Optimizing the future Java through collaboration

Vladimir Kozlov
Sandhya Viswanathan

Speakers

- Vladimir Kozlov, Oracle

  vladimir.kozlov@oracle.com
- Sandhya Viswanathan, Intel

  sandhya.viswanathan@intel.com
- You

  We encourage questions and discussions

Legal Disclaimers

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.
  A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.
- Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.
- The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.
- Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm
- Intel, the Intel logo, Intel Xeon, and Xeon logos are trademarks of Intel Corporation in the U.S. and/or other countries.
- Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families: Go to: Learn About Intel® Processor Numbers http://www.intel.com/products/processor_number

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

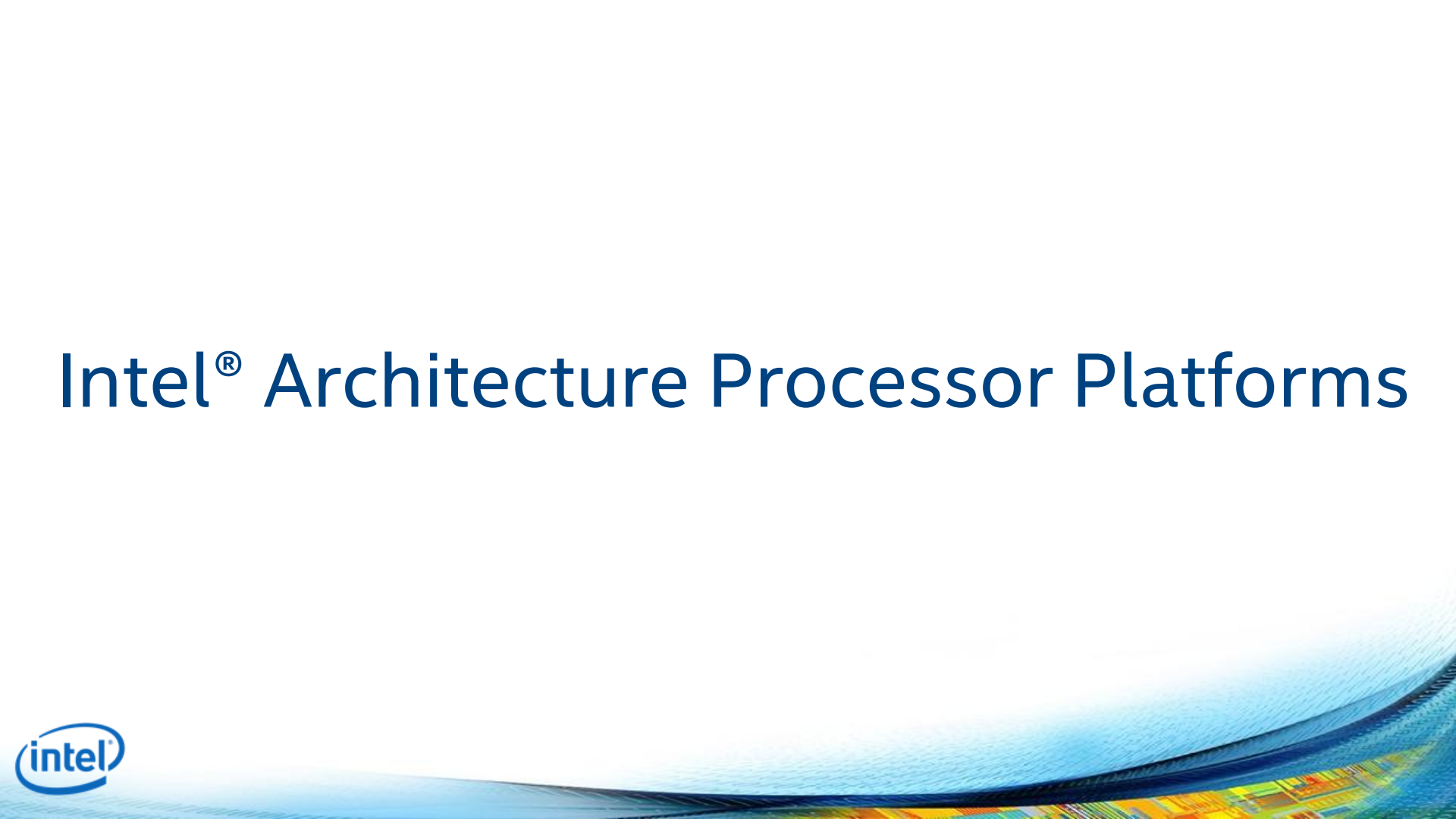
Legal Disclaimers – Continued

- Some results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.
- 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.
- Intel does not control or audit the design or implementation of third party benchmarks or Web sites referenced in this document. 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.
- Relative performance is calculated by assigning a baseline value of 1.0 to one benchmark result, and then dividing the actual benchmark result for the baseline platform into each of the specific benchmark results of each of the other platforms, and assigning them a relative performance number that correlates with the performance improvements reported.
- SPEC, SPECint, SPECfp, SPECrate, SPECpower, SPECjbb, SPECompG, SPEC MPI, and SPECjEnterprise* are trademarks of the Standard Performance Evaluation Corporation. See http://www.spec.org for more information.
- TPC Benchmark, TPC-C, TPC-H, and TPC-E are trademarks of the Transaction Processing Council. See http://www.tpc.org for more information.
- Intel® Advanced Vector Extensions (Intel® AVX)* are designed to achieve higher throughput to certain integer and floating point operations. Due to varying processor power characteristics, utilizing AVX instructions may cause a) some parts to operate at less than the rated frequency and b) some parts with Intel® Turbo Boost Technology 2.0 to not achieve any or maximum turbo frequencies. Performance varies depending on hardware, software, and system configuration and you should consult your system manufacturer for more information.
- Intel® Advanced Vector Extensions refers to Intel® AVX, Intel® AVX2 or Intel® AVX-512. For more information on Intel® Turbo Boost Technology 2.0, visit http://www.intel.com/go/turbo

