Using Microsoft Visual Studio 2008 for OpenMP Programming

Christian Terboven
terboven@rz.rwth-aachen.de

Center for Computing and Communication
RWTH Aachen University

WinHP³C

PPCES 2009
March 24th, RWTH Aachen University

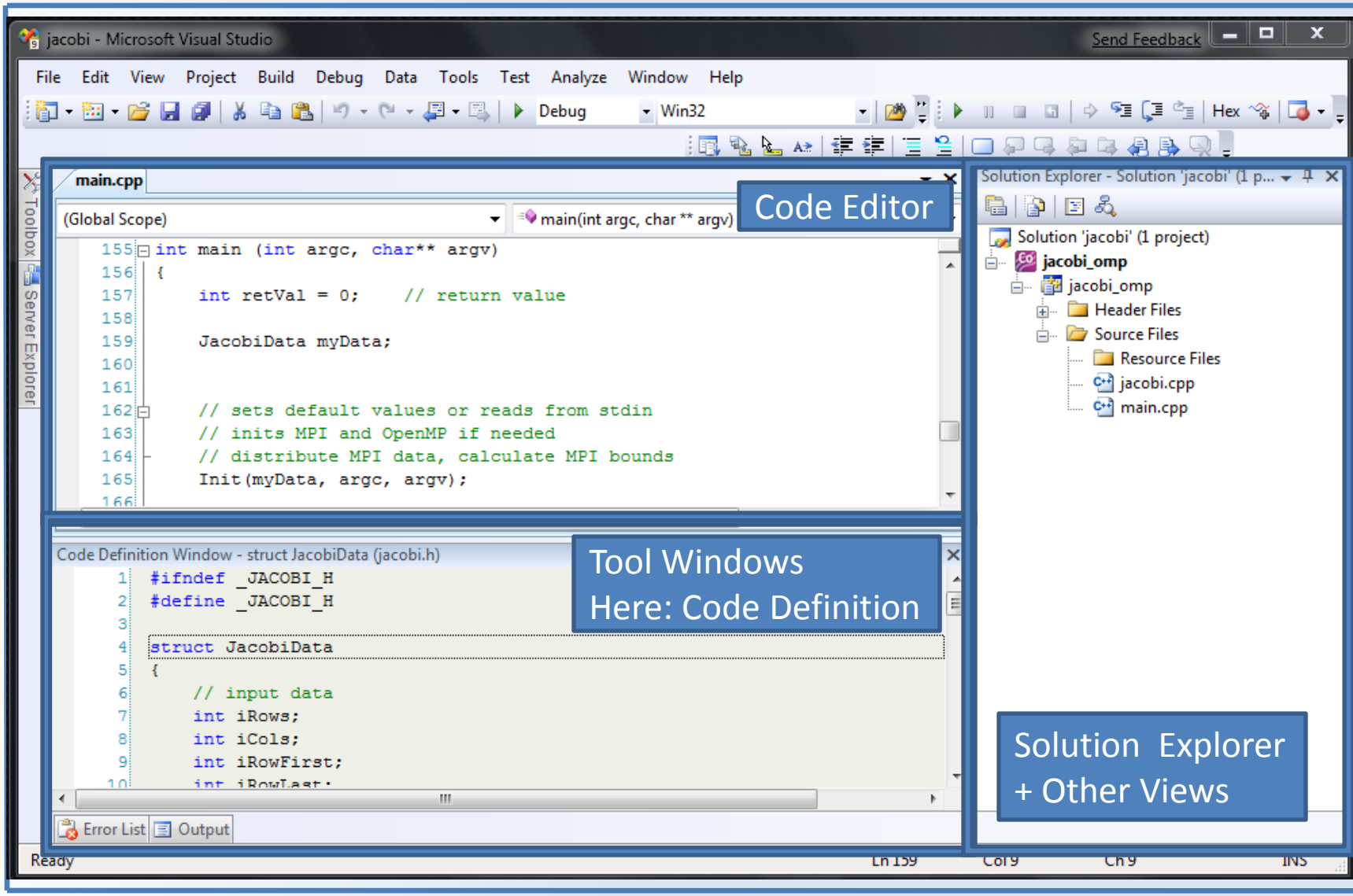
Agenda

- Project Management
- The Microsoft C/C++ compiler
- The Intel C/C++ and Fortran compiler
- Using OpenMP
- Demo

VS2008 Overview (1/2)

- Introduction into using Visual Studio 2008
  - Only text-mode programs are considered here, as HPC applications typically do not use GUIs
  - VS2008 offers great support for GUI development on Windows
- VS2008 provides good support for Parallel Programming:
  - Support for OpenMP for Shared-Memory parallel compilation
  - Debugging of parallel programs: OpenMP and MPI
  - Architecture-specific compiler optimizations
- The first start of VS2008 may take a few minutes as the help system has to be updated. You are prompted to choose from several pre-defined sets of VS2008 configuration options: I typically choose the C++ settings.

