백서

oneAPI 가속 컴퓨팅을 향한 크로스 아키텍처, 멀티 벤더 솔루션

- 기술은 틈새 시장부터 대규모 시장까지 확장할 수 있는 표준이 필요합니다.
- 개발자들은 독점 솔루션을 대신할 수 있는 개방형 멀티벤더 대안을 요구하고 있습니다.
- oneAPI는 개발자가 선택할 수 있는 개방적이고 신뢰할 수 있는 솔루션입니다.

저자

Joe Curley
소프트웨어 및 고급 기술 부서 부사장
인텔 소프트웨어 제품 및 에코시스템 부서
제너럴 매니저(GM)

Sanjiv Shah
소프트웨어 및 고급 기술 부서 부사장
인텔 소프트웨어 부서 개발자
제너럴 매니저(GM)

소프트웨어 표준화를 통한 가속기술 활성화

가속기 기술은 엔드포인트에서 데이터 센터에 이르기까지 컴퓨팅 인프라 전반에 걸쳐 더욱 많은 관심을 받고 있습니다. 폭발적으로 증가하는 데이터 관리부터 시간이 중요한 비즈니스 프로세스에 이르기까지 다양한 사용자 요구로 인한 에너지 예산이 기하급수적으로 증가하면서 컴퓨팅 성능은 더 중요해지고 있습니다.

지난 10년 동안은 프로그래머블 그래픽 가속기인 GPU의 사용에 대한 논의가 주를 이루었지만, 오늘날의 가속기 아키텍처는 GPU이상으로 훨씬 더 다양합니다. 현재 배포되었거나 개발 중인 인공 지능(AI), 고성능 컴퓨팅(HPC), 암호화, 데이터 이동 및 입출력(IO), 통신을 위한 전문 가속기는 슈퍼스칼라, 벡터, 데이터 플로우/스페이셜, 매트릭스, 새롭게 떠오르는 뉴로모픽 및 양자 등 다양한 아키텍처를 사용합니다. 아키텍처의 다양성 외에도 가속기 구현은 기존의 입출력(IO) 연결 디바이스부터 코히어런트 메모리 디바이스, CPU 콤플렉스와 긴밀하게 통합된 디바이스에 이르기까지 다양합니다.

가속기를 대량으로 도입하는 데 있어 가장 중요한 과제는 가속기 소프트웨어 개발의 프로세스, 시간, 비용, 그리고 유지 보수입니다. 가속기의 주요 장점은 특정 작업을 보다 효율적으로 수행하도록 설계되었다는 점입니다. 데이터 병렬 컴퓨팅을 위해 프로그래밍 할 수 있는 GPU부터 데이터 해싱이나 딥 러닝과 같은 특정 작업을 위해 특별히 제작된 ASIC에 이르기까지 일반적인 프로그래밍이 가능한 정도는 다양하지만 이러한 가속기의 프로그래밍 도구와 모델은 대부분 개별 디바이스별로 특화되어 있습니다. 디바이스 전문화를 통해 특정 작업에 최적화된 성능을 구현할 수는 있지만 개발자가 한 가속기에 맞도록 개발한 내용을 다른 가속기에 이식하여 그대로 사용할 수는 없습니다.

새로운 기술은 종종 특정 공급업체의 폐쇄적인 아키텍처로 초기에는 성공을 거두지만, 기술이 널리 보급되기 위해서는 여러 공급업체가 참여한 표준이 등장하여 결국에는 독점적인 선구자를 대체하게 됩니다. 이러한 자연스러운 순환에는 공급업체 종속의 경제적 결과를 포함하여 단일 벤더에게 의존하는데 따른 기술적 및 경제적 위험, 솔루션 구축, 배포 및 유지 관리 비용 등 여러가지 요인이 영향을 미칩니다. 이러한 사실은 희소한 기술을 보유한 직원의 필요성으로 인해 더욱 복잡해 집니다. 결과적으로 고객, 최종 사용자, 소프트웨어 개발자 및 커뮤니티는 가속 컴퓨팅을 위한 표준화된 솔루션을 찾기 위해 인텔과 협력하게 되었습니다.

oneAPI란?

oneAPI는 업계 전반에 걸쳐서 프로세서 및 가속기 아키텍처에 공통된 개발자 경험을 제공하는 개방형 표준 통합 프로그래밍 모델입니다.

oneAPI는 오픈 규격, 오픈 소스 구현 및 인텔 강화 구현이라는 세 가지 요소로 구성됩니다.

