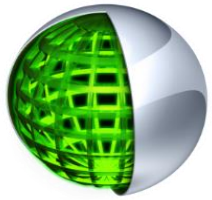
CUDA paralelno programiranje – biblioteke

CUDA platforma za paralelna izračunavanja

Programming Approaches

- Libraries — **"Drop-in" Acceleration**
- Directives — **Easily Accelerate Apps**
- Programming Languages — **Maximum Flexibility**

Development Environment



Nsight IDE
Linux, Mac and Windows
GPU Debugging and Profiling

CUDA-GDB debugger
NVIDIA Visual Profiler

Open Compiler Tool Chain

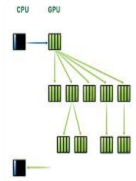
Enables compiling new languages to CUDA platform, and CUDA languages to other architectures

Hardware Capabilities

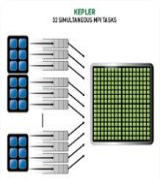
Izvor: Nvidia

SMX

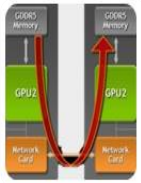
Dynamic Parallelism



HyperQ



GPUDirect



Tri načina za ubrzavanje aplikacija

aplikacije

biblioteke	direktive	programski jezici
cuFFT, cuDNN...	OpenACC	CUDA C, OpenCL C
"drop-in" ubrzavanje	lako ubrzavanje aplikacija	maksimalna fleksibilnost

Biblioteke

- Često korišćeni algoritmi se tipično organizuju u biblioteke
  - biblioteke se razvijaju od strane eksperata za performanse
  - mogu da ih koriste "obični" programeri, čime se značajno skraćuje proces razvoja i ubrzavaju aplikacije
- Uglavnom programer korisničke aplikacije ne mora uopšte da razmišlja o paralelizmu
  - paralelizam je "ugrađen" u biblioteku

Biblioteke

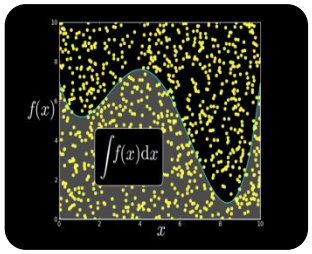
- Nedostaci primene biblioteka:
  - ponekad ih je teško prilagoditi specifičnim kontekstima
  - umeju biti složene i, s toga, razumevanje njihove implementacije može biti teško
- Prednosti primene biblioteka:
  - lako se koriste
  - "drop-in" – prate se standardni API-iji, čime je omogućeno ubrzanje sa minimalnim izmenama u kodu
  - programski kod je uglavnom visokog kvaliteta
  - pružaju visoke performanse

CUDA biblioteke

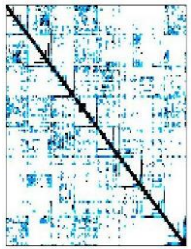
- cuBLAS – BLAS (Basic Linear Algebra Subprograms) biblioteka
- cuSPARSE – biblioteka za rad sa retko-posednutim matricama
- cuDNN – biblioteka za rad sa dubokim neuronskim mrežama
- cuFFT – biblioteka za brze Fourier transformacije
- cuRAND – Random Number Generation (RNG) bibilioteka
- NPP – performantne primitive za obradu slike i videa
- Thrust – šablonski C++ paralelni algoritmi i strukture podataka

