Webinar: Intel® System Studio 2015

Accelerate Development for Embedded, Mobile, and the Internet of Things

Robert Mueller-Albrecht
Technical Consulting Engineer
Intel SSG Developer Products Division

Intel's Vision

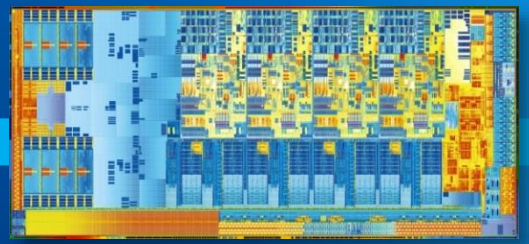
If it is smart and connected, it is best with Intel.

Data Center

Client

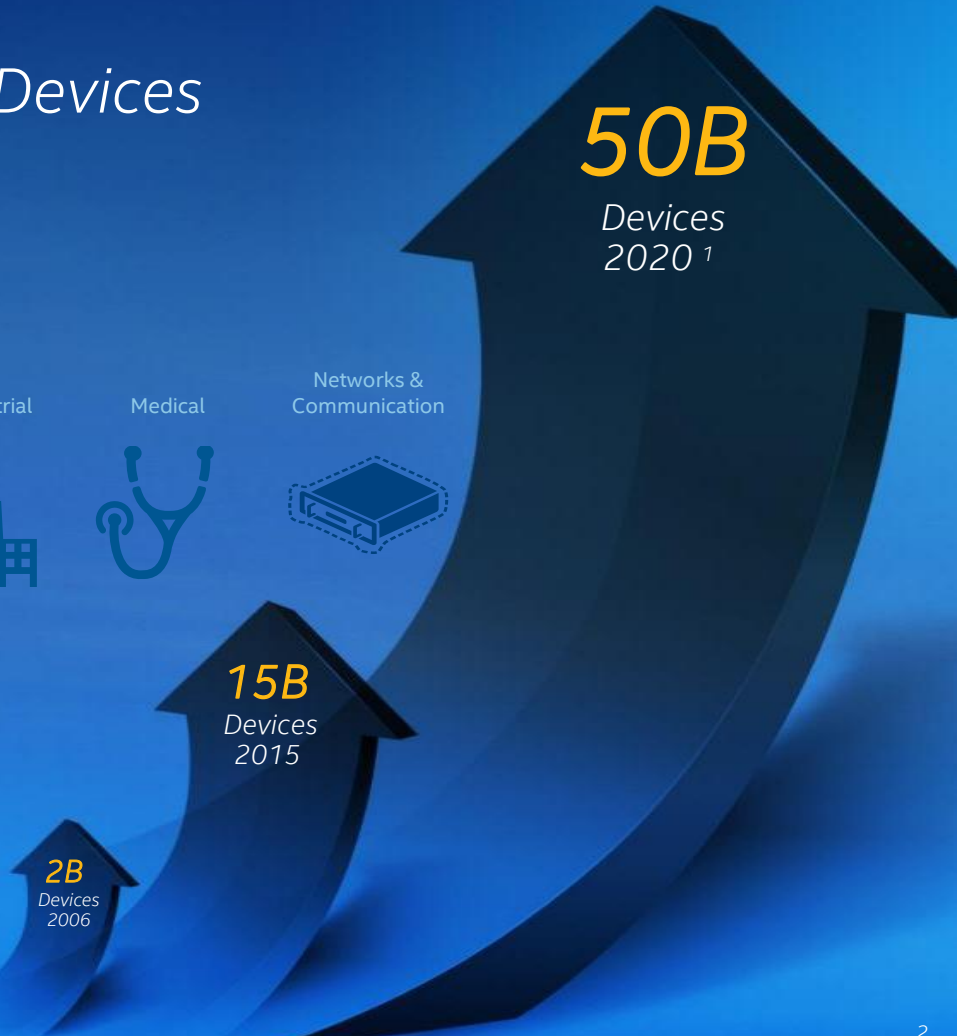
Ultra-Mobile

Wearables/IoT



Explosion Of Connected Smart Devices

... is driving an unprecedented need for efficient tools to meet shorter development cycles



Intel® System Studio

provides deep system-wide insight into power, performance, and reliability that helps accelerate time to market of Intel Architecture-based mobile and embedded systems and embedded applications

[1]IDC*, Intel, United Nations*

Intel® System Studio 2015 Overview

Deep system-wide insight into power, performance, and reliability that helps accelerate time to market of Intel Architecture-based mobile and embedded systems and embedded applications

Accelerate Time To Market

Speed-up development and testing with deep hardware and software insight

Strengthen System Reliability

Enhance system stability using in-depth system-wide debuggers and analyzers

Boost Power Efficiency and Performance

Boost system power efficiency and performance using system-wide analyzers, compilers and libraries

Embedded or Mobile System

intel Quark™

intel inside ATOM™ | intel inside CORE™i3 | intel inside CORE™i5 | intel inside CORE™i7 | intel inside XEON™

[1] *Linux*, Embedded Linux, Wind River* Linux*, Yocto Project*, Tizen**

Learn more at: http://intel.ly/system-studio

Intel® System Studio 2015 Overview

Deep system-wide insight into power, performance, and reliability that helps accelerate time to market of Intel Architecture-based mobile and embedded systems and embedded applications

DEBUGGERS & TRACING
System | Application

ANALYZERS
Power & Performance CPU / Graphics | Memory & Threading

COMPILER & LIBRARIES
C/C++ Compiler | Image, Signal, Media, Data & Math Processing

JTAG Interface

System & Application code running on Linux* [1], Android* or Windows*

Embedded or Mobile System

intel Quark™

intel inside ATOM™ | intel inside CORE™i3 | intel inside CORE™i5 | intel inside CORE™i7 | intel inside XEON™

[1] Linux*, Embedded Linux, Wind River* Linux*, Yocto Project*, Tizen*

Now also Android Lollipop* and 64-bit ready*

Optimization Notice

Intel® System Studio Workflow

Build & Optimize

System Software and Embedded Applications

Eclipse* / Workbench*/VS*

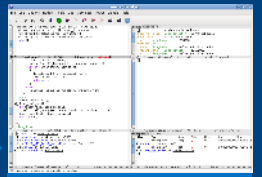
- Intel® Integrated Performance Primitives
- Intel® Math Kernel Library
- Threading Library

→ Intel® C++ Compiler incl. Intel® Graphics Technology offload

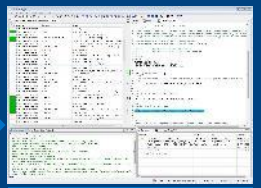
Use highly-optimized libraries and optimizing compiler technology

Debug & Trace

Applications



Intel-enhanced GDB Application Debugger



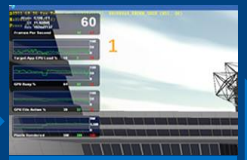
Intel® System Debugger

System Software

Analysis

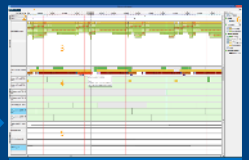
CPU/GPU workloads

During run-time

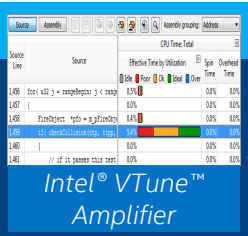


Intel® System Analyzer

Or Offline

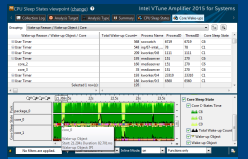


Intel® Platform Analyzer



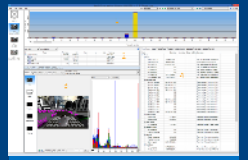
Intel® VTune™ Amplifier

Code performance on the CPU



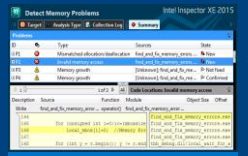
Intel® Energy Profiler

SoC system power



Intel® Frame Analyzer

OpenGL ES, DirectX graphics performance



Intel® Inspector

Application robustness (memory leaks)

Build, Debug/Trace and Optimize System and Embedded Application Code with One Integrated Solution – Intel® System Studio

Intel® System Studio 2015 Components

Category	Component	Target OS Support: Linux* [1, 5] Composer Edition	Linux* [1, 5] Professional Edition	Linux* [1, 5] Ultimate Edition	Android* [5] Composer Edition	Android* [5] Professional Edition	Android* [5] Ultimate Edition	Windows* Composer Edition	Windows* Professional Edition	VxWorks* Composer Edition
Host Operating Systems		Linux*, Windows*			Linux*, Windows*			Windows*		Linux*, Windows*
Integrated Development Environment		Eclipse*, Wind River* Workbench*			Eclipse*			Visual Studio*		Wind River* Workbench*
Compiler & Libraries	Intel® C++ Compiler	✓	✓	✓	✓	✓	✓	✓	✓	✓ [2]
	Intel® Integrated Performance Primitives	✓	✓	✓	✓	✓	✓	✓	✓	✓ [2]
	Intel® Math Kernel Library	✓	✓	✓				✓	✓	
	Intel® Threading Building Blocks	✓	✓	✓	✓	✓	✓	✓	✓	
Application Debugger	Intel-enhanced GDB* Application Debugger	✓	✓	✓	✓	✓	✓			
Analyzers	Intel® VTune™ Amplifier for Systems		✓	✓		✓	✓		✓	
	Intel® Energy Profiler		✓ [6]	✓ [6]		✓	✓		✓	
	System Analyzer					✓	✓		✓	
	Platform Analyzer [4]					✓	✓		✓	
	Frame Analyzer [4]					✓	✓		✓	
	Intel® Inspector for Systems		✓	✓					✓	
System Debugger	Intel® System Debugger (JTAG) [3]			✓			✓			

[1] Linux*, Embedded Linux, Wind River* Linux*, Yocto Project*, Tizen*
[2] Delivered with Wind River* VxWorks* platform*
[3] Via Intel® ITP-XDP3 probe, OpenOCD*, Macraigor* usb2demon* and EDKII* for UEFI*
[4] Available on Windows* host only
[5] Linux* and Android* target support available in a single product
[6] For detailed processor support please visit: https://software.intel.com/en-us/intel-energy-profiler

"With the new Intel® System Studio 2015, we improved the user experience of our recently launched Android based tablet tolino tab 8" (optimized for eReading) drastically by a factor of 3x (200ms vs. 500-700ms); which reduced the CPU workload and the resulting power consumption by at least the same factor."*
Dirk Hofmann, Chief Product Owner, Deutsche Telekom

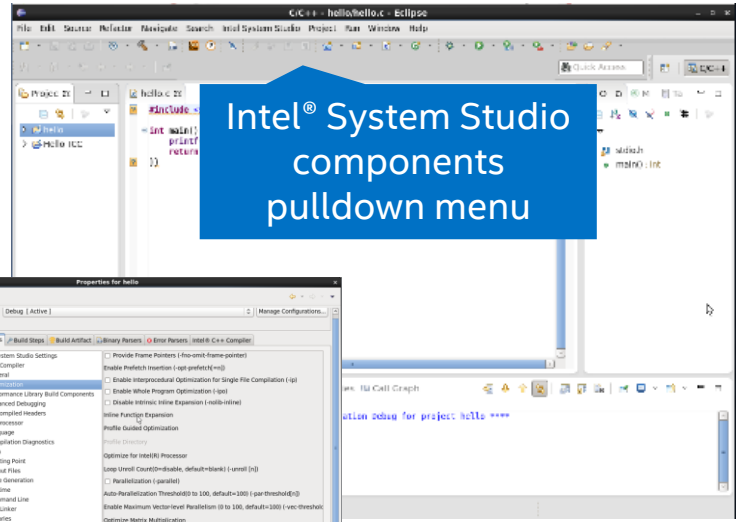
"It is well apparent that if a new customer will develop Intel Architecture based products, then Intel® System Studio tools are essential for success..."
Wayne Merrill, Manager, International Dept., Flatoak Co., Ltd./JAPAN

"I am very satisfied with Intel System Studio 2015 product. The C++ compiler is very fast and complete. The development tools that are part of these suites are very useful and they help detecting performance issues quite easily."
Eduardo Quintana, SFTWY CDI Ltda., Microsoft Partner

IDE integration

Comprehensive Eclipse* IDE Integration

- *Full integration of Intel® C++ Compiler into cross-build environments under Eclipse* IDE*
- *Integrates Intel® System Debugger & Intel® VTune™ Amplifier for Systems into Eclipse* IDE*
- *Integration of Intel enhanced GDB**



