System & Application Software Performance Tuning for Devices Powered by the Intel® Atom™ Processor

**Agenda**

- Market Segment and Tools Overview
- System Software Development
- Application Performance Tuning

**Intel® Tools Cover All These Device Categories**

- Consumer electronics
- Mobile Internet Devices
- Netbooks/Nettops
- Embedded
  - Intel® Media processor CE3100
  - Intel® Atom™ processor Zxx series
  - Intel® Atom™ processor Nxx series
  - Intel Atom™ processor Zxx series
  - Intel® Atom™ processor CE4100
  - Intel® Atom™ processor Zxx series

**Intel® Software Development Tools Coverage**

- Windows®
- Linux®
- RTOS

**Application Suite**

- For ISVs and Moblin Community – tune Moblin applications for more performance and extend battery life of Intel® Atom™ processor powered devices

**Embedded Suite**

- For OEM/ODMs (+ their key ISVs) and OSVs – use a complete tools solution with a sophisticated JTAG debug solution for embedded system and application software design

http://software.intel.com/software/products/atomtools
Intel® Software Development Tools
For Intel® Atom™ Processors

- Outstanding **performance**
  - Increased application software performance can help to extend battery life time

- Intel® architecture customization increases **productivity & efficiency**
  - Find and fix issues faster with full GUI driven system-level JTAG and application debugging tools

- Technology **alignment**
  - Latest Intel® Atom™ Processor and chipset support
  - NDA Tools BETA programs for next generation silicon

- Excellent **customer support**

Moblin Software Development Tools

**Moblin**
Open Source Linux® SW Platform for Mobile & Embedded Devices including Mobile Internet Devices (MID’s), Netbooks, Automotive In-Vehicle Infotainment Systems

**The Moblin SDK**
- Development guides, tutorials, sample code, API references
- Compliance Tools
- Project generator
- GNU Tools
- Moblin Image Creator 2
- PowerTop

**Intel® Software Development Tool Suite**
- Intel® C++ Compiler
- Intel® Integrated Performance Primitives Library
- Intel® JTAG Debugger
- Intel® Application Debugger
- Intel® VTune™ Performance Analyzer

Intel® Tool Suites complement the open source Moblin SDK

**Intel® C++ Software Development Tools**

**Intel® Embedded Software Development Tool Suite**

**Intel® Application Software Development Tool Suite**

**Intel® Embedded Software Development Tool Suite**

- Open Source Software Development Tools
- Intel® Embedded Software Development Tool Suite for Intel® Atom™ processor

**Intel® Application Software Development Tool Suite**

**Agenda**

- Market Segment and Tools Overview
- System Software Development
- Application Performance Tuning

**Intel® Tools for System Development**

**Cross Development**
- Different host and target hardware
- Cross compile on host
- Download and debug with JTAG Debugger

**Intel® C++ Compiler**
- Build performance critical OS components and drivers
- Optimize for fast execution and fast OS switch into low power mode

**Intel® JTAG Debugger**
- Debug and identify issues in bootloader
- Debug and identify issues in OS kernel
- Debug and identify issues in device drivers

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

Intel® Tools – a complete solution with more performance, and latest technology alignment
Using Intel® C++ Compiler for OS Kernel Development

- Install Intel® C++ Compiler into build environment
  - Use protected OS image build environment like Moblin Image Creator 2
- Modify component makefiles to use ICC instead of GCC for parts that
  - Are multimedia or data volume, or data stream driven
  - Have a lot of direct interaction with user interface
  - Note: OS kernels are highly optimized code. Recompile using different compiler – "hard work with limited benefit"
- Improve overall OS responsiveness and end-user experience

Use Intel® C++ Compiler for spot optimizations in System Software, e.g. performance critical drivers, codecs, etc.

Intel® JTAG Debugger - Target Connection

- System Software Development == bare metal programming
- JTAG based debugging is the only solution
- JTAG connector on the target HW required – to access
  - CPU registers
  - SoC components / peripheral registers
- An intelligent probe - e.g. Intel® XDP3 JTAG I/F probe connects host system with the target

RCP Eclipse GUI based JTAG Debugger

- Ensure OS image is on the target
- Connect JTAG Debugger to Intel® Atom™ Processor
  - $ ./xdb.sh
- Load OS image symbol information into debugger
- Set HW breakpoint at label "start_kernel"
  - some memory locations may not be mapped as valid yet or may be read-only:
    - XDB> set opt /hard=on
- Run target platform until basic platform initialization through firmware/BIOS is complete
  - System stops at “start_kernel”
- Step through kernel initialization and single step as you please
- Run to &mwait_idle to debug fully initialized OS
Intel® JTAG Debugger is recommended for OEM/OSVs who need to customize, debug and validate OS kernels.