Primeri GPU-ubrzanih biblioteka

NVIDIA cuBLAS



NVIDIA cuRAND



NVIDIA cuSPARSE

NVIDIA NPP

GPU VSIPL

Vector Signal Image Processing

CULA tools

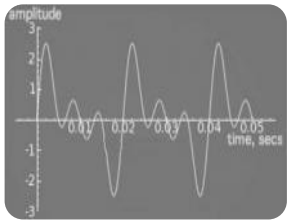
GPU Accelerated Linear Algebra

MAGMA

ICL UT

Matrix Algebra on GPU and Multicore

open source initiative



NVIDIA cuFFT

ROGUE WAVE SOFTWARE

IMSL Library

ArrayFire Matrix Computations

CUSP

Sparse Linear Algebra

open source initiative

C++ STL Features for CUDA

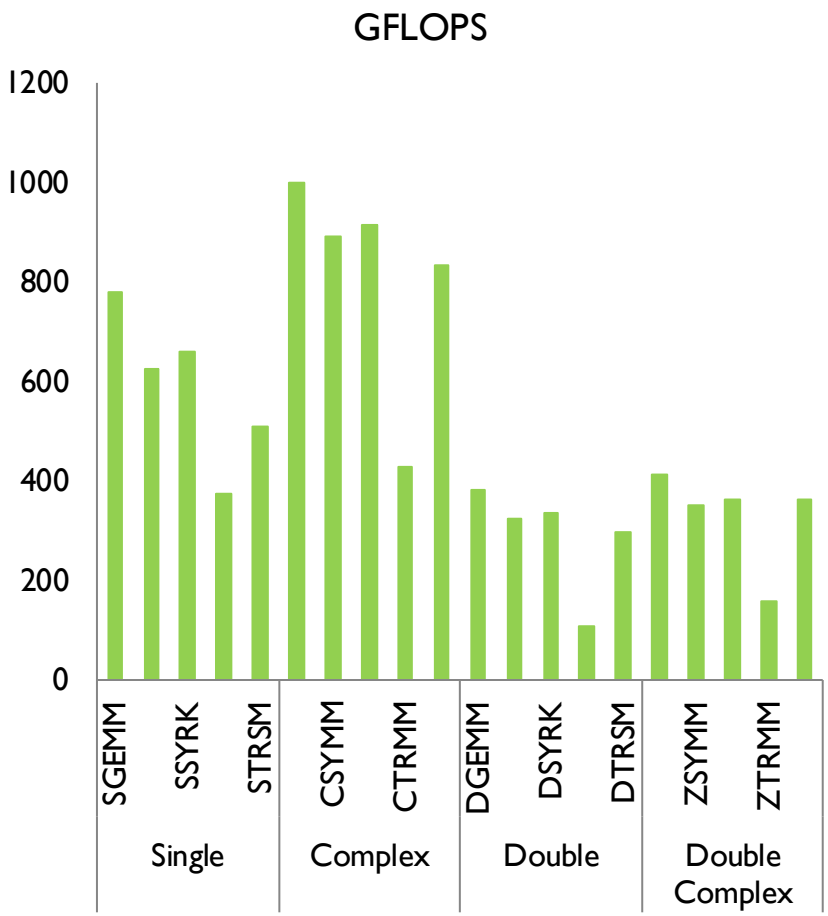
open source initiative

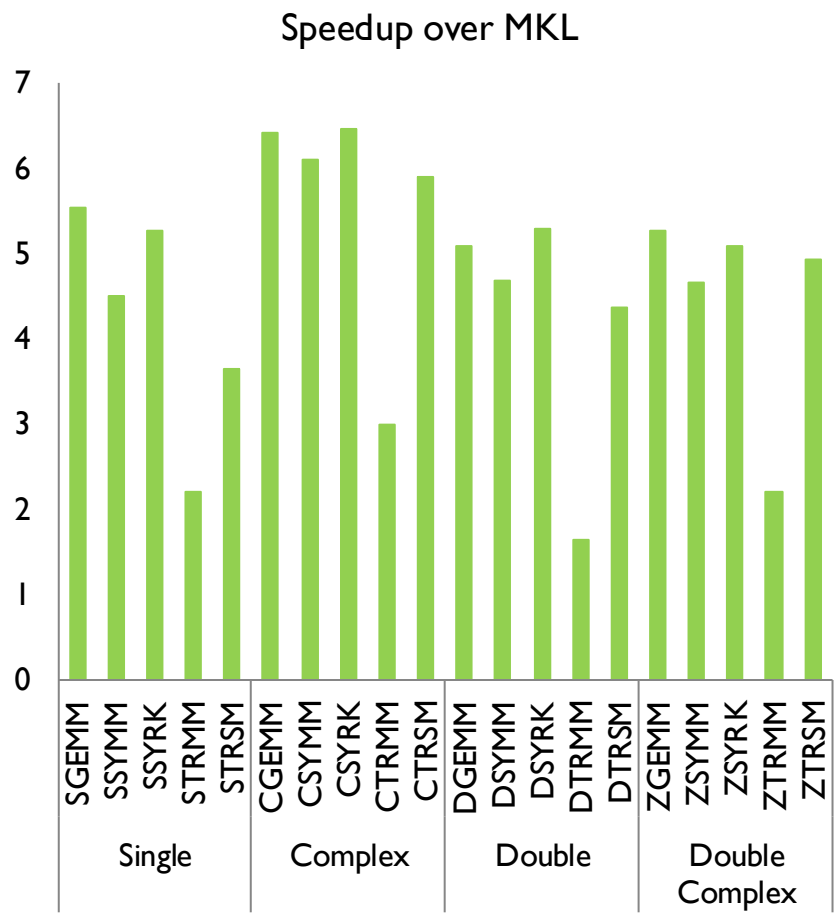