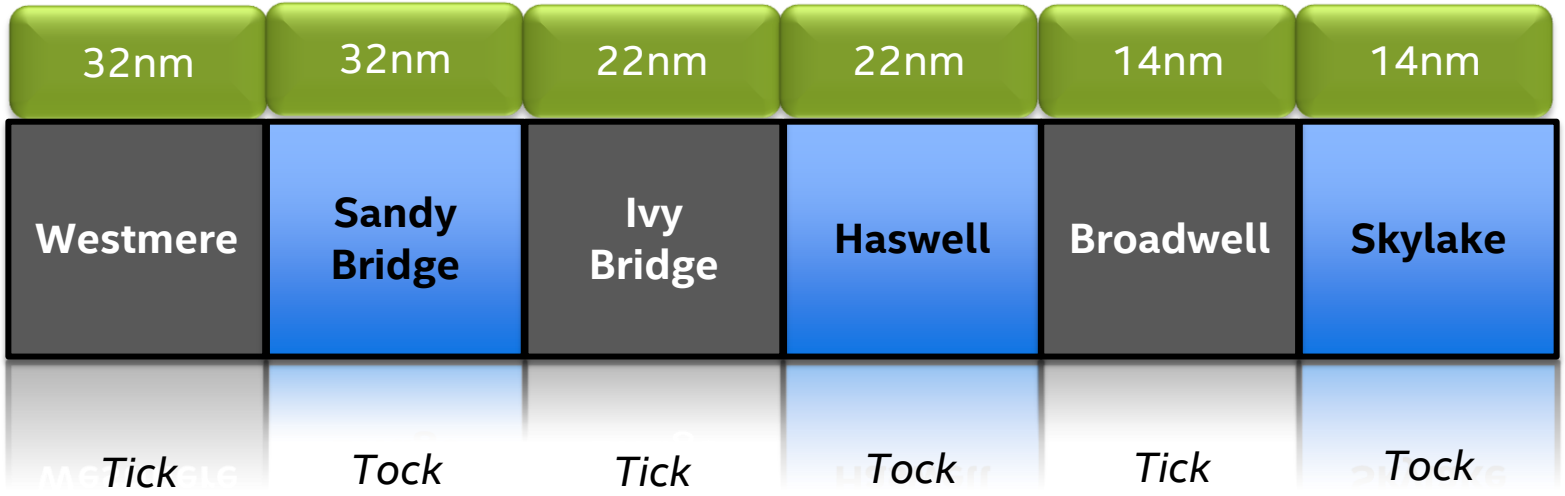
Agenda

- Intel® Architecture Processor Platforms
- Processor Features and JVM Optimizations
- JVM Optimizations Timeline and Plans
- Java Performance

Intel® Architecture Processor Platforms

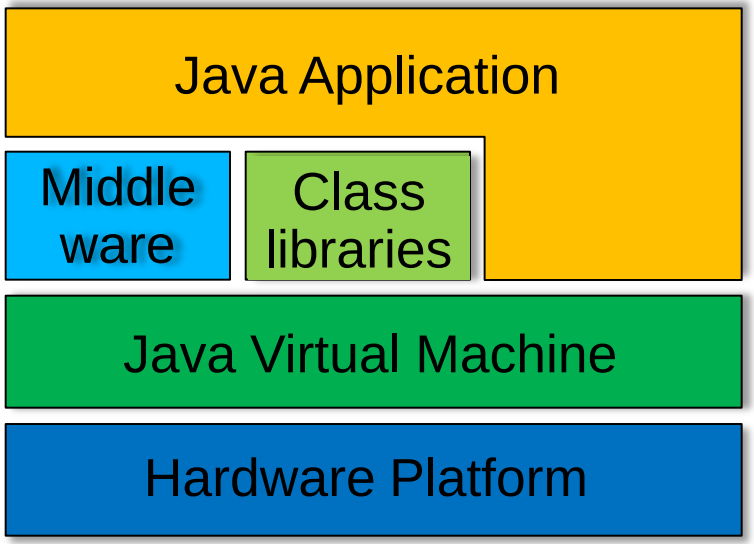


Tick/Tock Development Model



Processor Features and JVM Optimizations

Performance Optimization Overview



- Java optimized for latest IA features
  - JVM/JIT optimizations
  - JRE optimizations
- Java provides a runtime layer
  - Just upgrade to latest JVM
  - And get the benefit of optimizations

Optimization Areas

- Performance (AVX/AVX2)
- Scalability (TSX)
- Security (Crypto acceleration)
- IO (CRC acceleration)

Processor Features and JVM Optimizations
AVX/AVX2 SIMD Instructions

AVX/AVX2 Platform Feature

- AVX in SandyBridge processors
  - 256-bit wide floating point SIMD
  - 256-bit YMM registers
  - 8 single precision or 4 double precision FP operations
  - Three-operand instructions

    dest = src1 + src2

    instead of

    dest = dest + src
- AVX2 in Haswell processors
  - 256-bit wide integer SIMD
  - Bit Manipulation Instructions

JVM SIMD Optimizations

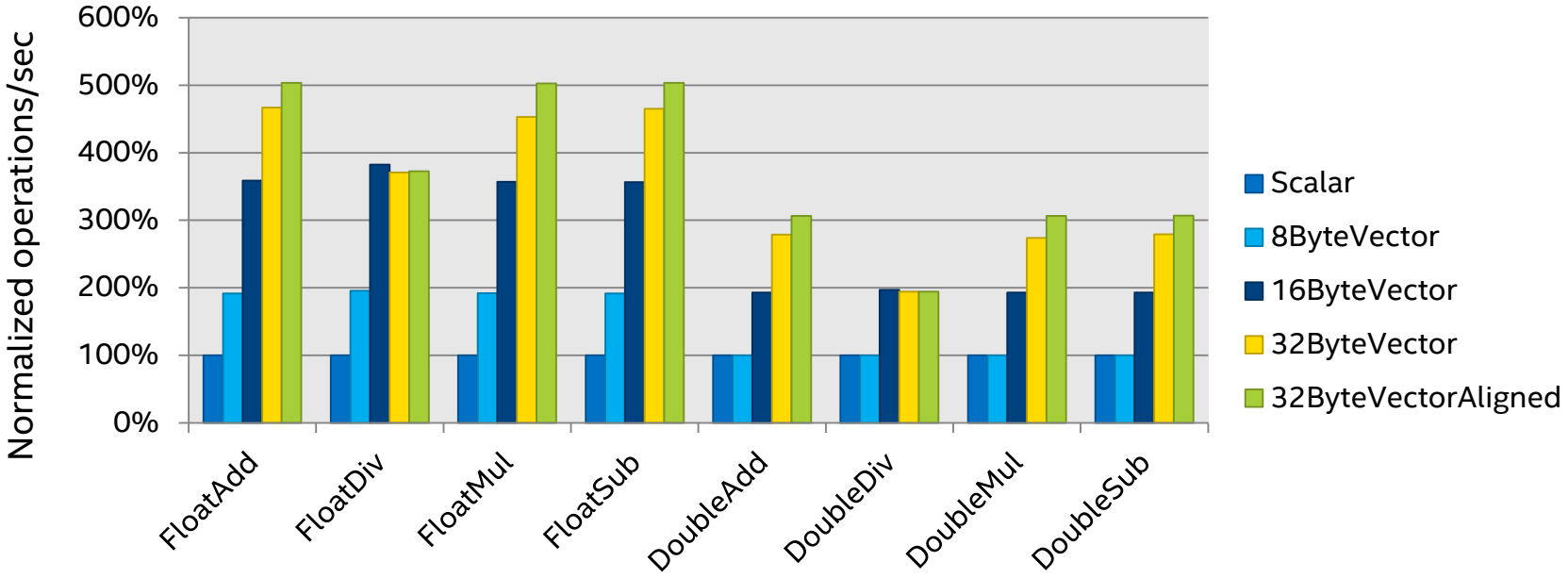
- JVM/JIT optimizations for AVX/AVX2
  - SIMD code generation improvements
  - Three operand instructions generation
  - Intrinsics and stubs

