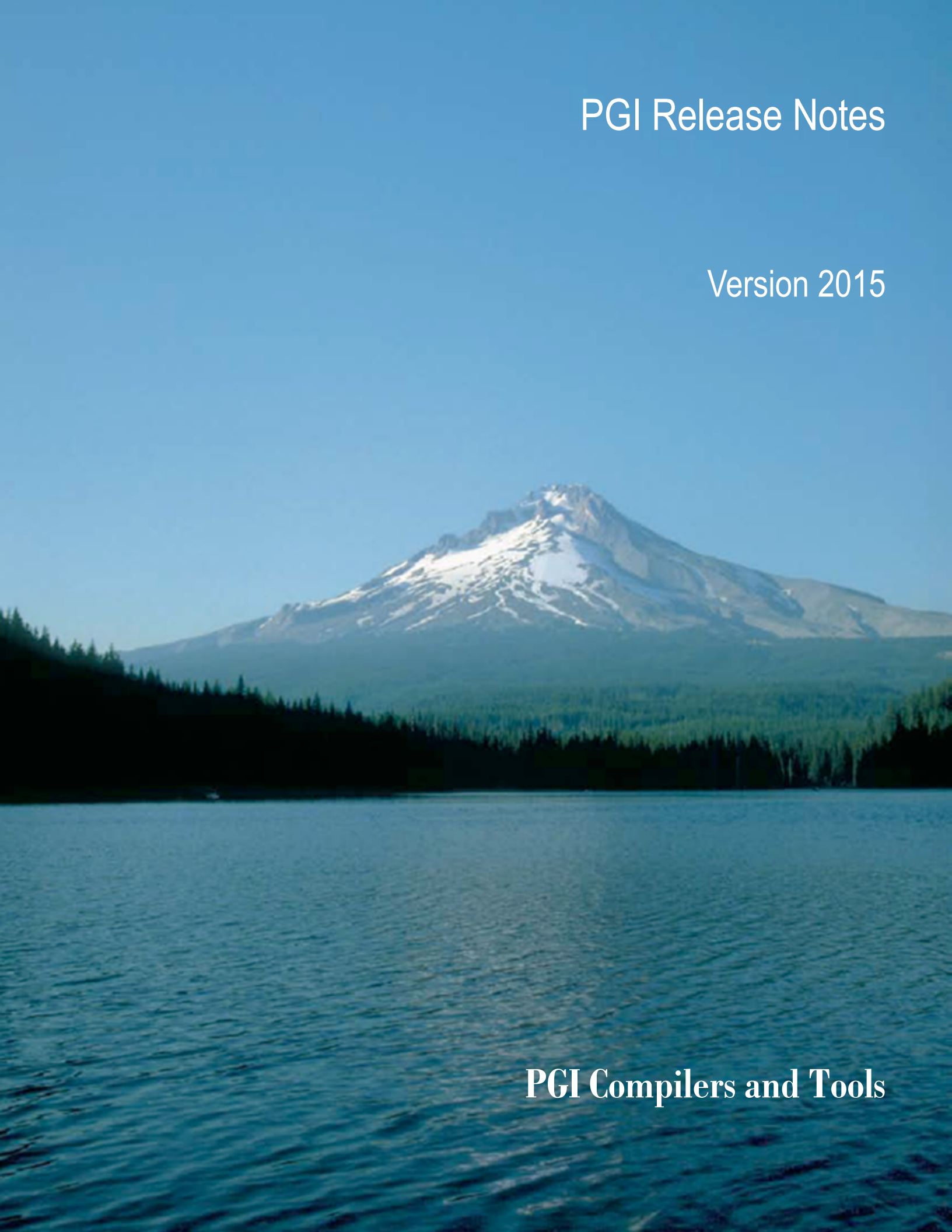
PGI Release Notes

Version 2015



PGI Compilers and Tools

TABLE OF CONTENTS

LIST OF TABLES

Chapter 1. RELEASE OVERVIEW

Welcome to Release 2015 of PGI Workstation™, PGI Server™, and the PGI CDK® Cluster Development Kit®, a set of compilers and development tools for 32-bit and 64-bit x86-compatible processor-based workstations, servers, and clusters running versions of the Linux operating system. PGI Workstation and PGI Server are also available for the Apple OS X and Microsoft Windows operating systems.

This document describes changes between previous releases of the PGI compilers and tools as well as late-breaking information not included in the current version of the PGI Compiler User's Guide.

1.1. Product Overview

PGI Workstation, PGI Server, and the PGI CDK include exactly the same PGI compiler and tools software. The difference is the manner in which the license keys enable the software.

1.1.1. Licensing Terminology

The PGI compilers and tools are license-managed. Before discussing licensing, it is useful to have common terminology.

