GPU accelerated Quantum ESPRESSO

Pietro Bonfà
Department of Mathematical, Physical and Computer Sciences, University of Parma, Italy
14 Jan. 2020 - Jülich
Topics

What’s in this presentation:

✓ Evolution of **GPU acceleration** in QE: strategies and motivation.

✓ **Domain Specific Libraries**, that you may want to check for your own codes.

✓ A few details about the programming model used for QE.

✗ How to run QE on GPU accelerated platforms

✗ How to develop QE on GPU accelerated platforms
Acknowledgements

https://gitlab.com/QEF/q-e-gpu

MaX CoE

Mission: materials science codes ready for exascale computing.

www.flapw.de
What is QE

QUANTUM ESPRESSO
is an integrated suite of Open-Source computer codes for electronic-structure calculations and materials modeling at the nanoscale. It is based on density-functional theory, plane waves, and pseudopotentials.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry’s phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.
- Phonons for 2D heterostructures (reference).

Spectroscopic properties:
- K-, L1 and L2,3-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurboTDDFT package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Quantum Transport:
- Ballistic Transport ( PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Quantum Transport:
- Ballistic Transport (PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, B3LYP);

Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.

Spectroscopic properties:
- K- and L1,2,3-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurbQDFT package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Quantum Transport:
- Ballistic Transport (PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.
- Phonons for 2D heterostructures (reference)

Spectroscopic properties:
- \( \text{K}^- \), \( \text{L}_1 \) and \( \text{L}_2,3 \)-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurboDFTD package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Quantum Transport:
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Ab-initio molecular dynamics
- Ballistic Transport ( PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.
- Phonons for 2D heterostructures (reference)

Spectroscopic properties:
- $K_-$, $L_1$ and $L_2,3$-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurboTDDFT package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Quantum Transport:
- Ballistic Transport ( PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code.
- Kubo-Greenwood electrical conductivity using the KGEC code.
## What is QE

### Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

### Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

### Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

### Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

### Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.

### Quantum Transport:
- Ballistic Transport (PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
-Effective charge and dielectric tensor.

Spectroscopic properties:
- $K^-, L_1$ and $L_{2,3}$-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurboTDDFT package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Quantum transport:
- Ballistic Transport (PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the Want code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code;
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- VdW corrections (DFT-D) or nonlocal VdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Structural Optimization:
- GDIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.

Quantum Transport:
- Ballistic Transport (PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the WanT code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code.
- Kubo-Greenwood electrical conductivity using the KGEC code.
What is QE

Ground-state calculations:
- Self-consistent total energies, forces, stresses;
- Kohn-Sham orbitals;
- Separable norm-conserving and ultrasoft (Vanderbilt) pseudo-potentials, PAW (Projector Augmented Waves);
- Several exchange-correlation functionals: from LDA to generalized-gradient corrections (PW91, PBE, B88-P86, BLYP) to meta-GGA, exact exchange (HF) and hybrid functionals (PBE0, B3LYP, HSE);
- vdW corrections (DFT-D) or nonlocal vdW functionals (vdw-DF);
- Hubbard U (DFT+U);
- Berry's phase polarization;
- Spin-orbit coupling and noncollinear magnetism.

Structural Optimization:
- GDIIS with quasi-Newton BFGS preconditioning;
- Damped dynamics.

Transition states and minimum energy paths
- Nudged Elastic Band method;
- Meta-Dynamics, using the PLUMED plug-in.

Ab-initio molecular dynamics
- Car-Parrinello Molecular Dynamics (CP package);
- Born-Oppenheimer Molecular Dynamics (PWscf package).

Response properties (density-functional perturbation theory):
- Phonon frequencies and eigenvectors at any wavevector;
- Full phonon dispersions; inter-atomic force constants in real space;
- Translational and rotational acoustic sum rules;
- Effective charges and dielectric tensors;
- Electron-phonon interactions;
- Third-order anharmonic phonon lifetimes, using the D3Q package;
- Infrared and (non-resonant) Raman cross-sections;
- EPR and NMR chemical shifts, using the QE-GIPAW package.
- Phonons for 2D heterostructures (reference).