JIT SIMD code generation

- JIT Compiler supports super-word vectorization framework

  "Exploiting Superword Level Parallelism with Multimedia Instruction Sets"
  by Samuel Larsen and Saman Amarasinghe
- Vectorization extended to 128/256 bit for SSE/AVX/AVX2
- Support for vector float & integer arithmetic operations
- Support for vector logical (and, or, xor) operations
- Destination alignment via loop peeling

Java SIMD Floating Point Performance



- Size of arrays: 1024 elements

Java SIMD Floating Point Performance

Scimark LU

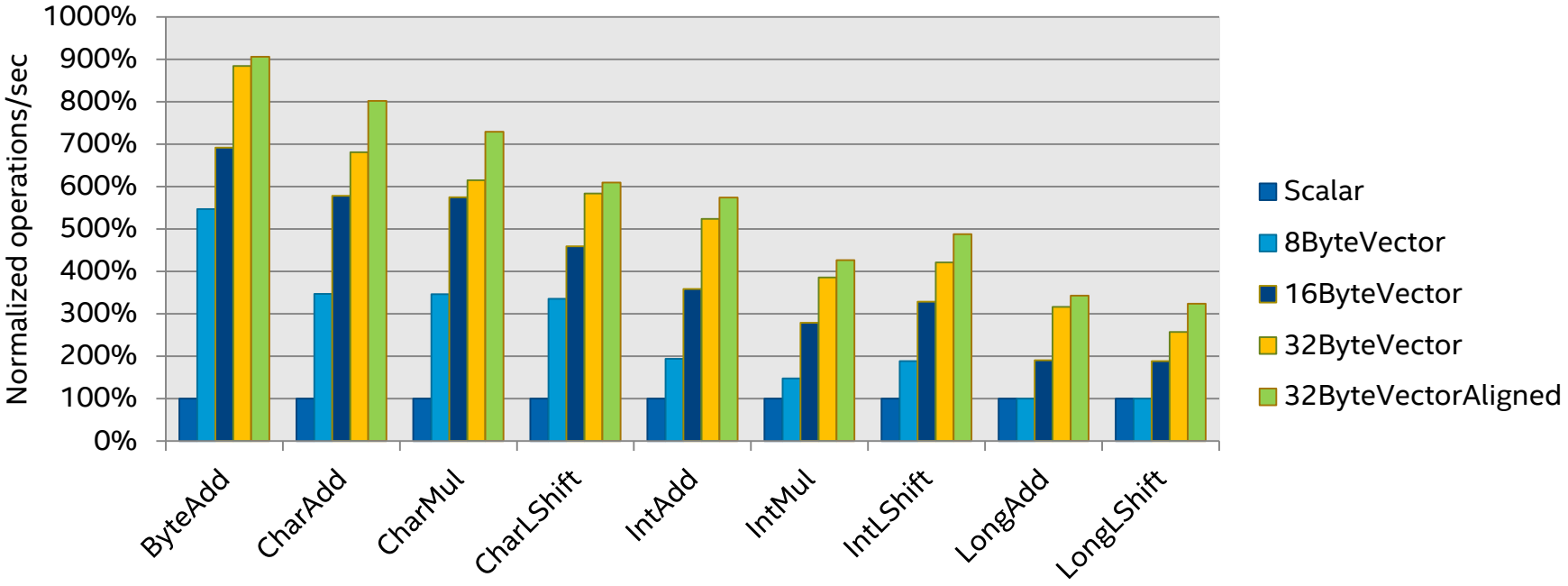
Snippet from spec.benchmarks.scimark.lu.LU.factor()

```
for (int ii=j+1; ii<M; ii++) {
    double Aii[] = A[ii];
    double Aj[] = A[j];
    double AiiJ = Aii[j];
    for (int jj=j+1; jj<N; jj++)
        Aii[jj] -= AiiJ * Aj[jj];
}
```

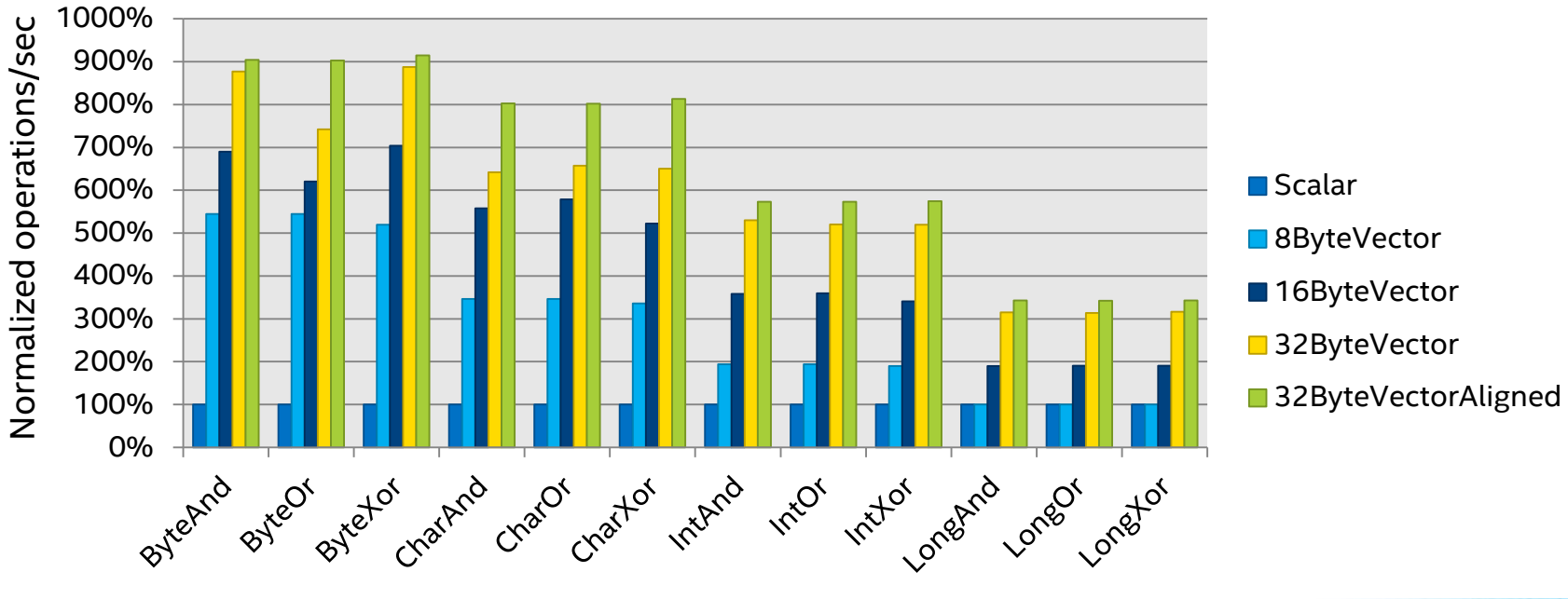
- ~40% performance gain
  - Vectorization of scimark.lu.LU.factor()
  - Three operand instructions generation