기존 표준을 기반으로 하는 oneAPI 사양은 커뮤니티 개발자와의 공개 협업을 통해 개발되었습니다. 새로운 하드웨어 아키텍처 또는 소프트웨어 언어에 대한 oneAPI의 빠른 사용을 지원하기 위해 oneAPI 구성 스펙에 대한 오픈 소스 구현이 이루어졌습니다. oneAPI는 다음과 같은 개발자들의 요구에 부응합니다.

- 단일 벤더, 단일 아키텍처 소프트웨어의 경제적 및 기술적 단점 방지
- 다양한 하드웨어에서 성능을 제공하는 동시에 생산성 향상
- 현재는 물론 미래에도 신뢰할 수 있는 개발 환경 제공

전체 사양과 사용에 문제가 없는 오픈 소스 코드 저장소에 대한 링크는 oneAPI 커뮤니티 웹 사이트(https://oneapi.io)에서 확인이 가능합니다. oneAPI 업계 이니셔티브는 생태계 전반에 걸쳐 호환 가능한 oneAPI 구현에 대한 더 많은 협업을 권장하고 있습니다.

인텔의 oneAPI 제품은 oneAPI 사양을 구현하고 소프트웨어 개발자를 위한 표준에 기반한 툴과 구성 요소를 제공합니다. 인텔의 oneAPI 제품은 개별 구성 요소 또는 쉽게 액세스할 수 있는 툴킷으로 제공되며, 여기에는 모든 주요 oneAPI 구성 요소가 포함된 인텔® oneAPI 기본 툴킷과 HPC, AI 및 분석, 사물 인터넷(IoT), 고급 렌더링을 위한 도메인에 최적화된 툴킷이 포함됩니다.

개방형 표준을 바탕으로 구축된 신뢰할 수 있는 기반

oneAPI는 인텔의 풍부한 고성능 컴파일러 및 라이브러리를 활용하여 개방형 표준을 구축하여 가속 컴퓨팅 배포와 관련된 위험을 줄여줍니다. 예를 들어, oneAPI Math Kernel Library (oneMKL)는 수십 년 동안 광범위하게 사용되고 개선된 인텔® 수학 커널 라이브러리를 통합한 것으로, Evans Data 2021 개발자 설문조사[1]에 따르면 업계에서 가장 널리 사용되는 고성능 수학 라이브러리입니다. 또한, oneAPI는 Fortran, C/C++, OpenMP 및 MPI와 같은 기존 HPC 프로그래밍 표준은 물론 Python 및 최적화된 풍부한 Python 라이브러리와도 상호 운용이 가능하여 레거시 코드와 쉽게 통합할 수 있습니다.

개발자는 성능과 정확성을 분석하고 기존 코드를 가속기로 오프로드하는 데 도움이 되는 고급 디버거 및 프로파일러 세트를 사용할 수 있습니다. 독점적인 CUDA 코드를 오픈 SYCL 언어(khronos.org/sycl)로 마이그레이션 하는 데 도움이 되는 도구[2]도 있습니다. 이러한 장점 덕분에 oneAPI는 2021년 11월에 최고의 HPC 프로그래밍 도구와 기술[3]에 부여하는 HPCwire Readers 초이스 어워드 (Choice Award Best HPC Programming Tool or Technology)를 수상했습니다.

oneAPI는 개방형 표준을 준수함으로써 기존 코드의 가속화를 용이하게 할 뿐만 아니라 개발자가 자신의 코드가 미래의 아키텍처에서 실행될 것이라는 확신을 가질 수 있도록 해 줍니다.

