The Process of Threading: Multithreading a Sequential Application for a Multi-core World

A Dr. Dobb’s Journal
Vendor Perspectives NetSeminar
Sponsored by Intel

Tuesday, July 25, 2006
9AM PT / 12PM ET
Threading for Intel Platforms
Process and tools

Software and Solutions Group (SSG),
Developer Products Division (DPD)

David Mackay, Ph.D.
Manager threading tools consulting and support
Inside the Intel® Core™
Microarchitecture
What is Multi-Core

Multiple, externally visible processors on a single die where the processors have independent control-flow, separate internal state and no critical resource sharing

Also called Chip-level Multi-Processing (CMP)
Intel® Processor and Platform Evolution for the Next Decade

Source: "Platform 2015: Intel® Processor and Platform Evolution for the Next Decade"
Multi-core ramp accelerates in 2006

**Intel® dual-core volume exiting 2006**

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop Performance</td>
<td>&gt;70% Dual-Core</td>
</tr>
<tr>
<td>Server</td>
<td>&gt;85% Dual-Core</td>
</tr>
<tr>
<td>Mobile Performance</td>
<td>&gt;70% Dual-Core</td>
</tr>
</tbody>
</table>

**Data is run-rate exiting the year. All products and dates are preliminary and subject to change without notice.**

Intel single core to multi-core shipments

By 2007, the majority of Intel® platforms will be multi-core
Applications must consider Multi-core Architectures
Application Performance on Multi-core Processors

- Application Performance on Multi-core processors is increased by effectively using processor cores
- Application threading can effectively utilize all processor cores
  - Threads are recognized by the Operating System (OS) as a unit that can be executed independent of another thread
  - Operating Systems are ready now for multiple threads
- Threads increase utilization of multiple cores
  - Application Performance Increases
Intel® Software Development Products
Introduction

Intel® Software Development Products help you use the power of your software to unleash the full potential of the hardware

http://www.intel.com/software/products

Performance
- Extract the maximum application performance from Intel processor based systems
- Simplifies taking advantage of new capabilities like multi-core and Intel® EM64T

Compatibility
- Compatible with popular development environments including Microsoft Visual Studio on Windows*, GCC on Linux*, XCode on Mac OS*
- 32-bit and 64-bit processor support with one package
- Linux*, Windows*, Mac OS* support

Support
- One year of unlimited technical support and upgrades with purchase
- Get answers from the engineers that know how to develop software on Intel Architecture

Intel Software Development Products offer Performance, Compatibility, and Support
Multi-Core Software Development
The Importance of Parallelism & Threading

Parallelize: Unleash the Potential!

• Path to performance
  – Native threads
  – Threaded libraries
  – Compiler generated threads

• Do Nothing:
  – Benefits Still Visible
  – Operating systems ready for multi-processing/tasking
  – Background tasks benefit from more compute resources
  – Miss opportunity to gain magnitudes of performance benefits

competitive pressures = demand for parallel applications
Multi-core Software Support

Use of multiple cores—very important to us and our customers

Three key challenges:

• Scalability
• Correctness
• Ease of programming
Multi-core Software Support

How will we help more developers use parallelism?

Intel® Thread Checker
pinpoints latent threading errors

Intel® Thread Profiler

Intel® VTune™ Performance Analyzer
two ways to gain
precise insight into threaded
code – application/lock level
or application/system level

We help make it easier

Correctness

Copyright © 2006, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel® Thread Checker
Create Threads Faster

Key Features

• Detects challenging data races and deadlocks
• Pinpoints errors to the source code line
• Works on standard debug builds without recompiling
• Recommends modules to instrument by usage (windows* product)
• Scriptable interface for test environment integration
• Supports 32 and 64-bit applications

* Intel and the Intel logo are registered trademarks of Intel Corporation. Other brands and names are the property of their respective owners.
Intel® Thread Profiler 3.0 for Windows*  
Optimize Threads Faster

Key Features

• Identifies synchronization objects that impact performance
• Highlights thread workload imbalances
• Shows the number of cores utilized
• Pinpoints issues to the source code line
• Maximizes application time spent in parallel regions
• Supports 32 and 64-bit applications
Amdahl’s Law

Describes the upper bound of parallel speedup (scaling)

\[ T_{\text{parallel}} = \left\{ (1-P) + \frac{P}{n} \right\} T_{\text{serial}} + O \]