**Debugging Linux® OS kernel**

- **Kernel**
  - start_kernel
- **Firmware/BIOS**
  - mwait_idle

**Trace Support**

- Hardware feature of Intel® Atom™ Processor
- Enables viewing of execution history
- Identify the root cause for exceptions

**Branch Trace Buffer**

- On-chip (Intel® JTAG Debugger)
- Memory allocated (Intel® Application Debugger)

**Kernel or Application Source Code**

- Send Branch Trace Information To Debugger

**Localize Configuration Issues with Instruction Trace**

- **C/C++ Source Window**
- **Trace Window**
- **Assembler Window**

**Chipset Peripheral Registers**

- **Intel® System Controller Hub US15W**
  - ~400 Peripheral Registers

**Validating Peripheral Register Settings Can Be Quite Complex**

**CPU & Chipset Specific Register Access**

- Show and change the content of all processor & chipset registers
- Convenient access to architectural registers - analyze register changes after instruction execution

**Bitfield Editor**

- Graphical representation of peripheral registers and bit fields with online documentation
- Easy and fully documented access to all processor registers and peripherals
- Change register contents on the fly, without re-compilation

Note: Intel® JTAG Debugger requires the XDP3 JTAG hardware interface from Intel
CPU & Chipset Specific Register Access

Bitfield Editor

Linux® OS Awareness – System Debug

Monitor kernel modules and system threads
Access status information
Debugging of Linux® memory images

Be aware of all relevant platform software stack interactions

System Software Debugging Receipt

- GCC for kernel build, ICC for performance critical code
- Compile kernel with debug info
- Connect target through JTAG I/F
- Set hardware breakpoint at "start_kernel"
- Run target to complete firmware/BIOS init
- Debug kernel
  - Execution trace to find errors that are hard to detect
  - Use translation table feature to resolve segmentation faults
- Inspect SoC/chipset peripheral registers to validate low-level drivers
- Use Flash feature to burn image into Flash memory

Use Intel® JTAG Debugger for in-depth system software debugging with full Si/SoC/chipset awareness

Agenda

- Market Segment and Tools Overview
- System Software Development
- Application Performance Tuning

Performance Optimization Principles

Re-compile
  - +SSE3_ATOM (Atom switch / in-order scheduler)
  - IPO (interprocedural optimization)
  - PGO (program guided optimization)
  - OpenMP (works on multicore/HT only) – source modification

Implement library functions
  - Highly optimized multimedia/math library functions
  - OpenMP compiled (works on multicore/HT only)
  - Update application source code & build environment

Modify source code
  - Identify C and ASM – source spot optimization opportunities
  - Analyze results – update sources, rebuild, analyze again

Intel® Tool Suites provide a complete spectrum of performance optimization methodologies
Identify Optimization Opportunities

Get the best performance out of an application, by

- Identifying optimization opportunities using the Intel® VTune™ Performance Analyzer

Questions to ask
- Where do I spend most of my execution time?
- Where do small optimizations have the biggest impact?
- What hardware bottlenecks and dependency stalls can be easily avoided?

Intel® VTune™ Performance Analyzer Identifies hard to find performance bottlenecks

<table>
<thead>
<tr>
<th>Line Number</th>
<th>Source</th>
<th>Counters</th>
<th>Instructions Retired</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Features
- Low overhead sampling
- No instrumentation required
- Monitor processor events like cache misses etc.
- View results in source or assembly

Usage Model
- Two components
  - Intel® VTune™ Performance Analyzer on host
  - Sampling Collector on the target
- Collect data on target and analyze it on the host

Sampling - How To Find Hotspots

- Pick an event to sample and configure PMU
  - Cache misses, branch mis-predictions, Dependency/pipeline stalls
- Start SEP sampling routine and application
- Performance Management Unit (PMU) periodically interrupts the processor

PMU

- Event 1
- Event 2
- Event 3
- Event 4
- Event 5

SEP == ISR

- Collect
- Execution address in memory (CS:IP)
- OS process and thread ID
- Executable module loaded at that address
- Write
- Information into *.TBS file

Numbers in counters define sampling rate
Take Advantage of Sampling Data

The Intel® VTune™ Performance Analyzer tells you which module, function or routine could use some improvement.

Focus your application optimization efforts where it counts – Intel® VTune™ Performance Analyzer helps to analyze applications without source and binary instrumentation.