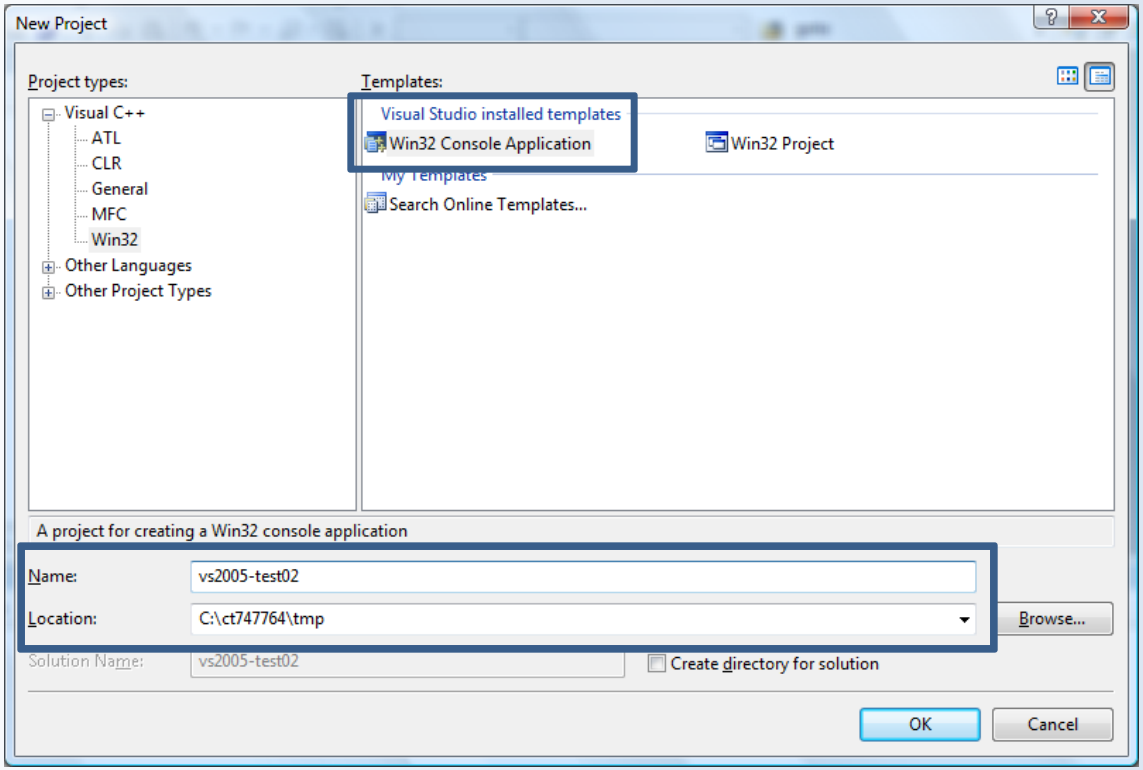
VS2008 Overview (2/2)



Visual Studio: Project Management (1/5)

- Everything that you do in Visual Studio will take place within the context of a *Solution*.
  - A Solution is a higher-level container for other items, for example a *Project*. Any other kind of file type can also be added to a Solution, for example documentation items.
  - A Solution can not contain another Solution.
  - Solutions group and apply properties across projects.
- A *Project* maps one to one with a compiler target.
  - A Project organizes the code.
- To start your work, a new Project has to be created with *File → New → Project…*

Visual Studio: Project Management (2/5)



Visual C++ → Win32 → Win32 Console Application

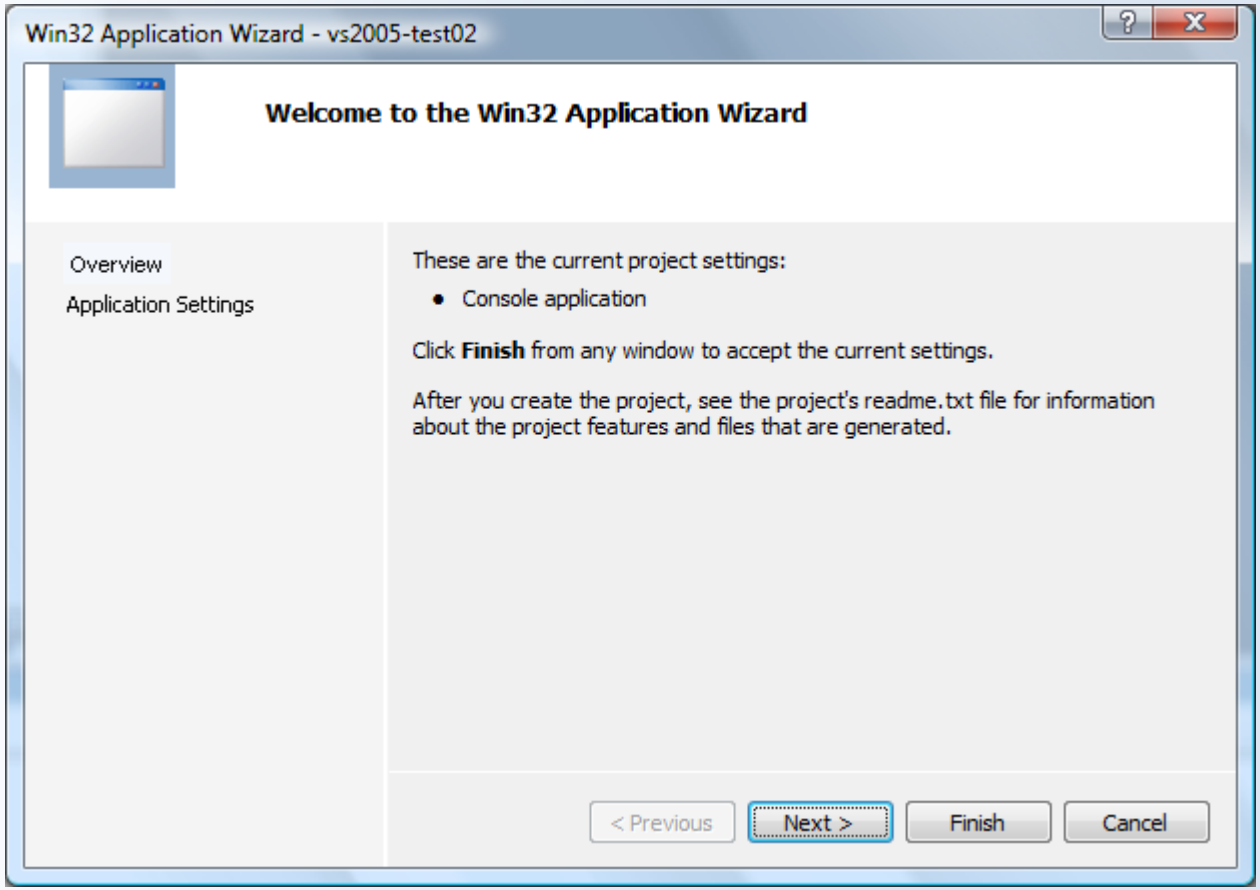
Name of the Project, A Solution of the same name is created as well

