for temporary, local execution by clients. OS streaming involves creating and storing a disk image on a server and loading it on the client via the network at boot time. In application streaming, the server stores strategically packaged client applications and distributes them to the clients as users access the application. When a user invokes an application, the server sends the first application execution block to the client, allowing the user to immediately become productive. It sends additional blocks in the background.

Streaming can work with virtualization technologies. For example, streamed applications can execute within a Virtual Machine (VM) on a client or server.

Another computing model is Virtual Hosted Desktops (VHD). In the VHD model, the client desktop environment runs within a virtual machine on a server. The server distributes the user interface to the client hardware using Remote Desktop Protocol (RDP). All processing occurs on the server within the VMs.

Benchmark Study

To evaluate the impact of these computing models, Intel Information Technology constructed a performance study to characterize backend utilization under a typical user workload. The load included standard office suite applications, plus concurrent playing of a video in a media player. We focused on the impact to backend resources by capturing server and network metrics as the load was scaled from one to 20 simultaneous clients.

We evaluated several software delivery configurations.

- Streamed OS with embedded applications: We created an image of office applications traditionally installed on the OS. The server streamed the image to diskless clients.
- Streamed OS with streamed applications: We created an OS image. At boot, the server streamed the OS to diskless clients. As the client accessed these applications, the server streamed the necessary software on-demand. Applications were run in virtual containers. No changes were made to the underlying OS in terms of registry settings or installed DLLs.
- VHD with embedded applications: For each client, we created a VM on the server that ran the OS with installed applications. The server distributed the user interface using RDP.
- VHD with streamed applications: For each client, we created a VM on the server that ran the OS. We packaged the suite of applications and streamed them to the client VM on-demand. Again, the user interface was distributed to the clients using RDP.
- OS running locally with embedded applications: We configured traditional clients with the OS and office applications installed locally on the client hard disk.
- OS running locally with streamed applications: We configured basic traditional clients with only the OS installed locally. Applications were packaged and streamed to the clients. Applications were run locally in virtual containers.

Technical Architecture

Benchmarking took place in the Intel IT lab environment. All software delivery configurations used the same server, client, and network hardware to help ensure a consistent infrastructure for the study. Table 1 lists the hardware and software used. Figure 1 (on the next page) illustrates the infrastructure used in the study.

Hardware/Software	Specifications
Servers	<ul style="list-style-type: none"> ▪ 4-socket Intel® Xeon® processor 7100^A series (dual-core; 8 cores total) ▪ 32 GB RAM ▪ RAID storage array
Diskless Clients (Qty. 20)	<ul style="list-style-type: none"> ▪ Intel® Core™2 Duo processor E6400^A 2.13 GHz ▪ 2 GB RAM
Traditional Clients (Qty. 20)	<ul style="list-style-type: none"> ▪ Intel® Core™2 Duo E6400 2.13 GHz ▪ 1-2 GB RAM
Network	<ul style="list-style-type: none"> ▪ 1 Gbps wired network in the lab ▪ Streamed OS tests configured to multicast the image using UDP, when applicable
OS Streaming Software	<ul style="list-style-type: none"> ▪ Ardence 4.0.1
Application Streaming Software	<ul style="list-style-type: none"> ▪ AppStream 5.2 ▪ AppStream was used with Altiris SVS* 1.4 to run application in virtual containers on the clients
Virtualization Software	<ul style="list-style-type: none"> ▪ Microsoft Virtual Server* 2005 R2 SP1
Virtual Desktop Software	<ul style="list-style-type: none"> ▪ Remote Desktop Client v6.0
Client OS	<ul style="list-style-type: none"> ▪ Microsoft Windows* XP
Test Applications	<ul style="list-style-type: none"> ▪ Microsoft Office* 2003 (Microsoft Word*, Excel*, PowerPoint*, and Access*) ▪ Adobe Acrobat Reader* 7 ▪ Photoshop* CS2 ▪ Windows Media Player* 10
Data Capture Software	<ul style="list-style-type: none"> ▪ Perfmon* for Windows ▪ Perfmon* for Windows Server* 2003
Mouse/Keyboard Capture Software	<ul style="list-style-type: none"> ▪ JitBit Macro Recorder*