- **License** - a legal agreement between NVIDIA and PGI end-users, to which users assent upon installation of any PGI product. The terms of the License are kept up-to-date in documents on pgroup.com and in the $PGI/<platform>/<rel_number> directory of every PGI software installation.
- **License keys** - ASCII text strings that enable use of the PGI software and are intended to enforce the terms of the License. License keys are generated by each PGI end-user on pgroup.com using a unique hostid and are typically stored in a file called `license.dat` that is accessible to the systems for which the PGI software is licensed.
- **PIN** - Personal Identification Number, a unique 6-digit number associated with a license. This PIN is included in your PGI order confirmation. The PIN can also be found in your PGI license file after **`VENDOR_STRING=`**.
- **License PIN code** - A unique 16-digit number associated with each PIN that enables users to "tie" that PIN to their pgroup.com user account. This code is provided by PIN owners to others whom they wish tied to their PIN(s).

1.1.2. Licensing Options

PGI offers licenses for either x64+accelerator or x64 only platforms. PGI Accelerator™ products, the x64+accelerator platform products, include support for the directive-based OpenACC programming model, CUDA Fortran and PGI CUDA-x86. PGI Accelerator compilers are supported on all Intel and AMD x64 processor-based systems with either CUDA-enabled NVIDIA GPUs or select AMD GPUs and APUs, running Linux, OS X, or Windows. OS X accelerator support is available only on NVIDIA GPUs.

1.1.3. PGI Workstation and PGI Server Comparison

- All PGI Workstation products include a node-locked single-user license, meaning one user at a time can compile on the one system on which the PGI Workstation compilers and tools are installed. The product and license server are on the same local machine.
- PGI Server products are offered in configurations identical to PGI Workstation, but include network-floating multi-user licenses. This means that two or more users can use the PGI compilers and tools concurrently on any compatible system networked to the license server, that is, the system on which the PGI Server license keys are installed. There can be multiple installations of the PGI Server compilers and tools on machines connected to the license server; and the users can use the product concurrently, provided they are issued a license key by the license server.

1.1.4. PGI CDK Cluster Development Kit

A cluster is a collection of compatible computers connected by a network. The PGI CDK supports parallel computation on clusters of 64-bit x86-compatible AMD and Intel processor-based Linux workstations or servers with or without accelerators and interconnected by a TCP/IP-based network, such as Ethernet.

Support for cluster programming does not extend to clusters combining 64-bit processor-based systems with 32-bit processor-based systems.

1.2. Release Components

Release 2015 includes the following components:

- PGFORTRAN™ native OpenMP and auto-parallelizing Fortran 2003 compiler.
- PGCC® native OpenMP and auto-parallelizing ANSI C11 and K&R C compiler.
- PGC++® native OpenMP and auto-parallelizing ANSI C++11 compiler.
- PGPROF® MPI, OpenMP, and multi-thread graphical profiler.
- PGDBG® MPI, OpenMP, and multi-thread graphical debugger.
- MPICH MPI libraries, version 3.1.3, for 64-bit development environments (Linux and OS X only).

64-bit linux86-64 MPI messages are limited to <2GB size each.

- Microsoft HPC Pack 2012 MS-MPI Redistributable Pack (version 4.1) for 64-bit and 32-bit development environments (Windows only).
- LAPACK linear algebra math library for shared-memory vector and parallel processors, version 3.4.2, supporting Level 3 BLACS (Basic Linear Algebra Communication Subroutines) for use with PGI compilers. This library is provided in both 64-bit and 32-bit versions for AMD64 or Intel 64 CPU-based installations running Linux, OS X, or Windows.
- ScaLAPACK 2.0.2 linear algebra math library for distributed-memory systems for use with MPICH, Open MPI, MVAPICH, and the PGI compilers on 64-bit Linux and OS X for AMD64 or Intel 64 CPU-based installations.
- A UNIX-like shell environment for 32-bit and 64-bit Windows platforms.
- FlexNet license utilities.
- Documentation in PDF and man page formats.

1.2.1. Additional Components for PGI CDK

The PGI CDK for Linux includes additional components available for download from the PGI website, but not contained in the installation package:

- MVAPICH2 MPI libraries, version 2.0 available for 64-bit development environments.
- Open MPI libraries, version 1.8.4 for 64-bit development environments with support for NVIDIA GPUDirect. Requires CUDA 6 or later.

1.2.2. MPI Support

You can use PGI products to develop, debug, and profile MPI applications. The PGPROF® MPI profiler and PGDBG® debugger included with PGI Workstation and PGI Server are limited to 16 local processes. The MPI profiler and debugger included with PGI CDK support up to 64 or 256 remote processes, depending on the purchased capabilities.

1.3. Terms and Definitions

This document contains a number of terms and definitions with which you may or may not be familiar. If you encounter an unfamiliar term in these notes, please refer to the online glossary at http://www.pgroup.com/support/definitions.htm

These two terms are used throughout the documentation to reflect groups of processors:

AMD64

A 64-bit processor from AMD™ designed to be binary compatible with 32-bit x86 processors, and incorporating new features such as additional registers and 64-bit addressing support for improved performance and greatly increased memory range. This term includes the AMD Athlon64™, AMD Opteron™, AMD Turion™, AMD Barcelona, AMD Shanghai, AMD Istanbul, AMD Bulldozer, and AMD Piledriver processors.