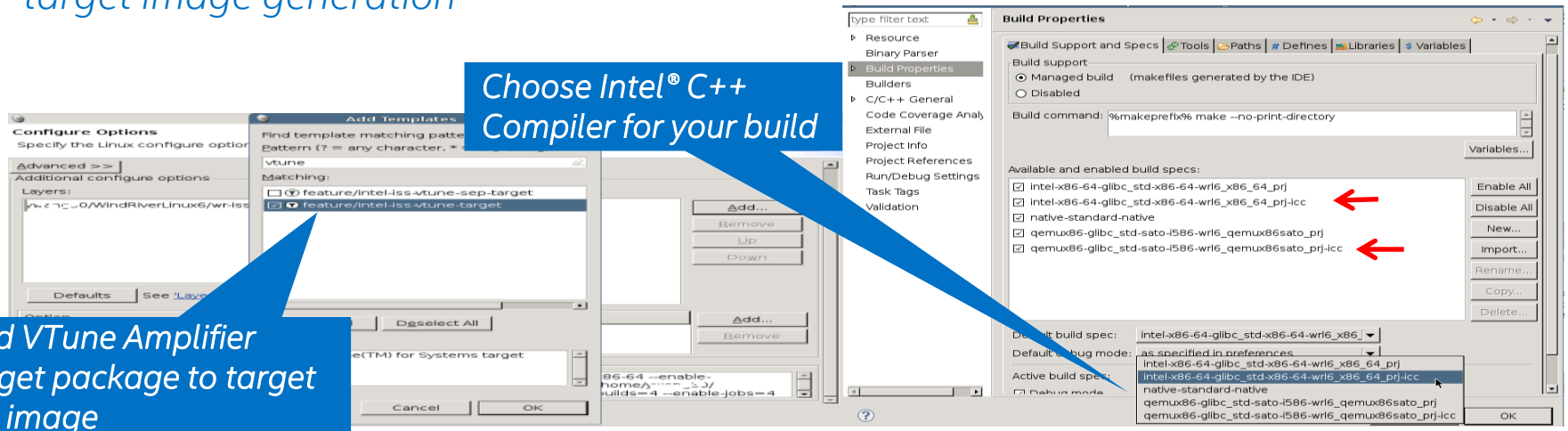
Easy configuration of Intel® System Studio components (build system, analyzers, debugger) within the IDE

Develop Fast Code Faster, Efficiently

Tight Wind River* Workbench Integration

NEW

- *Seamlessly switch between Intel® C++ Compiler and GCC**
- *Full sysroot and secondary toolchain layer support*
- *Wind River* build environment recipes*
- *Launch Intel® System Debugger & Intel® VTune™ Amplifier for Systems from Workbench**
- *Deploy runtime libraries, VTune™ Amplifier sampling collector automatically as part of target image generation*



Intel® Architecture Performance and Debug Solution for Wind River Linux**

Windows* Target Support

Plug-in to Microsoft® Visual Studio*

the standard IDE for Windows-based development*

Composer Edition

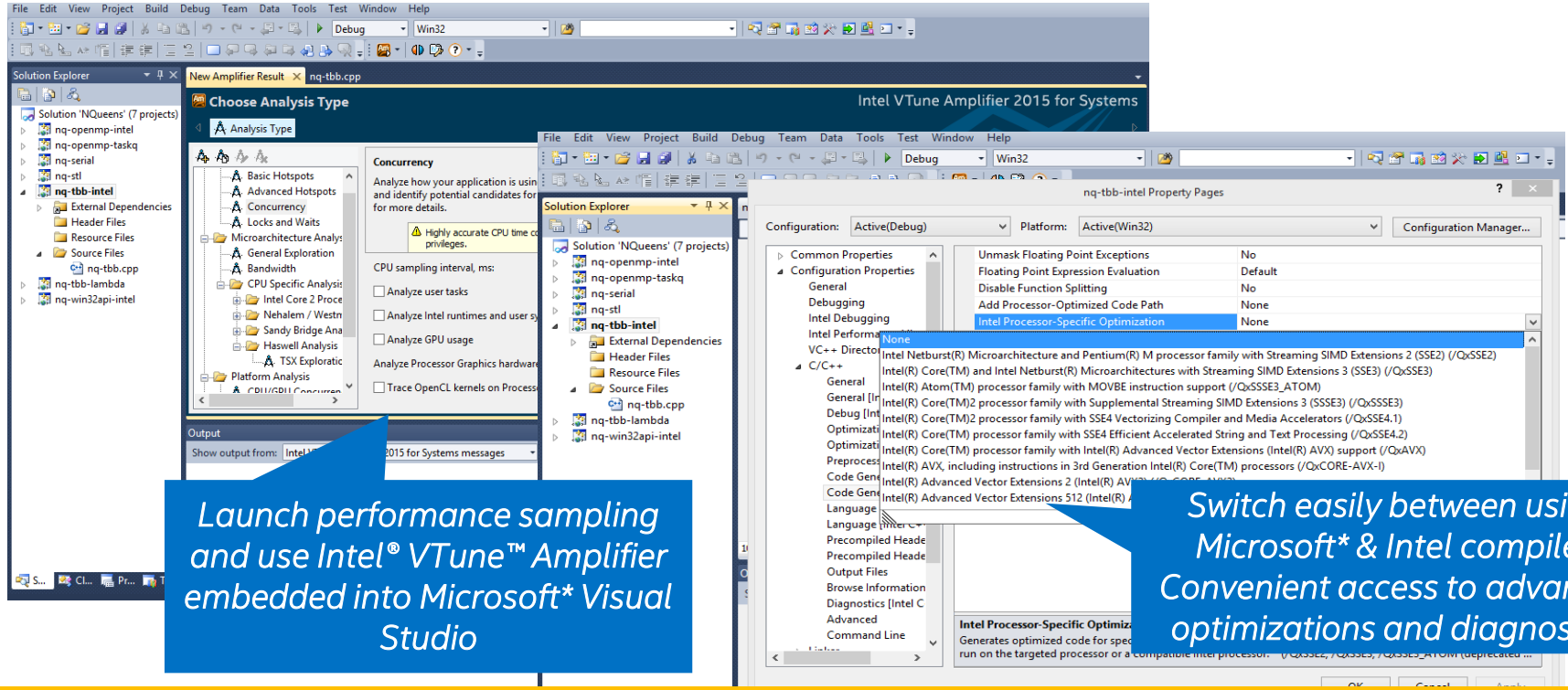
Professional Edition

- *Intel® C++ Compiler and Libraries for improved performance*
- *Intel® Integrated Performance Primitives, Intel® Math Kernel Library, Intel® Threading Building Blocks performance libraries for improved development efficiency*
- *Intel® Inspector for Systems for advanced code correctness checking*
- *Intel® VTune™ Amplifier for Systems for advanced performance profiling*
- *Intel® Energy Profiler to increase energy efficiency*

Develop Fast, Energy-efficient Embedded Windows Applications*

Intel® System Studio 2015 for Windows*

NEW



Fully Microsoft Visual Studio integrated solution to optimize Windows* applications*

Compiler and libraries

Compiler & Libraries Overview

C/C++Compiler

Image, Signal, Media, Data & Math Processing

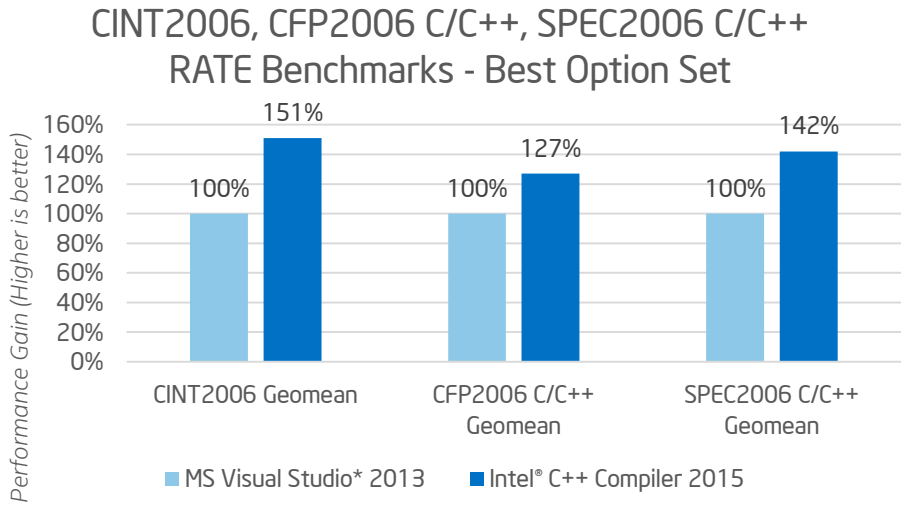
	Intel® C++ Compiler	*Intel® Integrated Performance Primitives*	*Intel® Math Kernel Library*	*Intel® Threading Building Blocks*
What does it provide	*Build highly optimized executables and libraries for Intel® Architecture*	*Performance optimized software building blocks*	*Advanced math and statistics functions*	*Scalable building blocks for advanced data and task parallelism*
Key Purpose	*Optimize performance of critical data and compute intensive workloads*	*Build highly optimized and easy to maintain signal and media processing codecs*	*Implement highly optimized solutions for data manipulation and data processing*	*Design and implement advanced task and data parallelism for highly parallel synchronized workloads*
How	*Rebuild critical workloads with architectural and parallelism optimizations*	*Integrates rich set of function primitives easily callable from workload*	*Advanced optimized libraries provide building blocks for complex math and data computation*	*Highly flexible C++ templates and synchronization primitives*

Build and Performance Solutions for Embedded Cross-Development in a World of Connected Devices

Intel® C++ Compiler Benchmarks on Windows* Targets Estimated Performance Difference

COMPILER & LIBRARIES

C/C++ Compiler | Image, Signal, Media, Data & Math Processing



Compilers
Intel® C++ Compiler for IA-32 applications, Version 15.0.0.007 Build 20140905
Intel® C++ Intel(R) 64 Compiler for Intel(R) 64 applications, Version 15.0.0.007 Build 20140905
Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1 for x86*
Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1 for x64*

Hardware Name	*nsticlew612.ins.intel.com*
Platform	*Haswell i7-4770K*
Hardware	*Microsoft* Windows* 7 SP1*
RAM	*Intel® Core™ i7-4770K CPU @ 3.50GHz*
HDD	*16GB*
	1TB

Benchmarks
CINT2006 Geomean
CFP2006 C/C++
SPEC2006 C/C++ Geomean

NOTE: 32-bit compilers for CINT2006 in RATE mode were used, as in SPEC publications

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance test, such as CINT2006, CFP2006 C/C++*, SPEC2006 C/C++*, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Benchmark Source: Intel Corporation. For more complete information about compiler optimizations, see our Optimization Notice.*

Intel® C++ Compiler for Windows Based Embedded Devices Provides A Performance Gain Up To **50%** Compared To Industry Leading C++ Compiler*

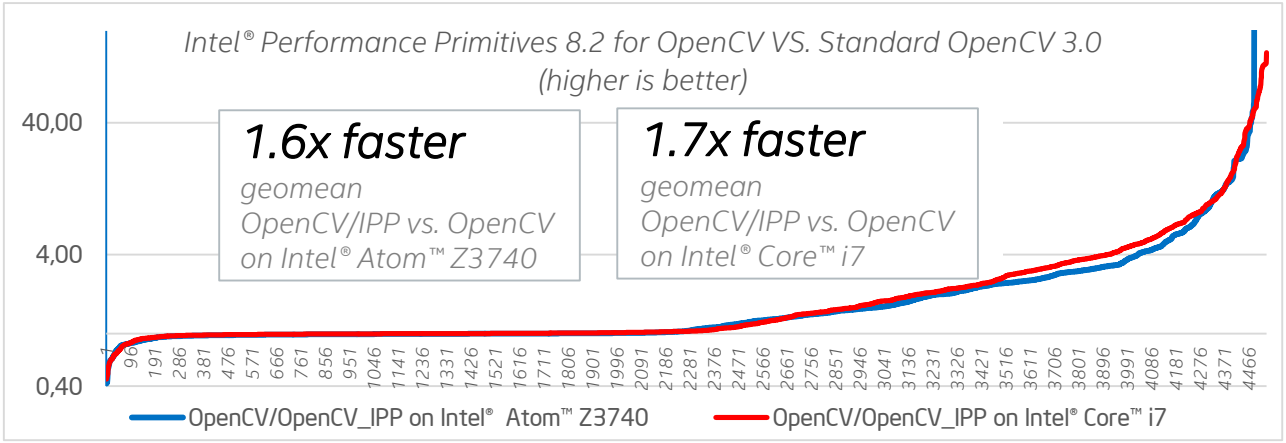
Optimization Notice

Intel® Integrated Performance Primitives for OpenCV*

COMPILER & LIBRARIES

C/C++ Compiler | Image, Signal, Media, Data & Math Processing

OpenCV *(Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. An open source version is available which calls IPP functions to gain more performance. Download: http://opencv.org/opencv-3-0-alpha.html*