Table 1: Test Hardware and Software

Category	Metrics
Logical Disk	<ul style="list-style-type: none"> ▪ % Disk, Read, and Write Times ▪ Avg. Disk Queue Length
Physical Disk	<ul style="list-style-type: none"> ▪ Disk Reads and Writes/sec
Memory	<ul style="list-style-type: none"> ▪ Pages, Faults, DZ Faults, T Faults/sec ▪ Pool Paged and Non-paged Bytes
Network Interface	<ul style="list-style-type: none"> ▪ Bytes Total, Sent, Received/sec ▪ Packets Total, Sent, Received/sec
Network Segment	<ul style="list-style-type: none"> ▪ % Net Utilization, Queue Length, Discards, and Errors
Processor	<ul style="list-style-type: none"> ▪ % Processor, Privileged, User, Interrupt, Idle Time, and DPC Time ▪ Interrupts and C1-C3 Transitions/sec
System	<ul style="list-style-type: none"> ▪ Proc Queue and Context Switches/sec

Table 2: Server Metrics

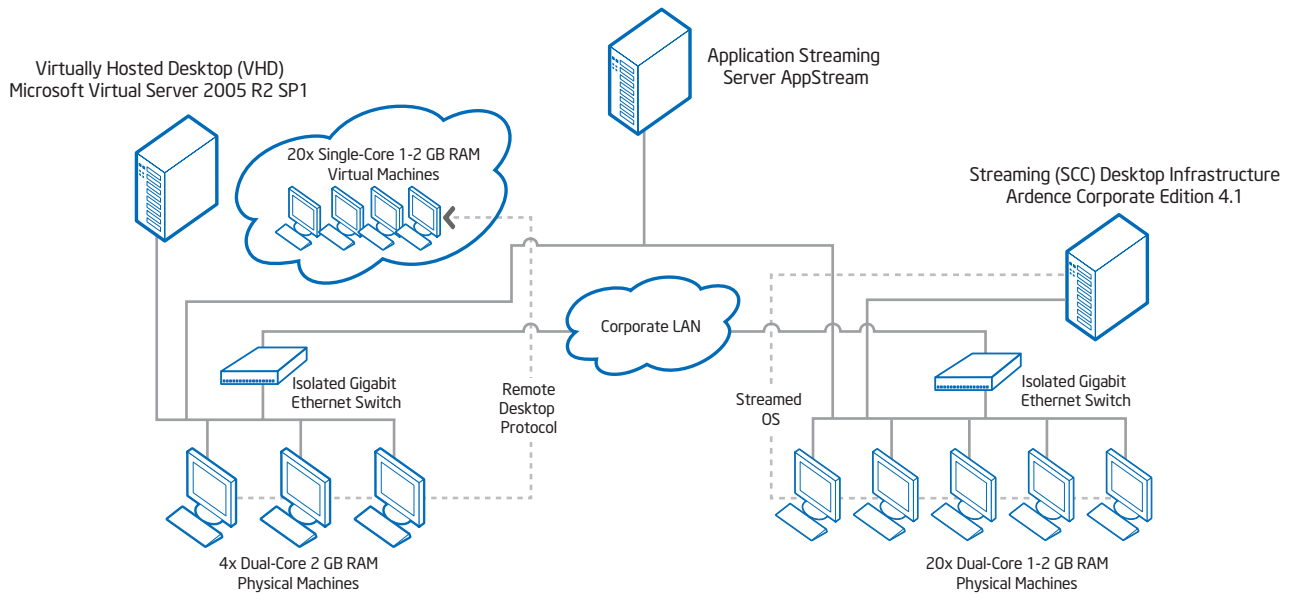


Figure 1: Conceptual Architecture Diagram

We also created a test harness to automate the benchmarking and to provide a consistent “office script” across the configurations.