Location where the folder structure is created

Win32 Console Application:
No GUI design support of Visual Studio enabled.

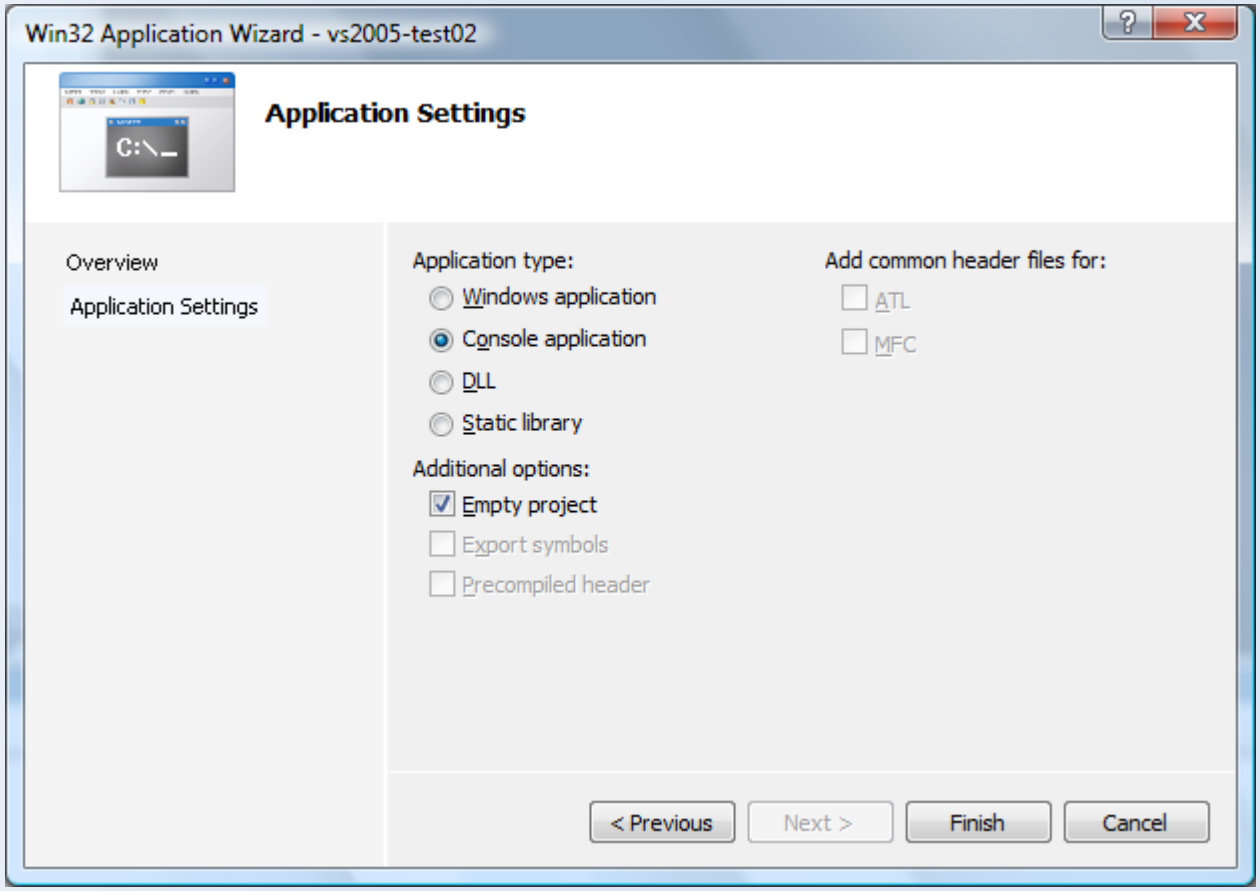
Visual Studio: Project Management (3/5)