Spectroscopic properties:
- \(K\)-, \(L_1\) and \(L_2,3\)-edge X-ray Absorption Spectra (XSpectra package);
- Time-Dependent Density Functional Perturbation Theory (TurboTDFFT package);
- Electronic excitations with Many-Body Perturbation Theory, using the YAMBO package;
- Electronic excitations with Many-Body Perturbation Theory (GWL package).

Quantum Transport:
- Ballistic Transport ( PWCOND package);
- Coherent Transport from Maximally Localized Wannier Functions, using the WanT code;
- Maximally-localized Wannier functions and transport properties, using the WANNIER90 code.
- Kubo-Greenwood electrical conductivity using the KGEC code.

...and several post processing tools by many research groups!
What is QE

- QUANTUM ESPRESSO is an initiative coordinated by the QUANTUM ESPRESSO Foundation, with the participation of SISSA, CINECA, ICTP, EPFL and many partners in Europe and worldwide.

- QUANTUM ESPRESSO is not a single application for quantum simulations; it is rather a distribution of packages performing different tasks and meant to be interoperable.

- Free as in GPLv2 and open development.
What is QE

- Runs from standalone workstation to massively parallel systems.
  
  ```
  ./configure && make all
  ```

- Large scientific user base, vehicle for new methods, new theories and new science.
  - 600k lines of Fortran
  - V6.5 -> 10k downloads
  - >50 contributors
  - 1600+ registered users
  - ...

- Simplify transition of new science to HPC environment.
What is QE
What is QE

Some of the time consuming workloads of many packages are already encapsulated in a number of libraries, namely

LAXLib  FFTXlib  KS_Solvers

FFTW, MKL, ESSL, ...
Profiling

PWscf (CPU version) running on a single KNL node with 64 MPI processes (best time to solution).

- Serial Code - $4.23e+04$ sec: 71.8%
- OpenMP - 0 sec: 0%
- MPI calls - $1.56e+04$ sec: 28.1%
Profiling

*PWscf* (CPU version) running on a single KNL node with 64 MPI processes (best time to solution).

Matrix multiplication size

- Serial Code: $4.23 \times 10^4$ sec, 71.8%
- OpenMP: 0 sec, 0%
- MPI calls: $1.56 \times 10^4$ sec, 28.1%
Profiling

**PWscf** (CPU version) running on a single KNL node with 64 MPI processes (best time to solution).

Not a single "portion" of the code takes the majority of the wall-time.

Depending on the input, 3D-FFT dominant or LA dominant.
Past and present QE GPU ports

Porting effort carried out by MaX and supported by NVIDIA.

CUDA C based plugin for QE 5.x (pw.x) developed by F. Spiga and I. Girotto.

Independent CUDA Fortran based port of QE 6.1 (pw.x) developed by F. Spiga and NVIDIA. Provides much better performance, limited features implemented.
QE-GPU-Plugin

✓ ✓ Self contained

phiGEMM: a CPU-GPU library for porting Quantum ESPRESSO on hybrid systems

Filippo Spiga, Ivan Girotto
Irish Centre for High-End Computing (ICHEC), Dublin, Ireland

20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2012

DOI: 10.1109/PDP.2012.72
QE-GPU-Plugin

✓✓ Self contained
✓ Good performance
QE-GPU-Plugin

✓ ✓ Self contained
✓ Good performance
QE-GPU-Plugin

✓ ✓ Self contained
✓ Good performance
✗ Boilerplate code
Porting strategy

Desiderata:

- Fortran: CUDA Fortran / OpenACC / OpenMP $\geq 4.5$.
- Explicit memory management:
  - derived types
  - generic hybrid architecture support.
- Preserve modularity.
- Preserve user experience.