We created a script that executed several typical user tasks in the same sequence for the tests. For streaming tests, the script executed as follows:

1. Start Script – 20-second pauses between steps.
2. Open Windows Media Player* and start playback of a 12 MB, 15 minute video; move the task to the background.
3. Open Word*; type 1.5 pages of text; copy and paste twice; save; close Word.
4. Open Acrobat Reader*; open Word document; copy/paste twice; insert two pictures into Word document; save; close Word; close Acrobat Reader.
5. Open Photoshop*; open 1 MB JPG; apply artistic transformations to JPG; close without saving.
6. Open Excel* doc, copy and paste multiple times to Word using Paste “Special” (metafile format for graphs) and other options; type paragraph of text; insert picture from file; save Word document; close Word; save Excel document; close Excel.

7. Open PowerPoint*; create a single slide with two text blocks and four resized pictures from file; save document; close PowerPoint.

8. Close Windows Media Player; End Script.

For VHD tests, we executed the same script, except for steps 2 and 5, due to configuration issues discussed later.

The tests executed the script on one client, five clients, ten clients, and 20 clients. For multiple clients, the scripts launched 15 seconds apart to stagger the workload. We added wait times to the script to slow down execution and simulate a “human pace.” This approach also enabled the separation of generated data by task, creating identifiable utilization peaks.

To ensure that we captured enough data for proper analysis, we executed the tests multiple times and averaged three statistically significant test runs. We placed time boxes around the utilization curves, as execution times were almost identical (at human pace). We then derived average utilization for each test run for selected metrics.

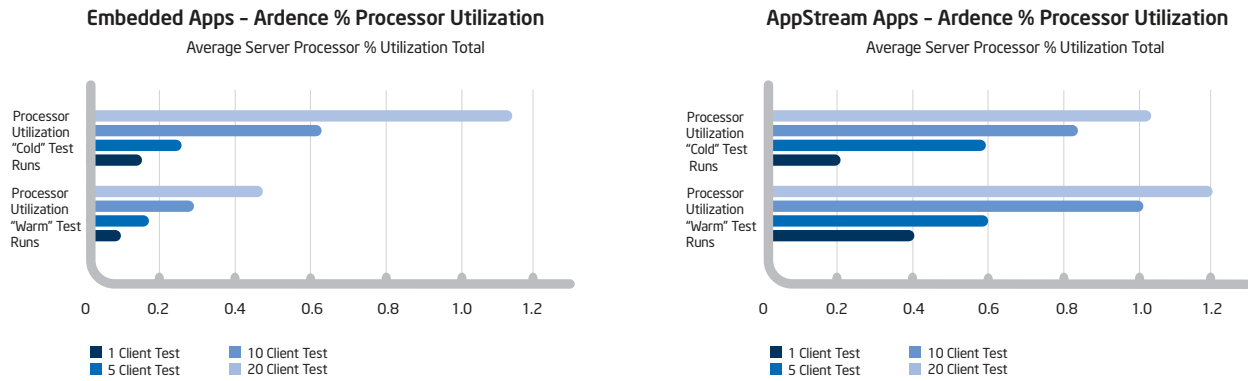


Figure 2: Server Processor Utilization for Streaming

In the streaming model, we configured the OS and applications to cache locally in memory and write back to the server for disk activity and file saves. Once loaded, applications remained resident in memory for subsequent use. We defined a “cold” run as a state when the OS and applications were accessed for the first time. We designated a “warm” run as a state when the OS and applications were already memory resident in the client. We gathered and compared data for both cold and warm runs for the streaming OS models. The concept of “warm” and “cold” test runs is not applicable to the VHD model.