Intel 64

A 64-bit Intel Architecture processor with Extended Memory 64-bit Technology extensions designed to be binary compatible with AMD64 processors. This includes Intel Pentium 4, Intel Xeon, Intel Core 2, Intel Core 2 Duo (Penryn), Intel Core (i3, i5, i7), both first generation

(Nehalem) and second generation (Sandy Bridge) processors, as well as Ivy Bridge and Haswell processors.

1.4. Supported Platforms

There are six platforms supported by the PGI Workstation and PGI Server compilers and tools. Currently, PGI CDK supports only 64-bit Linux clusters.

- **32-bit Linux** — includes all features and capabilities of the 32-bit Linux operating systems running on an x64 compatible processor. 64-bit Linux compilers will not run on these systems.
- **64-bit Linux** — includes all features and capabilities of the 64-bit Linux operating systems running on an x64 compatible processor. Both 64-bit and 32-bit Linux compilers run on these systems.
- **32-bit Windows** — includes all features of the 32-bit Windows operating systems running on either a 32-bit x86 compatible or an x64 compatible processor. 64-bit Windows compilers will not run on these systems.
- **64-bit Windows** — includes all features and capabilities of the 64-bit Windows version running on an x64 compatible processor. Both 64-bit and 32-bit Windows compilers run on these systems.
- **32-bit OS X** — supported on 32-bit Apple operating systems running on either a 32-bit or 64-bit Intel-based Mac system. 64-bit OS X compilers will not run on these systems.
- **64-bit OS X** — supported on 64-bit Apple operating systems running on a 64-bit Intel-based Mac system. Both 64-bit and 32-bit OS X compilers run on these systems.

1.5. Supported Operating System Updates

This section describes updates and changes to PGI 2015 that are specific to Linux, OS X, and Windows.

1.5.1. Linux

Linux download packages are organized in PGI 2015 so that you can download a 64-bit Linux compiler package for installation on 64-bit Linux machines, and/or you can download a 32-bit package that installs on 32-bit and 64-bit Linux systems.

- CentOS 5.2+, including CentOS 7
- Fedora 4+, including Fedora 21
- RHEL 4.8+, including RHEL 7
- SLES 10+, including SLES 11 SP 3
- SuSE 9.3+, including openSuSE 13.2
- Ubuntu 8.04+, including Ubuntu 14.10

1.5.2. OS X

PGI 2015 for OS X supports most of the features of the 32-bit and 64-bit versions for Linux environments. Except where noted in these release notes or the user manuals, the PGI compilers and tools on OS X function identically to their Linux counterparts.

- Supported versions are OS X versions 10.7 (Lion) and newer, including 10.10 (Yosemite).

1.5.3. Windows

PGI products for Windows support most of the features of the 32-bit and 64-bit versions for Linux environments. PGI products on all Windows systems include the Microsoft Open Tools but also require that the Microsoft Windows 8.1 Software Development Kit (SDK) be installed prior to installing the compilers.

The PGI C++ compiler for Windows is deprecated and will no longer be available as of the PGI 16.1 release.

PGI 2015 requires the Windows 8.1 SDK, even on Windows 7 and Windows 8.

These Windows operating systems are supported in PGI 2015:

- Windows Server 2008 R2
- Windows 7
- Windows 8
- Windows 8.1
- Windows Server 2012

PGI releases no longer include support for Windows XP, Windows Server 2003, or Windows Server 2008.

1.6. Getting Started

By default, the PGI 2015 compilers generate code that is optimized for the type of processor on which compilation is performed, the compilation host. If you are unfamiliar with the PGI compilers and tools, a good option to use by default is `-fast` or `-fastsse`.

These aggregate options incorporate a generally optimal set of flags for targets that support SSE capability. These options incorporate optimization options to enable use of vector streaming SIMD instructions for 64-bit targets. They enable vectorization with SSE instructions, cache alignment, and flushz.

The contents of the `-fast` or `-fastsse` options are host-dependent.

The following table shows the typical `-fast` and `-fastsse` options.

Table 1 Typical -fast and -fastsse Options

Use this option...	To do this...
-O2	Specifies a code optimization level of 2.
-Munroll=c:1	Unrolls loops, executing multiple instances of the original loop during each iteration.
-Mnoframe	Indicates to not generate code to set up a stack frame. **Note** With this option, a stack trace does not work.
-Mlre	Indicates loop-carried redundancy elimination.
-Mpre	Indicates partial redundancy elimination

-fast for 64-bit targets and -fastsse for both 32-bit and 64-bit targets also typically include the options shown in the following table:

Table 2 Additional -fast and -fastsse Options

Use this option...	To do this...
-Mvect=sse	Generates SSE instructions.
-Mscalarsse	Generates scalar SSE code with xmm registers; implies -Mflushz.
-Mcache_align	Aligns long objects on cache-line boundaries. **Note** On 32-bit systems, if one file is compiled with the -Mcache_align option, then all files should be compiled with it. This is not necessary on 64-bit systems.
-Mflushz	Sets SSE to flush-to-zero mode.
-M[no]vect	Controls automatic vector pipelining.