Hardware
Intel® Core™ i7-4771 CPU @ 3.50GHz 16GB, Intel Atom™ Z3740

Platform
Microsoft Windows* 8.1*

Software
Intel® Integrated Primitives 8.2 for OpenCV (ICV)*
OpenCV 3.0 Alpha
Standard OpenCV test framework

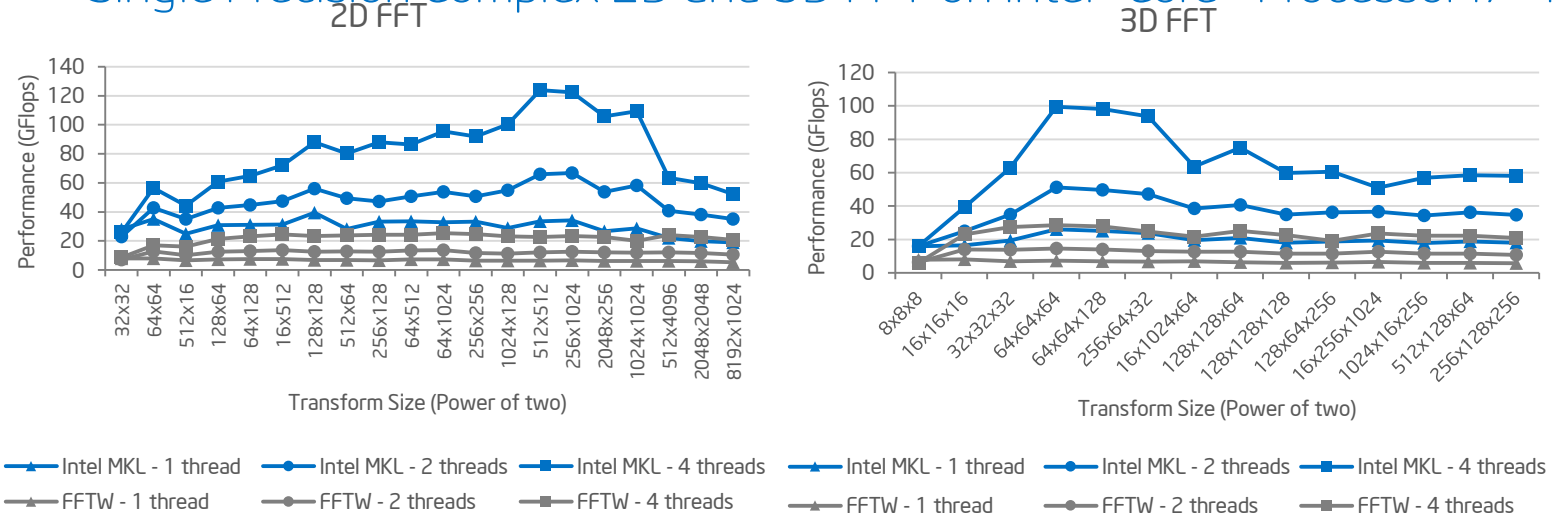
Test case
174 of 270 OpenCV functions call ICV functions.
4500 of 8000 performance tests (functions' with parameters changed in the loops) have ICV calls.
Test system config: --perf_min_samples=100 --perf_force_samples=100 parameters to stick to 100 samples per test (to diminish influence of random extremes).

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance test, such as OpenCV test framework are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Benchmark Source: Intel Corporation. For more complete information about compiler optimizations, see our Optimization Notice.

OpenCV 3.0 Performance Increases with Intel® Performance Primitives Optimizations Up To Factor **1.6x** (geomean)*

FFT Performance Using Intel® MKL vs. FFTW*

Single Precision Complex 2D and 3D FFT on Intel® Core™ Processor i7-4770K



Hardware
Intel® Core™ Processor i7-4770K, Quad-core CPU (8MB LLC, 3.5GHz), 8GB of RAM

Platform
RHEL 6.1 GA x86_64

Software
Intel® Math Kernel Library (Intel® MKL) 11.2
FFTW 3.3.4;*

Benchmark Source: Intel Corporation.

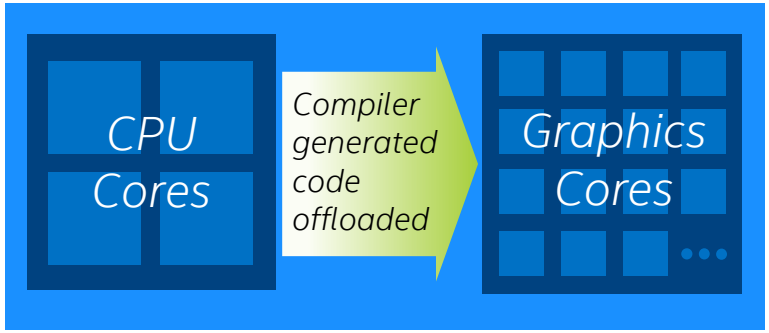
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance test, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Benchmark Source: Intel Corporation. For more complete information about compiler optimizations, see our Optimization Notice.

Intel® Math Kernel Library Helps to Boost the Performance of 2D FFT Image Processing Algorithms Significantly Over Open Source FFTW Library*

Offload Compute-intensive Code To Integrated Graphics Cores

- Compiler-generated code executed across CPU and graphics cores with simple #pragma
- Employ Intel® Cilk™ parallel extensions for highly parallel execution across graphics cores

Intel® Core™ Processors and Intel Xeon Processors with Intel HD or Intel Iris Pro Graphics



Example: Offloading Using Perfectly Nested _Cilk_for Loops

```
float (* A)[k] = (float (*)[k])matA;
float (* B)[n] = (float (*)[n])matB;
float (* C)[n] = (float (*)[n])matC;

#pragma offload target(gfx) if (do_offload) \
      pin(A: length(m*k)), pin(B: length(k*n)), pin(C: length(m*n))

      __Cilk_for (int r = 0; r < m; r += TILE_m) {
            __Cilk_for (int c = 0; c < n; c += TILE_n) {
                  …
            }
      }
```

For Excellent Image And Signal Processing Performance In Embedded Applications

Accelerate Build and Design
Intel® System Studio

COMPILER & LIBRARIES

C/C++ Compiler	Image, Signal, Media, Data & Math Processing

Intel® C++ Compiler

- Highly optimized build with Intel® C++ Compiler
- Seamless integration into your cross-build environment

Intel® Threading Building Blocks

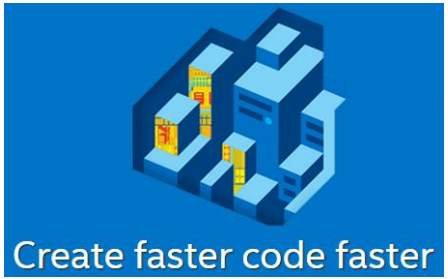
- Advanced parallel programming models for tuning concurrent programming flows

Intel® Integrated Performance Primitives

- Function primitives building blocks for media and signal processing

Intel® Math Kernel Library

- Highly optimized library for math and data analytics and manipulation



Accelerate Performance and Time-to-Market

Debuggers

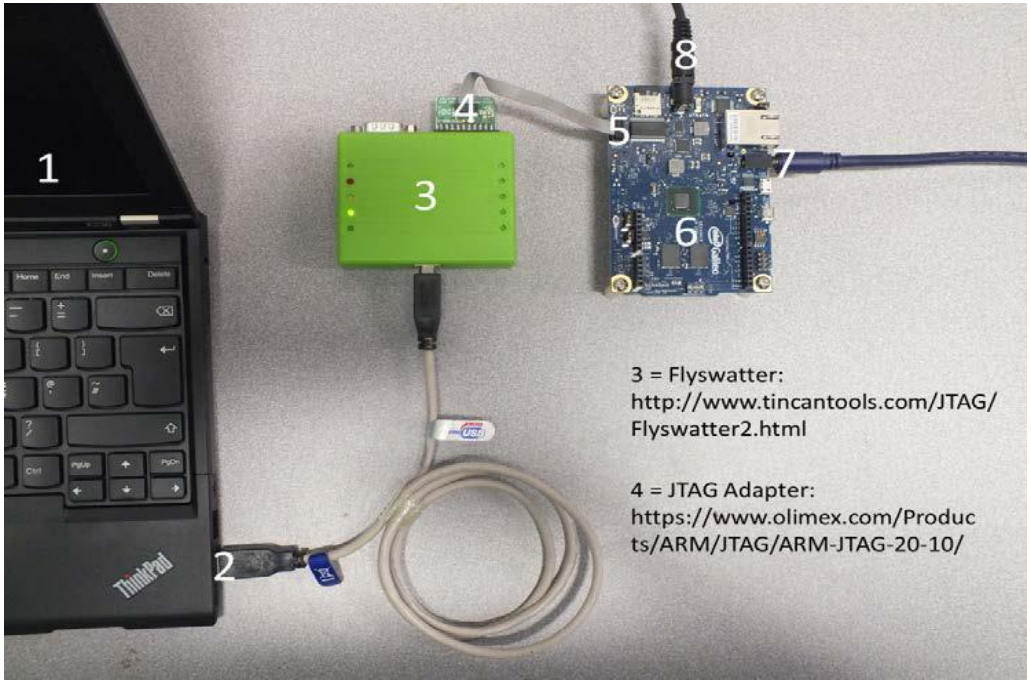
Debugger & Trace Overview

System — Application

	Intel® System Debugger	*SVEN Technology SDK*	*Intel-enhanced GDB* Application Debugger*
What does it provide	*Interactive system software runtime issue debug with deep architectural, UEFI and OS insight*	*Static instrumentation based data event tracing across Linux* kernel and application space*	*Remote interactive application cross-debug with process specific branch trace and data race debug capabilities*
Key Purpose	*Identify and fix deterministic system software defects*	*Identify and fix timing sensitive and non-deterministic data processing issues (especially across user space/kernel space boundary)*	*Identify and fix deterministic application software defects*
How	*Symbolic source level JTAG and debug agent run-control based stepping and execution flow analysis*	*Log and analyze data event traces in trace viewer and correlate to execution flow and time stamps*	*Use remote debug agent for symbolic source level stepping. Use process specific branch trace store to augment callstack.*

Deep insight into program flow and architecture for the entire software stack helps to resolve issues faster

Intel® System Debugger and Intel® Quark SoC (2)



Recommended setup for debugging with OpenOCD*

- Host System
- U
- JTASB 2.0 male-male A-B cableG Probe
- ARM-JTAG-20-10 Adapter
- JTAG Port
- Intel® Galileo Board
- Serial Cable to view boot process
- Power Supply

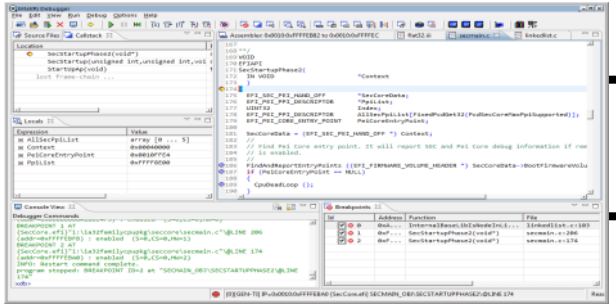
Intel® System Debugger

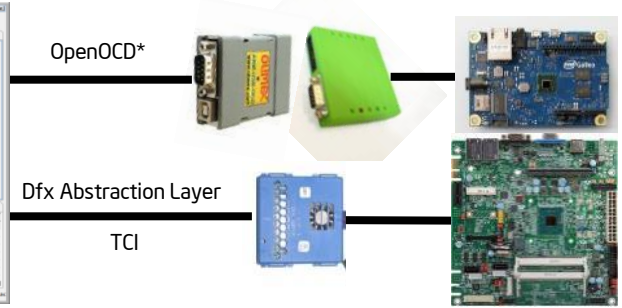
DEBUGGERS & TRACING

System | Application

Key Features

- Linux* and Windows* host
- JTAG debug for Intel® Atom™, Core™, Xeon® & Quark™ SoC-based platforms
- EFI/UEFI Firmware, bootloader debug, Linux* OS awareness
- In depth visualization of memory configuration, system state and register sets
- Dynamically loaded Linux kernel module debug
- LBR & Intel® Processor Trace On-Chip instruction trace support