독점적인 종속으로부터의 자유

oneAPI는 단일 벤더 또는 하드웨어 아키텍처에 종속(lock-in)되지 않는 독점적이고 폐쇄적인 "벽으로 둘러싸인 닫힌 정원(walled gardens)"에 대한 개방형 대안을 제공합니다.

이를 통해 하드웨어 및 소프트웨어 개발자 커뮤니티는 또 다른 개발자 에코시스템을 구축해야 하는 부담 없이 새로운 가속기를 제공할 수 있습니다. 예를 들어, Fujitsu는 oneAPI Deep Neural Network Library (oneDNN)[4]를 사용하여 Fugaku 슈퍼컴퓨터에서 선도적인 MLPerf 벤치마크 성능[5]을 달성했습니다. 또한, oneAPI는 시스템 개발자가 특정 문제에 가장 적합한 아키텍처를 선택하고 다양한 하드웨어 솔루션에서 가속기 혁신의 이점을 누릴 수 있게 해줍니다.

개발자들은 다음 하드웨어 플랫폼을 위해 소프트웨어를 다시 작성하는 대신 차세대 과학 기술 또는 더욱 혁신적인 기술을 개발하는데 시간을 투자할 수 있습니다.

최첨단 가속 하드웨어의 모든 가치를 실현하는 생산성 및 성능

oneAPI는 개발자가 여러 가속기 아키텍처에서 성능을 제공하는 단일 코드 베이스를 유지 관리할 수 있도록 하여 소프트웨어 개발을 간소화하고 개발 및 유지 관리 비용을 절감합니다.

컴파일러와 API의 라이브러리 구현을 최적화함으로써 개발자는 최신 하드웨어의 혁신적인 기능을 활용할 수 있습니다.

oneAPI의 성공 비결은 크로노스 그룹(Khronos Group)이 개발한 SYCL 언어를 채택한 것입니다. SYCL은 최신 C++를 확장하여 단일 가속기 아키텍처용으로 설계된 언어와 달리 이기종 병렬 처리를 지원합니다. 또한, oneAPI는 oneMKL과 같은 기존 라이브러리에 대한 표준 인터페이스를 제공합니다. 개발자들은 인텔의 CPU와 GPU가 아닌 공급업체에 최적화된 가속기 라이브러리를 일반적인 오픈소스 개발 방법을 사용하여 oneAPI에 성공적으로 적용시켰습니다.

마지막으로, oneAPI의 개방적인 특성은 생산적인 혁신을 위한 방향을 제공합니다. 개방형 인터페이스에서 오픈 소스 구현 및 인프라에 이르기까지, oneAPI는 개발자들이 생산적이고 현대적인 개방형 방식을 통해 새로운 언어 프론트엔드와 새로운 하드웨어 백엔드를 독립적으로 혁신하고 개발할 수 있도록 지원합니다.