For best performance on processors that support SSE instructions, use the PGFORTRAN compiler, even for FORTRAN 77 code, and the -fastsse option.

In addition to -fast and -fastsse, the -Mipa=fast option for interprocedural analysis and optimization can improve performance. You may also be able to obtain further performance improvements by experimenting with the individual -Mpgflag options that are described in the PGI Compiler Reference Manual, such as -Mvect, -Munroll, -Minline, -Mconcur, -Mpfi, -Mpfo, and so on. However, increased speeds using these options are typically application and system dependent. It is important to time your application carefully when using these options to ensure no performance degradations occur.

Chapter 2.
NEW AND MODIFIED FEATURES

This chapter provides information about the new or modified features of Release 2015 of the PGI compilers and tools.

2.1. What's New in Release 2015

Further information about most of these topics is included in the following sections.

15.9 Updates and Additions

- Beta support for the OpenACC parallel programming model on multicore x86 CPUs; whereas the `-ta=tesla` and `-ta=radeon` compiler options offload OpenACC parallel loops and regions to execute on a GPU, the new `-ta=multicore` compiler option maps OpenACC loops and regions to execute in parallel on multicore x86 CPUs.
  - Compile OpenACC programs for parallel execution across all cores of a multicore CPU or multi-socket CPU server
  - Incrementally parallelize applications for multicore CPUs and GPUs using the OpenACC KERNELS directive and PGI compiler optimization and parallelization feedback feature
  - Use a uniform parallel programming model across CPUs and GPUs in Fortran, C and C++
  - Write scalable OpenACC source code that will compile and run in parallel on NVIDIA GPUs, Radeon GPUs or multicore CPUs
- CUDA 7.0 toolkit support is now default; the CUDA 7.5 toolkit is integrated with this release and supported using a compile- and link-time option
- Numerous PGI Accelerator compiler improvements including:
  - Auto-privatization of scalar and array variables in OpenACC compute regions
  - New conditional sentinels for CUDA Fortran (!@CUF) and OpenACC (!@acc) with functionality similar to the OpenMP !$ conditional sentinel

- Support for CUDA 7.0 and 7.5 Thrust headers
- A new cuRAND Fortran module to enable calls to cuRAND with CUDA Fortran DEVICE arrays
- An updated version of the CUDA 7.0 cuFFT library
- Support for Fortran computed goto in OpenACC regions

- Platform updates
  - Support for Windows 10
  - OpenMPI 1.8.8 is now included in the PGI CDK Cluster Development Kit
  - 32-bit applications are no longer supported with CUDA-x86

- Over 30 user-requested fixes and updates.

15.7 Updates and Additions

- PGI Fortran/C/C++ Compilers
  - Incremental Fortran 2008 features: new iso_c_binding function C_sizeof, new iso_Fortran_env functions compiler_options and compiler_version, allocating polymorphic variables using the MOLD= specifier, associating non-pointer actual arguments with pointer dummy arguments in procedure calls.
  - Improved SIMD vectorization of loops with logical operations and expressions involving constants
  - Improved performance of intrinsic function calls where one or more input arguments are compile-time constants

- PGI Accelerator Fortran/C/C++ OpenACC Compilers
  - Support for Maxwell GPUs including Titan X
  - The default compute capability list for OpenACC and CUDA Fortran compilation for NVIDIA Tesla targets is now cc20, cc30, cc35, and cc50, generating device code for Fermi, Kepler K10 and K20, and Maxwell. The generation of device code can be time consuming, so you may notice an increase in compile time. The default can be overridden; refer to Default Compute Capability for details.
  - Improved support for calling the NVIDIA cuBLAS and cuRAND libraries from PGI compilers on Linux x86-64. Examples are included in the CUDA-Libraries directory under the PGI 2015 examples area.
  - The `-ta=tesla:beta` flag in PGI 15.7 will enable 128-bit load and store operations in the generated code, which can produce dramatic performance improvements when working on short structures in the innermost loop. This feature is currently under beta test, and may become the default in a future release.

- Other Features and Additions
  - Enhanced debugger support for setting breakpoints in shared objects
  - Enhanced communication of program load status in the debugger

- Updated JRE bundled with PGI packages to version 1.8u45
- Moved Eclipse Plug-in for C/C++ compilers into separate download package

15.5 Updates and Additions

- The OpenACC Unified Memory Evaluation Package is now included with all Linux 64-bit installation packages. Assent of a Beta License Agreement is still required. Please review the README_UM_EVAL.txt file in the doc directory before use.
- Improved OpenACC user experience via enhanced compiler messages.
- Bounds-checking is not supported in device code; the -Mbounds option has been disabled.
- Argument removal during IPA linking is now disabled for accelerator code.
- OpenACC Fortran for NVIDIA GPUs and CUDA Fortran now supports list-directed i/o to the default output unit (`PRINT *` or `WRITE(*,*)`) in device kernels. See the CUDA Fortran Programming Guide and Reference, section 3.6.7, for currently supported data types.
- A number of problems are corrected in this release. Refer to http://www.pgroup.com/support/release_tprs.htm for a complete and up-to-date table of technical problem reports fixed in recent releases of PGI compilers and tools. This table contains a summary description of each problem as well as the version in which it was fixed.