Example: Intel® Galileo, Intel® Quark SoC X1000

Example: , Intel® Atom™ Processor E3825

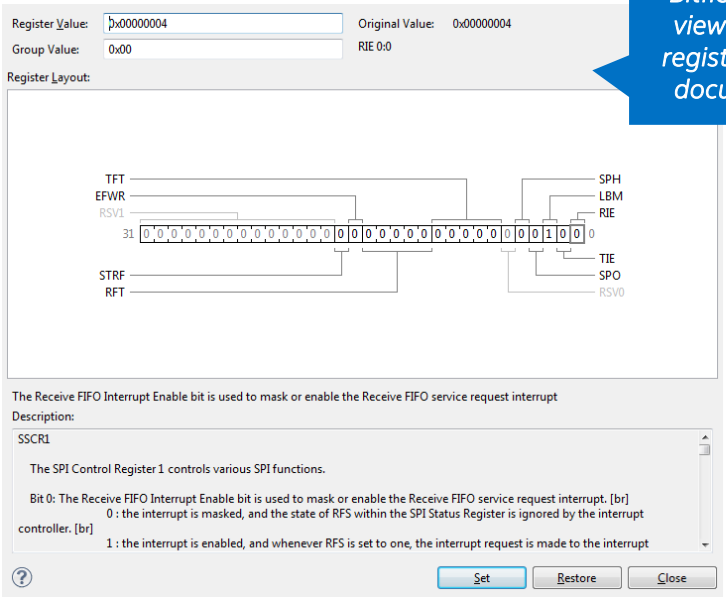
Intel® ITP-XDP 3BR for all Intel platforms
Additionally TinCanTools Flyswatter2* and Olimex* ARM-USB-OCD-H for Intel® Quark™ SoC

Complete system debug solution provides deep Insights into memory and system configuration

System Debugging For Intel® Quark™ Platforms

- Supports connection via low-cost OpenOCD*-based JTAG devices
- Insight to Intel® Quark™ SoCs

JTAG devices available for <$100

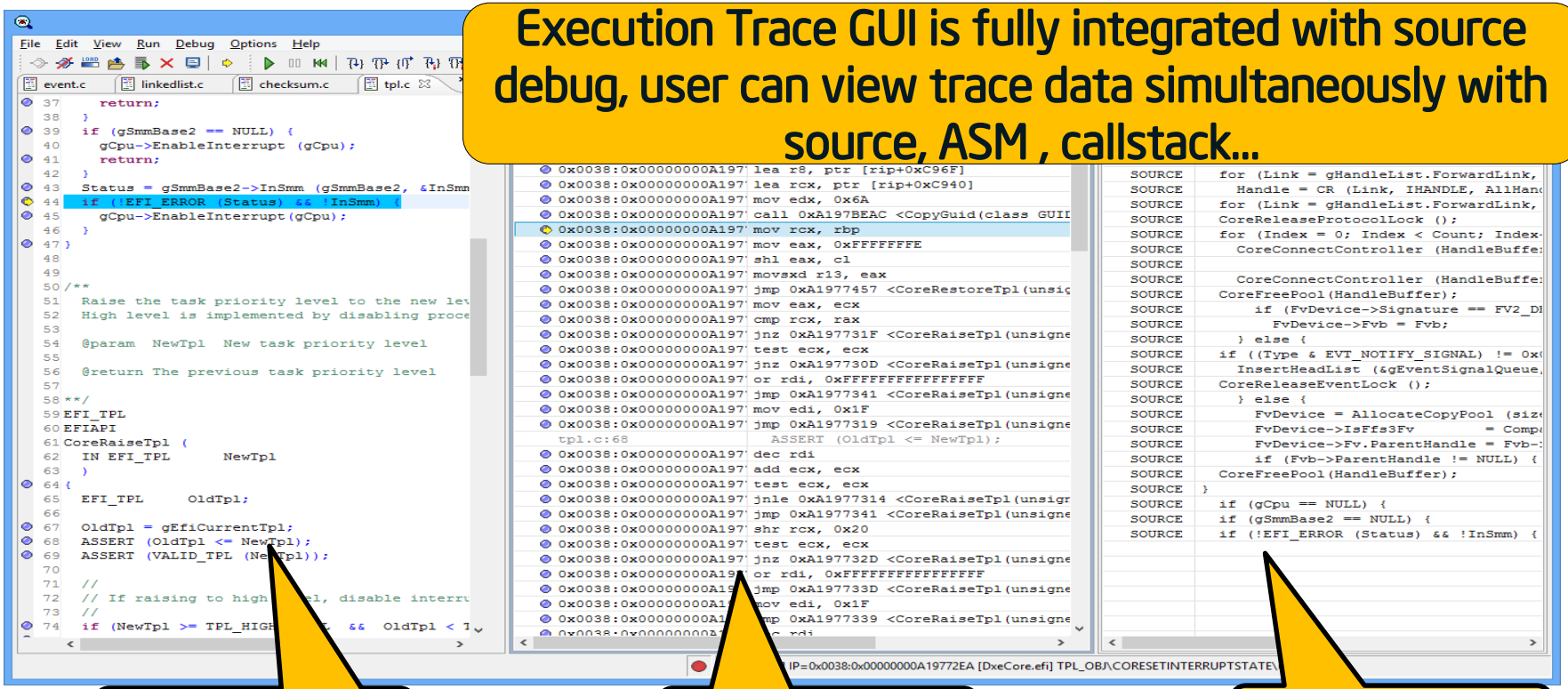


Bitfield Editor to view peripheral registers, incl. full documentation

Full Featured, Low-cost System Debug For Intel® Quark™ Platforms

Execution Trace GUI Overview

Execution Trace GUI is fully integrated with source debug, user can view trace data simultaneously with source, ASM , callstack…

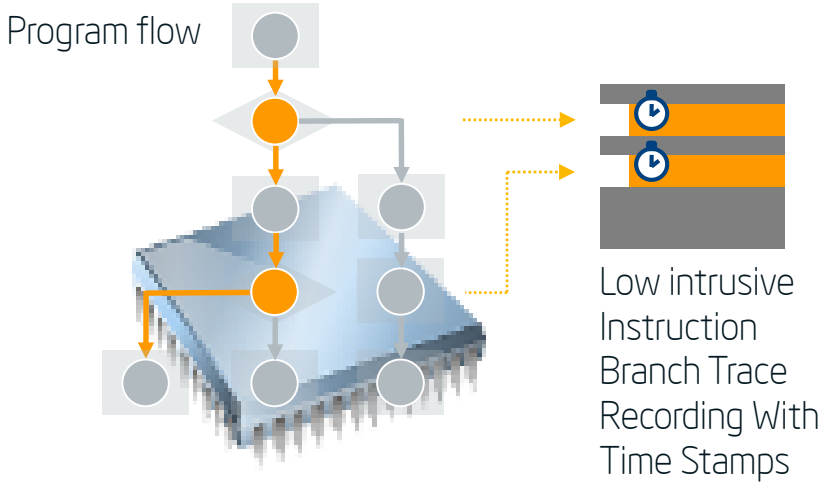


Low-overhead Processor Trace In New Intel® Core™ M Processor

Intel® System Studio low-overhead hardware-assisted tracing capabilities help developers isolate non-deterministic errors

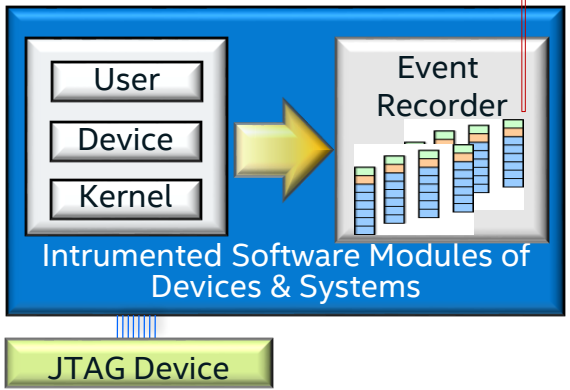
Intel® System Debugger 2015 For Program Execution History & Debugging

- *Intel® Processor Trace - low-overhead hardware based tracing capability on chip*
- *Now exposed with new **Intel® Core™ M processors***



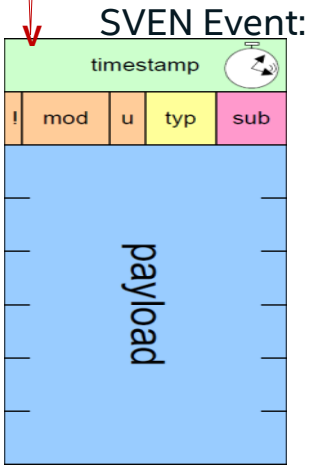
Isolate and resolve defects quickly

SVEN SoC Trace – low overhead technology for static instrumentation of key SoC components



- Event tracing with JTAG
- Smart breakpoints
  - Interrupt execution on key events

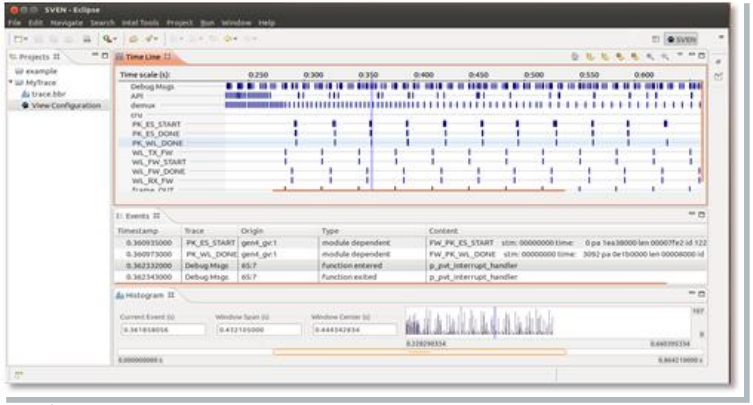
- Visualize, Analyze and Debug system event traces
  - Command line interface
  - Graphical interface



SVEN - A Stethoscope for your System System & SoC trace through JTAG

Trace Visualization

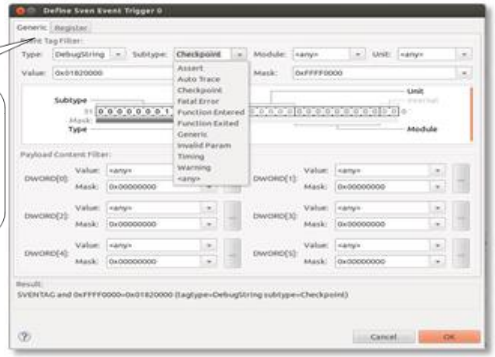
- Advanced navigation, search & filter
- Graphical and textual event display
- User controlled trace line grouping



Smart Event Triggers

- Live JTAG system debug with event tracing
- Smart breakpoints that interrupt execution on trace e nt calls
- Set smart breakpoints for in-depth analysis
  For example:
  - Break on any event from the USB driver
  - Break on any Debug String that starts with "E
  - Break if register X is accessed
  - Break if register X bits [7-9] have value 0b101

- Timeline view
- Search & Filter
- Smart Event Trigger definition



Enhance system stability through powerful JTAG & event tracing

Software Instrumentation Example

```
static void ipc_message_received(
    struct HostIPC_Handler           *hipc,
    struct Host_IPC_ReceiveQueue     *rcv_q,
    const struct _IPC_MessageHeader  *mh,
    const char                       *message,
    unsigned int                      message_size )
{
    struct Host_IPC_QueuedMessage    *msg;

    DEVH_FUNC_ENTER(hipc->devh);

    /** Queue the message for reading with HostIPC_GetNextInboundMessage() */
    DEVH_ASSERT( hipc->devh, (message_size > 0) );
    DEVH_ASSERT( hipc->devh, (message_size <= CONFIG_IPC_MESSAGE_MAX_SIZE) );
    DEVH_ASSERT( hipc->devh, (mh->ipc_mh_dst_qnum < CONFIG_IPC_HOST_MAX_RX_QUEUES) );

    devh_ReadReg32( hipc->devh, CONFIG_IPC_ROFF_DOORBELL_STATUS );

    if ( NULL != (msg = HostIPC_GetFreeMessage(hipc)) )
    {
        if ( message_size > CONFIG_IPC_MESSAGE_MAX_SIZE )
            message_size = CONFIG_IPC_MESSAGE_MAX_SIZE;

        msg->mh = *mh;  /* copy the header */
        memcpy( msg->msg, message, message_size );

        /* Add to inbound messages */
        OS_LIST_ADD_TAIL( &msg->node, &rcv_q->inbound_msgs );
    }
    else
    {
        DEVH_WARN( hipc->devh, "HIPC_RX_OVF" );
    }

    DEVH_FUNC_EXIT(hipc->devh);
}
```

WHO ?
DID WHAT ?
AND WHEN ?