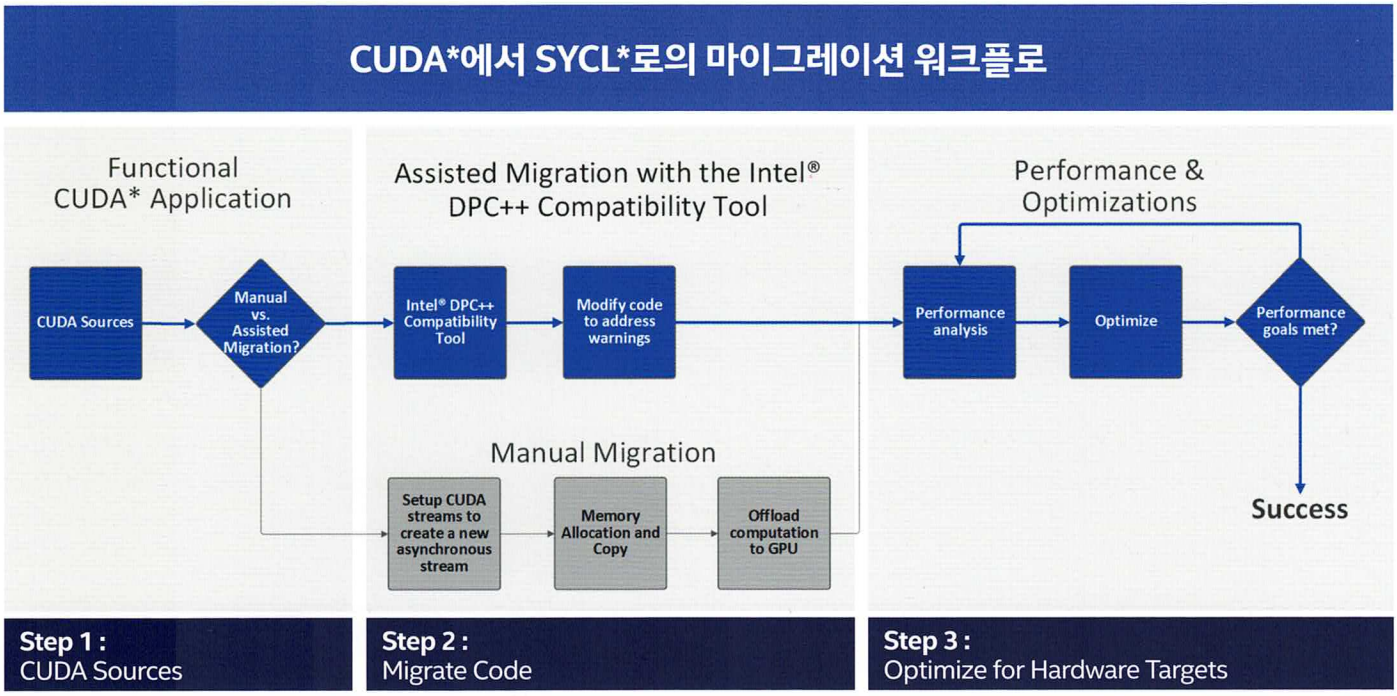
oneAPI는 하나의 아키텍처에 국한된 하드웨어별 모델이나 생산성과 성능을 맞바꾸는 해석된 언어와 달리 다양한 가속기에 걸쳐 고성능 애플리케이션을 배포할 수 있는 더 빠른 경로를 제공합니다.

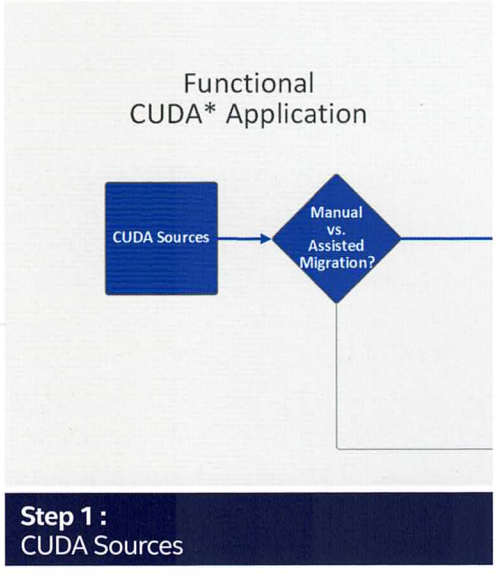
요약

기술은 틈새시장에서 대규모로 확장할 수 있는 표준이 필요합니다. 개발자 커뮤니티는 가속 컴퓨팅을 위한 독점적 프로그래밍 모델에 대한 대안을 요청했고, oneAPI는 그 대안을 제공했습니다.

oneAPI 이니셔티브는 개발자가 선택할 수 있는 개방적이고 신뢰할 수 있는 경로를 제공합니다.