- One might like or dislike wizards …

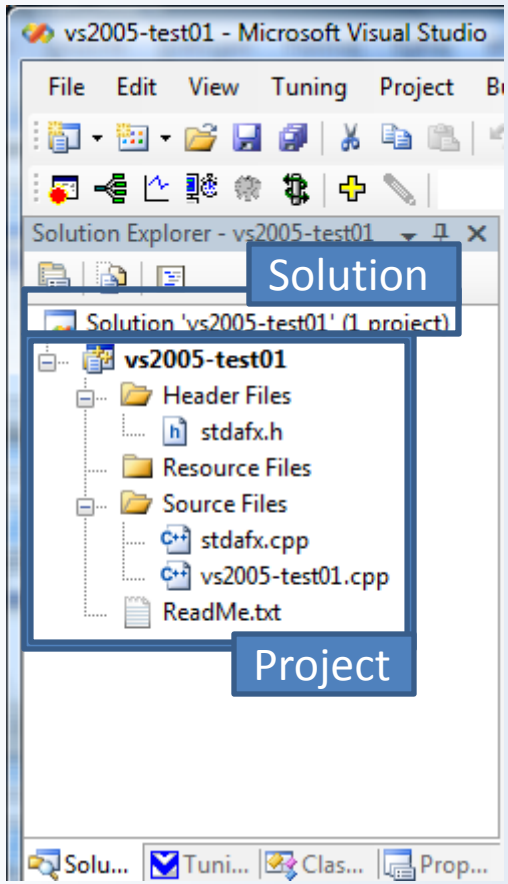


Visual Studio: Project Management (4/5)

- Choose *Empty project* if you already have source files.
- Make sure that *Precompiled header* is de-selected if you select *Empty project*.



Visual Studio: Project Management (5/5)



- In many cases, the shortest way to a desired operation can be found by right-clicking on a GUI element, using the context menu.
- Adding existing source code items (files) to a project: right-click on the Project (not the Solution !) and *Add → Existing Item…*
- Adding new items: right-click on the Project and *Add → New Item…*
- The folders (e.g. *Source Files*) do not have any other meaning than aiding you in structuring the files in a project. They do not map to physical folders. You can create your own folders.

Directory layout of Visual Studio solutions

- The executable is created in the directory of the active configuration during the build process.

- Directory structure of a solution:

<top level>	Given user directory
<project name>	Created by VS2005 / VS2008
Debug	Configuration: *Debug*
Release	Configuration: *Release*
x64	Platform: *x64* (64bit for Amd64/Intel64)
Debug	Configuration: *Debug*
Release	Configuration: *Release*

Agenda

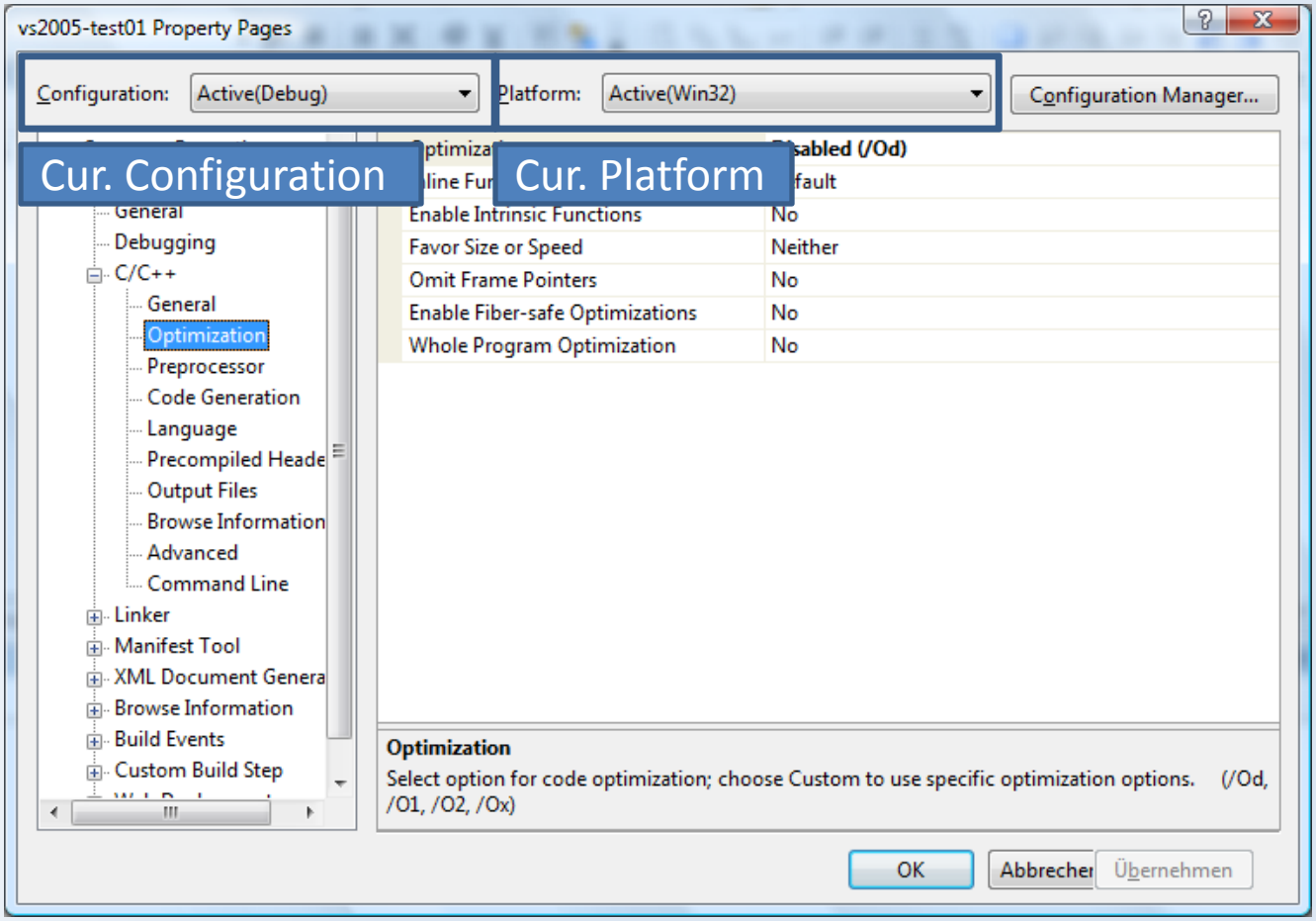
- Project Management
- The Microsoft C/C++ compiler
- The Intel C/C++ and Fortran compiler
- Using OpenMP
- Demo