Scaling = \( \frac{T_{\text{serial}}}{T_{\text{parallel}}} \)

\( n \) = number of processors
Amdahl’s Law

Describes the upper bound of parallel speedup (scaling)

\[
T_{\text{parallel}} = \left( 1 - \frac{P}{n} \right) + \frac{P}{n} = \frac{1}{0.75} = 1.33
\]

\[
\text{Scaling} = \frac{T_{\text{serial}}}{T_{\text{parallel}}} = \frac{1.0}{0.75} = 1.33
\]
Amdahl’s Law

Describes the upper bound of parallel speedup (scaling)

\[ T_{\text{parallel}} = \left\{ (1 - P) + \frac{P}{n} \right\} T_{\text{serial}} + O \]

Scaling = \( \frac{T_{\text{serial}}}{T_{\text{parallel}}} \) \( \frac{1.0/0.5}{2.0} = 2.0 \)

More parallel code improves scaling
Case Study
Prime Number Generation

main

FindPrimes(start,end)

TestForPrimes(number)

NO

ShowProgress

YES

Increment Count

8 primes found between 1 and 20 in 0.00 secs

See backup foils for system configuration
A Generic Development Cycle

Analyze
- VTune™ Performance Analyzer

Design (Introduce Threads)
- Intel® Performance libraries: IPP and MKL
- OpenMP* (Intel® Compiler)
- Explicit threading (Win32*, Pthreads*)

Debug for correctness
- Intel® Thread Checker
- Intel® Debugger

Tune for performance
- Intel® Thread Profiler
- VTune™ Performance Analyzer

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

**Where to thread?**

### Call Graph

![Call Graph Image]

<table>
<thead>
<tr>
<th>Module</th>
<th>Thread</th>
<th>Function</th>
<th>C. Calls</th>
<th>Self Time</th>
<th>Total Time</th>
<th>Callers</th>
<th>Callees</th>
<th>Module Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primebase.exe</td>
<td>Thread_0(784)</td>
<td>mainCRTStartup</td>
<td>600,017</td>
<td>1,087,686</td>
<td>1,087,924</td>
<td>1</td>
<td>13</td>
<td>c:\Documents an</td>
</tr>
<tr>
<td>Primebase.exe</td>
<td>Thread_0(784)</td>
<td>main</td>
<td>1</td>
<td>8</td>
<td>1,087,815</td>
<td>1</td>
<td>5</td>
<td>c:\Documents an</td>
</tr>
<tr>
<td>Primebase.exe</td>
<td>Thread_0(784)</td>
<td>TestForPrime</td>
<td>600,000</td>
<td>1,081,869</td>
<td>1,081,869</td>
<td>1</td>
<td>1</td>
<td>c:\Documents an</td>
</tr>
<tr>
<td>Primebase.exe</td>
<td>Thread_0(784)</td>
<td>GetCommandLineArguments</td>
<td>1</td>
<td>0</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>c:\Documents an</td>
</tr>
<tr>
<td>Primebase.exe</td>
<td>Thread_0(784)</td>
<td>_atexit</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>c:\Documents an</td>
</tr>
</tbody>
</table>

## Diagram

![Diagram Image]
Analysis
What is the expected speedup?

Scaling(2P) = \frac{100}{(99.7/2 + 0.3)} = \sim 1.99X
Analysis
Which threading model?

Rapid prototyping with OpenMP*

Fork-join parallelism

*Other names and brands may be claimed as the property of others
Design (Introduce Threads)

```c
#pragma omp parallel for
for( int i = start; i <= end; i+= 2 )
{
    if( TestForPrime(i) )
        globalPrimes[gPrimesFound++] = i;
    // ShowProgress(i, range);
}
```
Design (Introduce Threads)

```c
#pragma omp parallel for
for( int i = start; i <= end; i+= 2 )
{
    if( TestForPrime(i) )
        globalPrimes[gPrimesFound++] = i;
    // ShowProgress(i, range);
}
```

OpenMP

Defined by the for loop

Create threads here for this parallel region
Design (Introduce Threads)

Scaling of 1.4X
Debug for Correctness

C:\Documents and Settings\David Mackay\My Documents\PTT\2RACE\Debug>primerace.exe 1 1200000

92700 primes found between 1 and 1200000 in 0.81 secs

C:\Documents and Settings\David Mackay\My Documents\PTT\2RACE\Debug>primerace.exe 1 1200000