가속 컴퓨팅을 위한 크로스 아키텍처, oneAPI 라면 가능합니다.





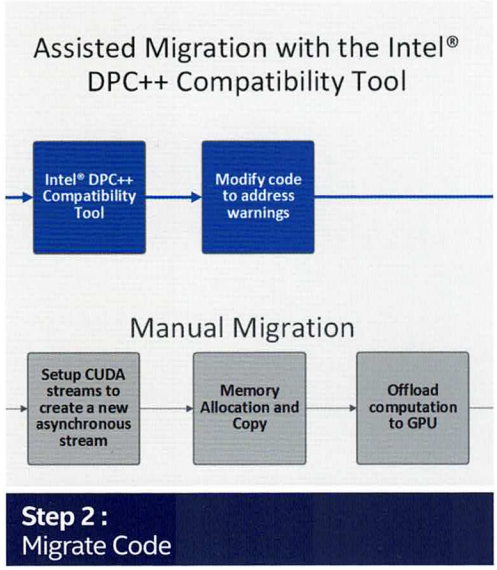
1단계 : CUDA 소스 마이그레이션 방법 결정

SYCL로 마이그레이션하려면 작동하는 CUDA 애플리케이션이 있는지 확인하십시오. 다음 중 하나를 통해 CUDA 소스를 마이그레이션할 수 있습니다. CUDA와 SYCL 코드를 나란히 비교할 수 있는 인텔 DPC++ 호환성 도구를 사용하여 대부분의 SYCL 코드를 자동 생성합니다. CUDA 소스를 수동으로 분석하고 모든 특정 CUDA 호출을 동등한 SYCL 호출로 대체합니다.

인텔 DPC++ 호환성 도구는 일반적으로 코드의 90%-95%를 마이그레이션하고 마이그레이션을 완료하기 위해 수동 개입이 필요한 코드 영역에 대한 경고를 생성합니다.[6]

이 도구는 **<dpct/dpct.hpp>** 헤더 파일 에 정의된 도우미 함수를 사용합니다 . 이는 dpct 도우미 기능을 지원하기 위해 일부 SYCL 호출이 추가 레이어로 래핑되기 때문입니다. 수동으로 마이그레이션된 SYCL 코드는 CUDA 호출에 직접 매핑되는 SYCL 호출 및 구문을 사용합니다.

간단한 벡터 추가 샘플을 사용하여 마이그레이션을 다운로드하고 시도하십시오 .



2단계 : 코드 마이그레이션

이 단계에서는 **Assisted** 또는 **Manual** 방법을 사용하여 소스 코드를 SYCL로 마이그레이션합니다. 마이그레이션을 완료한 후 SYCL 소스 코드에 대한 개발 작업을 계속하십시오.

Assisted Migration

인텔 DPC++ 호환성 도구를 사용하여 기존 CUDA 코드를 SYCL로 마이그레이션합니다. 이 도구는 CUDA 언어 커널 및 라이브러리 API 호출을 이식하고 대부분의 CUDA 코드를 아키텍처 및 공급업체에서 이식 가능한 SYCL 코드로 마이그레이션합니다.

참고

SYCL에 대해 자세히 알아보고 인텔 DPC++ 호환성 도구가 소스 코드를 어떻게 변경했는지 이해하려면 CUDA에서 SYCL로 마이그레이션–Jacobi 반복 방법을 참조하십시오. 이 가이드는 Jacobi 샘플을 사용하여 CUDA와 SYCL 매핑 사이의 기술적인 세부 사항을 설명합니다.