opcode	operands
Loop:	
Vmovdqu	ymm8,ymmword ptr [rcx+rbp*8+10h]
vmovdqu	ymm9,ymmword ptr [rax+rbp*8+10h]
vmulpd	ymm7,ymm8,ymm2
vsubpd	ymm7,ymm9,ymm7
vmovdqu	ymmword ptr [rax+rbp*8+10h],ymm7
movsxd	rcx,ebp
vmovdqu	ymm8,ymmword ptr [rax+rcx*8+30h]
vmovdqu	ymm9,ymmword ptr [rcx+rcx*8+30h]
vmulpd	ymm7,ymm9,ymm2
vsubpd	ymm7,ymm8,ymm7
vmovdqu	ymmword ptr [rax+rcx*8+30h],ymm7
add	ebp,8
cmp	ebp,edi
jl	*Loop*

Java SIMD Integer Performance
Arithmetic Operations

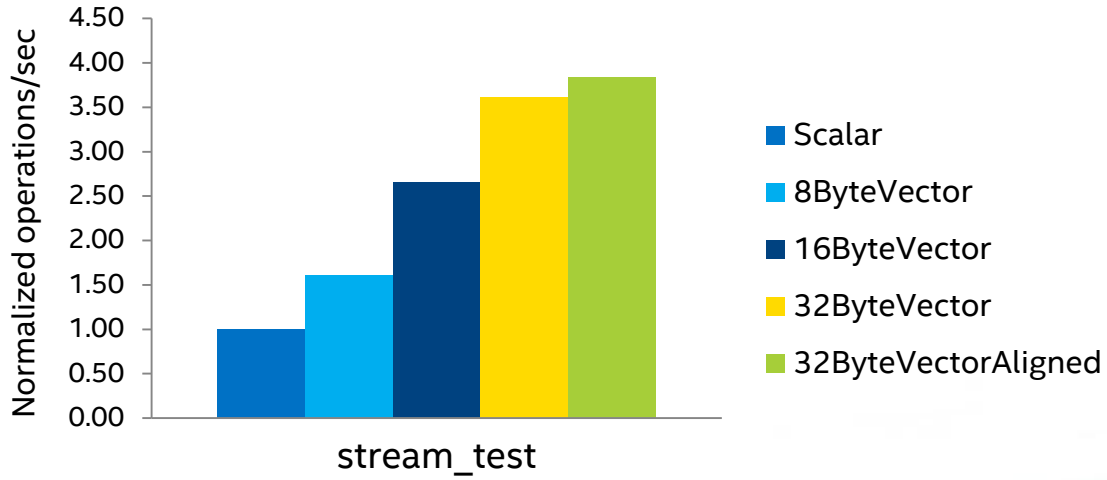


Java SIMD Integer Performance
Logical Operations



SIMD Examples: Java Streams

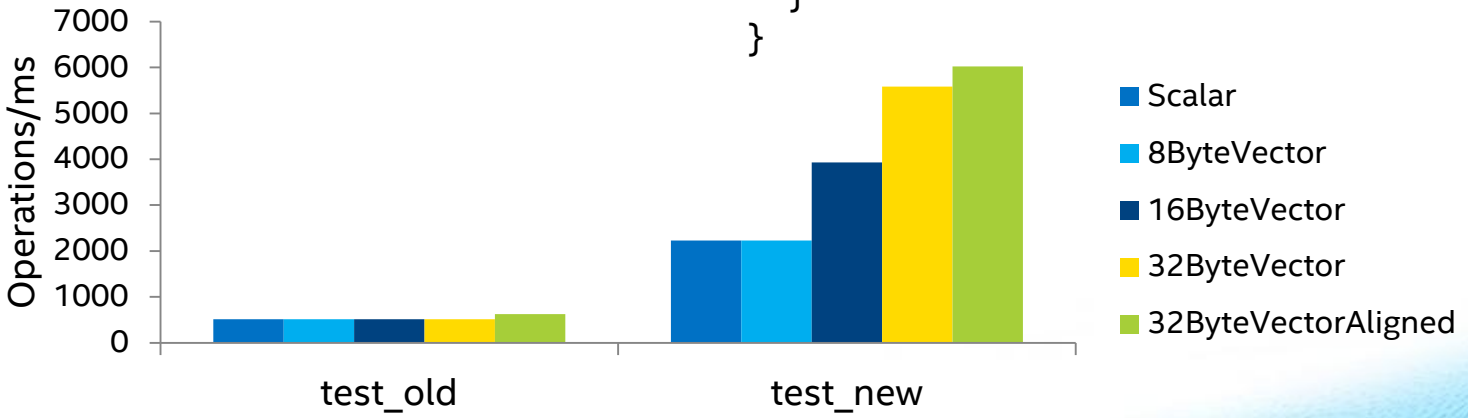
```
private static void stream_test(int[] out, int[] in1, int[] in2, int length)
{
  IntStream.range(0, length).forEach (i -> { out[i] = (in1[i] - in2[i]) * (in1[i] - in2[i]); });
}
```



SIMD Examples: 2 D Array

```
static void test_old(double[][] a,
                     double[][] b,
                     double[][] c) {
  for (int i=0; i<MMAX;i++) {
    for (int j=0;j<NMAX;j++) {
       a[j][i]= b[j][i] + c[j][i];
    }
  }
}
```

