

# USING GRAPHICS PROCESSING UNITS IN HIGH PERFORMANCE COMPUTING

- Modern GPUs are high-performance many-core processors capable to deliver very high computation and data throughput, comparable with supercomputers available few years ago:  
e.g. **NVIDIA Tesla C1060 = 240 cores = 0.92 teraflops**
- Software delivered by GPU vendors makes general-purpose computation on GPUs easier than ever:  
**NVIDIA CUDA, OpenCL**
- GPU massively parallelism - thousands of "light" threads running in parallel on GPU

# Realisation of Conjugate Gradient method on graphics hardware (1)

- One of most popular iterative methods for solving linear equations systems
- Coefficient Matrix – sparse, symmetric and positive-definite
- Main operations of the method:
  - sparse matrix vector multiplication – *most time-consuming operation of the CG method; possible to implement in many ways; available in CUDPP library; main problem – irregular memory access pattern*
  - scalar-vector multiplication, vector addition – *ideally match up with GPU architecture*
  - dot product – *second time-consuming operation of the CG method; also available in CUDPP - based on ParallelReduction algorithm*

# Realisation of Conjugate Gradient method on graphics hardware (2)

- The key to efficient realisation of Conjugate Gradient method on GPU is efficient implementation of sparse matrix-vector multiplication ( $SPM \times V$ )
- The CUDPP library makes the GPU programming easier but not fully efficient
- It is possible to reduce execution time by implementing the  $SPM \times V$  operation from the beginning
- Appropriate allocation of GPU memory makes memory access significantly faster (*modified\_CSR*)