15.4 Updates and Additions

- PGI C++ Compiler
  - GNU 4.9 compatibility
  - Incremental C++14 features: lambdas, binary literals, apostrophes as digital separators
  - GNU statement expressions with class type results
  - General performance improvements
- PGI Fortran Compiler
  - Incremental Fortran 2008 features: transformational Bessel functions, storage size intrinsic, complex inverse trigonometric intrinsics, function for C sizeof in iso_c_binding, find location in an array
- PGI Accelerator Fortran/C/C++ OpenACC Compilers
  - CUDA 6.5 targeted by default
  - Integrated CUDA 7.0 toolkit
  - New OpenACC SDK examples
- Other Features and Additions
  - Debugger breakpoints now support hit counts
  - Linux 64-bit link options now include -Meh_frame to preserve exception-handling frame information by default

15.3 Updates and Additions

- A number of problems are corrected in this release. Refer to http://www.pgroup.com/support/release_tprs.htm for a complete and up-to-date table of technical problem reports fixed in recent releases of PGI compilers and tools. This table contains a summary description of each problem as well as the version in which it was fixed.

15.1 Updates and Additions

- PGI C++ Compiler
  - PGC++ (pgc++) is now default on Linux and OS X
  - Full support for C++11 on Linux and OS X
  - Expanded support for OpenACC
  - OpenACC implicit deep copy preview
  - NVCC host compiler on Linux
- PGI Fortran Compiler
  - Incremental Fortran 2008 features
  - Support for automatic arrays in OpenACC routines and CUDA Fortran
  - New CUDA Fortran intrinsics
  - New CUDA Fortran cuSPARSE module
- PGI Accelerator OpenACC Fortran/C/C++ Compilers
  - Comprehensive OpenACC 2.0 support
  - Support for OpenACC CUPTI-based profiling
  - Support for CUDA 6.0/6.5 and NVIDIA Kepler K40/K80 GPUs
  - PGI OpenACC Unified Memory Evaluation Package for Linux (available as a separate download)
  - New OpenACC SDK examples
- Other Features and Additions
  - Updated pre-compiled MPICH, MVAPICH and Open MPI libraries
  - New top-level `PrgEnv-pgi` environment module; automatically load modules for PGI compilers, MPICH, and optionally NetCDF in one step
  - New operating system support including Ubuntu 14.10, Fedora 21, CentOS 7, RHEL 7 and OS X Yosemite

2.2. New and Modified Compiler Options

Starting with version 15.1, the default behavior for all 64-bit Linux compilers is to link against dynamic libraries. The previous default method was to link against static libraries. To override the new default behavior, link using either the `-Bstatic` or `-Bstatic_pgi` options.

Accelerator device code generation for the 64-bit compilers has changed to use nvvm by default. Use `-ta=tesla:nollvm` or `-Mcuda=nollvm` to use the legacy CUDA-C code generator.

Release 2015 supports a number of new command line options as well as new keyword suboptions for existing command line options.

New compiler options include the following:

- `-c11` — Use the C11 language (C/C++ only).
- `-c1x` — Equivalent to `-c11`.
- `-M[no]idiom` — Enable [disable] loop idiom recognition.
- `--install` — Rerun makelocalrc.

Modifications to the `-Mcuda` suboptions include the deprecation of `cc1x`, `cc1+`, `tesla` and `tesla+` as well as these changes:

- `charstring` — Enable limited support for character strings in GPU kernels.
- `[no]lineinfo` — Generate GPU line information (replaces `[no]lineninfo`).
- `[no]llvm` — The LLVM back end is now the default in 64-bit mode.

Modifications to the `-ta=tesla` suboptions include the deprecation of `cc1x`, `cc1+`, `tesla`, `tesla+` and `[no]required` as well as these changes:

- `[no]lineinfo` — Generate GPU line information (replaces `[no]lineninfo`).
- `[no]llvm` — The LLVM back end is now the default in 64-bit mode.

Modifications to the `-ta=radeon` suboptions include the deprecation of `[no]required` as well as these changes:

- `[no]lineinfo` — Generate GPU line information (replaces `[no]lineninfo`).
- `[no]llvm` — The LLVM/SPIR back end is now the default in 64-bit mode.
- `spir` — Use the LLVM/SPIR back end, which is now the default in 64-bit mode.

Modifications to the `-acc` suboptions include the deprecation of the `[no]required` suboption.