- Prepare full featured and well tested CUDA-enabled libraries containing performance critical kernels.
- Directive based acceleration of QE's applications.
Porting strategy

\[ A\psi = \lambda B\psi \]

**LAXLib**
- Task: parallel linear algebra
- Libs: ELPA, MKL, cuBLAS, cuSOLVER, ESSL, ...

**FFTXlib**
- Task: parallel distributed FFT
- Libs: FFTW, MKL, ESSL, cuFFT, ...

**KS_Solvers**
- Task: iterative solvers
- Libs: LAXLib, MKL, cuBLAS, ...

**Programming model**
- Directive based programming

**Objective**
- Optimize memory duplication, allocation, and synchronization.
- Optimize computational efficiency and concurrency.
CUDA Fortran

😊 Fortran equivalent for CUDA C++

😊 Syntax is similar to CUDA, but more concise.

😊 Full set of libraries and interfaces.

😢 Complete syntax only on PGI compilers.

😢 Partial implementation on IBM compilers, useless for QE.
CUDA Fortran

attributes(global) subroutine increment(a, b)
    implicit none
    integer, intent(inout) :: a(:)
    integer, value :: b
    integer :: i, n
    i = blockDim%x*(blockIdx%x-1) + threadIdx%x
    n = size(a)
    if (i <= n) a(i) = a(i)+b
end subroutine increment

CUDA kernels in Fortran

call vaddkernel <<<(N+31)/32,32 >>> (A,B,C,N)

cuda equivalent syntax

Full support for Fortran intrinsic types.
CUDA Fortran

Allocation done by the host, according to “device” attribute

Just copy (no need for cuda APIs for sync. copies).

```fortran
module mm
    real, device, allocatable :: a(:)
    real, allocatable :: b(:)
    attributes(device) :: b

    real, device, allocatable :: a(:,:), c
    allocate( a(1:n,1:m), STAT=ivar )
    ! CHECK ivar
    allocate(c)
    ... 
    deallocate( a, c )
end module mm

contains
    attributes(global) subroutine s( b )
end subroutine s
```
CUDA Fortran

Allocation done by the host, according to “device” attribute

Just copy (no need for cuda APIs for sync. copies).
CUDA Fortran

Fortran interfaces for most (all?!?) NVIDIA Cuda Runtime and NVIDIA Libraries.
CUDA Fortran

Cuf kernels, directive based automatic kernel generation:

```
program incTest
  use cudafor
  implicit none
  integer, parameter :: n = 256
  integer :: a(n), b
  integer, device :: a_d(n)
  a = 1
  b = 3
  a_d = a
  !$cuf kernel do <<<*,*>>>
  do i = 1, n
    a_d(i) = a_d(i)+b
  enddo
  a = a_d
  if (all(a == 4)) write(*,*) 'Test Passed'
end program incTest
```
Cuf kernels, a few simple rules:

- Scalars are private by default
- Reduction automatically detected (only scalars)
- Only loop based constructs (possibly nested)
QE-GPU CUDA Fortran

✓ Single programming language: Fortran + CUDA Fortran
QE-GPU CUDA Fortran

✓ Single programming language: Fortran
QE-GPU CUDA Fortran

✓ Single programming language: Fortran + CUDA Fortran

✓ Very good performance (shown at the end of this presentation).

✗ Code duplication:
  • Performance (eg. preserve cache blocking optimizations)
  • Preserve both CPU and GPU version.
Is it that different?