Intel® C++ Compiler

<table>
<thead>
<tr>
<th>Compiler Features</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Significantly faster than GCC High performing code maps directly into application quality and battery lifetime</td>
</tr>
<tr>
<td>In-order scheduler</td>
<td>Compiler optimization switch that re-arranges/optimizes application code to be executed with best performance on Intel's Low-power Intel® Architecture technology Better performance of system- and application software helps to reduce power consumption of a mobile device</td>
</tr>
<tr>
<td>Profile Guided Optimization</td>
<td>Multi-stage optimization method with feedback loop Improves application performance by reducing instruction-cache thrashing, reorganizing code layout, shrinking code size, and reducing branch mispredictions</td>
</tr>
<tr>
<td>GCC Compatibility</td>
<td>Intel Compiler provides GCC language extensions and is source and binary code compatible with GCC Saves efforts in porting/re-using existing code</td>
</tr>
</tbody>
</table>

Need For In-order Scheduler Support - avoid dependency stalls

Consider code sequence: 
- a = b * 7;
- c = d * 7;

Representative assembly:
1 movl b, %eax
2 imull $7, %eax
3 movl %eax, a
4 movl d, %edx
5 imull $7, %edx
6 movl %edx, c

Processor cycles

- 10%
- 23%
- 45%
- 81%

Model instruction pipeline and avoid dependency stalls by using the in-order scheduler feature

Need For In-order Scheduler Support - avoid dependency stalls

Compiler switch –xSSE3_ATOM enables the in-order scheduler, which may improve application’s performance behavior

C/C++ Compiler Benchmark

Estimated Relative Performance To GCC 4.5.0 (GCC 4.5.0 = 1.0)

- 10%
- 23%
- 45%
- 81%

Use Intel® C++ Compiler for higher performance on Intel® Atom™ processors
Estimated by measurement on internal systems based on the following configuration assumptions:

- Source: Intel estimates as of September 9, 2009
- Basis of comparison: Intel estimates for current version of Intel and GCC compilers as of September 9, 2009

Compilers:
- Intel® C++ Compiler 11.1 for Linux® (icc)
- GCC 4.5.0

Hardware:
- Form factor: Mini-ITX / micro-ATX compatible
- Integrated Intel® Atom™ Processor 330 (1.6 GHz / 1MB L2 Cache / 533 MHz System Bus), v6.12.2
- Memory: 2GB, Hard disk: 40GB
- Chipset: i945GC and ICH7
- Audio: Realtek ALC662 audio codec (0.1 channel HD audio)
- Video: Intel® Graphics Media Accelerator 950 & S-video output support
- I/O Control: SMSC LPC47M997 based Legacy I/O controller for serial, parallel, and PS/2 ports
- LAN control: 10/100/1000 Mbits/sec LAN subsystem using the Realtek LAN adapter device

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information see www.intel.com/software/products

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

Note:
- * 178:galge: GCC 4.5.0: Assumes use of -fno-strict-aliasing
- * 252:comp: GCC 4.5.0: Assumes use of -march=athlon
- * 255:vortex: GCC 4.5.0: Assumes use of -mcpu=m64 -ffixed-form -ffixed-line-length=132 EXTRA_LD_FLAGS = -mcpu=athlon

Other estimates for current version of Intel and GCC compilers as of September 9, 2009

Intel® C++ Compiler 11.1 for Linux® VS. GCC 4.5.0:
- Compiler benchmarks based on SPECfp are based on C/C++ applications only
- Compiler switches used for estimates:
  - -O2
  - -o2
  - -O3
  - -o3
  - -ffast
  - -m32
  - -ipo
  - -no
  - -strict
  - -mpc64
  - -form
  - -length
  - -prof_use
  - -all
  - -mcpu=m64 -ffixed-form -ffixed-line-length=132 EXTRA_LD_FLAGS = -mcpu=athlon

Intel encourages all of its customers to visit the referenced Web sites or others where similar performance benchmarks are reported and confirm whether the referenced benchmarks are accurate and reflect performance of systems available for purchase.
Summary

- Intel Software Development Tool Suites for OEMs, OSVs, ("Embedded Suite") and ISVs ("Application Suite") cover the entire cycle of SW development
- Intel® Tool Suites for Intel® Atom™ Processors complement the open source Moblin SDK
- Intel Tool Suites provide a complete spectrum of performance optimization methodologies (compiler switches, IPP multimedia libs, performance bottleneck analysis with VTune)
- Intel® C++ Compiler for spot optimizations in System Software, e.g. performance critical drivers, codecs, and applications in general
- Intel JTAG debugger for in-depth system software debugging with full Si/SoC/chipset awareness

Information

- Check the web for more details on both the "Embedded" and "Application" tool suites
  www.intel.com/software/products/atomtools
- Download your 30days try-and-buy evaluation version
- Articles and Documentation:
- Knowledge Base Articles:

Additional sources of information on this topic:

- Additional info in the Moblin community
  - www.moblin.org
  - www.moblinzone.com