2.3. New and Modified Fortran Functionality

- Incremental Fortran 2008 features including SIMPLY CONTIGUOUS, ERF/ERFC, inverse hyperbolic functions, elemental and transformational Bessel functions, storage size intrinsic, gamma, log_gamma, hypot, complex inverse trigonometric intrinsics, function for C sizeof in iso_c_binding, and find location in an array.
- Enhanced ANSI-like preprocessing for Fortran.
- Enhanced SIMD vectorization: conditionals, vectorization profitability analysis.

2.4. New and Modified C/C++ Functionality

- PGC++ (pgc++) is now default on Linux and OS X. Features include GNU compatible name mangling and language features supporting g++ versions 4.2–4.9.
- Full support for C++11 on Linux and OS X. New C11 features include:

  - Lambdas
  - Alignof/alignas
  - No return
  - Type generics
  - Thread-local storage (TLS)
  - Long double using x87 80-bit arithmetic
- Incremental support for C++14 features including lambdas, binary literals, and apostrophes as digital separators.
- OpenACC class methods implicitly compiled for device execution, reference class/struct members allowed in data clauses and compute regions, manual deep copy of classes/structs with enter/exit data pragmas.
- OpenACC implicit deep copy using CUDA Unified Memory on NVIDIA GPUs (preview feature available in a separate download package).
- pgc++ is also now supported as an NVCC host compiler on Linux.
- Enhanced SIMD vectorization: conditionals, runtime pointer checking, vectorization profitability analysis.

The pgcpp C++ compiler is deprecated and will no longer be available as of the PGI 16.1 release. On Linux and OS X, the pgcpp C++ compiler has been replaced by the pgc++ C++ compiler. pgc++ offers substantial compatibility with GNU C++, including most language features and object code compatibility; it does not provide command-line options compatibility. On Windows, the pgcpp C++ compiler will not be replaced.

2.5. New and Modified Tools Functionality

This section provides information about the debugger, PGDBG, and the profiler, PGPROF.

- Hit count breakpoints have been added to PGDBG; this feature provides the ability to stop execution when the breakpoint hit count is equal to, greater than or a multiple of a specified number.
- Support for disassembling the AVX-2 instruction set has been added to both PGDBG and PGPROF.
- Performance improvements have been made to PGDBG's execution speed, on Windows in particular.

2.6. New and Modified CUDA Fortran Functionality

NVIDIA is dropping support for 32-bit CUDA toolkits. PGI's CUDA support plan will follow:

32-bit Systems

- Linux: CUDA 6.5 will be the default version of CUDA through PGI 15.10

- OS X: None
- Windows: CUDA 6.5 will be the default version of CUDA through PGI 15.10

PGI will stop supporting CUDA on 32-bit systems in 2016.

64-bit Systems

- Linux: Support for CUDA 6.5, 7.0, 7.5
- OS X: Support for CUDA 7.0, 7.5
- Windows: Support for CUDA 7.0, 7.5

Beginning in PGI 15.4, changes and enhancements have been made to CUDA Fortran default stream handling:

- Two new constants which can be used as stream values have been added to the cudafor module:

```
INTEGER(KIND=CUDA_STREAM_KIND), PARAMETER :: cudaStreamLegacy    = 1
INTEGER(KIND=CUDA_STREAM_KIND), PARAMETER :: cudaStreamPerThread = 2
```

- The `cudaGetStreamDefault` function has been renamed to `cudaforGetDefaultStream`.
- The `cudaSetStreamDefault` function has been renamed to `cudaforSetDefaultStream`.
- Streams for the sum(), maxval(), and minval() intrinsics which operate on device data can now be controlled with the functions `cudaforReductionSetStream` and `cudaforReductionGetStream`.
- If a nonzero default stream is set using `cudaforSetDefaultStream`, it will now be used for CUF Kernels and global subroutine launches if no other stream is explicitly set in the launch configuration.

For further information, please refer to the CUDA Fortran Programming Guide.

Other changes made in PGI 15.4:

- Support for CUDA 6.5/7.0 and NVIDIA Kepler K40/K80 GPUs.
- Updated cuSPARSE module with interfaces for changes in CUDA 7.0 cuSPARSE library.

These changes were made in PGI 15.1:

- New tuned host intrinsics which operate on device or managed data: sum, maxval, minval.
- New cuSPARSE module with interfaces to cuSPARSE library.
- Support for CUDA 6.0/6.5 and NVIDIA Kepler K40/K80 GPUs.
- Support for Fortran automatic arrays in CUDA Fortran kernels.
- Support for global static data in Accelerator device code.
- Full support for CUDA Unified Memory on NVIDIA GPUs.

2.7. New Features in PGI Accelerator OpenACC Compilers

Multicore Support

PGI 15.9 includes the Beta option `-ta=multicore`, to set the target accelerator for OpenACC programs to the host multicore. This will compile OpenACC compute regions for parallel execution across the cores of the host X86 processor or processors. The host multicore will be treated as a shared-memory accelerator, so the data clauses (copy, copyin, copyout, create) will be ignored and no data copies will be executed.

