GET ONBOARD WITH THE INTEL® PARALLEL STUDIO XE 2018 BETA
## Intel® Parallel Studio XE

**Create Faster Code... Faster**

### Composer Edition

**Build**
- Compilers & Libraries
  - C / C++ Compiler
    - Optimizing Compiler
  - Fortran Compiler
    - Optimizing Compiler
  - Intel® TBB
    - C++ Threading Library
  - Intel® Distribution for Python*
    - High Performance Scripting
  - Intel® MKL
    - Fast Math Kernel Library
  - Intel® IPP
    - Image, Signal & Data Processing
  - Intel® DAAL
    - Data Analytics Library

### Professional Edition

**Analyze**
- Analysis Tools
  - Intel® VTune™ Amplifier
    - Performance Profiler
  - Intel® Inspector
    - Memory & Thread Debugger
  - Intel® Advisor
    - Vectorization Optimization & Thread Prototyping

### Cluster Edition

**Scale**
- Cluster Tools
  - Intel® MPI Library
    - Message Passing Interface Library
  - Intel® Trace Analyzer & Collector
    - MPI Tuning & Analysis
  - Intel® Cluster Checker
    - Cluster Diagnostic Expert System

---

**Intel® Architecture Platforms**

Operating System: Windows*, Linux*, MacOS®


© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. *Other names and brands may be claimed as the property of others. For more complete information about compiler optimizations, see our Optimization Notice.
CREATE FASTER HPC AND CLOUD SOFTWARE
WHAT'S NEW IN INTEL® PARALLEL STUDIO XE 2018 BETA

Modernize Code for Performance, Portability and Scalability on the Latest Intel® Platforms

- Use fast Intel® AVX-512 instructions on Intel® Xeon® and Xeon Phi™ processors.
- Parallelize and vectorize C++ STL easily using Parallel STL*.
- Intel® Advisor - Roofline finds high impact, but under optimized loops
- Intel® Distribution for Python* - Faster Python* applications
- Stay up-to-date with the latest standards and IDE:
  - C++2017 draft parallelizes and vectorizes C++ easily using Parallel STL*
  - Full Fortran* 2008, Fortran 2015 draft
  - OpenMP* 5.0 draft, Microsoft Visual Studio* 2017
- Support for Intel® Omni-Path Architecture

Flexibility for Your Needs

- Application Snapshot - Quick answers: Does my hybrid code need optimization?
- Intel® VTune™ Amplifier – Profile private clouds with Docker* and Mesos* containers, Java* daemons

And much more*...


* See Release Notes for the full list with further updates and new features.
Intel® Compilers: C/C++ & Fortran*
### INTEL® COMPILERS

<table>
<thead>
<tr>
<th>As a software developer, I care about:</th>
<th>...and my challenges are:</th>
<th>Intel compilers offer:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong> – I develop applications that need to execute FAST</td>
<td>Taking advantage of the latest hardware innovations</td>
<td>Developers the full power of the latest x86-compatible processors and instruction sets</td>
</tr>
<tr>
<td><strong>Productivity</strong> – I need productivity and ease of use offered by compilers</td>
<td>Finding support for the leading languages and programming models</td>
<td>Support for the latest Fortran, C/C++, and OpenMP* standards; compatibility with leading compilers and IDEs</td>
</tr>
<tr>
<td><strong>Scalability</strong> – I develop and debug my application locally, and deploy my application globally</td>
<td>Maintaining my code as core counts and vector widths increase at a fast pace</td>
<td>Scalable performance without changing code as newer generation processors are introduced</td>
</tr>
</tbody>
</table>
Optimized for Latest HW, Language Standards
Intel® C, C++, and Fortran* Compilers 18.0 Beta

Common Updates

- Optimizations for latest Intel® Xeon® and Intel® Xeon Phi™ processors
- Initial OpenMP* 5.0 draft support
- Microsoft* Visual Studio* 2017 integration
- Lightweight hardware-based profile-guided optimization