92689 primes found between 1 and 1200000 in 0.86 secs

C:\Documents and Settings\David Mackay\My Documents\PTT\2RACE\Debug>primerace.exe 1 1200000

92707 primes found between 1 and 1200000 in 0.80 secs

C:\Documents and Settings\David Mackay\My Documents\PTT\2RACE\Debug>primerace.exe 1 1200000

92735 primes found between 1 and 1200000 in 0.83 secs

C:\Documents and Settings\David Mackay\My Documents\PTT\2RACE\Debug>
Intel® Thread Checker

Identification of data safety issue:

Dead-Locks, Race Conditions etc

Supported environments:

OpenMP*

Native thread-API on Microsoft Windows* systems (Win32 Threads) and Linux* (PThreads)

Analysis:

Instrumented application

Dynamic monitoring while application runs
# Debug for Correctness

![Intel Thread Checker interface](image)

## Intel® Thread Checker - Activity: 10:11 AM, 2006 May 30 (TC: primerace.exe)

<table>
<thead>
<tr>
<th>Relation/ID</th>
<th>Short Description</th>
<th>Severity</th>
<th>Description</th>
<th>Count</th>
<th>Filtered</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1</td>
<td>Read -&gt; Write data race</td>
<td>x</td>
<td>Memory write at “Primerace.cpp”:112 conflicts with a prior memory read at “Primerace.cpp”:112 (anti dependence)</td>
<td>375</td>
<td>False</td>
</tr>
<tr>
<td>1/2</td>
<td>Write -&gt; Read data race</td>
<td>x</td>
<td>Memory read at “Primerace.cpp”:112 conflicts with a prior memory write at “Primerace.cpp”:112 (flow dependence)</td>
<td>376</td>
<td>False</td>
</tr>
<tr>
<td>1/3</td>
<td>Write -&gt; Write data race</td>
<td>x</td>
<td>Memory write at “Primerace.cpp”:112 conflicts with a prior memory write at “Primerace.cpp”:112 (output dependence)</td>
<td>376</td>
<td>False</td>
</tr>
<tr>
<td>2/4</td>
<td>Thread termination</td>
<td>1</td>
<td>Thread termination at “Primerace.cpp”:108 - includes stack allocation of 3145728 and use of 4096 bytes</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>3/5</td>
<td>Thread termination</td>
<td>1</td>
<td>Thread termination at “Primerace.cpp”:108 - includes stack allocation of 1048576 and use of 4096 bytes</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>4/6</td>
<td>Thread termination</td>
<td>1</td>
<td>Thread termination at “Primerace.cpp”:113 - includes stack allocation of 1048576 and use of 8192 bytes</td>
<td>1</td>
<td>False</td>
</tr>
</tbody>
</table>

*Other brands and names are the property of their respective owners*
Debug for Correctness

Intel® Thread Checker - Activity: 10:11 AM, 2006 May 30 (TC: primeace.exe)

Memory read at "Primeace.cpp" 112 conflicts with a prior memory write at "Primeace.cpp" 112 (flow dependence)

Address | Line | Source
--- | --- | ---
0x1391 | 108 | #pragma omp parallel for
0x1392 | 109 | for( int i = start; i <= end; i += 2 )
0x135A | 110 | if( TestForPrime(i) )
0x1364 | 111 | globalPrimes[qPrimesFound++] = i;

// ShowProgress(i, range);

int main(int argc, char **argv)
{

Address | Line | Source
--- | --- | ---
0x1391 | 108 | #pragma omp parallel for
0x1392 | 109 | for( int i = start; i <= end; i += 2 )
0x135A | 110 | if( TestForPrime(i) )
0x1364 | 111 | globalPrimes[qPrimesFound++] = i;

// ShowProgress(i, range);

int main(int argc, char **argv)
{
#pragma omp parallel for
for( int i = start; i <= end; i += 2 )
{
    if( TestForPrime(i) )
#pragma omp critical
    globalPrimes[gPrimesFound++] = i;

    // ShowProgress(i, range);
}
Debug for Correctness

```
C:\Documents and Settings\David Mackay\My Documents\PTT\3Correct\Release>prinecorrect.exe 1 1200000

92938 primes found between 1 and 1200000 in 0.83 secs
```

Copyright © 2006, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners*
Intel® Thread Profiler

Detection of thread-specific bottlenecks:
- Load imbalance
- Contention on synchronization objects
- Threading overhead