Visual Studio Configurations (1/3)

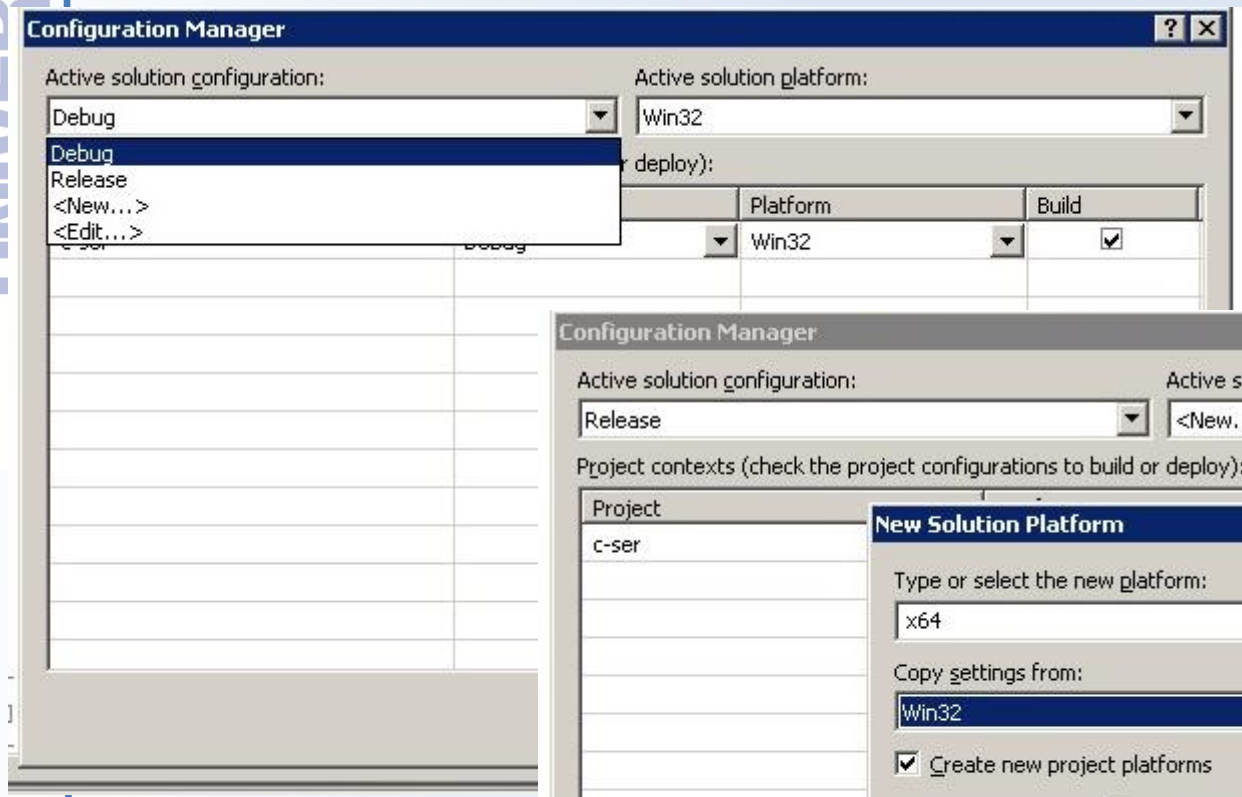
- The set of compiler options is managed in a *Configuration*.
- There are two configurations pre-defined: *Debug* and *Release*.
  - Debug: typical options for debugging, no optimization.
  - Release: debugging still possible, some optimization options.
- The compile process can be triggered by right-clicking on the project and choosing *Build*. Or from the menu: *Build → Build <projectname>*.
- *Build → Build Solution* builds all projects in the solution.
- During and after the compile process compiler output (informational messages, warnings, errors) is displayed in the tool window.
- By double-clicking on such a message, the cursor jumps to the corresponding place in the code.

Visual Studio Configurations (2/3)