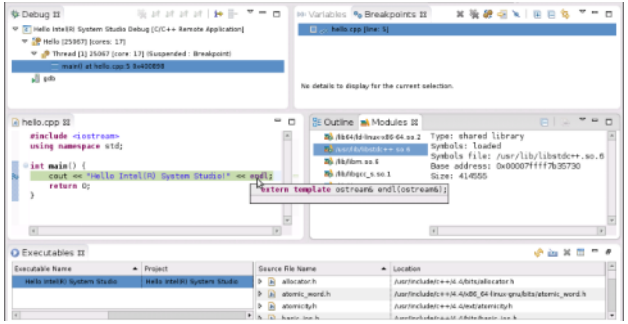
Intel® System Studio
Application Debug and Trace

DEBUGGERS & TRACING

System | Application

Intel-enhanced GNU* GDB

- Plug into existing Eclipse* IDE for increased productivity
- Debug issues where symptoms are not visible immediately
- Remote debug with branch trace and data race detection
- Intel® enhanced GDB* Debugger with pre-build binaries for Yocto Project* and WR Linux* targets



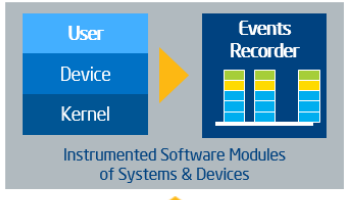
SVEN SDK

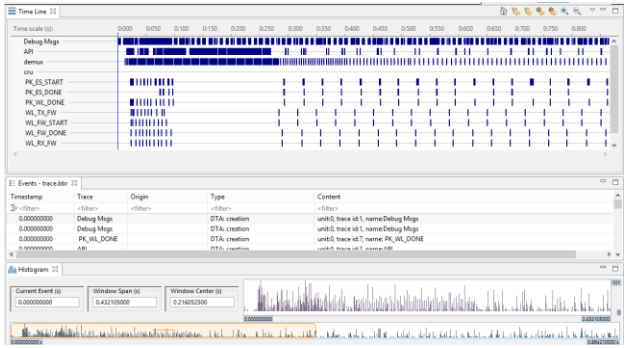
Detailed SoC & CPU System View

- Multiple cores (IA, DSP, other)
- User and kernel code

Ultra-low Overhead Sampling

- Can remain in production builds
- Around 1/10 of a microsecond
- Well defined event structure





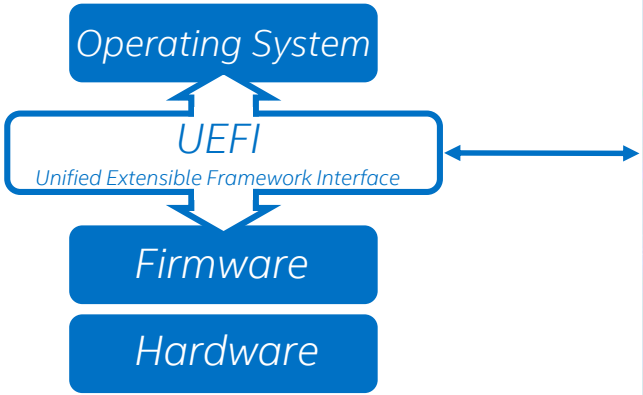
Intel® enhanced debug and trace solutions help to resolve issues fast

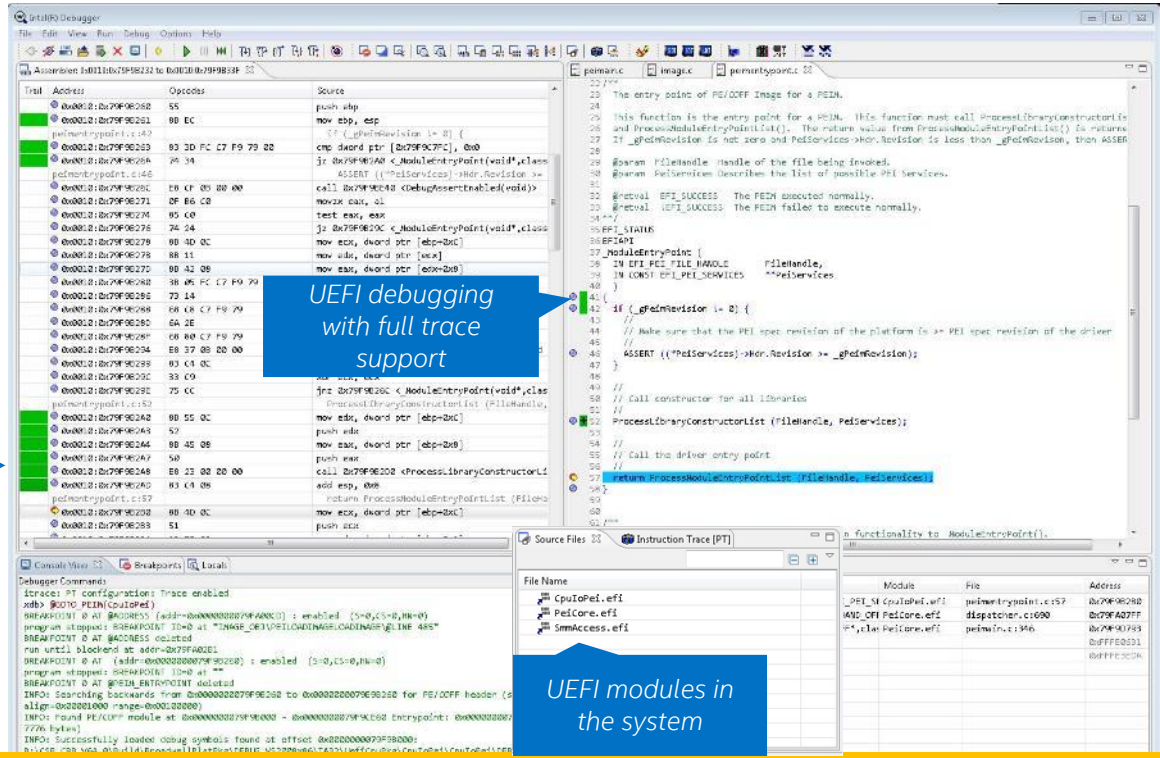
Intel® System Debugger
Advanced UEFI Debugging

DEBUGGERS & TRACING

System | Application

- *Full access and visibility to UEFI through JTAG or USB connections*
- *Helps quickly isolate non-deterministic bugs*





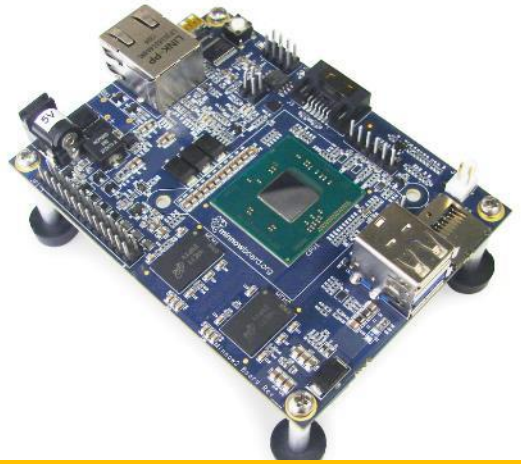
Improve System Boot Time And Reliability

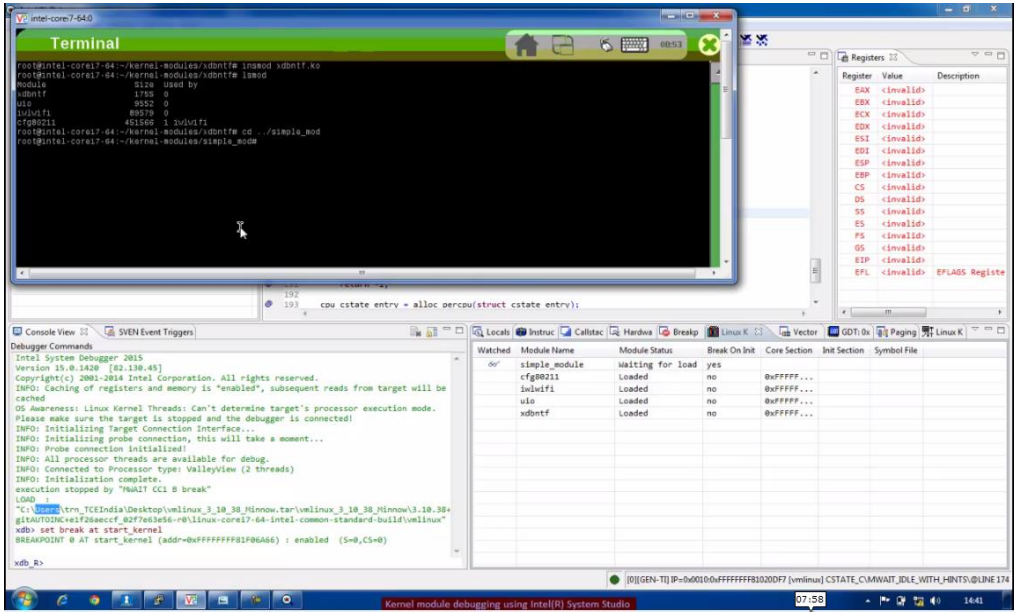
Linux* Kernel Module Device Debug on Intel® Atom™ E38xx - MinnowBoard MAX

DEBUGGERS & TRACING

System | Application

Video Demo





Dynamically loaded device driver debug without instrumentation

Accelerate Debug and Trace with Intel® System Studio

DEBUGGERS & TRACING

System | Application

Intel-enhanced GNU* GDB

- Process specific instruction trace and interactive data race debug
- Accelerates finding and resolving system software reliability issues

Intel® System Debugger

- Deep insight into application and system software
- Understanding memory and platform configuration
- Full-features low-cost JTAG debug solutions for makers and IoT edge devices

SVEN Technology SDK

- Ultra-low overhead data event tracing for hard to reproduce problems

NEW

JTAG devices available for <$100

Accelerate Reliability and Time-to-Market

Analyzers

Analyzer Overview

Power & Performance
CPU / Graphics

Memory & Threading

	Intel® VTune™ Amplifier for Systems	*Intel® Energy Profiler*	*Intel® Inspector for Systems*	*System Analyzer*	*Platform Analyzer*	*Frame Analyzer*
What does it provide	*In-depth analysis of CPU performance*	*System-wide energy analysis*	*Memory and thread analyzer*	*Overview of system performance with real-time view*	*Overview of system performance with offline view*	*Single-frame graphics analysis for DirectX and OpenGL ES workloads*
Key Purpose	*Optimize compute workloads for CPU*	*Boost power efficiency and battery life*	*Enhance system and application robustness*	*Optimize graphical workloads across CPU/GPU*	*Optimize graphical workloads across CPU/GPU*	*Optimize graphical workloads at the detailed draw call level*
How	*Uses system counters and sampling to identify performance hot spots*	*Uses processor sleep states to identify power sapping execution*	*Identifies memory and threading conflicts*	*Make real-time experiments with graphics*	*Make offline analysis of CPU and GPU metrics and workloads*	*Experiment with graphics workloads to improve performance and visual quality*

Comprehensive Platform Analysis Capabilities To Boost Power Efficiency, Performance And Robustness

Intel® VTune™ Amplifier for Systems
Performance Profiling for Embedded and Mobile Devices

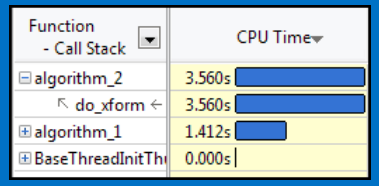
ANALYZERS

Power & Performance
CPU / Graphics

Memory & Threading

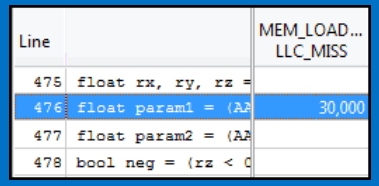
Where is my system ...

Spending Time?



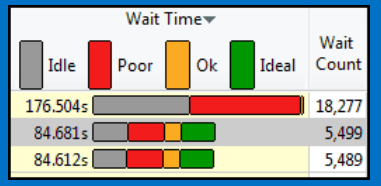
- Focus tuning on functions taking time
- See call stacks
- See time on source

Wasting Time?



- See cache misses on your source
- See functions sorted by # of cache misses

Waiting Too Long?