C/C++

- Full C11 and C++14, initial C++ 17
- Parallel STL* for parallel and vector execution of the C++ STL

Fortran*

- Full Fortran* 2008, initial Fortran 2015
- Visual Studio* 2015 shell
Intel® Distribution for Python* – High-Performance Python
FASTER: NUMPY* AND SCIPY*
INTEL® DISTRIBUTION FOR PYTHON* 2018 BETA

Latest Beta Libraries
- Intel® Math Kernel Library (Intel® MKL)
- Intel® Data Analytics Acceleration Library (Intel® DAAL)
- Intel® MPI Library
- Intel® Threading Building Blocks (Intel® TBB)
- Intel® Integrated Performance Primitives (Intel® IPP)

NumPy/SciPy Acceleration
- Accelerated arithmetic & transcendental functions (NumPy*)
- Faster FFT (NumPy & SciPy*)
- Memory management optimization for NumPy

OpenCV* Supported
- Included with Distribution
- Accelerated with Intel® IPP
Intel® Math Kernel Library – Fast Math Library
Intel® Threading Building Blocks - C++ Threading Library
Intel® Integrated Performance Primitives - Image, Signal, and Data Processing
# New and Enhanced Math Functions

## Intel® Math Kernel Library 2018 Beta

<table>
<thead>
<tr>
<th>BLAS</th>
<th>BLAS Batch API</th>
<th>LAPACK</th>
<th>New Vector Math Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>New integer multiplication routines</td>
<td>BLAS group &amp; batch: Efficiency &amp; performance</td>
<td>Aasen’s Algorithm</td>
<td>24 New Algorithms</td>
</tr>
<tr>
<td>Improved SGEMM optimization</td>
<td>New - Batched triangular solve matrix</td>
<td>Factorization &amp; solver routines</td>
<td>Richer choice of optimized routines</td>
</tr>
<tr>
<td>Convolution &amp; inner product optimizations</td>
<td>Faster GEMM_BATCH</td>
<td>Improved performance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Good for medium-to-small matrices</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

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

For more complete information about compiler optimizations, see our Optimization Notice.
**Generic Parallel Algorithms**

- Efficient scalable way to exploit the power of multi-core without having to start from scratch.

**Flow Graph**

- A set of classes to express parallelism as a graph of compute dependencies and/or data flow

**Concurrent Containers**

- Concurrent access, and a scalable alternative to containers that are externally locked for thread-safety

**Synchronization Primitives**

- Atomic operations, a variety of mutexes with different properties, condition variables

**Task Scheduler**

- Sophisticated work scheduling engine that empowers parallel algorithms and the flow graph

**Timers and Exceptions**

- Thread-safe timers and exception classes

**Threads**

- OS API wrappers

**Thread Local Storage**

- Efficient implementation for unlimited number of thread-local variables

**Memory Allocation**

- Scalable memory manager and false-sharing free allocators

---

*Other names and brands may be claimed as the property of others. For more complete information about compiler optimizations, see our Optimization Notice.*

© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
Parallel Standard Template Library

- Latest draft standard C++ 2017 support
- Method to parallelize & vectorize (w/compiler) with minimal code change
- Can specify execution policies
- Policies are customizable for different targets
- Fully utilize multithreading capability of Intel® TBB
- Combined with Intel® C/C++ Compiler to enable vectorization

```markdown
<table>
<thead>
<tr>
<th>Exec Policy</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>Sequential</td>
</tr>
<tr>
<td>par</td>
<td>Parallel</td>
</tr>
<tr>
<td>par_vec</td>
<td>Vectorized Parallel</td>
</tr>
<tr>
<td>my_policy_target</td>
<td>Some target HW</td>
</tr>
</tbody>
</table>
```

Example:

```cpp
for_each(exec policy, begin, end, function)
```
SUPPORT FOR LATEST HW, FASTER COMPRESSION, & MORE
INTEL® INTEGRATED PERFORMANCE PRIMITIVES 2018 BETA