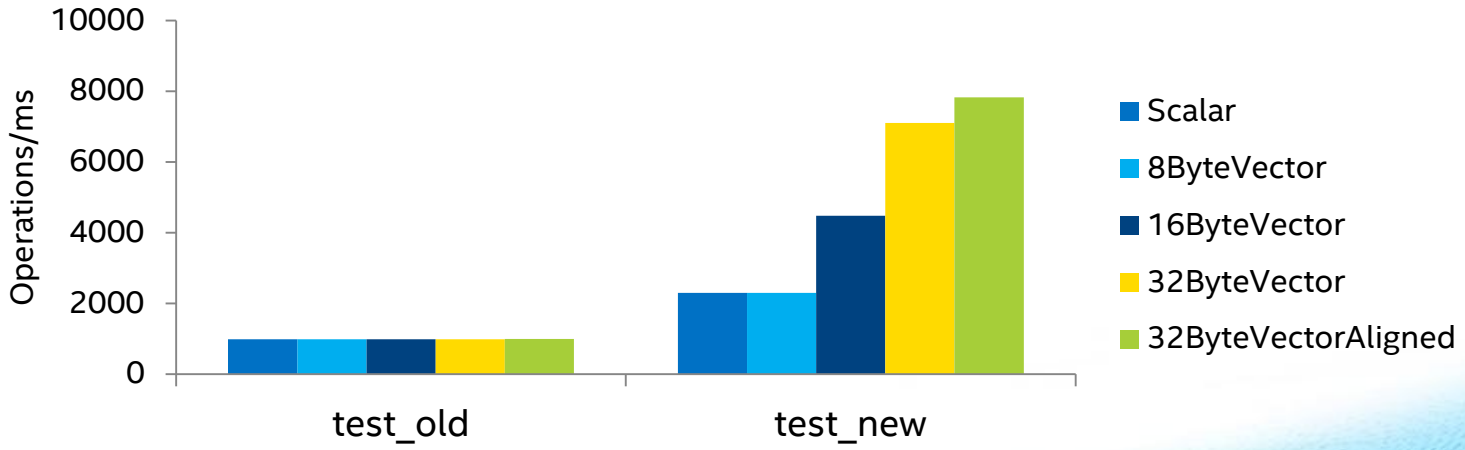
```
static void test_new(double[][] a,
                     double[][] b,
                     double[][] c) {
  for (int i=0; i<NMAX;i++) {
    for (int j=0;j<MMAX;j++) {
       a[i][j]= b[i][j] + c[i][j];
    }
  }
}
```



SIMD Examples: Conditional Statements

```
static void test_old(double[] a,
                     double[] b,
                     double[] c) {
  for (int i=0;i<MMAX;i++) {
    if (i == 100)
       a[i] = 1.0;
    else
       a[i]= b[i] + c[i];
  }
}
```

```
static void test_new(double[] a,
                     double[] b,
                     double[] c) {
  for (int i=0;i<MMAX;i++) {
     a[i]= b[i] + c[i];
  }
  a[100] = 1.0;
}
```



JVM SIMD Optimizations: String and Array Intrinsics

- Array and String JRE intrinsic methods & stubs extended to use AVX/AVX2
  - System.arraycopy
  - Arrays.fill, Arrays.equals
  - String.compareTo, String.equals
  - String.indexof
- Performance on micro benchmark with 1024 elements
  - Up to 50% improvement for aligned data
  - Further improvements seen for String.compareTo due to algorithm improvement

JVM SIMD Optimizations: ISO Encoding

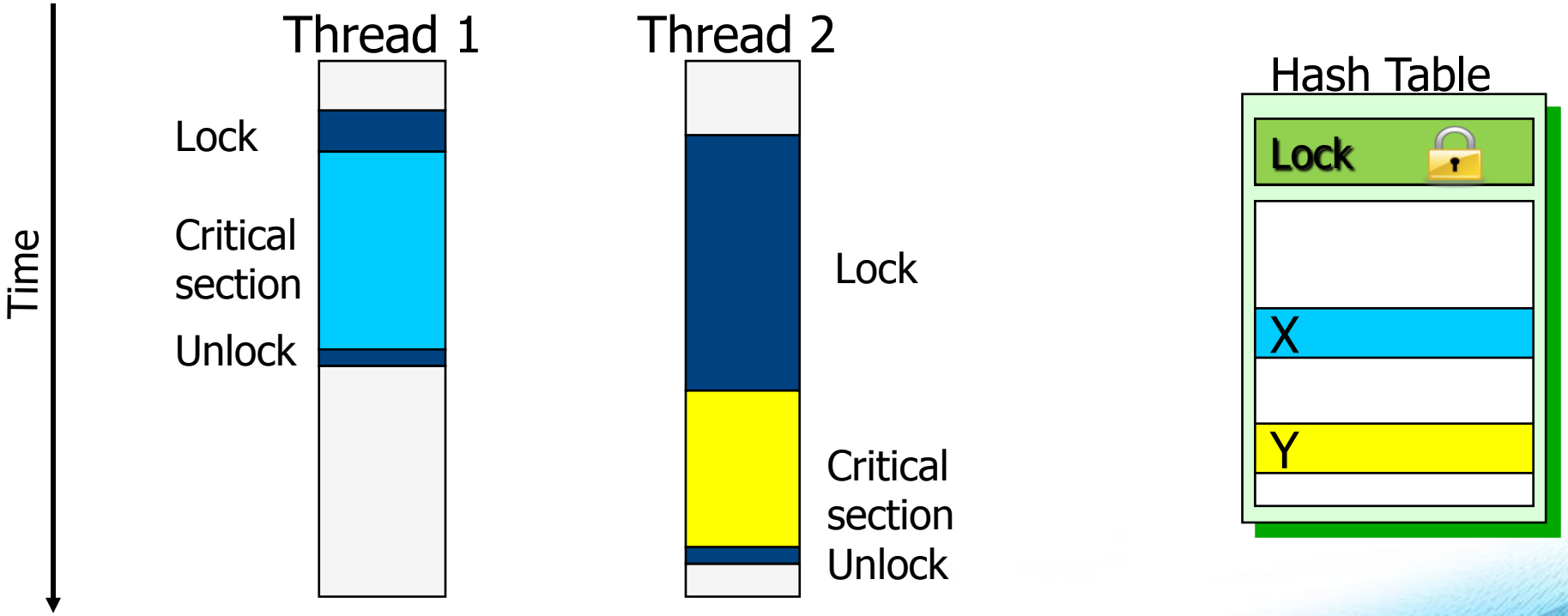
- SIMD used for ISO 8859 encoder intrinsic
  - Intrinsic for ISO_8859_1::encodeISOArray() method
  - Using SSE/AVX 16 chars encoded at a time
  - 32 characters encoded using AVX2
  - Micro-benchmark: 5X gain with SSE/AVX, 7X gain with AVX2
  - ISO 8859 encoder used in enterprise applications