Server Utilization Results

For server utilization, we evaluated four relevant scenarios:

- Streamed OS with embedded applications
- Streamed OS with streamed applications
- VHD with embedded applications
- VHD with streamed applications

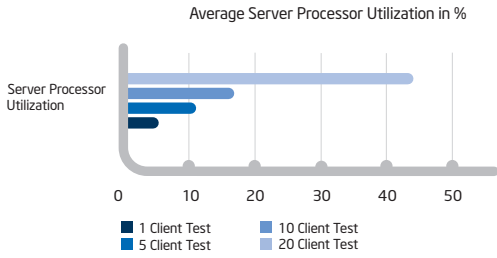
Server processor utilization was very efficient for streaming (Figure 2). Cold test runs for OS streaming with embedded applications used 1.2 percent of the server processor to run 20 clients. Warm run utilization improved to 0.5 percent for the same 20 clients. This illustrates the positive impact of local caching and its effectiveness.

Application streaming had comparable performance to embedded applications. About 1 percent of the processor is used for 20 clients. However, less than 1 percent of the processor was used for cold runs and almost 1.2 percent was used for warm runs. Server processor utilization increased unexpectedly for warm runs. This may have been caused by the system overhead to determine whether system updates are required at the client, or utilization is so low that the data could have been impacted by noise.

In both scenarios, streaming used a very small portion of the server processor, even when the clients were scaled. Cold and warm runs were not significantly different.

While testing the VHD model, we discovered that the Photoshop and video tasks induced significant latency in script execution due to processor spikes in the VHD host. This latency caused script run failures when the number of test VHDs was equal to or exceeded the number of physical cores in the host server. We also discovered lack of support for standard desktop functionality, such as the alt+tab keyboard combination, all 3D graphics, and high processor utilization applications. In addition, exogenous demand from a single VHD affected the entire host, as in the case of the “mystify” screen saver. With this screen saver active on only one VHD, all other virtual desktops were brought to a standstill. Therefore, the concurrent video and all Photoshop tasks were excluded from the script.

Virtualized Desktop Embedded Apps - Processor Utilization



Virtualized Desktop AppStream Apps - Server Utilization

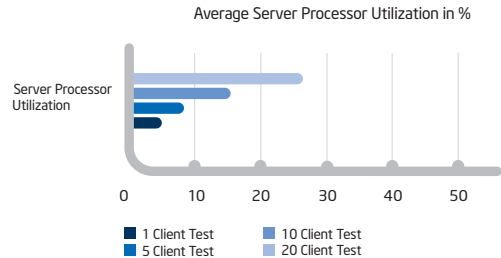


Figure 3: Server Processor Utilization for VHD

The VHD model had much higher server utilization than streaming, even with the script modifications (Figure 3). In the case of embedded applications, one client used 10 percent of processor. Processor utilization increased significantly when the number of VMs exceeded the number of cores on the server. 20 clients used 45 percent of processor.

When we streamed applications to VMs, server processor utilization dramatically improved. 20 clients used 25 percent of the processor, compared to the 45 percent when not streamed. The likely explanation for the improvement is that application streaming has a much smaller footprint that does not rely on the OS. All configurations are local to the execution container, which involves less processing.

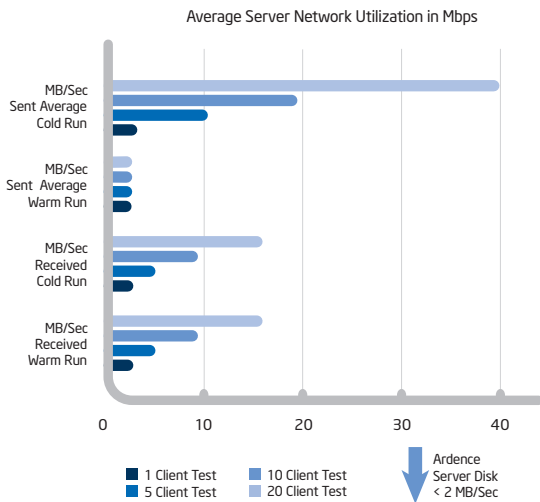
Network Utilization Results

For network utilization, we evaluated six relevant scenarios:

- Streamed OS with embedded applications
- Streamed OS with streamed applications
- VHD with embedded applications
- VHD with streamed applications
- OS running locally from a hard disk with embedded applications
- OS running locally from a hard disk with streamed applications

Network traffic for OS streaming consumed the entire 1 Gbps network for about 35 seconds when concurrently rebooting 20 clients. It appears as though OS streaming will use all available

Embedded Apps - Ardence Network Utilization



AppStream Apps - Ardence Network Utilization

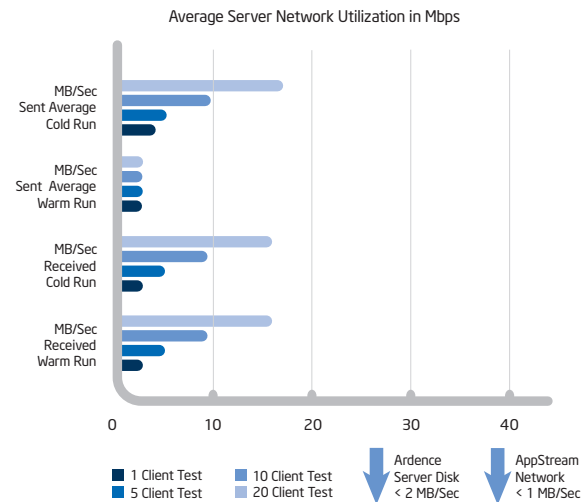


Figure 4: Network Utilization for Streaming

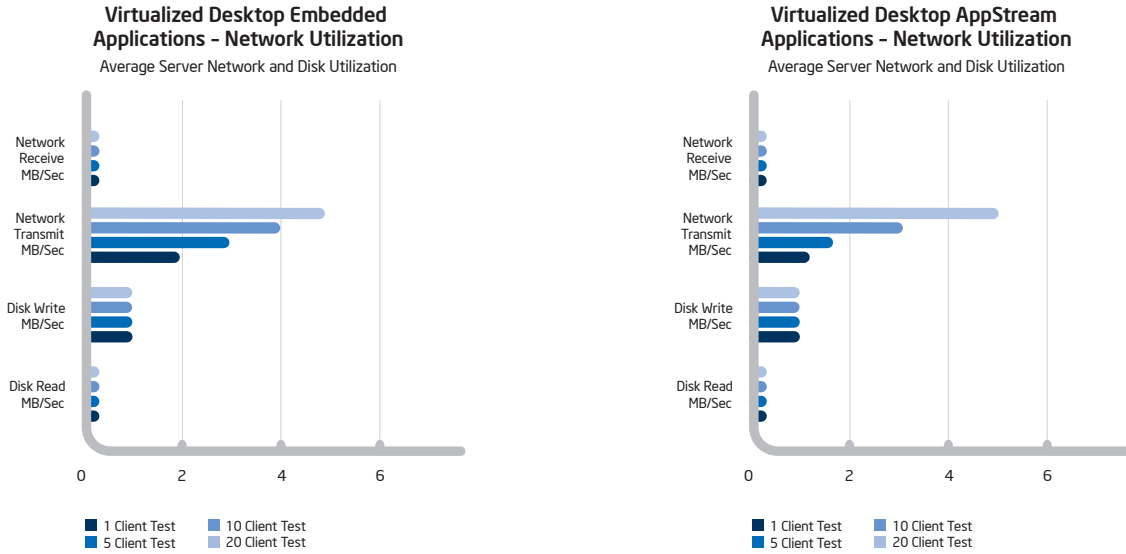


Figure 5: Network Utilization for VHD

bandwidth at reboot time, slowing network operations and creating a utilization spike. One way to mitigate network limitations and poor user response is to pre-determine a strategy for staggering any required rebooting for a widespread OS refresh or patch. An effective network design can help by structuring it into segments and strategically deploying streaming servers.

