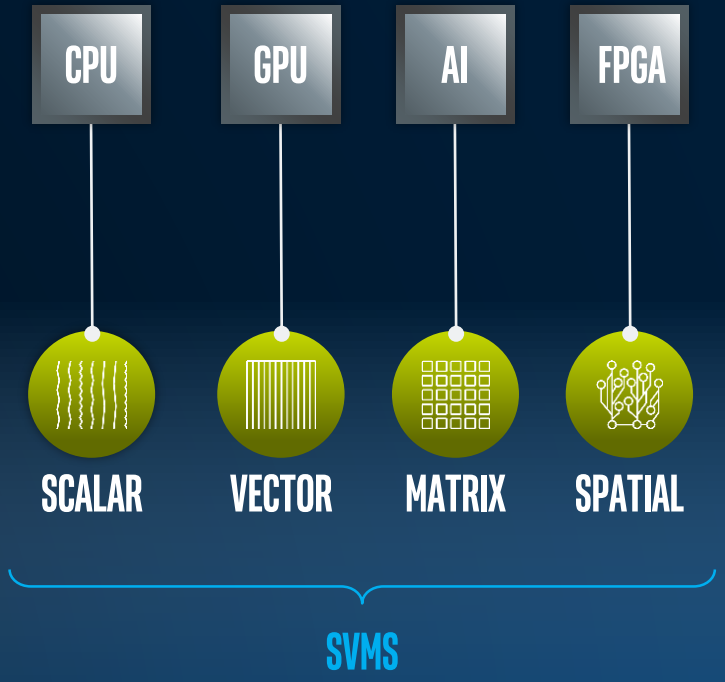
Why a single C++ API makes sense for Heterogeneous Compute

J.D. Patel
Jayesh.Patel@Intel.com

DIVERSE WORKLOADS REQUIRE DIVERSE ARCHITECTURES

The future is a **diverse** mix of scalar, vector, matrix, and spatial **architectures** deployed in CPU, GPU, AI, FPGA and other accelerators



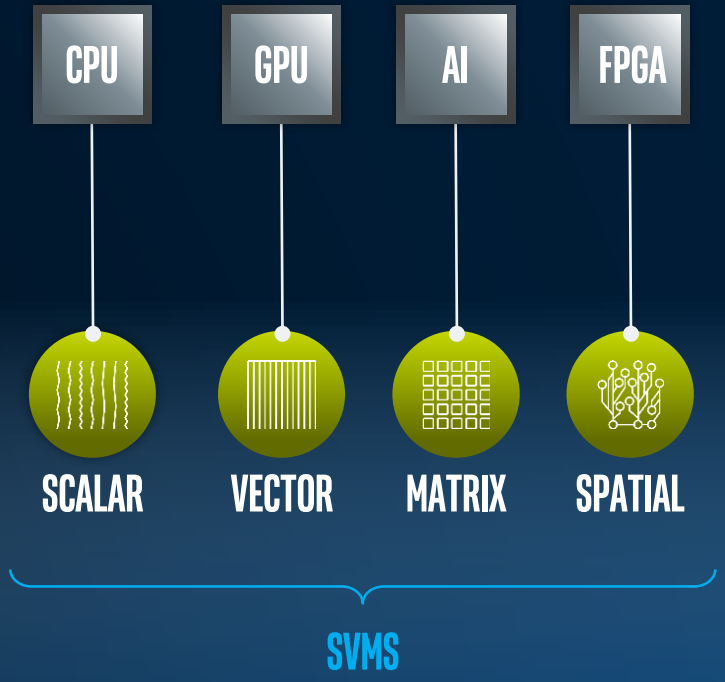
PROGRAMMING CHALLENGE

Diverse set of data-centric hardware

No common programming language or APIs

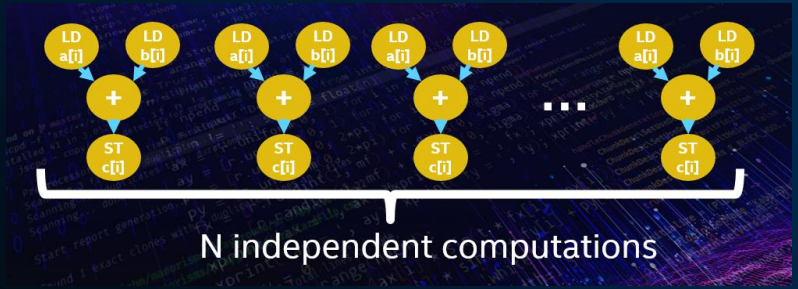
Inconsistent tool support across platforms

Each platform requires unique software investment



HOW DO SVMS ARCHITECTURES SUPPORT PARALLELISM

Compiler and runtime map the N independent computations to the data parallel hardware.