cuBLAS i cuSPARSE

- cuBLAS – rad sa gusto-posednutim matricama na GPU
- cuSPARSE – rad sa retko-posednutim matricama na GPU
- Standardni deo CUDA Toolkit
- Kompletna BLAS implementacija sa korisnim proširenjima
  - podržava152 standardne rutine
  - rad sa realnim brojevima u jednostrukoj i dvostrukoj preciznosti
  - rad sa kompleksnim brojevima u jednostrukoj i dvostrukoj preciznosti

cuBLAS Level 3 performanse

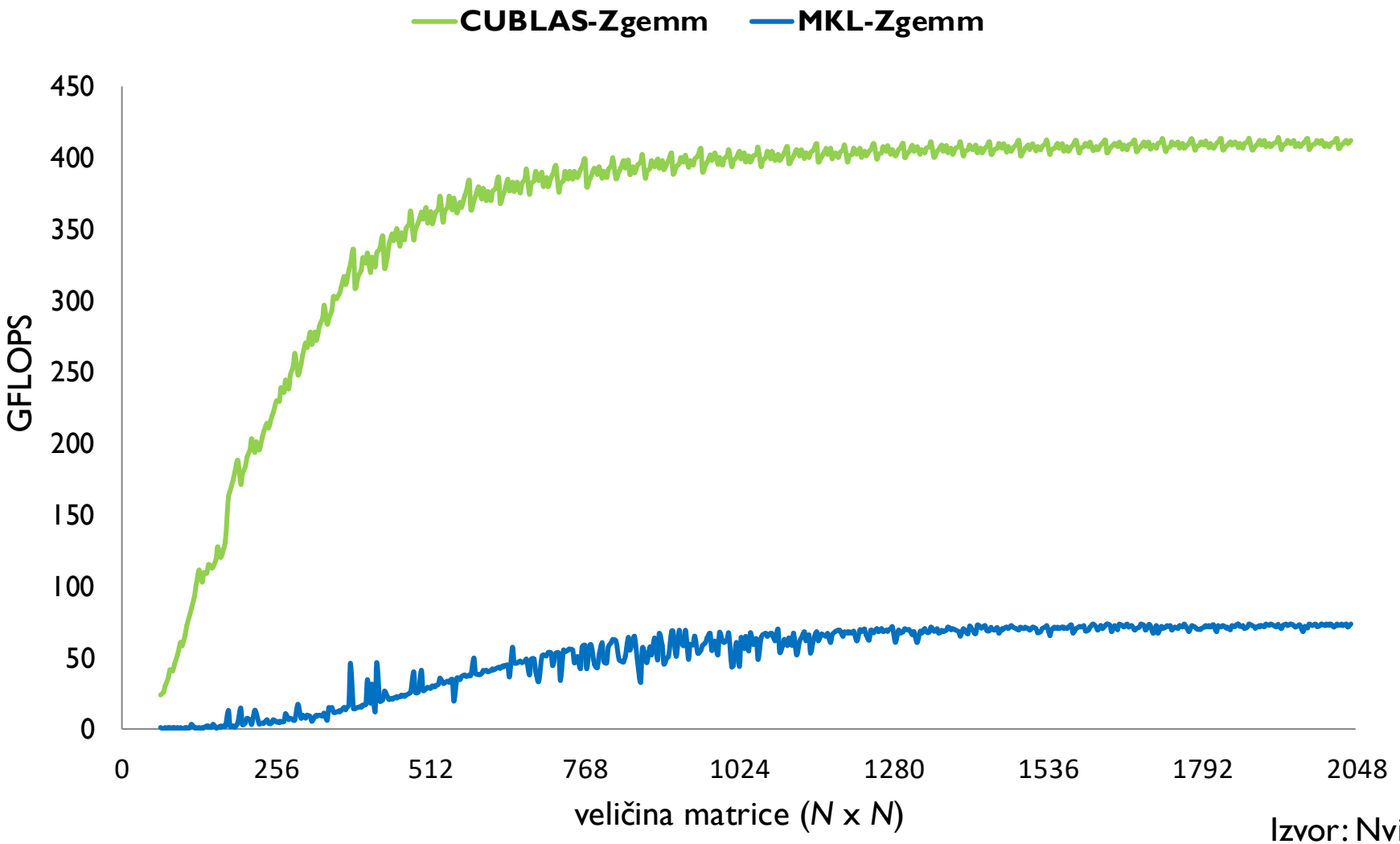
Do 1 TFLOPS održivih performansi i **6x** ubrzanje u odnosu na Intel MKL