- See locks by wait time
- Red/Green for CPU utilization during wait

- Linux & Windows hosted cross sampling for Embedded & Mobile target devices
- Low overhead & easy to use
- No special recompiles

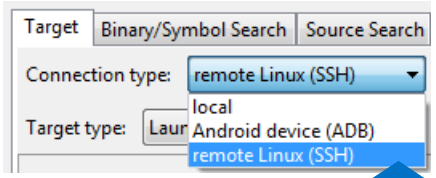
Advanced system profiling for scalable performance

Enhanced Intel® VTune™ Amplifier For Systems

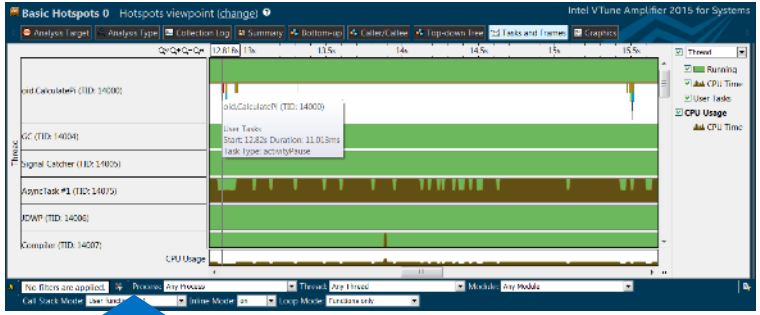
ANALYZERS

Power & Performance CPU / Graphics | Memory & Threading

- *Easy remote target collection through graphical interface*
- *Android* 64-bit Lollipop* and Android* Run-Time (ART) support*
- *Correlate Android Systrace and Linux* Perf or F-Trace information with other performance data*
- *Advanced Android graphics and life cycle events analysis*
- *Drivers for Java performance analysis now pre-installed into Android Lollipop simplifying usage*



Graphical interface to collect data on a remote Linux system via SSH or Android* via ADB*



Correlate Android Framework information with other VTune™ performance data

Improve Performance and End-User Experience of Intel® Architecture-based Devices

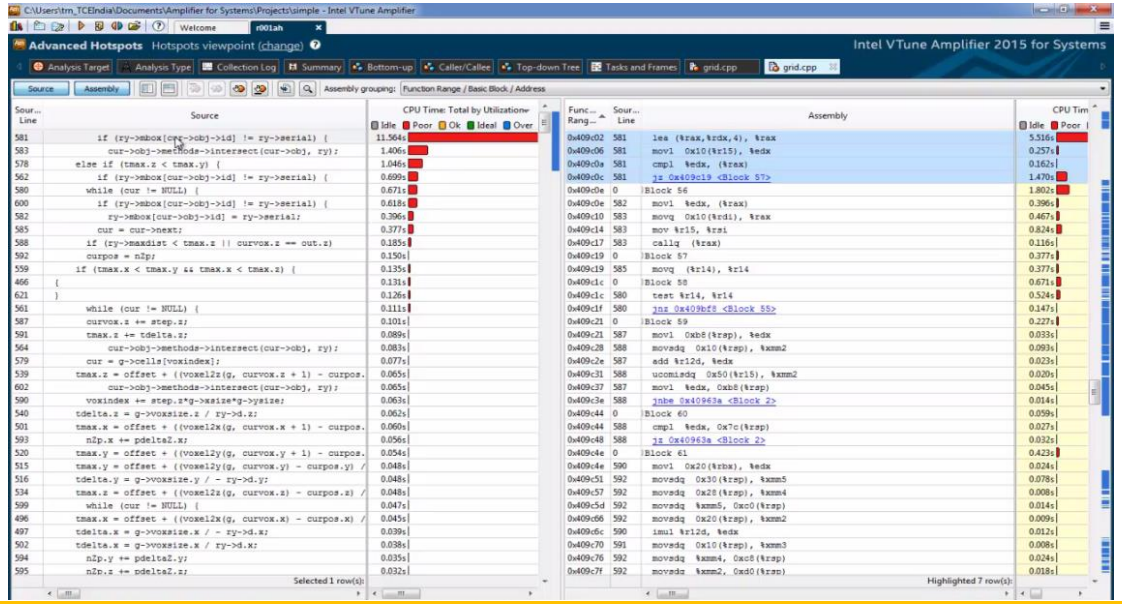
Optimization Notice

Intel® VTune™ Amplifier for Systems Remote Performance Data Sampling

ANALYZERS

Power & Performance CPU / Graphics | Memory & Threading

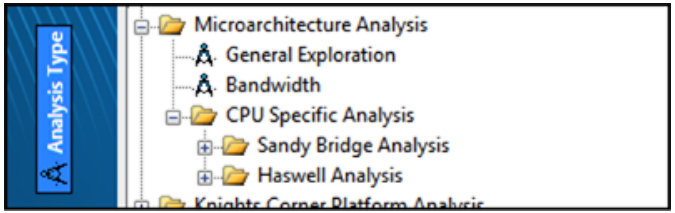
Video Demo

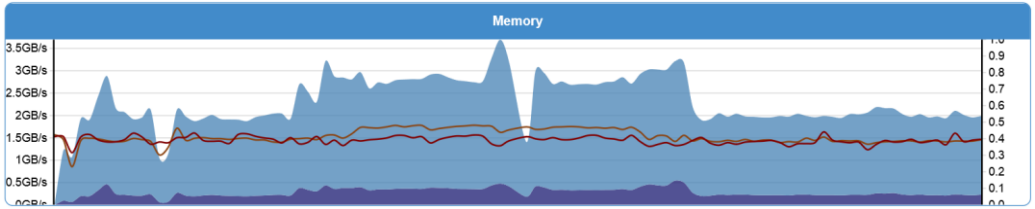


Boost performance with remote system-level and application level analysis

Intel® VTune™ Amplifier for Systems added bandwidth analysis

- *Preset profiles provide an easy "point and shoot" set-up. No memorizing complex event names. Advanced profiles like bandwidth analysis, cache analysis and branch mis-predictions find tuning opportunities*
- *Now with SoC, DDR memory and GPU bandwidth analysis*





More performance insights into Intel® Architecture SoC components and peripherals

Intel® System Studio Graphics Analysis Tools

ANALYZERS

Power & Performance CPU / Graphics | Memory & Threading

- Real-time system-level performance analysis for Intel-based Android and Windows devices
- Immediate experiments and overwrites enable developers to isolate CPU and GPU performance problems
- Metrics for CPU, GPU, API, memory, network, power, etc.
- Frame-level graphics tuning for DirectX and OpenGL ES workloads

Optimize CPU and Intel® HD Graphics performance with real-time profilers

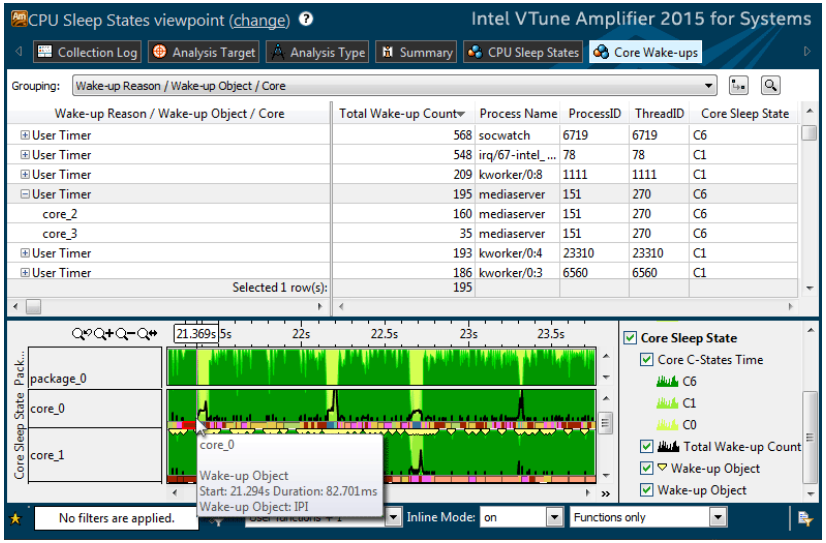
Intel® Energy Profiler

Enhanced Intel® Energy Profiler

ANALYZERS

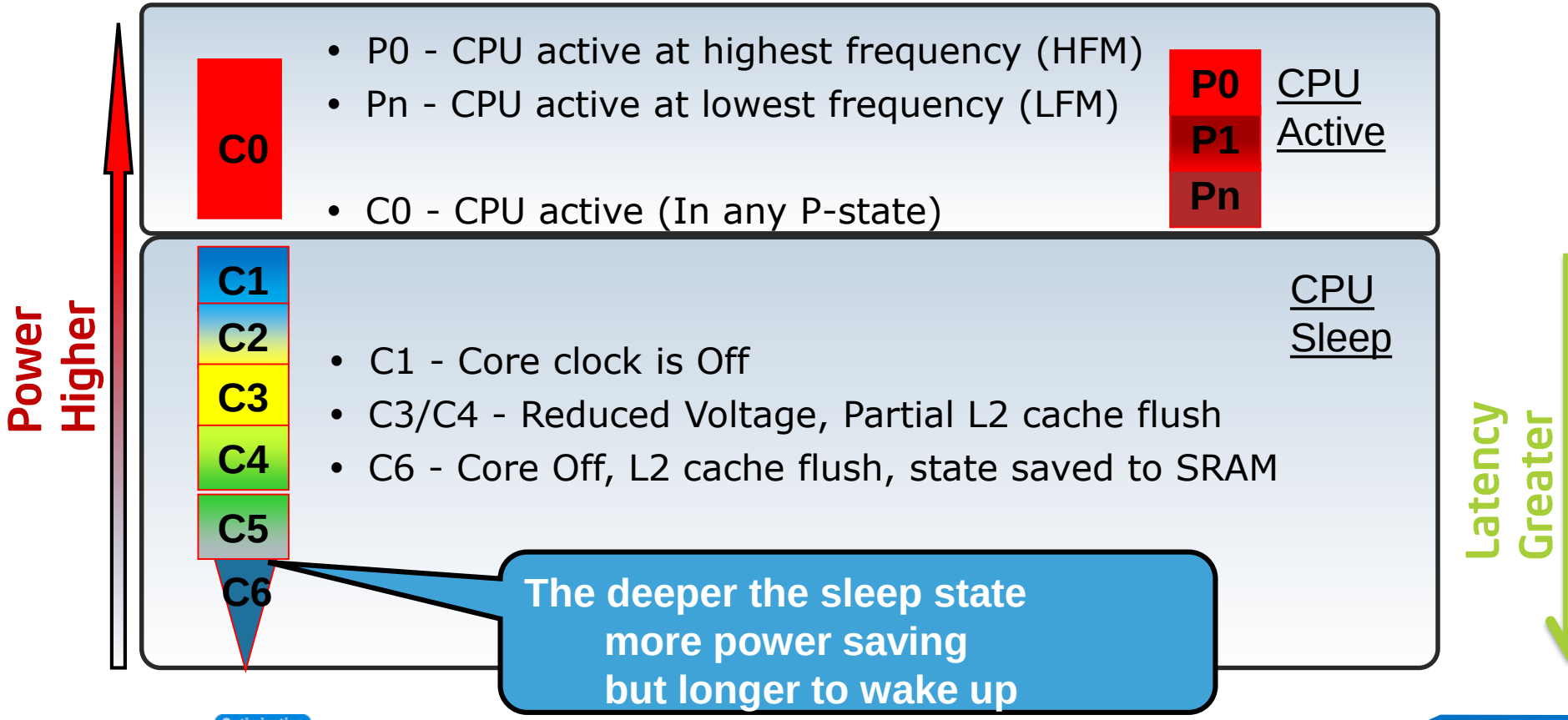
Power & Performance CPU / Graphics | Memory & Threading

- *Available now for Windows* targets*
- *Supports new Intel® Core™ M processor*
- *Correlates system activity to source code to identify power sapping implementations*
- *New power data for graphics processors and DRAM self-refresh*



Improve Power Efficiency for Intel® Architecture-based Devices

CPU C-States / P-States



Components of power analysis

Idle vs. Active

- Do nothing efficiently
- Hurry up and get idle.

    e.g. Multi-threading (distributing work evenly across cores)