Processor Features and JVM Optimizations
Transactional Memory

Intel TSX

- Restricted Transactional Memory Support
- Provides fine grain lock performance with coarse grain locks
- Instruction set extension for IA
  - XBEGIN <fallback_handler>
  - XEND
  - XABORT <abort_code>
  - XTEST
- Hardware manages transactional updates – All or none

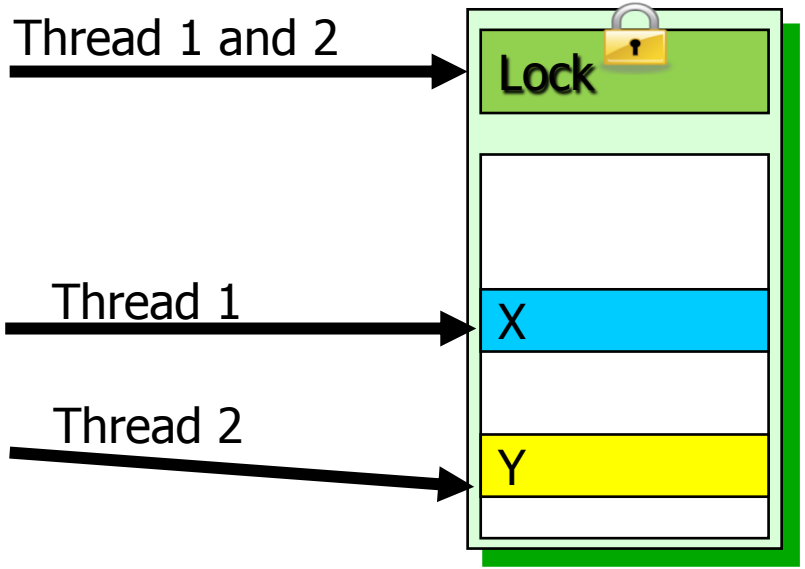
Intel TSX


Time
Thread 1
Lock
Critical section
Unlock
Thread 2
Lock
Critical section
Unlock
Hash Table
Lock
X
Y

Lock Contention Vs Data Conflict

Data conflicts limit concurrency, not lock contention

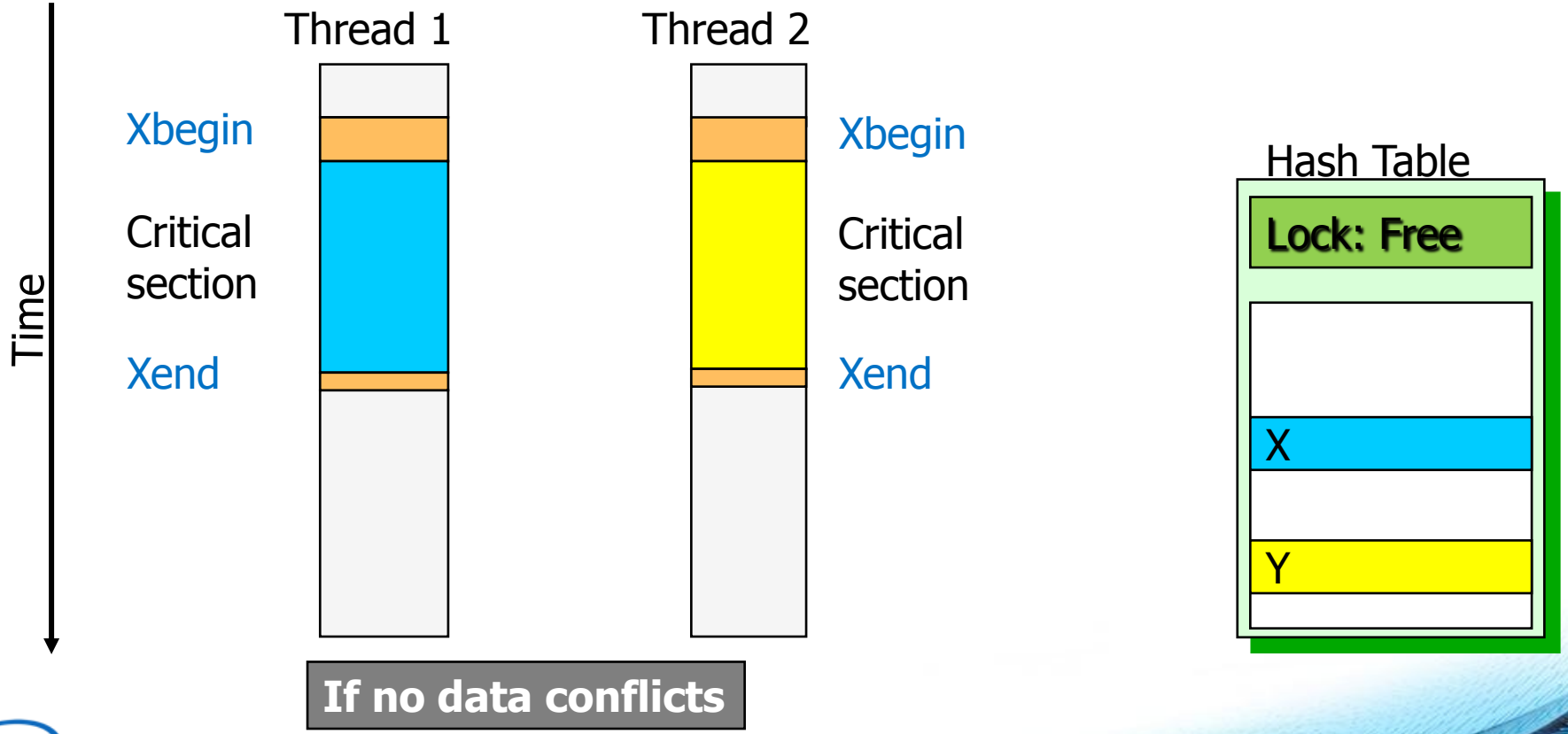
Lock contention present



No data conflicts

Focus on data conflicts, not lock contention

TSX Locking: Concurrent Critical Section



TSX Locking in Java

- Applicable to Java synchronized methods and statements
- Synchronization implemented using TSX
- Useful for highly contended locks
- JVM option to enable -XX:+UseRTMLocking
- Abort statistics collection and display
- Support for retries, auto tuning and manual tuning

