A few examples as introduction to CUDA programming.

Each example directory contains a *Makefile* and **make run** should compile, link and run that code.
All Makefiles include specific compiler/environment variables via **include ../${COMPILER}default.mk** with COMPILER from {NVCC_, GCC_, CLANG_ ICC_}.
The chosen compiler can be set directly in the *Makefile* or more flexible as option in **make run COMPILER=NVCC_** .

firstSteps/

data_mv_RO.cu
original example by Ruetsch/Oster: Getting Started with CUDA (old but still a great introduction)
Just moving some data from CPU to GPU, increment on GPU, move result to CPU and check the result

data_mv_GH.cu
similar as above but in a more C++ style.

first_steps.cu
similar as above but with additional device functions (run on GPU)

first_unified.cu
similar as above using unified memory

matvec.cu
dense matrix-vector multiplication (CPU + several GPU versions)
comparison also with cuBLAS implementation (Basic Linear Algebra Subroutines on GPU).

Choose example by commenting/uncommenting appropriate line in *Makefile*.

skalar/

Inner product of two vectors calculated on GPU, see also older hints.
Scalar_{0-3}.cu are steps in developing the CUDA code following the excellent presentation by Mark Harris.

skalar_3_fast.cu
Final CUDA-code for inner product (without the complete unrolling by M. Harris)

skalar_3_fast_WrongTiming.cu
Demonstrates wrong timing for CUDA code (look for comment // **WT**)

skalar_4.cu
Uses cublasDdot

skalar_5.cu
Uses thrust library (STL for CUDA)

densematrices_libs/

Use of cuBLAS without the nvcc compiler from NVIDIA, i.e. compile your code with your favorite compiler and link the CUDA libraries additionally.

Comparison of Matrix*Matrix multiplication on

- CPU, one core (own code)
- CPU, all cores (own code)
- CPU BLAS, all cores
- GPU

for double and float data.

Some available targets in the Makefile:

run

clean

doc

info

cache

mem

prof