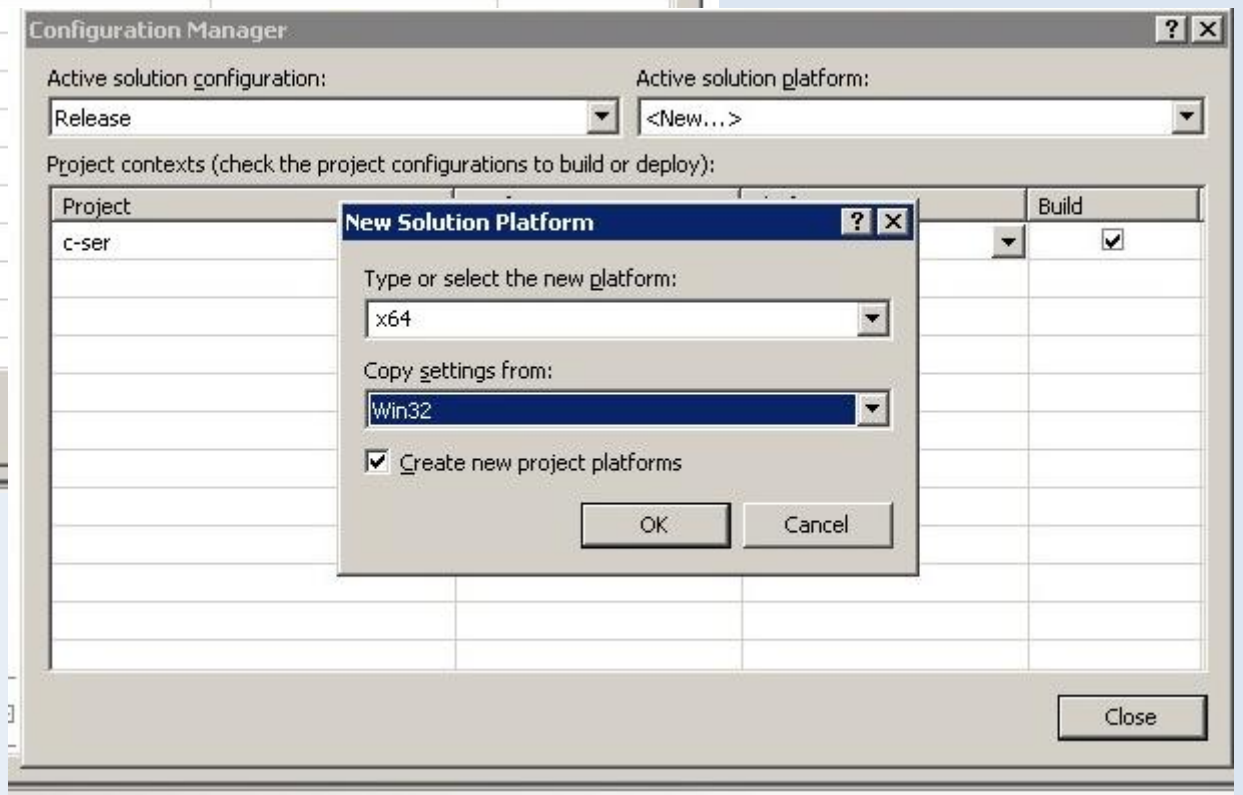
- Right-clicking on a project and choosing Properties leads to the project configuration dialog.



Visual Studio Configurations (3/3)

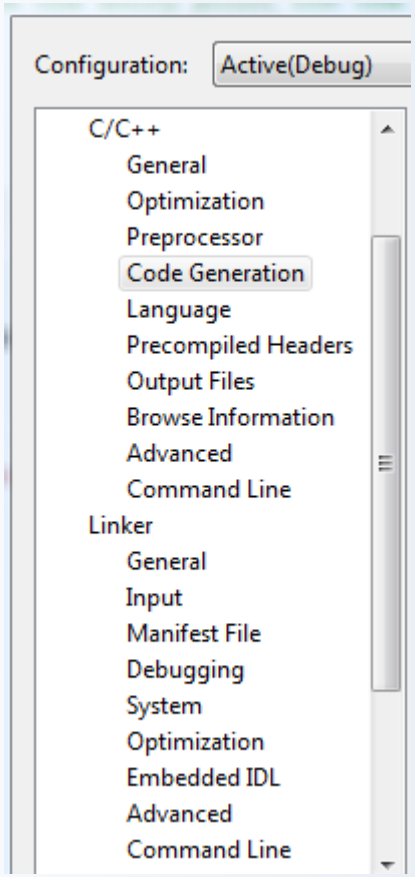


- Only Win32 and x64 are supported, not Itanium.



- You can create your own configurations.

Microsoft C/C++-specific settings



- Important General Settings:
  - C/C++ → General
    - Addition Include Directories: Include Path
  - Linker → General
    - Additional Library Directories: Library Path
  - Linker → Input
    - Additional Dependencies: Libraries to be used
- Important Optimization Settings:
  - C/C++ → Optimization
    - Optimization: General Optimization Level
    - Inline Function Expansion: Inlining
  - C/C++ → Code Generation
    - Enable Enhanced Instruction Set: Vectorization

Agenda

- Project Management
- The Microsoft C/C++ compiler
- The Intel C/C++ and Fortran compiler
- Using OpenMP
- Demo

Using the Intel C/C++ compiler

1. Create a (Win32) project using the Microsoft C/C++ compiler
2. Right-click on the Solution or on the Project
3. Intel Parallel Studio Installed (`cluster-win-beta` + `-lab`): Intel Parallel Composer → Use Intel C++
4. Intel Parallel Studio Not Installed (`cluster-win`): Use Intel(R) C++