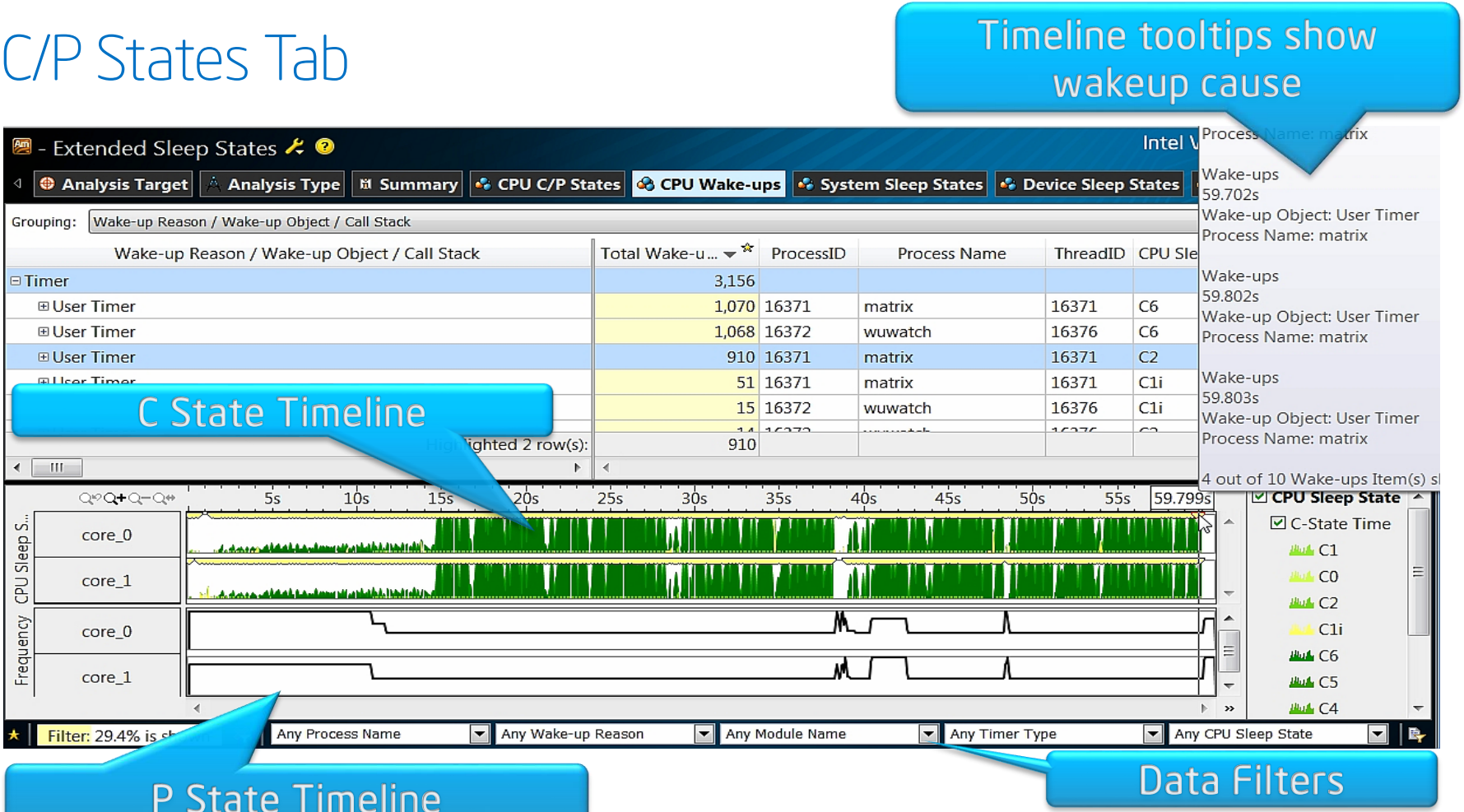
Optimize Sleep Behavior

- Minimize sporadic wakeups.
- Schedule all periodic activities from the app into same wakeup period.
- What is waking h/w from low power states? Why?

Optimize Utilization

- What is active? *Why* is it active?
- Minimize Polling Loops. Use event driven framework when possible.
- Turn devices off. Open devices can prevent the system from entering power saving state.

C/P States Tab



What C-State residency numbers tell you

Residency in the deepest C-state should be >95% @ idle

If you see high residency in C0 state

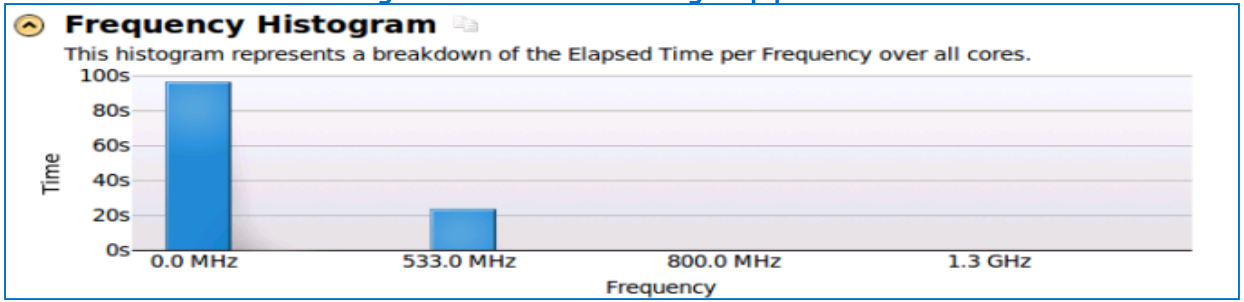
- CPU is executing instruction
- Next Step: Active Analysis

If you see high residency in the intermediate C-states

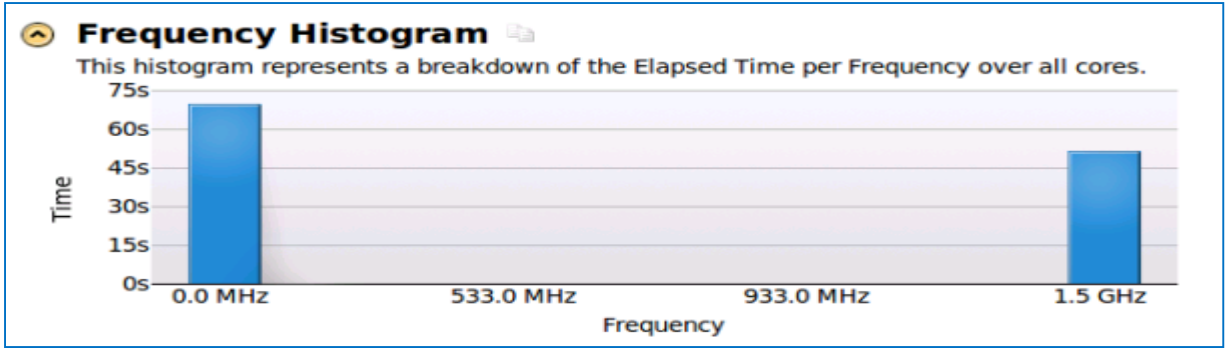
- Frequent active/idle transition is keeping CPU from entering deepest C-state. Possible causes are:
  - Application scheduling periodic timer with short period activity
  - Application waits for interrupts (from device, IPI) very frequently

Comparisons

Native Video Playback vs 3rd Party Application



Native Video Playback

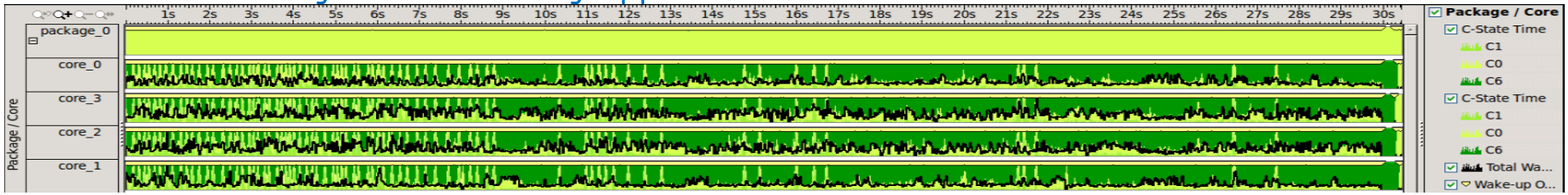


3rd Party Application

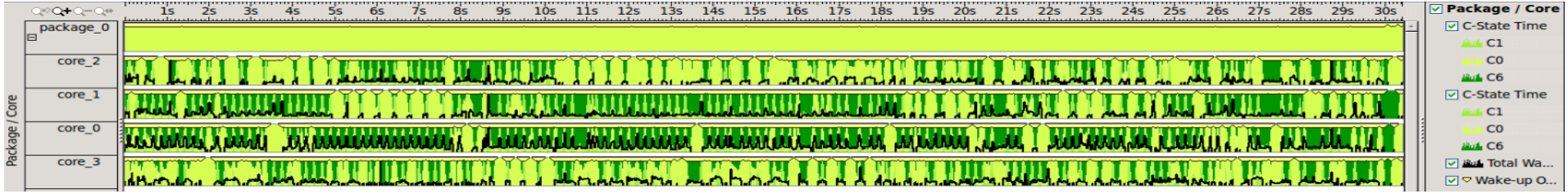
Which is "Better"?