Izvor: Nvidia

Performanse ZGEMM i Intel MKL



Izvor: Nvidia

cuSOLVER

- Biblioteka visokog nivoa napisana nad cuBLAS i cuSPARSE – skup funkcija za rad sa gusto i retko-posednutim matricama i sistemima
- cusolverDN: glavni LAPACK gusti rešavač (engl. *solver*)
  - dense Cholesky, LU, SVD, QR
  - primene: optimizacija, računarska vizija, CFD
- cusolverSP
  - retki direktni rešavač i Eigensolver
  - primene: Njutnov metod, hemijska kinetika
- cusolverRF
  - retki refaktorizacioni rešavač
  - primene: hemija, ODJ, simulacija kola

cuFFT: Multi-dimenzionalne FFT

- Dostupna počev od CUDA 4.1
  - Fleksibilni I/O rasporedi podataka za sve vrste transformacija
    - sličan FFTW "Advanced Interface"
    - eliminiše dodatno transponovanje i kopiranje podataka
  - API je thread-safe i može se pozivati od strane više niti domaćina

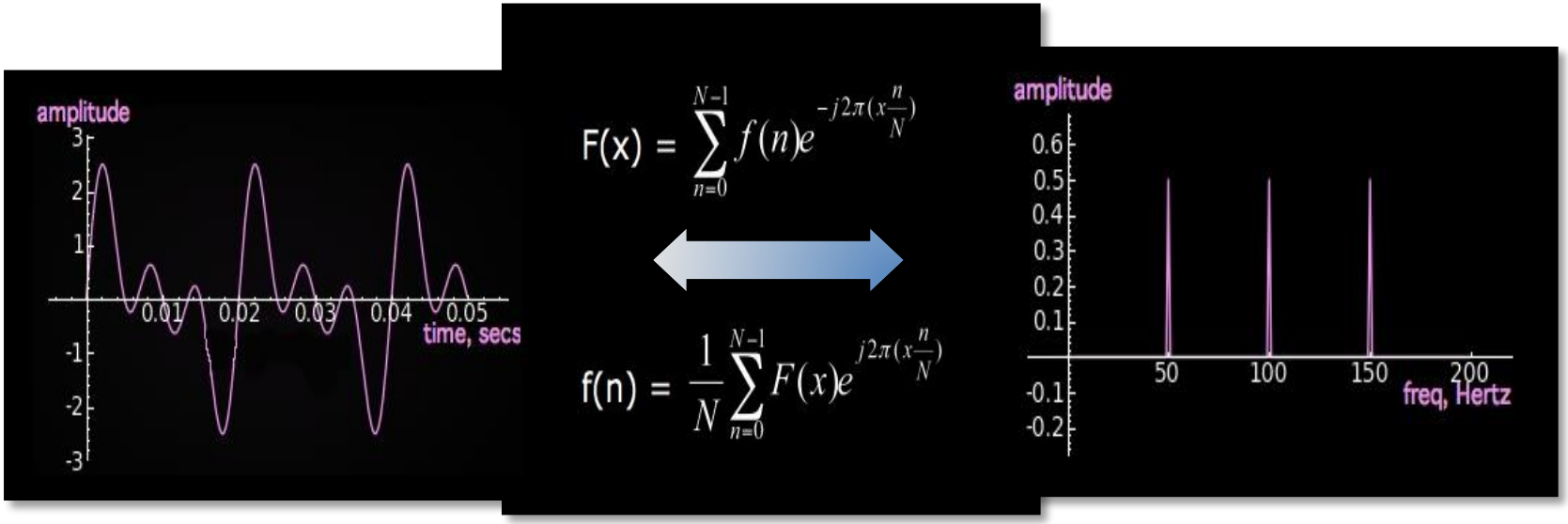
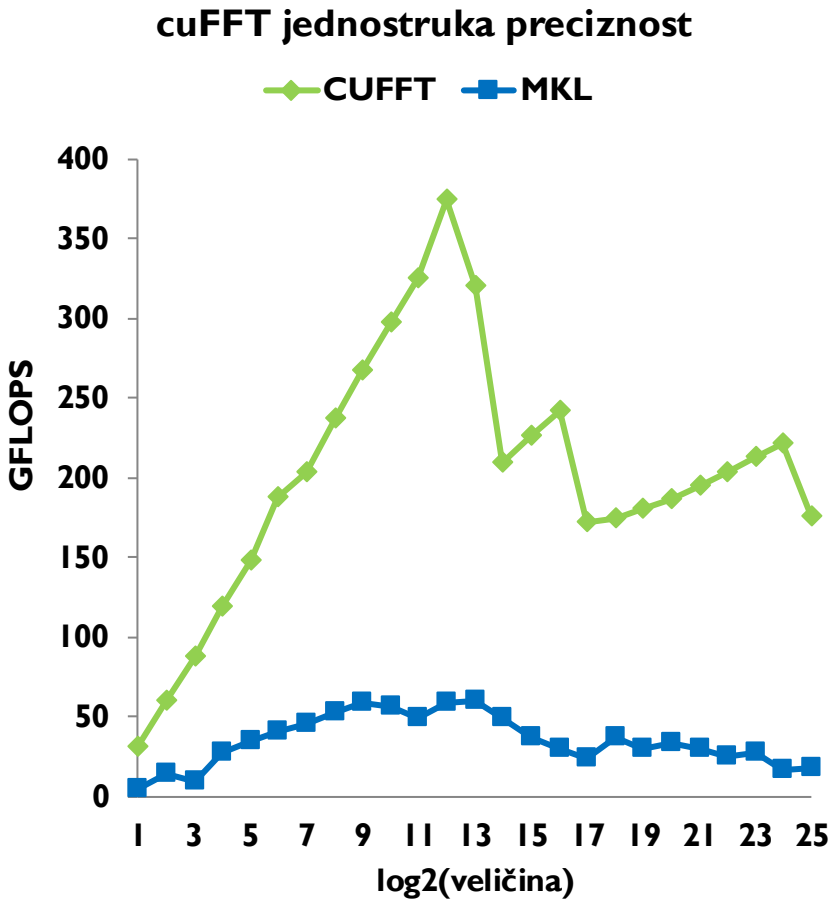
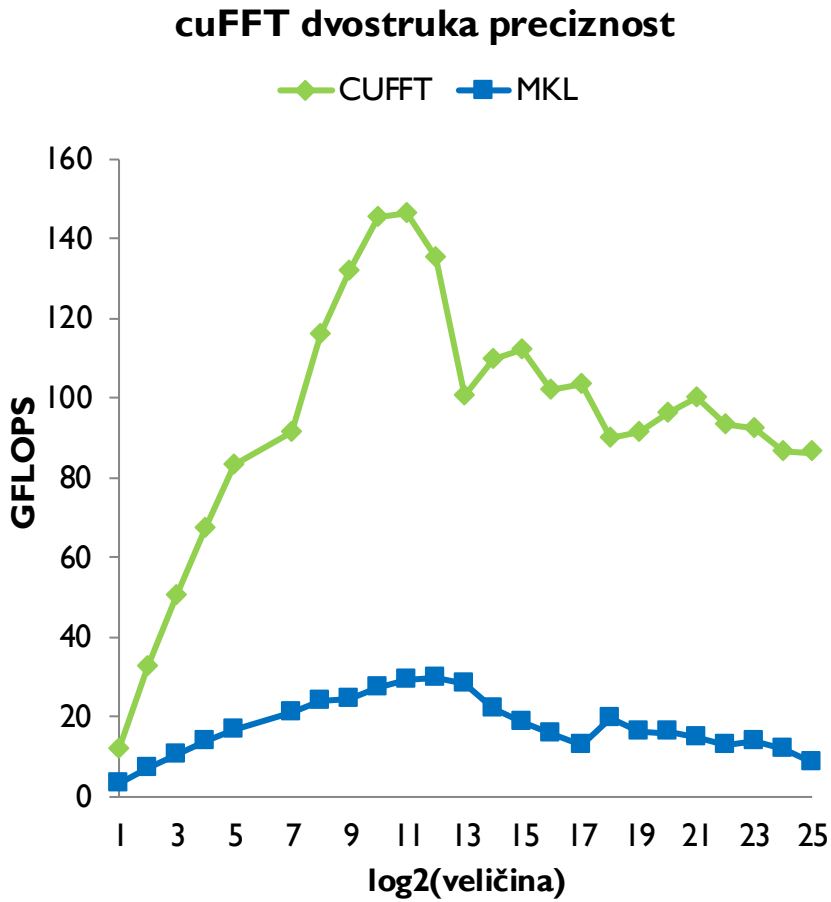