**Latest Hardware**
- Support for Intel® Xeon Phi™ processor

**Cryptography**
- No dependence on main Intel® IPP package nor version

**Compression**
- Optimization of LZO (lossless data compression algorithm) with SSE4.2 & AVX2
- Increased performance
ANALYSIS TOOLS

Intel® VTune™ Amplifier - Performance Profiler
Application Performance Snapshot – High-level Application Characterization
Intel® Advisor - Vectorization Optimization and Thread Prototyping
Intel® Inspector - Memory and Thread Debugger
INTEL® VTUNE™ AMPLIFIER PERFORMANCE PROFILER

- Accurate Data - Low Overhead
  - CPU, GPU, FPU, threading, bandwidth...
- Meaningful Analysis
  - Threading, OpenMP region efficiency
  - Memory access, storage device
- Easy
  - Data displayed on the source code
  - Easy set-up, no special compiles

“Last week, Intel® VTune™ Amplifier helped us find almost 3X performance improvement. This week it helped us improve the performance another 3X.”

Claire Cates
Principal Developer
SAS Institute Inc.

http://intel.ly/vtune-amplifier-xe
**BETTER: MPI, SNAPSHOT, CLOUD, PYTHON*, AND MORE**

**INTEL® VTUNE™ AMPLIFIER 2018 BETA**

<table>
<thead>
<tr>
<th>Latest Hardware</th>
<th>Better Metrics Easier Tuning</th>
<th>Container Support</th>
<th>Enhanced Profiling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for Intel® Xeon Phi™ processor (codenamed Knights Landing)</td>
<td>Better Application Snapshot: Merges MPI + Application data</td>
<td>Profile inside Docker* &amp; Mesos* containers</td>
<td>Locks &amp; waits analysis for mixed Python* and native code</td>
</tr>
<tr>
<td></td>
<td>Enhanced metrics for MPI applications: Imbalance &amp; critical path rank</td>
<td>Attach to Running Java* services &amp; daemons</td>
<td>Memory consumption analysis for Python, C/C++</td>
</tr>
</tbody>
</table>

© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

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

For more complete information about compiler optimizations, see our [Optimization Notice](#).
**OPTIMIZE PRIVATE CLOUD-BASED APPLICATIONS**

**PROFILE NATIVE & JAVA* APPS IN CONTAINERS WITH INTEL® VTUNE™ AMPLIFIER BETA**

**Profile Enterprise Applications**
- Native C, C++, Fortran*
- Attach to running Java* services (e.g., Mail)
- Profile Java daemons without restart

**Accurate, Low-Overhead Data Collection**
- Advanced hotspots and hardware events
- Memory analysis
- Accurate stack information for Java and HHVM*

**Popular Containers Supported**
- Docker*
- Mesos*

Software collectors (e.g., locks & waits) and Python* profiling are not currently available for containers.

---

© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others. For more complete information about compiler optimizations, see our [Optimization Notice](#).
APPLICATION PERFORMANCE SNAPSHOT
EASY PERFORMANCE SNAPSHOT FOR THREADED MPI APPLICATIONS

Quick & easy performance overview
- Does the app need performance tuning?

MPI and non-MPI Apps
- Distributed MPI with or without threading
- Shared memory applications

Popular MPI implementations supported
- Intel® MPI
- MPICH and Cray MPI

Richer metrics on computation efficiency
- CPU (processor stalls, memory access)
- FPU (vectorization metrics)

*Linux only
Faster Vectorization Optimization:
- Vectorize where it will pay off most
- Quickly ID what is blocking vectorization
- Tips for effective vectorization
- Safely force compiler vectorization
- Optimize memory stride

Breakthrough for Threading Design:
- Quickly prototype multiple options
- Project scaling on larger systems
- Find synchronization errors before implementing threading
- Design without disrupting development

Add Parallelism with Less Effort, Less Risk and More Impact

http://intel.ly/advisor-xe
NEW! ROOFLINE, FASTER ANALYSIS & MORE...
INTEL® ADVISOR – VECTORIZATION OPTIMIZATION

Roofline analysis helps you optimize effectively
- Find high impact, but under optimized loops
- Does it need cache or vectorization optimization?
- Is a more numerically intensive algorithm a better choice?