EXPECTATIONS OF UNIFIED PROGRAMMING MODEL/LANGUAGE

Cross-architecture support with extensibility for increased portability

Performance closer to respective native model/language/arch

Open standards based for increased productivity

WHY NOT AN EXISTING LANGUAGE?

Portable languages are either serial (C++, Python) or high-level (MATLAB*).

Data parallel languages are either proprietary (CUDA*) or low-level (OpenCL*).

Lack of commonality in code-bases and methodology, resulting in extra cost and delays

Data Parallel C++

Cross architecture

Performant

Open

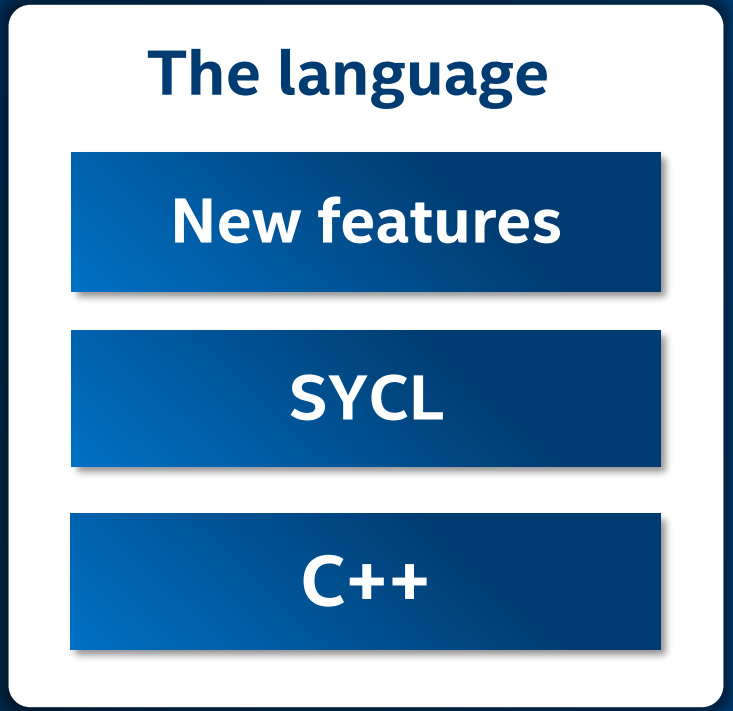
DATA PARALLEL C++

C++ language of choice for performance
Modern C++ with productivity features

SYCL* for cross-architecture support
Abstracts away boiler-plate OpenCL code
Interoperability with OpenCL maintained
Device-code restrictions apply
Modern features:
- Automatic scheduling of data movement
- Single-source compilation

New features open-sourced:
- Making SYCL more feature rich
- e.g. Unified shared memory support

The goal is to have all standards based:
new features → SYCL → C++



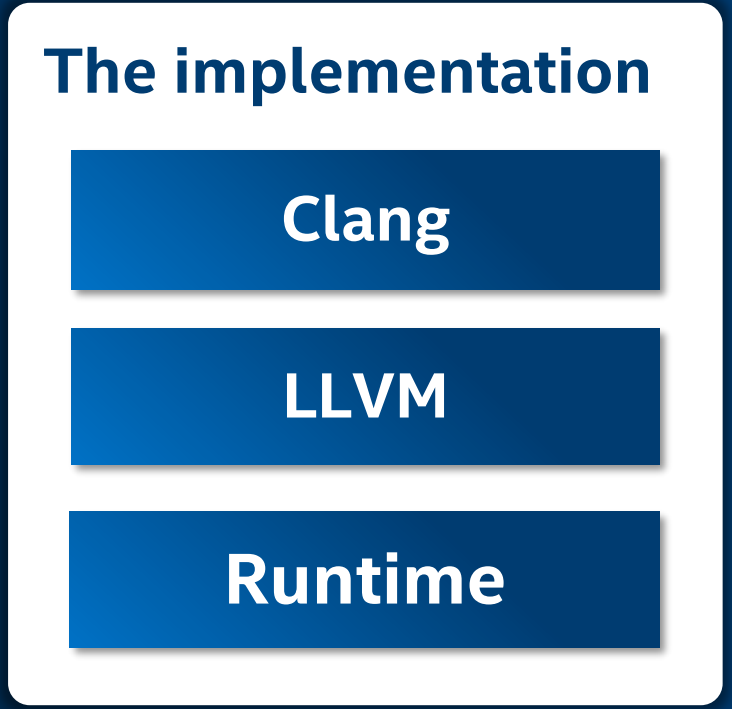
DATA PARALLEL C++

Intel-led open-source implementation

Popular open-source components

Broad industrywide adoption

Cross-architecture/vendor contributions



DATA PARALLEL C++

STANDARDS-BASED, CROSS-ARCHITECTURE LANGUAGE

Language to deliver uncompromised parallel programming productivity and performance across CPUs and accelerators