Image courtesy: Nvidia

FFT do 10x brži od MKL

1D se koristi u obradi signala i kao osnova za 2D i 3D FFT





Izvor: Nvidia

cuDNN

- CUDA biblioteka primitiva za duboke neuronske mreže
- cuDNN pruža visoko optimizovane implementacije za standardne rutine kao što su konvolucija unapred i unazad, prozivka, normalizacija i aktivacioni slojevi
- Deo je Nvidia Deep Learning SDK
- Koriste je radna okruženja kao što su Caffe, TensorFlow, Theano i Microsoft Cognitive Toolkit
- Izračunavanje konvolucije troši 80-90% ukupnog vremena obrade

Paralelno programiranje na GPU – OpenACC i OpenCL

OpenACC
Directives for Accelerators

OpenACC – Direktive

- OpenACC (www.openacc.org) je paralelni programski model zasnovan na direktivama, sa prenosivim performansama i usmeren ka korisniku
- Projektovan za prenos programa na heterogene HPC platforme sa značajno umanjenim programerskim naporom
- OpenACC specifikacija podržava C/C++ i Fortran, kao i više hardverskih arhitektura uključujući Intel x86 i Nvidia GPU
- OpenACC API pruža skup:
  - kompajlerskih direktiva (pragma)
  - bibliotečkih rutina
  - promenljivih okruženja

OpenACC – Direktive

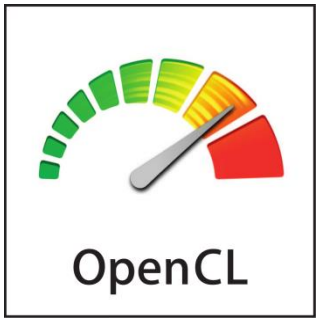
- OpenACC programiranje može početi sa sekvencijalnim programom
- Potom se dodaju OpenACC direktive
- Najveći deo detalja kod generisanja kernela i prenosa podataka prepušta se OpenACC prevodiocu
- OpenACC može se prevoditi putem ne-OpenACC prevodioca ignorisanjem pragma direktiva

OpenACC primer: matrično množenje

```
void computeAcc(float *P, const float *M, const float *N, int Mh, int Mw, int Nw)
{
   #pragma acc parallel loop copyin(M[0:Mh*Mw]) copyin(N[0:Nw*Mw]) copyout(P[0:Mh*Nw])
   for (int i=0; i<Mh; i++) {
      #pragma acc loop
       for (int j=0; j<Nw; j++) {
           float sum = 0;
           for (int k=0; k<Mw; k++) {
               float a = M[i*Mw+k];
               float b = N[k*Nw+j];
               sum += a*b;
           }
           P[i*Nw+j] = sum;
       }
    }
}
```

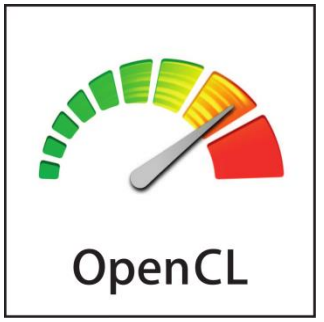
OpenACC
Directives for Accelerators

OpenCL



- Open Computing Language – OpenCL
- Originalno predstavljen 2009
- Standard razvijan od strane Khronos Group (Apple, Intel, AMD, Nvidia, Samsung...)
- Open source specifikacija standarda za razvoj heterogenih paralelnih aplikacija
  - Paralelni programi koji koriste skup različitih procesnih jedinica
  - Cilj je unificirati kako se paralelizam izražava, kako preneti izračunavanja na akceleratore kao što je GPU i održati prenosivost koda sa jedne platforme na drugu

OpenCL



- Prednosti u odnosu na CUDA:
  - nije vlasnička tehnologija
  - nije vezana isključivo za Nvidia GPU
- Nedostaci u odnosu na CUDA:
  - prenosivost, ali ne nužno i prenosivost performansi
  - CUDA je prilagođena Nvidia hardveru, tako da nudi više performanse
  - Nivo dostupnih programskih apstrakcija je nešto niži i pisanje i debagiranje programa je nešto složenije