Faster data collection
- Filter by module - Calculate only what is needed.
- Track refinement analysis – Stop when every site has executed

Make better decisions with more data/recommendations
- Intel MKL friendly – Is the code optimized? Is the best variant used?
- Function call counts in addition to trip counts
- Top 5 recommendations added to summary
- Dynamic instruction mix – Expert feature shows exact count of each instruction

Faster data collection
- MPI support in the command line dialog
**INTEL® INSPECTOR**

- Find and eliminate errors
  - Memory leaks, invalid access...
  - Races & deadlocks
  - C, C++ and Fortran (or a mix)
- Simple, Reliable, Accurate
  - No special recompiles
    - Use any build, any compiler
  - Analyzes dynamically generated or linked code
  - Inspects 3rd party libraries without source
  - Productive user interface + debugger integration
  - Command line for automated regression analysis
FEWER FALSE POSITIVES, LATEST OSS
INTEL® INSPECTOR 2018 BETA MEMORY AND THREAD DEBUGGER

**Fewer False Positives**
- Previous lock/unlock model expanded to unlocked, lock for read, lock for write
- C++ 17 std::shared_mutex added
- Windows SRW Locks added

**Latest OSs**
- Ubuntu* 16.10
- Fedora* 25

Find difficult to diagnose non-deterministic threading errors like race conditions

Example race condition:

<table>
<thead>
<tr>
<th>Correct Answer</th>
<th>Incorrect Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Thread 1</strong></td>
<td><strong>Thread 2</strong></td>
</tr>
<tr>
<td>Read count</td>
<td>✈️ 0</td>
</tr>
<tr>
<td>Increment</td>
<td>✈️ 0</td>
</tr>
<tr>
<td>Write count</td>
<td>✈️ 1</td>
</tr>
<tr>
<td>Read count</td>
<td>✈️ 1</td>
</tr>
<tr>
<td>Increment</td>
<td>1</td>
</tr>
<tr>
<td>Write count</td>
<td>✈️ 2</td>
</tr>
</tbody>
</table>

Example race condition:

<table>
<thead>
<tr>
<th>Read count</th>
<th>✈️ 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increment</td>
<td>✈️ 0</td>
</tr>
<tr>
<td>Write count</td>
<td>✈️ 1</td>
</tr>
</tbody>
</table>
CLUSTER TOOLS

Intel® MPI Library - Message Passing Interface Library
Intel® Trace Analyzer and Collector - MPI Tuning and Analysis
Intel® Cluster Checker - Cluster Diagnostic Expert System
INTEL® MPI AND INTEL® TRACE ANALYZER AND COLLECTOR

What

Intel's high-performance MPI communications profiler and analyzer for scalable HPC development

Why

- **Scale performance**: Perform on more nodes
- **Scale forward**: Multi-core and many-core ready
- **Scale efficiently**: Tune and debug on more nodes

How

- **Visualize**: Understand parallel application behavior
- **Evaluate**: Profiling statistics and load balancing
- **Analyze**: Automated analysis of common MPI issues
- **Identify**: Communication hotspots
**Better Performance and Support for Intel® Platforms**

**Intel® MPI Library and Intel® Trace Analyzer and Collector 2018 Beta**

<table>
<thead>
<tr>
<th>Latest Hardware</th>
<th>Faster Time to Results and Onto the Next Job</th>
<th>Expanded Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Latest Intel® Xeon® processor (codenamed Skylake)</td>
<td>- MPI start-up time</td>
<td>- OpenSHMEM* support in Intel® Trace Analyzer and Collector</td>
</tr>
<tr>
<td>- Latest Intel® Xeon Phi™ processors and coprocessors (codenamed Knights Landing)</td>
<td>- Finalization time for OFI/TMI</td>
<td></td>
</tr>
<tr>
<td>- Latest Intel® Omni-Path Architecture fabric</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Key Concepts:**