By default, `-ta=multicore` will generate code that will use all the available cores of the processor. If the compute region specifies a value in the num_gangs clause, the minimum of the num_gangs value and the number of available cores will be used. At runtime, the number of cores can be limited by setting the environment variable ACC_NUM_CORES to a constant integer value. If an OpenACC compute construct appears lexically within an OpenMP parallel construct, the OpenACC compute region will generate sequential code. If an OpenACC compute region appears dynamically within an OpenMP region or another OpenACC compute region, the program may generate many more threads than there are cores, and may produce poor performance.

The `-ta=multicore` option differs from the `-ta=host` option in that `-ta=host` generates sequential code for the OpenACC compute regions. In this release, `-ta=multicore` may not be used in conjunction with `-ta=tesla`, `-ta=radeon` or `-ta=host`.

Default Compute Capability

As of the 15.7 release, the default compute capability list for OpenACC and CUDA Fortran compilation for NVIDIA Tesla targets is cc20, cc30, cc35, and cc50. The generation of device code can be time consuming, so you may notice an increase in compile time. You can override the default by specifying one or more compute capabilities using either command-line options or an rcfile.

To change the default with a command-line option, provide a comma-separated list of compute capabilities to `-ta=tesla:` for OpenACC or `-Mcuda=` for CUDA Fortran.

To change the default with an rcfile, set the DEFCOMPUTECAP value to a blank-separated list of compute capabilities in the siterc file located in your installation's bin directory:

```
set DEFCOMPUTECAP=20 30 35 50;
```

Alternatively, if you don't have permissions to change the siterc file, you can add the DEFCOMPUTECAP definition to a separate .mypgirc file (mypgi_rc on Windows) in your home directory.

The PGI 2015 release implements the OpenACC 2.0 release, except as described below. In addition, the PGI 2015 release has implemented some features scheduled for the next version of OpenACC.

New OpenACC 2.0 Features

- Cache directive
- Auto loop clause
- Device_type clause
- Collapse clause
- Firstprivate clause
- Gang(num:) and gang(static:) loop subclauses
- Reduction in routine clause
- Use of complex data types in reduction clause

OpenACC 2.0 Missing Features

- The declare link directive for global data is not implemented.
- Nested parallelism (parallel and kernels constructs within a parallel or kernels region) is not implemented.

Present_or_

The PGI 2015 release has changed the behavior of the **copy**, **copyin**, **copyout** and **create** clauses so they will behave the same as **present_or_copy**, **present_or_copyin**, **present_or_copyout** and **present_or_create**. This will not change the behavior of any existing correct OpenACC program. A program that would fail with a runtime error message that data in a data clause is already present on the device will now continue to execute. This behavior will be part of the next OpenACC specification version.

Shortloop Clause

The PGI compilers allow an additional loop clause, **shortloop**. If the **shortloop** clause appears on a loop directive with the **vector** clause, it tells the compiler that the loop trip count is less than or equal to the number of vector lanes created for that loop. This means the value of the **vector()** clause on the loop directive in a kernels region, or the value of the **vector_length()** clause on the parallel directive in a parallel region will be greater than or equal to the loop trip count. This allows the compiler to generate more efficient code for the loop.

Expressions for Vector Length

Previously, the expression in the **vector_length** clause on the **parallel** directive, and the expression in the **vector** clause on the loop directive in a kernels region was limited to compile-time constant expressions. The PGI compilers now allow a runtime expression in this clause. If the value exceeds the limits of the target device, the program will fail at runtime. The current limit for NVIDIA CUDA devices is 1024 for the product of the vector length and the number of workers. The current limit for AMD Radeon devices is 256 for the product of the vector length and the number of workers.

C++ Class Member References In Data Clauses

This release allows C++ class data members to be referenced in OpenACC data clauses. In a class member function, the class member may appear in a data clause as if it were a local variable. The data member may also be referenced using pointer (−>) or member (.) notation. In all cases, when the parent class is also present on the device, the corresponding pointer in the device copy of the parent class is updated to point to the device copy of the data member.

```
template<typename T>
class myvect {
    T* mydata;
    size_t mysize;
public:
    inline T& operator[](int i) const { return mydata[i]; }
    void axpy(myvect<T>& x, T a) {
        size_t n = mysize;
        #pragma acc parallel loop present(mydata, x)
        for (int i = 0; i < n; ++i)
            mydata[i] += a*x[i];
    }
    . . .
};
```

Fortran Derived Type Member References In Data Clauses

This release allows Fortran derived type members to appear in OpenACC data clauses. The member may be an allocatable, pointer or direct array. If the member is an allocatable or pointer array and the parent derived type variable is also present on the device, the corresponding pointer in the device copy of the parent derived type variable is also updated to point to the device copy of the member.

```
type objects
  real, allocatable, dimension(:) :: temperature
  integer, allocatable, dimension(:) :: location
end type
    . . .
type(objects) :: x
    . . .
!$acc enter data copy(x, x%temperature)
    . . .
!$acc parallel loop
    do i = 1, n
       x%temperature(i) = findtemp(i)
    enddo
```

Support for Profiler / Trace Tool Interface

This release implements a preliminary version of the OpenACC Profiler/Trace Tools Interface. This is the interface used by the PGI profiler to collect performance measurements of OpenACC programs.