Traffic decreased over time and, overall, was better than expected for streaming (Figure 4). With embedded applications and 20 clients, cold runs generated 40 Mbps while warm runs sent 1 Mbps. Streaming the applications lessened the load on the network to 15 Mbps cold and 1 Mbps warm sent for 20 clients. In both scenarios, network received was 15 Mbps for cold and warm runs on 20 clients.

Network traffic for VHD was consistent and light for both embedded and streamed applications (Figure 5). 20 clients generated 5 Mbps for network sent and 1 Mbps for network received. There was a slight reduction in network traffic when applications were streamed.

Streamed applications used slightly more network than locally installed applications in clients that contain a hard disk and are running the OS locally (Figure 6). The results were better than expected, illustrating that application streaming does not generate an unusually heavy load on the network.

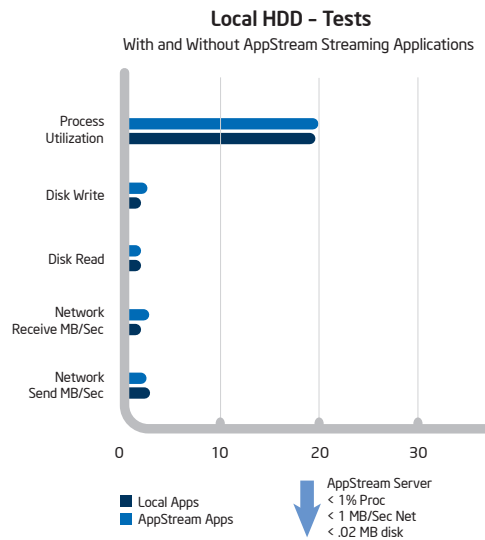


Figure 6: Network Utilization for Local OS

Model	Percent Server Processor Utilization (20 Clients)	Network Utilization (20 Clients)	Single Server Scaling (Est. Max Clients)	Caveats
VHD/Embedded App	45	0.5 Mbps	35 Clients	No multimedia or compute-intensive
VHD/Streamed App	25	0.5 Mbps	55 Clients	No multimedia or compute-intensive
Stream OS/Embedded App	1	5 Mbps	150 Clients	Reboot Caution
Stream OS/Streamed App	1	0.8 Mbps	150+ Clients	Reboot Caution

Table 3: Summary of Study Results

Conclusion

Based on our study of streaming and VHD computing models, it is apparent that each computing model is efficient for certain workloads, but also requires special considerations when designing around a particular model. Table 3 summarizes our results.

VHD offers excellent performance for certain workloads, allowing infrastructures to run as much as 20 times thinner. However, VHD, also places a substantial load on the server, up to 45 percent utilization for 20 clients. Scaled out further than 20 clients, we would expect to run out of server processor resources before we flooded our network capacity. The study shows the importance of system design for VHD to prevent overloading the server especially, if multimedia and other demanding applications will be used.

Streaming promises to be a very efficient model, with very low server disk and processor utilization. Over time, streaming performs efficiently on the network, comparable to locally installed OS and applications. During a massive reboot of clients, however, the network can be easily flooded as the OS is simultaneously downloaded to many clients. The study shows the importance of designing the network and client reboot strategies to mitigate slow startup times.

Acronyms

- DLL: Dynamic Link Library
- OS: Operating System
- VHD: Virtual Hosted Desktops
- RDP: Remote Desktop Protocol
- UDP: User Datagram Protocol
- VM: Virtual Machine

Authors

Catherine Spence is an Enterprise Architect with Intel Information Technology.
 Christian Black is a Systems Engineer with Intel Information Technology.

Special Recognition

Special thanks to Sabraish Kumar and Todd Christ for project guidance and technical support.

^a Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See www.intel.com/products/processor_number for details.

This paper is for informational purposes only. THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Intel disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Copyright © 2008 Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, and Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