- Allows code reuse across hardware targets, while permitting custom tuning for a specific accelerator
- Open, cross-industry alternative to single architecture proprietary language

Based on C++

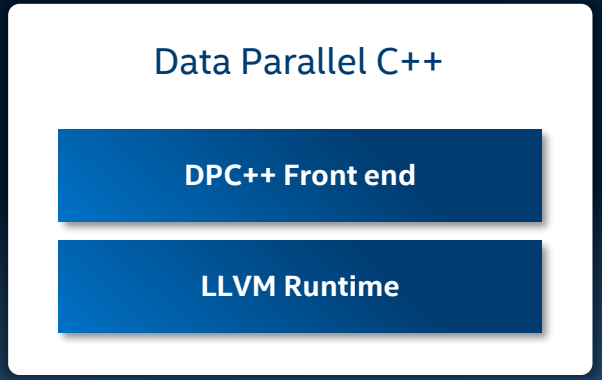
- Delivers C++ productivity benefits, using common and familiar C and C++ constructs
- Incorporates SYCL* from the Khronos* Group to support data parallelism and heterogeneous programming

Community Project to drive language enhancements

- Extensions to simplify data parallel programming
- Open and cooperative development for continued evolution

Builds upon Intel's years of experience in architecture and compilers

Custom-tuning for each architecture will still be required.



oneAPI: single unified programming model to deliver cross-architecture performance

INTEL ONEAPI CORE CONCEPT

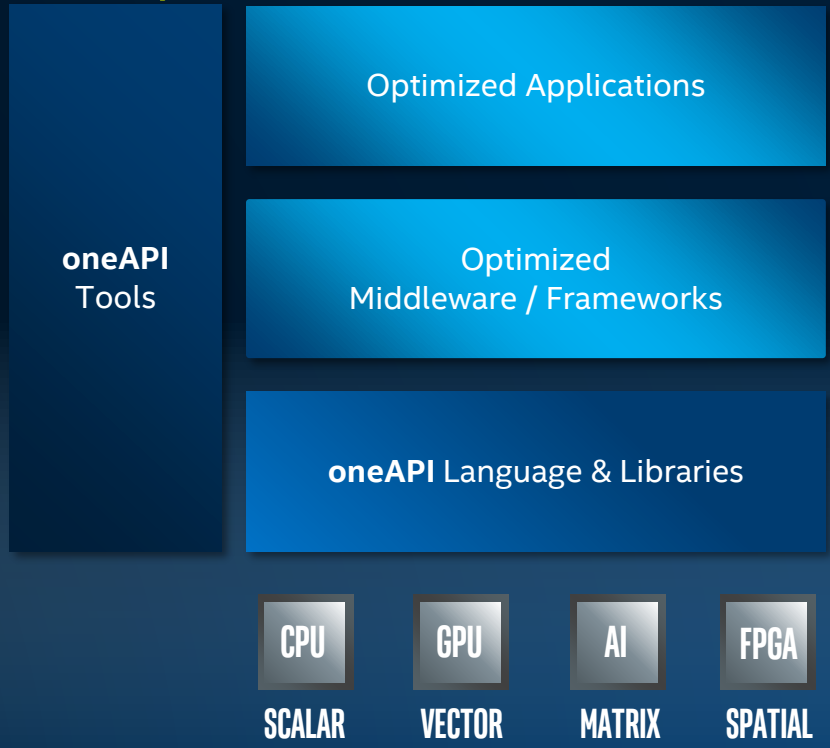
Project oneAPI delivers a unified programming model to simplify development across diverse architectures

Common developer experience across Scalar, Vector, Matrix and Spatial architectures (CPU, GPU, AI and FPGA)