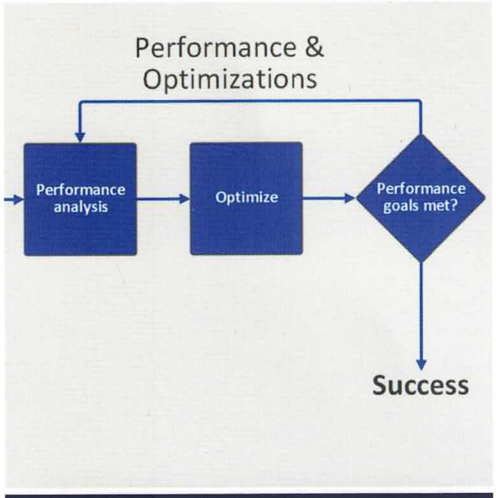
Manual Migration

수동으로 마이그레이션된 SYCL 코드는 CUDA 호출에 직접 매핑되는 실제 SYCL 호출 및 구문을 사용합니다. 이 방법은 더 깔끔하게 마이그레이션된 코드를 제공하고 코드를 더 쉽게 따라갈 수 있도록 합니다. 둘 사이의 코드 기능은 거의 동일합니다.

Jacobi 샘플을 사용하는 CUDA와 SYCL 매핑 사이의 기술적인 세부 사항은 CUDA에서 SYCL로 마이그레이션–Jacobi 반복 방법의 지침을 참조하십시오 . 이 안내서는 CUDA 및 SYCL의 기본 개념과 코드 마이그레이션을 위한 필수 용어를 설명합니다.

비동기 스트림과 메모리 할당 및 복사를 오프로드하고 설정하는 일반적인 단계가 있지만 실제 작업은 오프로드 계산에서 발생합니다. CUDA와 SYCL은 GPU에서 실행되는 오프로드 커널 생성에 대한 몇 가지 기본 개념을 공유합니다. SYCL 구문을 효율적으로 이해하려면 유사점과 차이점을 식별하여 이러한 많은 개념을 매핑하십시오.

- CUDA 스레드 블록 및 SYCL 작업 그룹
- SLM(공유 로컬 메모리) 액세스
- CUDA 스레드 블록 및 SYCL 배리어 동기화
- CUDA 협력 그룹 및 SYCL 하위 그룹
- CUDA 워프 프리미티브 및 SYCL 그룹 알고리즘
- CUDA 및 SYCL 원자성



Step 3 :
Optimize for Hardware Targets

3단계 : 하드웨어 대상에 맞게 최적화

이 단계에서 컴파일 및 실행되는 작업 코드가 있습니다. Intel® VTune™ Profiler 및 Intel® Advisor와 같은 Intel® 도구를 사용하여 Intel GPU용으로 마이그레이션된 코드를 최적화합니다.

이러한 도구는 애플리케이션 성능을 최적화하기 위해 개선해야 할 코드 영역을 식별하는 데 도움이 됩니다. 두 도구 모두 최적화 전략을 시각화하는 데 도움이 되는 그래픽 사용자 인터페이스를 포함합니다.

intel.

[1] 2021년 글로벌 개발 설문조사, 제 1 부, 에반스 데이터 코퍼레이션(Evans Data Corporation)
[2] 예제: 인텔® DPC++ 호환성 도구 및 https://www.iwocl.org/wp-content/uploads/iwocl-2019-dhpcc-tobias-stauber-resyclator-transforming-cuda-C-source-code-into-sycl.pdf
[3] www.hpcwire.com/off-the-wire/hpcwire-reveals-winners-of-the-2021-readers-and-editors-choice-awards-during-sc21/
[4] https://www.fujitsu.com/global/about/resources/publications/technicalreview/2020-03/article09.html 및
https://www.oneapi.io/event-sessions/ai-adeep-dive-into-a-deep-learning-library-for-the-a64fx-fugaku-cpu-meet-the-developer/
[5] MLPerf 벤치마크 성능: https://blog.fltech.dev/entry/2020/11/19/fugaku-onednn-deep-dive-en
[6] Intel 추정치는 2021년 9월 기준이며 Rodinia, SHOC(Scalable Heterogeneous Computing) 및 Pennant와 같은 70개의 HPC 벤치마크 및 샘플 세트에 대한 측정을 기반으로 합니다. 결과는 다를 수 있습니다.