OpenACC

CUDA Fortran kernel

! Is acc kernels loop present(qmod(ny), ylmk0(ny, lmaxq * lmaxq), qg, qrad, lp, lpl, ap) & ! Is acc num_workers(256) collapse(1) if(on_device)
! do ig = 1, ngy
!    !
!    qg(ig) = (0.0d0, 0.0d0)
!    qm = qmod(ig) * dqi
!    px = qm - int(qm)
!    ux = 1.0d0 - px
!    vx = 2.0d0 - px
!    wx = 3.0d0 - px
!    i0 = int(qm) + 1
!    i1 = i0 + 1
!    i2 = i0 + 2
!    i3 = i0 + 3
!    uxv = ux * vx * sixth
!    pwv = px * wx * 0.5d0
!   ! find angular momentum l corresponding to combined index lp
!   ! (l is actually l+1 because this is the way grad is stored, check init_us_1)
!   !
!   if (lp == 1) then
!     l  = 1
!     sig = CMPLX(1.0d0, 0.0d0)
!   elseif (lp <= 4) then
!     l  = 2
!     sig = CMPLX(-1.0d0, 0.0d0)
!   elseif (lp <= 9) then
!     l  = 3
!     sig = CMPLX(0.0d0, 0.0d0)
!   elseif (lp <= 16) then
!     l  = 4
!     sig = CMPLX(-1.0d0, 0.0d0)
!   elseif (lp <= 25) then
!     l  = 5
!     sig = CMPLX(1.0d0, 0.0d0)
!   elseif (lp <= 36) then
!     l  = 6
!     sig = CMPLX(0.0d0, -1.0d0)
!   else
!     sig = CMPLX(-1.0d0, 0.0d0)
!   endif
! work = qrad(i0, i1, l, np) * uxv * wx + &
! qrad(i1, i2, l, np) * pwv * vx + &
! qrad(i2, i3, l, np) * pwv * ux + &
! qrad(i3, i1, l, np) * px * uxv
! qg(ig) = qg(ig) * sig * CMPLX(ap, lp, lpl, jv) * ylmk0(ig, lp, work, 0.0d0, kind=DP)
! enddo

ig = threadIdxx+BlockSize*(threadIdxx+1)
if (ig <= ngy) then
    ! compute the indices which correspond to ih,jh
    dqy = 1.0_dp / dq
    qg(ig) = 0.0d0
    qm = qmod(ig) * dqi
    px = qm - int(qm)
    ux = 1.0d0 - px
    vx = 2.0d0 - px
    wx = 3.0d0 - px
    i0 = int(qm) + 1
    i1 = i0 + 1
    i2 = i0 + 2
    i3 = i0 + 3
    uxv = ux * vx * sixth
    pwv = px * wx * 0.5d0
    do lm = 1, lpl (ivl, jvl)
        lp = lp (ivl, jvl, lm)
        if (lp == 1) then
            l  = 1
            sig = CMPLX(1.0d0, 0.0d0, kind=DP)
        elseif (lp <= 4) then
            l  = 2
            sig = CMPLX(0.0d0, -1.0d0, kind=DP)
        elseif (lp <= 9) then
            l  = 3
            sig = CMPLX(1.0d0, 0.0d0, kind=DP)
        elseif (lp <= 16) then
            l  = 4
            sig = CMPLX(-1.0d0, 0.0d0, kind=DP)
        elseif (lp <= 25) then
            l  = 5
            sig = CMPLX(0.0d0, -1.0d0, kind=DP)
        elseif (lp <= 36) then
            l  = 6
            sig = CMPLX(-1.0d0, 0.0d0, kind=DP)
        else
            l  = 7
            sig = CMPLX(-1.0d0, 0.0d0, kind=DP)
        endif
        work = qrad(i0, i1, l, np) * uxv * wx + &
        qrad(i1, i2, l, np) * pwv * vx + &
        qrad(i2, i3, l, np) * pwv * ux + &
        qrad(i3, i1, l, np) * px * uxv
        qg(ig) = qg(ig) * sig * CMPLX((ylmk0 (ig, lp) * work + ap (lp, lpl, jv), np)))
Is it that different?
Libraries

LAXLib

FFTXLib

devXLib

ELPA

FFTW, MKL, ESSL, ...

CUDA

OpenMP
QE Libraries

- Full API support:

```fortran
IF( use_tg ) THEN
  CALL invfft ('tgWave', tg_psic_d, dfts )
  CALL tg_get_group_nr3( dfts, right_nr3 )
END IF
```

- Unit testing:

```shell
[master@node153 tests]$ mpirun -np 1 ./test_fft_scalar_gpu.x
fortran_tester: 0 error(s) for 32 test(s)
fortran_tester: all tests succeeded
```

```shell
[master@node153 tests]$ mpirun -np 4 ./test_fft_scatter_mod_gpu.x
fortran_tester: 51 error(s) for 224 test(s)
fortran_tester: tests failed
```
LAXLib

Solution of **dense eigenvalue problem** of real or complex hermitian matrices.