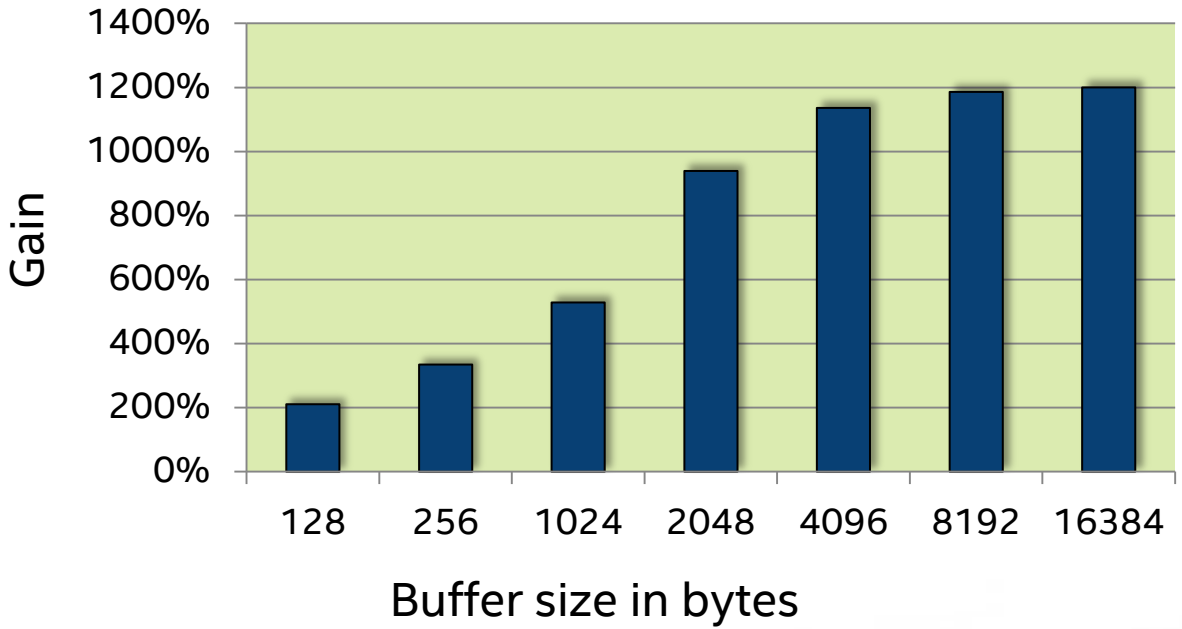
Processor Features and JVM Optimizations
CRC Acceleration

CRC Acceleration

- java.utill.zip CRC32 acceleration
  - Different polynomial than supported by IA CRC32 instruction
  - PCLMULQDQ instruction for checksum acceleration
  - 3x gain on SandyBridge for 4K buffer
  - 10x gain on Haswell for 4k buffer

  http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/fast-crc-computation-generic-polynomials-pclmulqdq-paper.pdf
- Used in checksum calculation during compression
- Used in Big Data workloads (Hadoop/HBASE) for checksum
  - 4% improvement for HiBench sort

CRC Acceleration



Processor Features and JVM Optimizations
Crypto Acceleration

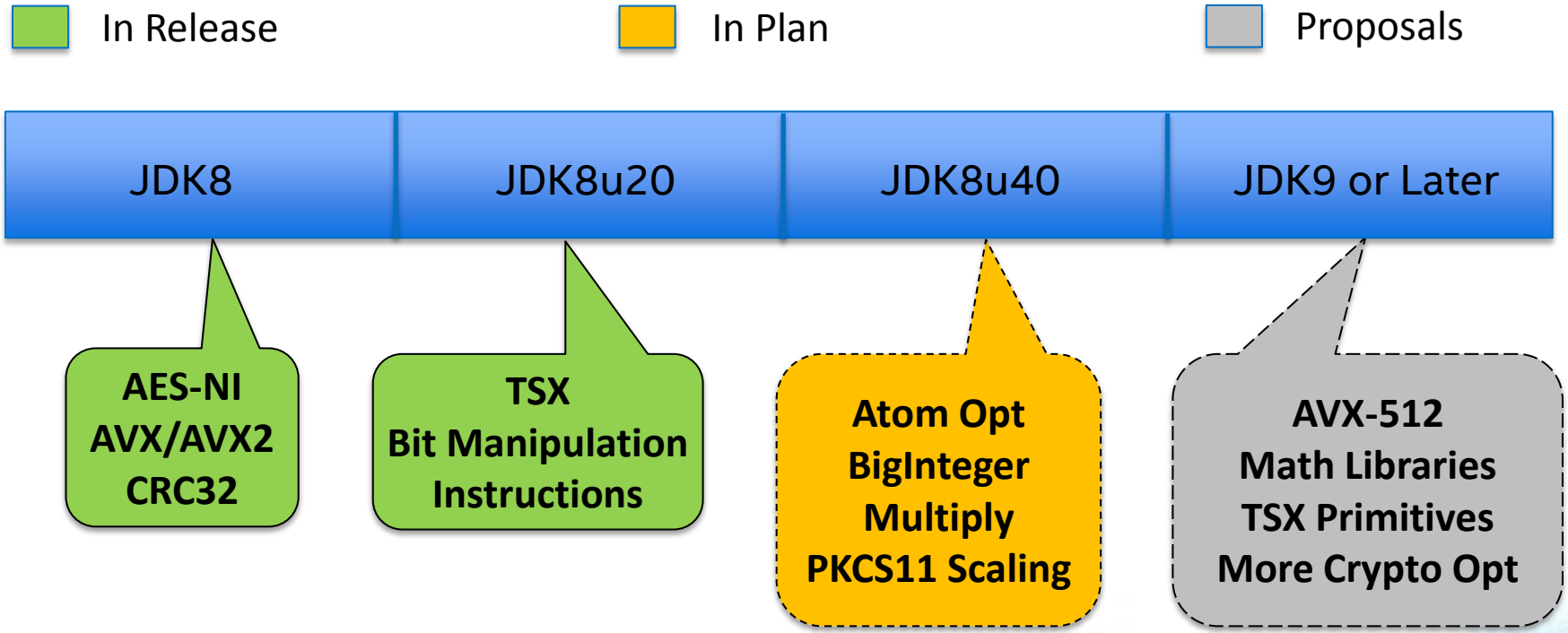
AES-NI in Java

- Stub for SunJCE crypto provider
- AES-NI instructions to accelerate AES encryption & decryption
- Used in Enterprise and Health care applications
  - 3x-10x performance gain on kernels
  - 10% gain on SPECjvm2008 crypto.aes
  - 5% gain on SPECjbb2013
  - GNAX health 50% gain for encryption and 60% for decryption

http://www.gnaxhealth.com/docs/Protecting_Healthcare_Data_Cloud_GNAX_Health_Intel.pdf