- **Symptoms** are subjective indications of health
- **Signs** are objective indications of health detected by direct observation
- **Diagnoses** are the identification of the root cause of an issue
- **Remedies** are methods to resolve an issue

<table>
<thead>
<tr>
<th>Concept</th>
<th>Human</th>
<th>Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symptom</td>
<td>Difficulty walking; ankle hurts</td>
<td>Job is running slow</td>
</tr>
<tr>
<td>Signs</td>
<td>- Range of ankle motion limited to 50% of normal&lt;br&gt;- Ankle severely inflamed compared to non-injured leg&lt;br&gt;- X-ray negative for fracture</td>
<td>- DGEMM performance is 50% of peak&lt;br&gt;- Zombie process is using 100% of the processor</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Sprained ankle</td>
<td>Zombie process is stealing cycles</td>
</tr>
<tr>
<td>Remedy</td>
<td>Ice ankle &amp; keep it elevated, take 500 mg of ibuprofen every 4-6 hours</td>
<td>Kill the zombie process</td>
</tr>
</tbody>
</table>
ENSURE HPC CLUSTER COMPONENTS WORK TOGETHER

**INTEL® CLUSTER CHECKER 2018 BETA**

**Latest Hardware and Middleware**
- Latest Intel® Xeon® processor (codenamed Skylake)
- Latest Intel® Xeon Phi™ processors and coprocessors (codenamed Knights Landing)
- Latest Intel® Omni-Path Architecture fabric
- Intel® Enterprise Edition for Lustre*

**Productivity Improvements**
- Simplified grouping of checks for extensibility
- Improved diagnostic output
- Support of multiple database sources for checks

**Integration and Frameworks**
- Validation of Intel® Scalable System Framework
- Validation of Intel® HPC Orchestrator integration

*Other names and brands may be claimed as the property of others.
# Intel® Parallel Studio XE

**CREATE FASTER CODE...FASTER**

<table>
<thead>
<tr>
<th>COMPOSER EDITION</th>
<th>PROFESSIONAL EDITION</th>
<th>CLUSTER EDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BUILD</strong></td>
<td><strong>ANALYZE</strong></td>
<td><strong>SCALE</strong></td>
</tr>
<tr>
<td>Compilers &amp; Libraries</td>
<td>Analysis Tools</td>
<td>Cluster Tools</td>
</tr>
<tr>
<td>C / C++ Compiler</td>
<td>Intel® VTune™ Amplifier</td>
<td>Intel® MPI Library</td>
</tr>
<tr>
<td>Optimizing Compiler</td>
<td>Performance Profiler</td>
<td>Message Passing Interface Library</td>
</tr>
<tr>
<td>Fortran Compiler</td>
<td>Intel® Inspector</td>
<td>Intel® Trace Analyzer &amp; Collector</td>
</tr>
<tr>
<td>Optimizing Compiler</td>
<td>Memory &amp; Thread Debugger</td>
<td>MPI Tuning &amp; Analysis</td>
</tr>
<tr>
<td>Intel® TBB</td>
<td>Intel® Advisor</td>
<td>Intel® Cluster Checker</td>
</tr>
<tr>
<td>C++ Threading Library</td>
<td>Vectorization Optimization</td>
<td>Cluster Diagnostic Expert System</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intel® Distribution for Python*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>High Performance Scripting</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Intel® Architecture Platforms**

Operating System: Windows*, Linux*, MacOS†*

---

**More Power for Your Code** - software.intel.com/intel-parallel-studio-xe

© 2017 Intel Corporation. All rights reserved. Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

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

For more complete information about compiler optimizations, see our Optimization Notice.
Legal Disclaimer and 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.

- Copyright © 2017, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

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
FASTER CODE.... FASTER

JOIN INTEL® PARALLEL STUDIO XE 2018 BETA

REGISTER AT intely/intel-parallel-studio-xe-2018-beta