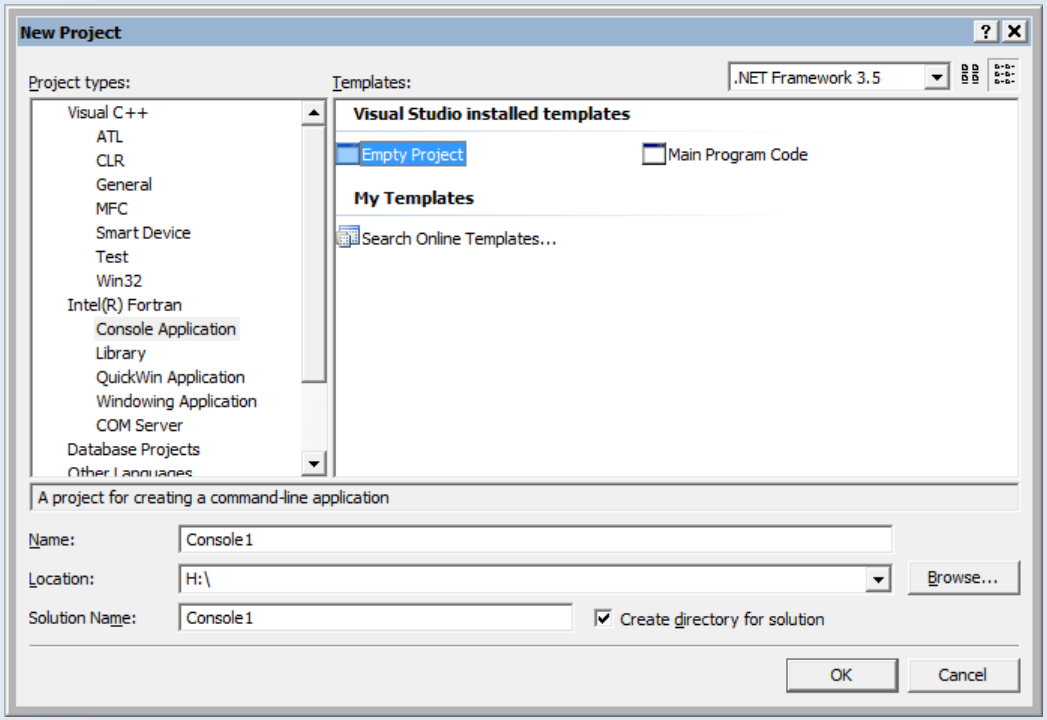
- Solution or Project can be converted back to use Microsoft C/C++ as well.

Intel C/C++-specific settings

- Most settings are the same as with Microsoft C/C++, but:
- Additional Optimization Settings:
  - C/C++ → Optimization
    - Generate Alternate Code Path and/or Use Intel(R) Processor Extensions: Optimization for specific CPU
    - Parallelization: Ask the compiler for automatic parallelization
  - C/C++ → Language
    - Recognize the Restrict Keyword: Enable C99 `restrict` (Tuning!)
    - Enable C++0x Support: Enable first C++0x features
  - Add to C/C++ → Command Line and Linker → Command Line:
    - /Qtcheck: Enable source instrumentation for Thread Checker
    - /Qtprofile: Enable source instrumentation for Thread Profiler

Using the Intel Fortran compiler

1. Open the Project Creation dialog via *File → New → Project…*
2. Select *Intel(R) Fortran* as project type



3. Typically select *Empty Project* as best-suited option

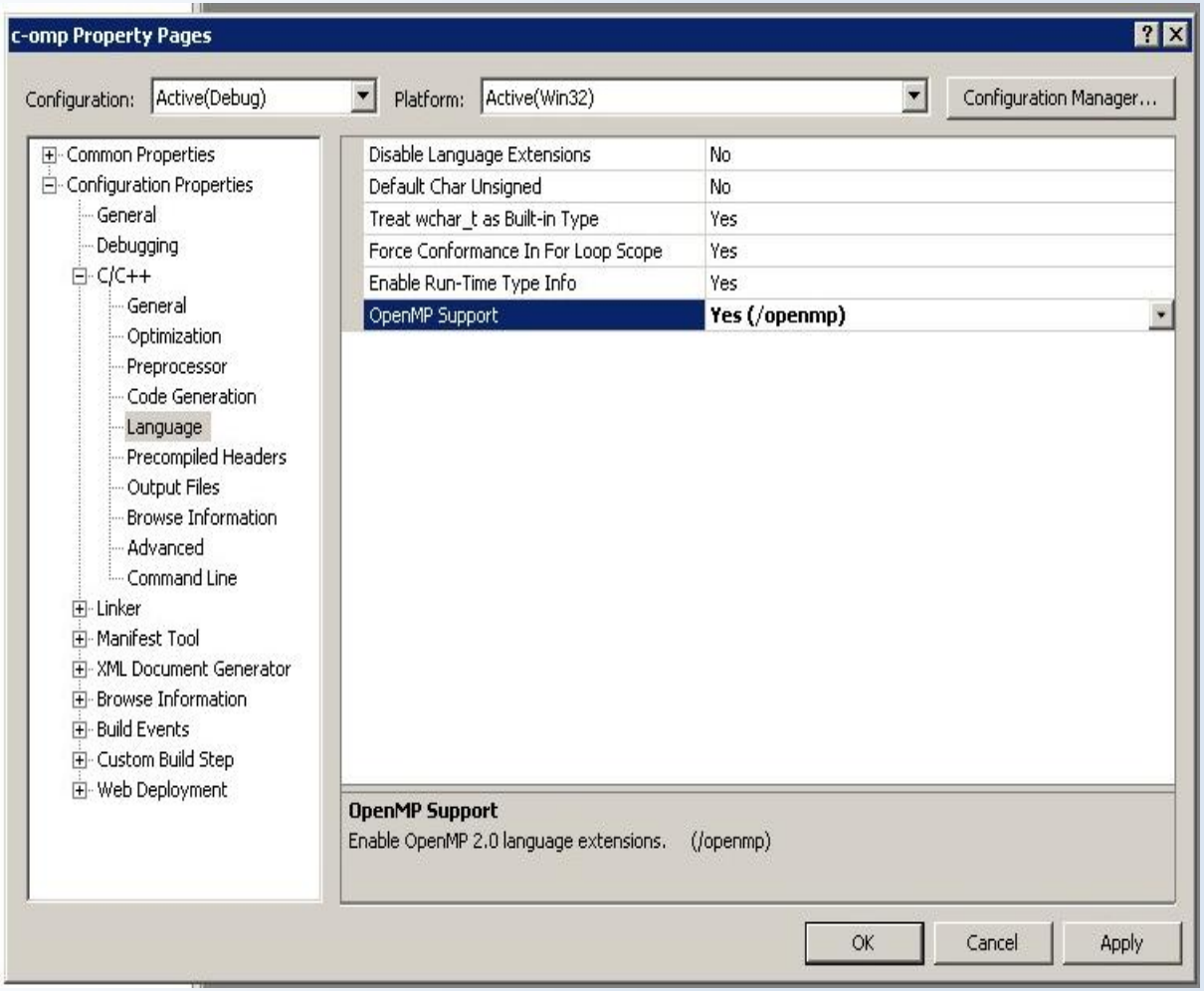
- Remaining project handling does not differ from C/C++!