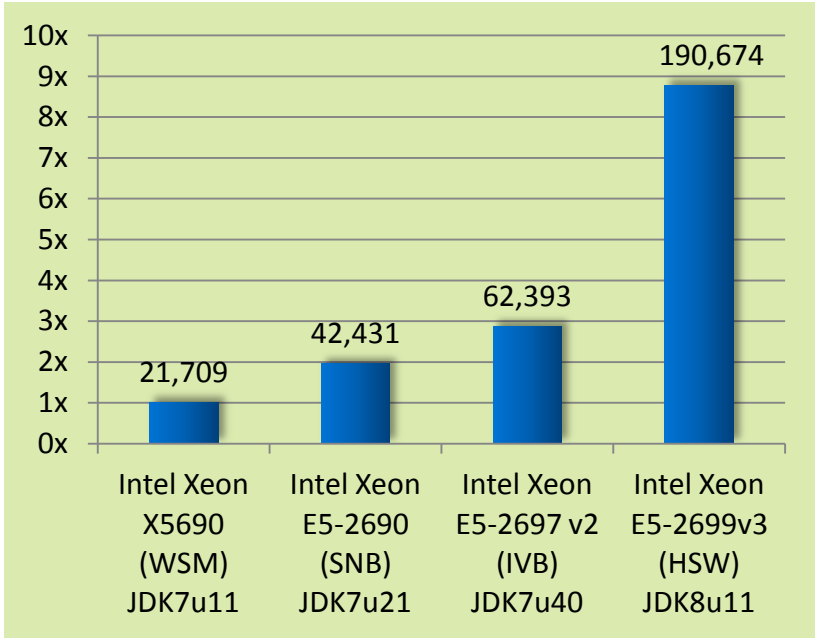
JVM Optimization Timeline & Proposals

Optimization Timeline & Proposals



Java Performance on Intel Architecture

SPECjbb2013 Performance



WSM
www.spec.org/jbb2013/results/res2013q1/jbb2013-20130205-00003.html

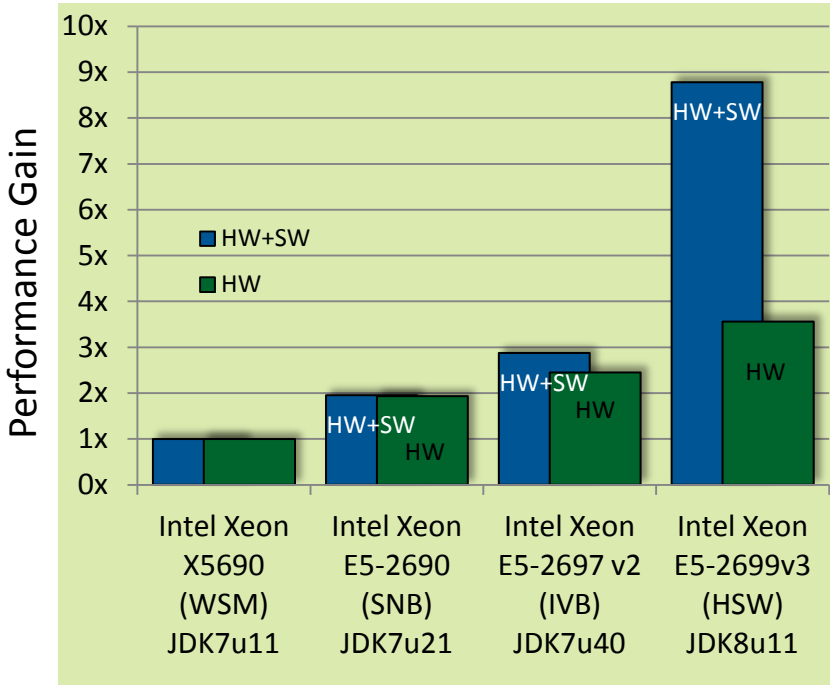
SNB
www.spec.org/jbb2013/results/res2013q3/jbb2013-20130723-00035.html

IVB
www.spec.org/jbb2013/results/res2013q4/jbb2013-20130917-00039.html

HSW
www.spec.org/jbb2013/results/res2014q3/jbb2013-20140902-00101.html

SPECjbb2013-MultiJVM max-jOPS performance gains across multiple generations of Intel® Xeon E5 platforms and Java Software

SPECjbb2013 Performance: HW/SW Gain



WSM
www.spec.org/jbb2013/results/res2013q1/jbb2013-20130205-00003.html

SNB
www.spec.org/jbb2013/results/res2013q3/jbb2013-20130723-00035.html
www.spec.org/jbb2013/results/res2013q2/jbb2013-20130403-00028.html

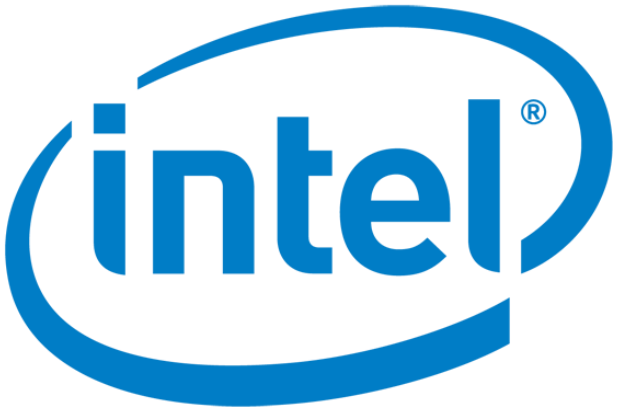
IVB
www.spec.org/jbb2013/results/res2013q4/jbb2013-20130917-00039.html
www.spec.org/jbb2013/results/res2014q3/jbb2013-20140818-00089.html

HSW
www.spec.org/jbb2013/results/res2014q3/jbb2013-20140902-00101.html
www.spec.org/jbb2013/results/res2014q3/jbb2013-20140818-00090.html

- HW only numbers are with JDK7u11 on all the platforms
- WSM performance is taken as base

SPECjbb2013-MultiJVM max-jOPS performance gains across multiple generations of Intel® Xeon E5 platforms and Java Software

Questions or suggestions?


intel