Supported environments:
- OpenMP*
- Native thread-API on Microsoft Windows* (Win32 Threads)
- Native thread-API on Linux* (PThreads)

Analysis:
- Binary and/or source instrumentation
- Profiled run-time libraries (-Openmp_profile)
Tune for Performance- Combination View
Tune for Performance

Threads are not balanced
Tune for Performance

Threads are not balanced

Reschedule from block to block cyclic
Threads are not balanced
Balance the workload for the threads

```c
#pragma omp parallel for schedule(static, 8)
for( int i = start; i <= end; i += 2 )
{
    if( TestForPrime(i) )
    #pragma omp critical
        globalPrimes[gPrimesFound++] = i;

    // ShowProgress(i, range);
}
```
Tune for Performance

Balance the workload for the threads

Command Prompt

C:\Documents and Settings\David Mackay\My Documents\PTT\4TUNE1\Release>prinetune1.exe 1 1200000

92938 primes found between 1 and 1200000 in 0.66 secs

Scaling achieved is 1.77X
Tune For Performance – balanced
Tune For Performance – Object focus
Tune For Performance – Object View
Tune for Performance
Performance Re-visited

Let’s look at the OpenMP locks...

```c
#pragma omp parallel for schedule (static,8) for( int i = start; i <= end; i += 2 )
{
   if( TestForPrime(i) )
      #pragma omp critical
      globalPrimes[gPrimesFound++] = i;

      ShowProgress(i, range);
}
```

Lock is in a loop
Performance Re-visited

Let’s look at the OpenMP locks...

```c
#pragma omp parallel for schedule (static,8)
for( int i = start; i <= end; i += 2 )
{
    if( TestForPrime(i) )
    {
        globalPrimes[InterlockedIncrement(&gPrimesFound)] = i;
    }
    ShowProgress(i, range);
}
```
Tune for Performance

All locks have been taken care of

C:\Documents and Settings\David Mackay\My Documents\PTT\5TUNE2\Release>primetune2.exe 1 1200000

92938 primes found between 1 and 1200000 in 0.63 secs

C:\Documents and Settings\David Mackay\My Documents\PTT\5TUNE2\Release>

Final Scaling 1.86
Tune for Performance
Multi-core Software Support Summary

Transition from sequential to multi-threaded, parallel application is supported by Intel® Software Products in every stage

* Analysis
* Design
* Implementation
* Debugging
* Tuning

Intel® Threading Tools offer unique features for developers, considerably reducing the effort to fully exploit multi-core architectures
“Taking advantage of Intel’s broad range of software development tools for threading and their extensive expertise, we were able to see a 70% performance increase when our mp3 and mp3 SURROUND codecs ran on Intel® Centrino® Duo mobile technology. This results in a greater listening experience for consumers.”

Rocky Caldwell, General Manager mp3 Licensing, Thomson

“In our development efforts, we hugely depend on Intel tools. We use the Intel compiler, the Threading Tools, and also VTune to make sure our codecs exploit the Intel platform as best as possible”

Markus Monig CEO MainConcept

“Intel’s Threading Tools have accelerated our development cycle dramatically. The Intel Threading Tools are now an integral part of our development process.”

Dana Batali Director of Pixar RenderMan Development

“Threading can be a challenging task, and time consuming as well. Intel really understands this and has great products for software developers.”

Bill Hensler Vice President of engineering Video Solutions for Adobe

“Threading can be a challenging task, and time consuming as well. Intel really understands this and has great products for software developers.”
Call to Action

Thread your software for performance on Intel multi-core platforms

Evaluate Intel Software Development products to improve your threaded software development productivity and performance

www.intel.com/software/products
Summary

- Applications must consider Multi-core Architectures
- Intel® Software Development Products offer Performance, Compatibility, and Support
- Applications take advantage of Multi-core by using multiple execution threads
  competitive pressures = demand for parallel applications
- Intel Software Development Products help deliver Multi-core application performance
  - Intel® C++ and Fortran Compilers
  - Intel® Vtune™ Performance Analyzers
  - Intel® Performance Libraries
  - Intel® Thread Checker and Intel® Thread Profiler
- Intel® Threading Tools offer unique features for developers, considerably reducing the effort to fully exploit multi-core architectures
We are optimizing RenderMan’s core to be very scalable for future multi-core architectures. Intel’s Threading Tools have accelerated our development cycle dramatically. Intel’s Thread Checker for example, helped identify potential threading issues very quickly, in days compared to weeks if done otherwise. Thread Profiler, on the other hand, has helped us understand threading performance problems so we could fix them to improve scalability. The Intel Threading Tools are now an integral part of our development process.