Primer: sabiranje vektora – CUDA C

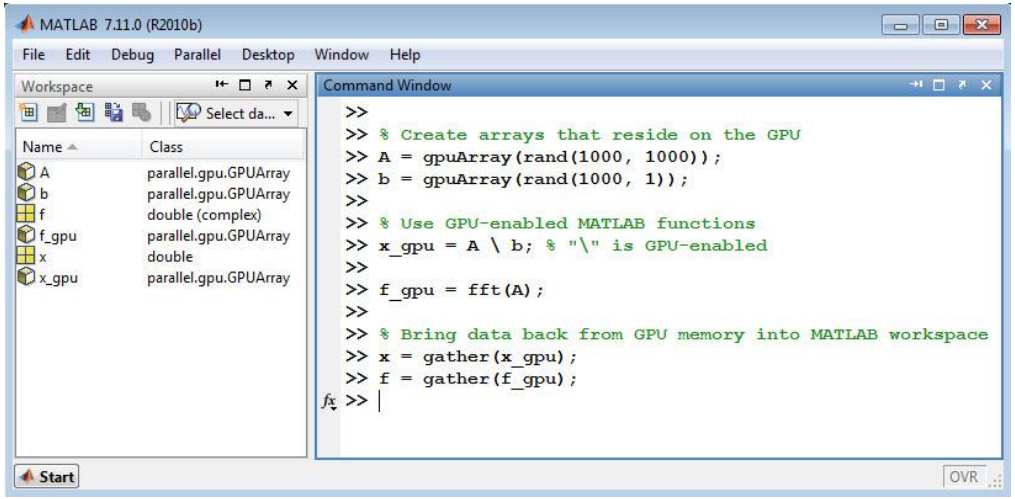
```
__global__ void addGPU(int* c,
                       const int* a,
                       const int* b)
{
    const unsigned int tid = threadIdx.x;
    c[tid] = a[tid] + b[tid];
}
```

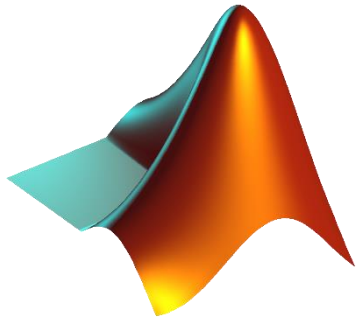
Primer: sabiranje vektora – OpenCL C

```
__kernel void addGPU(__global int* c,
                     __const int* a,
                     __const int* b)
{
    const unsigned int tid = get_global_id();
    c[tid] = a[tid] + b[tid];
}
```