Both **serial** and **distributed parallel** implementation.

Extract **all eigenpairs** or a **subset**.
QE Libraries

LAXLib

Initially custom code now part of CUDA 10.1.

Available at
https://gitlab.com/max-centre/components
QE Libraries

**FFTXlib**

Sparse FFT in reciprocal space.
Parallel, distributed, accelerated.
Both pencil and slab decomposition.

Available at https://gitlab.com/max-centre/components
The local potential contribution is computed more efficiently in real space:

\[
\psi_{ik}(G) \xrightarrow{FFT} \psi_{ik}(r) \\
[v_{KS}\psi_{ik}](r) = v_{KS}(r)\psi_{ik}(r) \\
[v_{KS}\psi_{ik}](r) \xrightarrow{FFT} [v_{KS}\psi_{ik}](G)
\]
QE Libraries

- For each band, FFT to real space, multiplication, FFT to reciprocal space.
  \[ \rightarrow \text{Many independent small 3D FFTs} \quad (10^1 \rightarrow 10^3) \]

Images from E. Pascolo, M. Sc. thesis
QE Libraries

For each band, FFT to real space, multiplication, FFT to reciprocal space.
→ Many independent small 3D FFTs ($10^1 \rightarrow 10^3$)
QE Libraries

- Many small 3D FFTs ($10^1 \rightarrow 10^3$)
- Overlap of communication and computation
- Batched work

![Bar chart showing FFTW time comparison between no batch and batched work for 1 node and 2 nodes.](chart.png)

- 4 bands 1D FFT
- Scatter
- 4 bands 1D FFT
- Scatter
- Alltoall
- 4 bands 2D FFT
- Alltoall
- 4 bands 2D FFT
QE Libraries

- QE allocates many small auxiliary workspaces. This impacts substantially the performances of the accelerated version of the code.
- Optimize memory allocation: GPU memory is limited.

```
USE buffer_module, ONLY : gpu_buffer
!
implicit none
!
REAL, POINTER :: work(:)
gpu_buffer%lock_buffer(work, 10, ierr)
[...]
gpu_buffer%release_buffer(work, ierr)
```

**devXlib**

Available at https://gitlab.com/max-centre/components
QE Codes

- GPU acceleration currently available for **PWscf**. CP and PHonon planned.
## Current status and evolution

**Application:** *pw.x*

<table>
<thead>
<tr>
<th>GPU version</th>
<th>Total Energy (K points)</th>
<th>Forces</th>
<th>Stress</th>
<th>Collinear Magnetism</th>
<th>Non-collinear magnetism</th>
<th>Gamma trick</th>
<th>US PP &amp; PAW</th>
<th>EXX</th>
<th>DFT+U</th>
<th>All other functionalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>v5.4</td>
<td>A</td>
<td>W</td>
<td>W</td>
<td>B (?)</td>
<td>U</td>
<td>A</td>
<td>A</td>
<td>?</td>
<td>W (?)</td>
<td>W (?)</td>
</tr>
<tr>
<td>v6.1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>U</td>
<td>W (*)</td>
<td>A</td>
<td>U</td>
<td>U</td>
<td>U (*)</td>
</tr>
<tr>
<td>v6.3</td>
<td>A</td>
<td>W</td>
<td>W</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>V6.5</td>
<td>A</td>
<td>A</td>
<td>W</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>W</td>
</tr>
</tbody>
</table>

**Accelerated, Working, Unavailable, Broken**
Benchmarks

**Marconi @ CINECA**
Model: Xeon E5-2697 v4 (BDW) @ 2.30 GHz
Cores: 2x18 = **36**
RAM: 128 GB/node

**Q3 2016**
1.3 TFLOPs

**Galileo @ CINECA**
Model: Xeon E5-2630 v3 (HSW) @ 2.40 GHz
Cores: 2x8 = **16**
Accelerators: **2 x K80**
RAM: 128 GB/node

**Q1 2015**
0.6 + 2x2.9 TFLOPs

**Piz Daint XC50 @ CSCS:**
Model: Xeon E5-2690 v3 (HSW) @ 2.60 GHz
Cores: 1x12 = **12**
Accelerators: **1 x P100**
RAM: 64 GB/node

**Q4 2016**
0.5 + 4.7 TFLOPs
**Benchmarks**

Best time to solution obtained with *pw.x* v6.2, with and without GPU support, and with the GPU port of v6.1 done by NVIDIA.