- Dana Batali, Director of RenderMan Development, Pixar

We used Intel Threading Tools including Intel Thread Profiler to realize improved threaded application performance of Omni Page 15 running on Intel multi-core platforms. We look forward to using Intel Thread Profiler with its critical path analysis and selective magnification of important time regions on future thread optimization projects.

- Gyorgy Varszegi, Scansoft (Nuance)

Intel Thread Checker was very useful for analyzing bottlenecks in our threaded code. Thread Profiler quickly pinpointed problem areas and showed us the reasons for the slowdown, so we were able to restructure the code for better threaded performance.

- Martin Watt, Software Architect, Alias

[Intel] Thread Checker helped us find non-thread safe functions that allowed two threads to modify the same memory without proper synchronization. We couldn’t have gotten the networking up and running as quickly and as efficiently without Thread Checker. Thread Checker is simply an awesome tool and we are not going to develop multi-threaded code without it.

- Doug Service, Director of Technology Development, Chris Stark, Software Engineer, Ritual Entertainment

Intel Thread Checker helped Siemens by identifying issues in software we develop and in software we purchase from third parties. We use Intel Thread Checker to improve the quality of our software and look forward to expanding the use of the tool in more of our software development groups.

- Andreas Dietrich, Research & Development Image Processing, Siemens Medical Solutions
Q&A
Please Submit Your Questions Now
Resources

Intel Software Development Products:
http://www.ddj.com/intel/51.htm

Intel Discussion Forums (including threading forum):
http://www.ddj.com/intel/52.htm

Intel Multicore Processing:
http://www.ddj.com/intel/53.htm

Intel Software College:
http://www.ddj.com/intel/54.htm
Back-up Reference Resources
Debug for Correctness

Intel® Thread Checker pinpoints notorious threading bugs like data races, stalls and deadlocks.
Prime Number Generation

Tune for Performance

Thread Profiler pin-points performance bottlenecks in threaded applications.
# Reference Materials

<table>
<thead>
<tr>
<th>Technical information, supported environments, white papers</th>
<th><a href="http://www.intel.com/software/products">http://www.intel.com/software/products</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Training (e.g. on-line, on-site, webcast)</td>
<td><a href="http://www.intel.com/cd/ids/developer/asm-o-na/eng/training/index.htm">http://www.intel.com/cd/ids/developer/asm-o-na/eng/training/index.htm</a></td>
</tr>
<tr>
<td>Course Catalog</td>
<td><a href="http://or1cedar.cps.intel.com/softwarecollege/CourseCatalog.asp">http://or1cedar.cps.intel.com/softwarecollege/CourseCatalog.asp</a></td>
</tr>
<tr>
<td>Technical Discussion Forums</td>
<td><a href="http://softwareforums.intel.com/ids">http://softwareforums.intel.com/ids</a></td>
</tr>
</tbody>
</table>
Purchase of Intel® Software Development Products includes one year of unlimited premier support

Intel® Premier Support includes:

- Primary support for all Intel Software Development Products
- Online access to Intel Premier Support Website
- Issue submission & tracking
- Product updates & related downloads
- FAQ’s, user forums, & other proactive notices

Support Comes Directly from Experts in Software Development at Intel

Registering for support was easy, and we value the security of knowing that Intel is there to help, even though we haven’t needed it so far.”

— Rob Hoffmann - Director of Marketing, NewTek, Inc.
Customer Testimonials

**Rogue Wave**
Using Intel's Math Kernel Library significantly boosted the performance of Rogue Wave's SourcePro Analysis on Intel processors. The optimized MKL routines easily linked in place of our non-optimized BLAS and LAPACK routines, and delivered the performance boost our customers demanded. SourcePro C++ products have demanding cross-platform portability requirements, and MKL worked seamlessly on Windows and Linux operating systems. I can't think of any reasons not to use the MKL
Randall Robinson, Development Manager

**SGI**
Intel's MKL vector math library is expected to help deliver as much as a 20% improvement in benchmark code for Cambridge University running on a 128 processor SGI Altix 3700 system.
Crispin Keable, Ph.D., Engineering Manager