Agenda

- Project Management
- The Microsoft C/C++ compiler
- The Intel C/C++ and Fortran compiler
- Using OpenMP
- Demo

Enabling OpenMP (1/3)

- OpenMP support has to be enabled in a configuration:



OpenMP-capable compilers:

- VS2005 C/C++
- VS2008 C/C++
- Intel C/C++
- Intel FORTRAN

Enabling OpenMP (2/3)

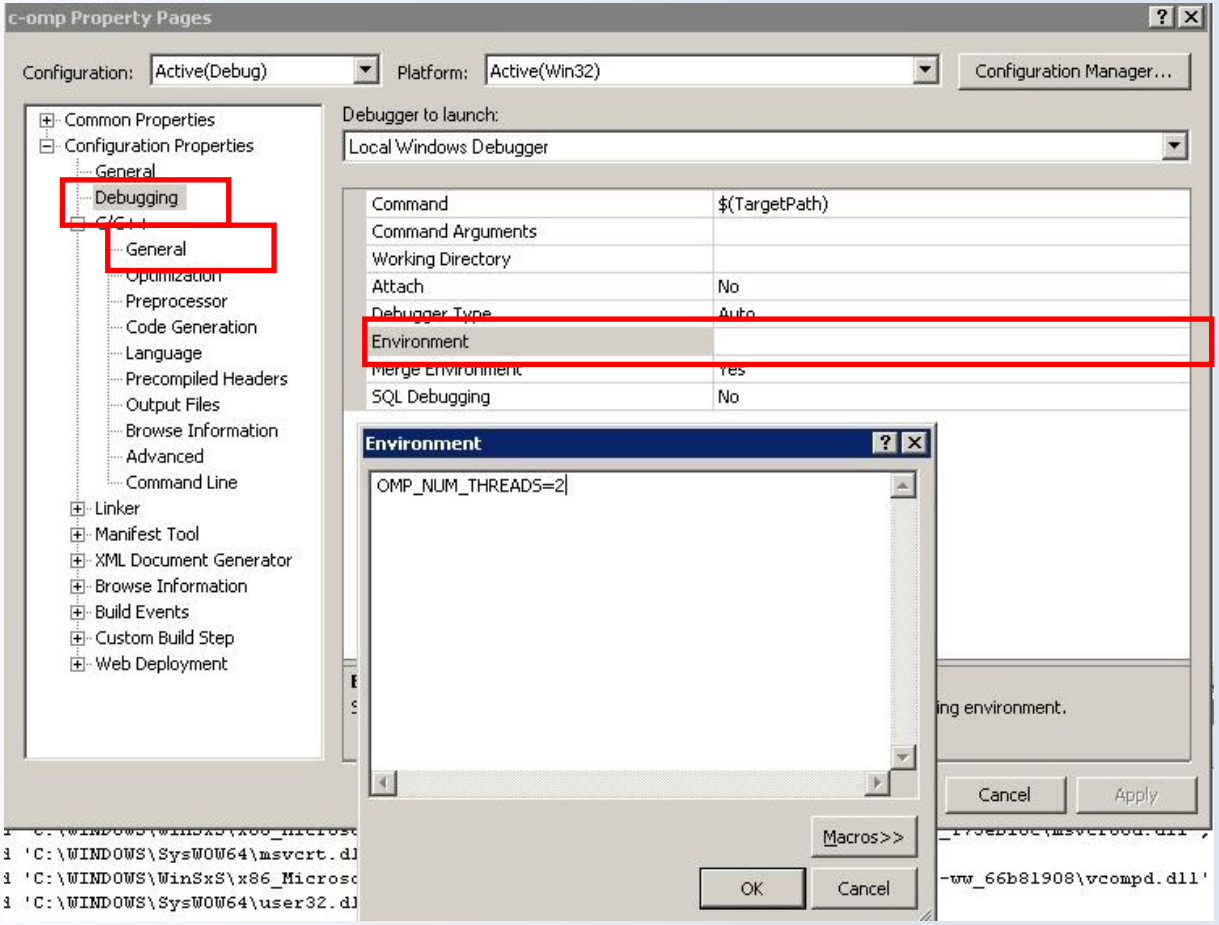
- Known problem with Visual Studio 2005 / 2008 and OpenMP:

- The message appears if an OpenMP program has been compiled with OpenMP support enabled, but `omp.h` had not been included.

- Solution: include `omp.h` in at least one file per project.

Enabling OpenMP (3/3)

- Setting the number of threads for debugging of OpenMP programs: set environment variable `OMP_NUM_THREADS`.



Agenda

- Project Management
- The Microsoft C/C++ compiler
- The Intel C/C++ and Fortran compiler
- Using OpenMP
- Demo

Demo

Jacobi-C++-omp

The End

Thank you for your attention!

Questions?