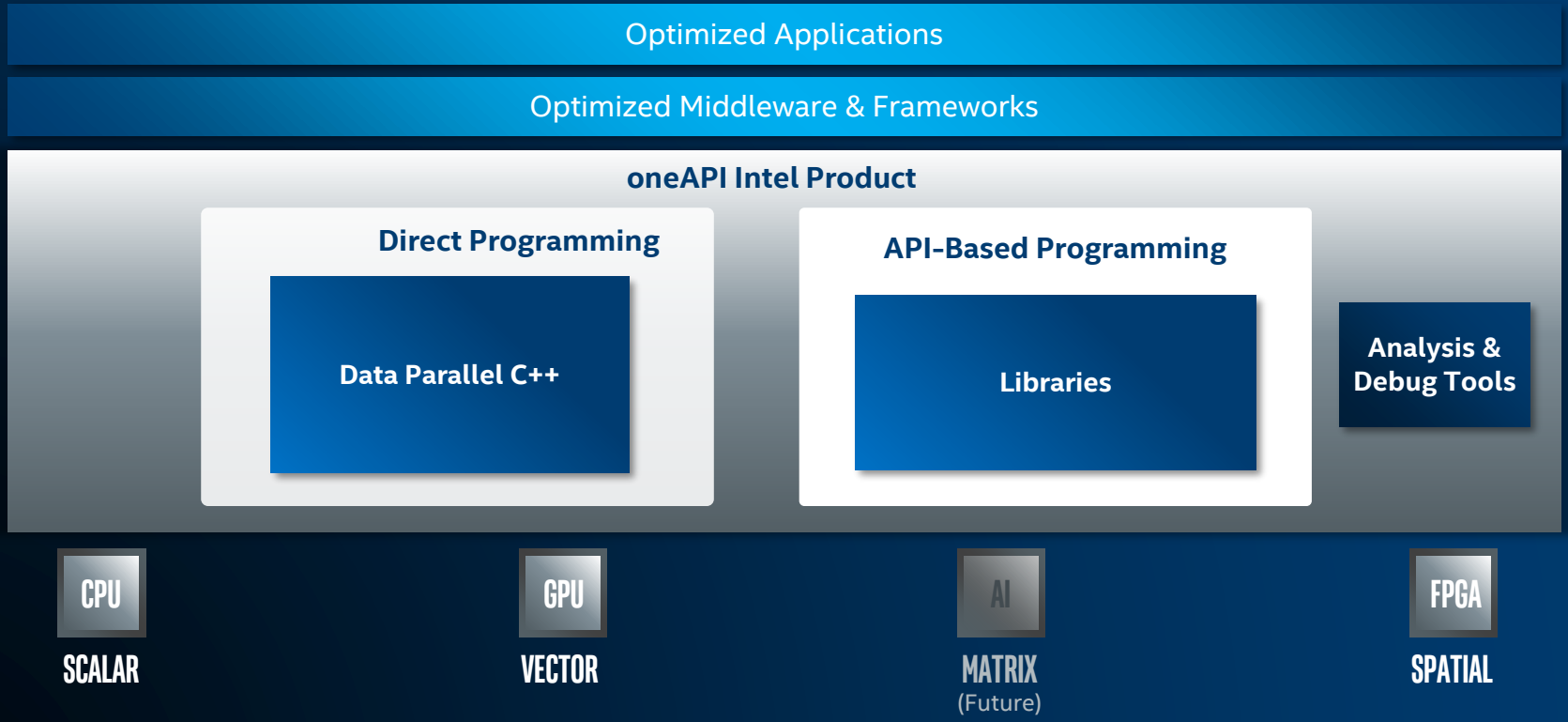
Uncompromised native high-level language performance

Device-specific tuning will still be required for max performance

Based on industry standards and open specifications



ONEAPI INTEL PRODUCT



Some capabilities may differ per architecture and custom-tuning will still be required.

SUMMARY

Diverse workloads for data-centric computing are driving the need for diverse compute architectures including CPUs, GPUs, FPGAs, and AI accelerators

oneAPI unifies and simplifies programming of Intel CPUs and accelerators, delivering developer productivity and full native language performance

oneAPI is based on industry standards and open specifications to encourage ecosystem collaboration and innovation

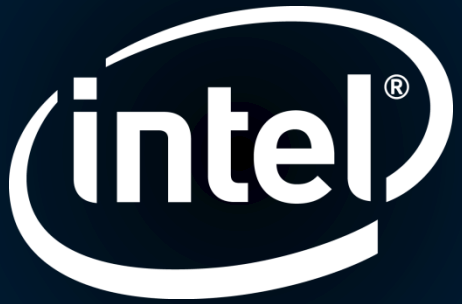
NOTICES & DISCLAIMERS

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.

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


intel®