**MSC Software**
Intel Math Kernel Library 6.0 has replaced much of our proprietary code for Itanium®-based systems running Linux® because of the performance benefits it delivers. The library also reduces the time our developers spend on maintaining existing code and enables them to focus on adding new functionality to our virtual product development applications.
Joe Griffin, Nastran® Linux® porting specialist

**ABAQUS**
By adopting the Intel MKL DGEMM libraries, our standard timing improved between 43% and 71%, which is very impressive
Matt Dunbar, Software Developer
## Intel® Thread Checker and Thread Profiler

### Feature Comparison

<table>
<thead>
<tr>
<th></th>
<th>Windows*</th>
<th>Linux*</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Intel® Thread Checker</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GUI/CLI – view Windows* results</td>
<td></td>
<td>CLI – view Linux results</td>
</tr>
<tr>
<td>Windows Threads Support</td>
<td></td>
<td>POSIX Threads Support</td>
</tr>
<tr>
<td>OpenMP* Support</td>
<td></td>
<td>Itanium® Processor Support</td>
</tr>
<tr>
<td>Intel® EM64T Support</td>
<td></td>
<td>Intel® EM64T Support</td>
</tr>
<tr>
<td><strong>Intel® Thread Profiler</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GUI/CLI – view Windows* results</td>
<td></td>
<td>GUI – view Linux results via Linux* RDC</td>
</tr>
<tr>
<td>Windows Threads Support</td>
<td></td>
<td>POSIX Threads Support</td>
</tr>
<tr>
<td>OpenMP* Support</td>
<td></td>
<td>Itanium® Processor Support</td>
</tr>
<tr>
<td>Intel® EM64T Support</td>
<td></td>
<td>Intel® EM64T Support</td>
</tr>
</tbody>
</table>

*Other brands and names are the property of their respective owners.
Learn more...Intel books

VTune™ Performance Analyzer Essentials
Measurement and Tuning Techniques for Software Developers

Intel® Integrated Performance Primitives
How to Optimize Software Applications Using Intel® IPP

Multi-Core Programming
Increasing Performance through Software Multithreading

Programming with Hyper-Threading Technology
How to Write Multithreaded Software for Intel® IA-32 Processors

The Software Optimization Cookbook
High-Performance Recipes for the Intel® Architecture

Scientific Computing on Itanium®-based Systems
### Who is parallelizing Software and using Intel Development Tools?

<table>
<thead>
<tr>
<th>Activision (Ravensoft)</th>
<th>Pinnacle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe</td>
<td>Pixar (Renderman)</td>
</tr>
<tr>
<td>Algorithmics</td>
<td>Paradigm</td>
</tr>
<tr>
<td>Alias</td>
<td>PTC</td>
</tr>
<tr>
<td>Autodesk</td>
<td>Red Hat</td>
</tr>
<tr>
<td>Business Objects</td>
<td>SAP</td>
</tr>
<tr>
<td>Cakewalk</td>
<td>SAS</td>
</tr>
<tr>
<td>CodecPeople</td>
<td>Siebel CRM</td>
</tr>
<tr>
<td>Computer Associates</td>
<td>Signet</td>
</tr>
<tr>
<td>Corel (WordPerfect)</td>
<td>Skype</td>
</tr>
<tr>
<td>Cyberlink</td>
<td>SLB</td>
</tr>
<tr>
<td>Discreet</td>
<td>SnapStream</td>
</tr>
<tr>
<td>IBM</td>
<td>Sonic (Roxio)</td>
</tr>
<tr>
<td>id Software</td>
<td>Sony</td>
</tr>
<tr>
<td>Landmark</td>
<td>Steinberg</td>
</tr>
<tr>
<td>Macromedia</td>
<td>SunGard</td>
</tr>
<tr>
<td>Mainconcept</td>
<td>Sybase</td>
</tr>
<tr>
<td>Maxon</td>
<td>Symantec</td>
</tr>
<tr>
<td>mental images</td>
<td>Thomson</td>
</tr>
<tr>
<td>Microsoft (Office Suite)</td>
<td>THQ</td>
</tr>
<tr>
<td>Midway</td>
<td>Ubisoft</td>
</tr>
<tr>
<td>MSC</td>
<td>UGS</td>
</tr>
<tr>
<td>Novell SUSE</td>
<td>Valve</td>
</tr>
<tr>
<td>Oracle</td>
<td>Valve</td>
</tr>
<tr>
<td>Pegasus</td>
<td>Valve</td>
</tr>
</tbody>
</table>

*Other brands and names are the property of their respective owners.*