Comparisons

Native Video Playback vs 3rd Party Application



Native Video Playback

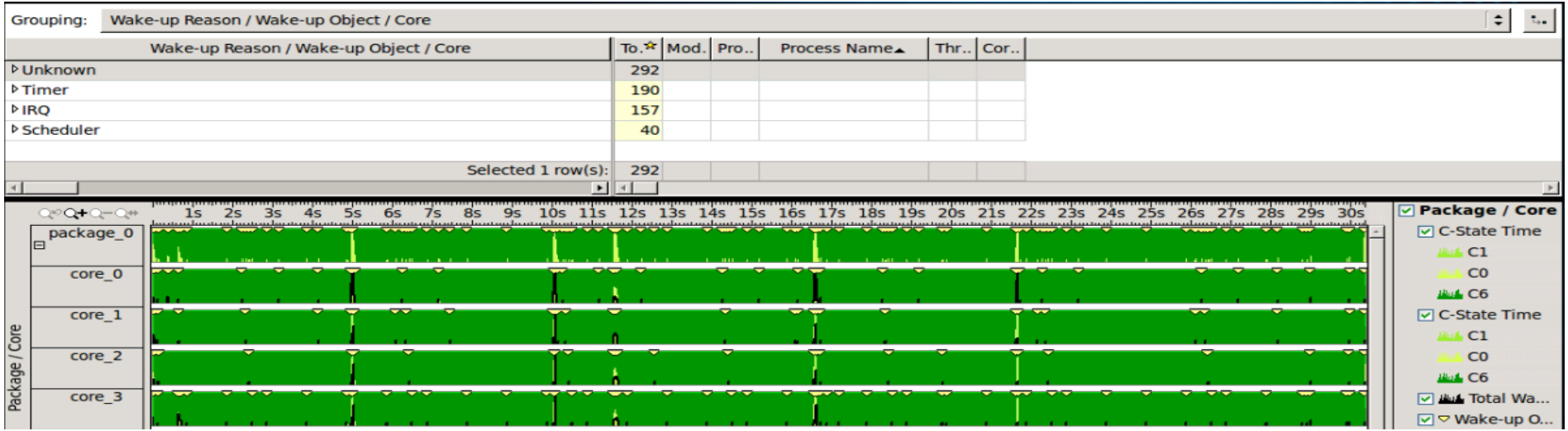


3rd Party Application

Much more time spent in C0 for the 3rd part application

Comparisons

Adding a new Kernel Driver

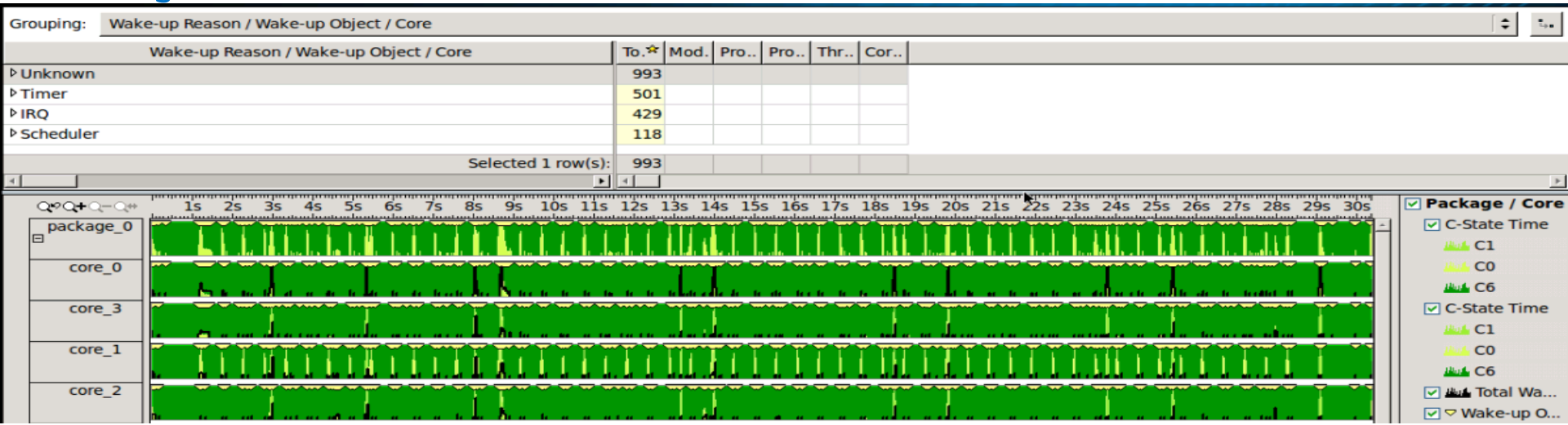


Idle Workload

Note the Wake-up Counts

Comparisons

Adding a new Kernel Driver



New Driver

Note the Wake-up Counts

Comparisons

Adding a new Kernel Driver

Reduce the kernel logging (printk) in driver

```
void my_timer_callback( unsigned long data )
{
   int i,j,ret;
   for (i=0;i<5000000;i++) {
     for (j=0;j<5000000;j++) {
        sum=sum+i;
   printk( "my_timer_callback called %d\n", count );
  }}
  // setup the timer again to fire  500ms
  if (count <50) {
     setup_timer( &my_timer, my_timer_callback, 0 );
     ret = mod_timer( &my_timer, jiffies +
msecs_to_jiffies(500) );
     if (ret) printk("Error in
my_timer_callback\n");
     count++;
  }
}
```

```
void my_timer_callback( unsigned long data )
{
   int i,j,ret;
   for (i=0;i<5000000;i++) {
     for (j=0;j<5000000;j++) {
        sum=sum+i;
   //printk( "my_timer_callback called %d\n", count );
  }}
  // setup the timer again to fire  500ms
  if (count <50) {
     setup_timer( &my_timer, my_timer_callback, 0 );
     ret = mod_timer( &my_timer, jiffies +
msecs_to_jiffies(500) );
     if (ret) printk("Error in my_timer_callback\n");
     count++;
  }
  else {
   printk( "callback called %d times\n",count);
  }
}
```

VTune Amplifier Comparison Feature

Summary

Intel® System Studio 2015 Value Proposition

Accelerate Time To Market	Strengthen System Reliability	Boost Power Efficiency and Performance	Category	Component
✓		✓	Compiler & Libraries	Intel® C++ Compiler
✓		✓		Intel® Integrated Performance Primitives
✓		✓		Intel® Math Kernel Library
✓		✓		Intel® Threading Building Blocks
✓	✓		Application Debugger	Intel-enhanced GDB* Application Debugger
✓		✓	Analyzers	Intel® VTune™ Amplifier for Systems
✓		✓		Intel® Energy Profiler
✓		✓		System Analyzer
✓		✓		Frame Analyzer
✓		✓		Platform Analyzer
✓	✓			Intel® Inspector for Systems
✓	✓		System Debugger	Intel® System Debugger (JTAG)

Next Steps – Getting Started

Download and evaluate

http://intel.ly/system-studio

Get support

User Forum:

- http://software.intel.com/en-us/forums/intel-system-studio

Intel Premier Support:

- https://premier.intel.com, Product: Intel® System Studio

Articles, User Guides and Getting Started:

- https://software.intel.com/en-us/articles/intel-system-studio-articles
- https://software.intel.com/en-us/articles/intel-system-studio-release-notes

Contact us for next generation Intel® Processor support:

- IntelSystemStudio@intel.com

Intel® System Studio 2015 Components

Category	Component	Target OS Support: Linux* [1, 5] Composer Edition	Linux* [1, 5] Professional Edition	Linux* [1, 5] Ultimate Edition	Android* [5] Composer Edition	Android* [5] Professional Edition	Android* [5] Ultimate Edition	Windows* Composer Edition	Windows* Professional Edition	VxWorks* Composer Edition
Host Operating Systems		Linux*, Windows*			Linux*, Windows*			Windows*		Linux*, Windows*
Integrated Development Environment		Eclipse*, Wind River* Workbench*			Eclipse*			Visual Studio*		Wind River* Workbench*
Compiler & Libraries	Intel® C++ Compiler	✓	✓	✓	✓	✓	✓	✓	✓	✓ [2]
	Intel® Integrated Performance Primitives	✓	✓	✓	✓	✓	✓	✓	✓	✓ [2]
	Intel® Math Kernel Library	✓	✓	✓				✓	✓	
	Intel® Threading Building Blocks	✓	✓	✓	✓	✓	✓	✓	✓	
Application Debugger	Intel-enhanced GDB* Application Debugger	✓	✓	✓	✓	✓	✓			
Analyzers	Intel® VTune™ Amplifier for Systems		✓	✓		✓	✓		✓	
	Intel® Energy Profiler		✓ [6]	✓ [6]		✓	✓		✓	
	System Analyzer					✓	✓		✓	
	Platform Analyzer [4]					✓	✓		✓	
	Frame Analyzer [4]					✓	✓		✓	
	Intel® Inspector for Systems		✓	✓					✓	
System Debugger	Intel® System Debugger (JTAG) [3]			✓			✓			

[1] Linux*, Embedded Linux, Wind River* Linux*, Yocto Project*, Tizen*
[2] Delivered with Wind River* VxWorks* platform*
[3] Via Intel® ITP-XDP3 probe, OpenOCD*, Macraigor* usb2demon* and EDKII* for UEFI*
[4] Available on Windows* host only
[5] Linux* and Android* target support available in a single product
[6] For detailed processor support please visit: https://software.intel.com/en-us/intel-energy-profiler

What Are You Developing For?

Systems and IoT

IA powered hardware, Embedded Systems, Mobile Systems, IoT

Hardware-based embedded progr.

- BIOS/UEFI/FW
- Kernel/OS
- Drivers
- Embedded Applications

Intel® System Studio

Responsiveness

Ultrabooks, OS X devices, Tablets, Phones

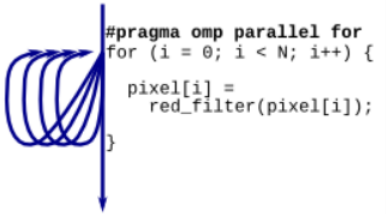
APP** Optimization

- Android
- Windows
- OS X

Intel® INDE

Scientific & Performance

HPCs, servers, clusters



Performance through Parallel Processing

- Parallel Processing
- Threading
- Message Passing

Intel® Parallel Studio XE

Multiple Devices

Ultrabooks, Tablets, Phones

Cross Device – Multiple APP stores

- Mobile Apps
- HTML5 technology
- Write - once

Intel® XDK

*** An APP is an application with a dedicated function, distributed through APP stores*

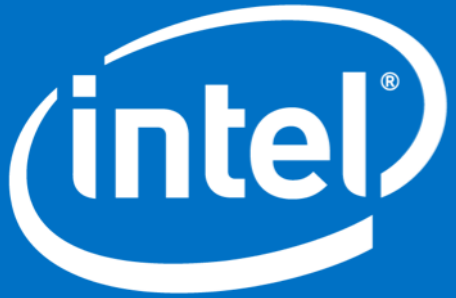
Intel® IoT Developer Kit – Product Brief

Intel® IoT Developer Kits are versatile, performance-optimized and fully integrated end-to-end IoT solutions supporting a variety of programming environments, tools, security, cloud connectivity and hardware such as Intel® Edison, Intel® Galileo and Intel® Gateway Solutions for IoT. The Intel IoT Developer Kit lowers the barriers to entry by combining a small, powerful and production ready hardware & software platform together with a comprehensive IoT Developer program and a highly engaged global developer community.

Software included in the IoT Dev Kit:

- The Yocto* Linux system
  - Provides resources for creating applications in various programming languages: C/C++, Python, Node.js and visual programming
- Integrated Development Environments (IDEs) and Tools
  - Allows developers to create, run and debug applications directly on the Galileo and Edison boards: Eclipse* (C/C++), Intel® XDK IoT Edition (Node.js), Wyliodrin* (Visual)
  - Intel® System Studio for IoT: development suite that provides deep hardware and software insights to speed development, testing and optimization
- Intel® IoT Analytics
  - Provides cloud APIs for data collection, data visualization, reports, rules engine and analytics
- Middleware libraries
  - Provides developers high level API access to Galileo and Edison boards and middleware libraries to easily control the various sensors and actuators

Optimization Notice


intel

Intel Compiler Benchmark Configuration Information

C++ benchmark configuration Info - SW Versions: Intel® C/C++ 14.0, Microsoft Visual C++ 2012 (Windows), GCC 4.8.1 (Linux); Hardware: HP ProLiant DL360 G8*, 2 x Intel® Xeon® processor E5-2670 (2.60GHz, 20480KB LLC, TurboBoost is on, HyperThreading is on), 128GB RAM, SAS; Windows Operating System: Windows 7 Enterprise, Service pack 1; Linux Operating System: Red Hat Enterprise Linux Server release 6.2 (Santiago), Kernel 2.6.32-220.el6.x86_64. ||| Compiler Options: INT Speed (Linux): Intel compiler 14.0- C: -xAVX -ipo -O3 -no-prec-div -static -parallel -opt-prefetch -auto-p32. C++: xAVX -ipo -O3 -no-prec-div -opt-prefetch -auto-p32. GCC 4.8.1- C: -m64 -Ofast -ffast-math -flto -march=native -mfpmath=sse -funroll-all-loops -static -ftree-parallelize-loops=16. C++: -m64 -Ofast -ffast-math -flto -march=native -mfpmath=sse -funroll-all-loops. ||| FP Speed (Linux): Intel Compiler 14.0- C: -xAVX -ipo -O3 -no-prec-div -static -parallel -opt-prefetch -ansi-alias. C++- xAVX -ipo -O3 -no-prec-div -static -opt-prefetch -ansi-alias. GCC 4.8.1- C:-m64 -Ofast -ffast-math -flto -march=native -mfpmath=sse -funroll-all-loops -static -ftree-parallelize-loops=16. C++: -m64 -Ofast -ffast-math -flto -march=native -mfpmath=sse -funroll-all-loops. ||| INT Speed (Windows): Intel compiler 14.0- C: -Qvc11 -Qstd=c99 -QxAVX -Qipo -O3 -Qprec-div- -Qopt-prefetch -Qparallel -Qauto-ilp32. C++: -Qvc11 -QxAVX -Qipo -O3 -Qprec-div- -Qopt-prefetch -Qcxx_features -Qauto-ilp32. Visual C++* 2012- C: /O2 /Ob2 /GL /arch:AVX /favor:EM64T /fp:fast /Qpar. C++: /O2 /Ob2 /GL /arch:AVX /favor:EM64T /fp:fast /Qpar -EHsc -GR. ||| FP Speed (Windows): Intel compiler 14.0- C: -Qvc11 -Qstd=c99 -QxAVX -Qipo -O3 -Qprec-div- -Qopt-prefetch -Qansi-alias -Qparallel. C++: -Qvc11 -QxAVX -Qipo -O3 -Qprec-div- -Qopt-prefetch -Qcxx_features -Qansi-alias. Fortran: -QxAVX -Qipo -O3 -Qprec-div- -Qopt-prefetch -Qparallel.

Fortran benchmark configuration Info - Compiler Versions: Intel® Fortran 14.0, PGI 13.6;, Absoft 13.0.3, gFortran 4.8.1 ; Hardware: Blue Hills ATX Media IVB Desktop DZ77BH-55K-IDD; Intel® Core™ i7-3770K CPU @ 3.50GHz, TurboBoost is on, HyperThreading is off, 16GB RAM; Windows Operating System: Windows 7 Enterprise, Service pack 1; Linux Operating System: Red Hat Enterprise Linux Server release 6.3 (Santiago); Kernel 2.6.32-279.el6.x86_64 ; Compiler Options (Windows and Linux): Intel Fortran compiler 14.0: ifort -O3 -fast -parallel -ipo -no-prec-div , PGI 13.6: pgf95 -fastsse -Munroll=n:4 -Mipa=fast,inline -Mconcur=bind . Polyhedron benchmark (www.polyhedron.com) performed by Intel Corp on August 20, 2013.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

The cost reduction scenarios described in this document are intended to enable you to get a better understanding of how the purchase of a given Intel product, combined with a number of situation-specific variables, might affect your future cost and savings. Nothing in this document should be interpreted as either a promise of or contract for a given level of costs.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804