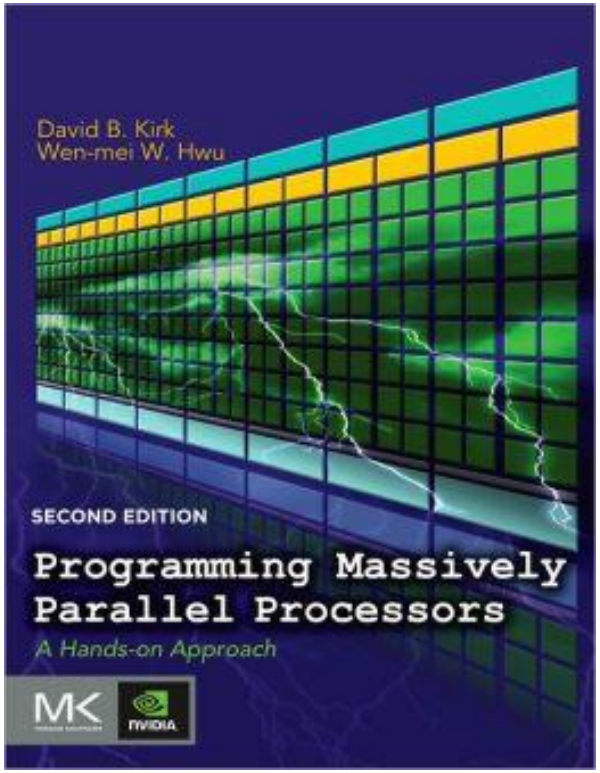
MATLAB Parallel Computing Toolbox

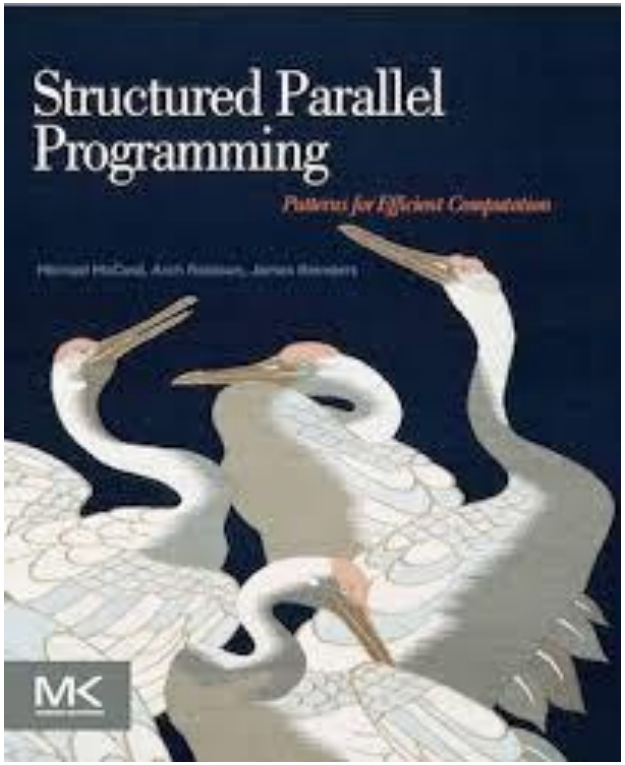
- Omogućava rešavanje složenih problema primenom višejezgarnih CPU, mnogojezgarnih GPU, kao i klastera, i to primenom programskih konstrukcija visokog nivoa - paralelnih for petlji, specijalnih tipova nizova i paralelizovanih numeričkih algoritama
- `gpuArray` za rad sa podacima koje obrađuje uređaj



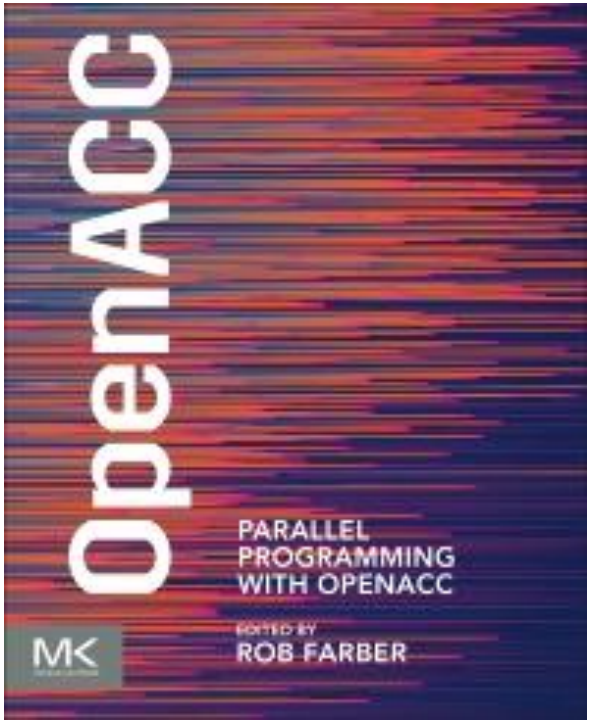


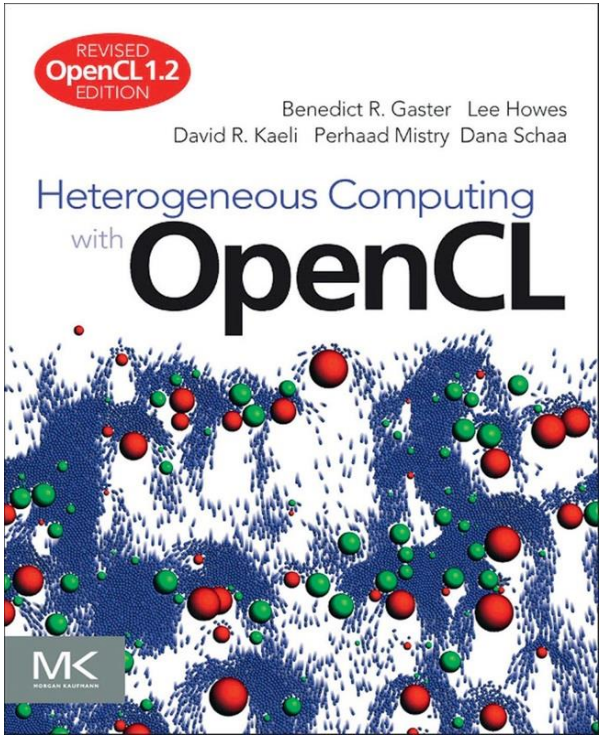
Literatura – optimizacija i paralelni obrasci





Literatura – OpenACC i OpenCL





Literatura – MATLAB i GPU Computing