**MnSi, bulk, ferromagnetic.**
64 atoms, 365 KS states, USPP.

**Piz Daint XC50 @ CSCS:**
Processors: 12-cores Intel Haswell 2.60 GHz
Accelerators: 1 NVIDIA P100
RAM: 64 GB/node

**Galileo @ CINECA**
Processors: 2*8-cores Intel Haswell 2.40 GHz
Accelerators: 2 NVIDIA K80
RAM: 128 GB/node
Benchmarks

Best time to solution obtained with \textit{pw.x} v6.3, with and without GPU acceleration, and with the GPU port of \textit{pw.x} v6.1 done by NVIDIA.

Ta$_2$O$_5$

Large test case, 2D, 26 k-points, 96 atoms, 326 KS states.
Benchmarks

Best time to solution obtained with $pw.x$ v6.3, with and without GPU acceleration, and with the GPU port of $pw.x$ v6.1 done by NVIDIA.
Benchmarks

Best time to solution obtained with \textit{pw.x} v6.3, with and without GPU acceleration, and with the GPU port of \textit{pw.x} v6.1 done by NVIDIA.
Benchmarks

Best time to solution obtained with \textit{pw.x} v6.4, with and without GPU acceleration.

Very large benchmark:
\textbf{AuCONH} benchmark: 586 atoms, 1531 KS

\textbf{~ 18x speedup}


diagram:
- CPU – XL
- CPU – GNU
- GPU

<table>
<thead>
<tr>
<th>Components</th>
<th>Processor</th>
<th>Count</th>
<th>Peak FLOPS</th>
<th>Peak AI FLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>9,216</td>
<td>9.96 PF</td>
<td>3.456 EF</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>27,648</td>
<td>215.7 PF</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>POWER9</td>
<td>2 x 18 x 256</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V100</td>
<td>6 x 18 x 256</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Courtesy of Dr. Ye Luo

\textit{pw.x} v6.4
HT Benchmarks

New Galileo @ CINECA
Model: Xeon E5-2697 v4 (BDW) @ 2.30 GHz
Cores: 1x18; RAM: 128 GB/node

Random structures from COD:
- Magnetic / Non Magnetic
- Metal / Insulator
- Efficiency / Accuracy
- K-points grid: Very fine / Fine / Normal / Gamma
HT Benchmarks

QE v6.5 best time to solution
Porting other QE applications

- **Preserve all functionalities**
  - Feature testing already available...enough?
  - More feature and unit tests
  - Create verification scheme

- **Preserve accelerated function modularity**
  - For debugging
  - For code maintainability
  - For simpler development

- **Directive based**
Lessons learnt

- Separation of concerns simplifies subsequent application porting
  - Reduces branches in the code.
  - Cleaner and simpler adoption of accelerated functions in new code.

- Modularization helps (duplicated) data management
  - Porting pushes modularization forward

- CUDA Fortran helped removing boilerplate.

- (Moving to GIT helped a lot)
Remaining challenges

- HW availability limits GPU programming:
  - Policy for contributions?
  - Ratio between CPU and GPU efforts?

- Different hybrid solutions about to appear.

- Is OpenMP the final answer?

- Source code duplication.
Remaining challenges

- HW availability limits GPU programming:
  - Policy for contributions?
  - Ratio between CPU and GPU efforts?

- Different hybrid solutions about to appear.

- Is OpenMP the final answer?

- Source code duplication.

Suggestions?
Thanks for your attention!