2.8. C++ Compiler

The pgcpp C++ compiler is deprecated and will no longer be available as of the PGI 16.1 release. On Linux and OS X, the pgcpp C++ compiler has been replaced by the pgc++ C++ compiler. pgc++ offers substantial compatibility with GNU C++, including most language features and object code compatibility; it does not provide command-line options compatibility. On Windows, the pgcpp C++ compiler will not be replaced.

2.8.1. C++ and OpenACC

This release includes support for OpenACC directives for the C++ compilers, pgc++ (Linux, OS X) and pgcpp. There are limitations to the data that can appear in data constructs and compute regions:

- Variable-length arrays are not supported in OpenACC data clauses; VLAs are not part of the C++ standard.
- Variables of class type that require constructors and destructors do not behave properly when they appear in data clauses.
- Exceptions are not handled in compute regions.
- Any function call in a compute region must be inlined. This includes implicit functions such as for I/O operators, operators on class type, user-defined operators, STL functions, lambda operators, and so on.

2.8.2. C++ Compatibility

PGI 2015 C++ object code is incompatible with prior releases.

All C++ source files and libraries that were built with prior releases must be recompiled to link with PGI 2015 or higher object files.

Starting with the PGI 15.1 release, optional packages that provide a C++ interface—such as the MPICH package included with all PGI products—now require the use of the pgc++ compiler driver for compilation and linking. This is a change from previous releases where pgcpp was the required driver.

Other affected packages include:

- mvapich2-2.0_x86_64.tar.gz
- netcdf-4.3.0_x86_64.tar.gz
- openmpi-1.8.4_x86_64.tar.gz

2.9. New and Modified Runtime Library Routines

PGI 2015 supports new runtime library routines associated with the PGI Accelerator compilers.

For more information, refer to *Using an Accelerator* in the User's Guide.

2.10. Library Interfaces

PGI provides access to a number of libraries that export C interfaces by using Fortran modules. These libraries and functions are described in the PGI Compiler User's Guide.

2.11. Environment Modules

On Linux, if you use the Environment Modules package (e.g., the **module load** command), then PGI 2015 includes a script to set up the appropriate module files.

Chapter 3. DISTRIBUTION AND DEPLOYMENT

Once you have successfully built, debugged and tuned your application, you may want to distribute it to users who need to run it on a variety of systems. This section addresses how to effectively distribute applications built using PGI compilers and tools.

3.1. Application Deployment and Redistributables

Programs built with PGI compilers may depend on runtime library files. These library files must be distributed with such programs to enable them to execute on systems where the PGI compilers are not installed. There are PGI redistributable files for all platforms. On Windows, PGI also supplies Microsoft redistributable files.

3.1.1. PGI Redistributables

The PGI 2015 Release includes these directories:

$PGI/linux86/15.9/REDIST
$PGI/linux86-64/15.9/REDIST
$PGI/win32/15.9/REDIST
$PGI/win64/15.9/REDIST

These directories contain all of the PGI Linux runtime library shared object files, OS X dynamic libraries, or Windows dynamically linked libraries that can be re-distributed by PGI 2015 licensees under the terms of the PGI End-user License Agreement (EULA). For reference, a text-form copy of the PGI EULA is included in the 2015 directory.

3.1.2. Linux Redistributables

The Linux REDIST directories contain the PGI runtime library shared objects for all supported targets. This enables users of the PGI compilers to create packages of executables and PGI runtime libraries that will execute successfully on almost any PGI-supported target system, subject to these requirements:

- End-users of the executable have properly initialized their environment.
- Users have set LD_LIBRARY_PATH to use the relevant version of the PGI shared objects.

3.1.3. Microsoft Redistributables

The PGI products on Windows include Microsoft Open Tools. The Microsoft Open Tools directory contains a subdirectory named `redist`. PGI 2015 licensees may redistribute the files contained in this directory in accordance with the terms of the PGI End-User License Agreement.

Microsoft supplies installation packages, `vcredist_x86.exe` and `vcredist_x64.exe`, containing these runtime files. These files are available in the `redist` directory.

Chapter 4. TROUBLESHOOTING TIPS AND KNOWN LIMITATIONS

This section contains information about known limitations, documentation errors, and corrections. Wherever possible, a work-around is provided.

For up-to-date information about the state of the current release, visit the frequently asked questions (FAQ) section on pgroup.com at www.pgroup.com/support/faq.htm

4.1. General Issues

Most issues in this section are related to specific uses of compiler options and suboptions.

- Object files created with prior releases of PGI compilers are incompatible with object files from PGI 2015 and should be recompiled.
- Using the `-g` option to generate debug information for CUDA Fortran has these limitations:
  - Only linux 64-bit platform supports this feature
  - No debug information is generated for boundaries for Fortran automatic arrays, adjustable dummy arrays, or assumed-shape dummy arrays.
  - No debug information is generated for GPU code after a **`!CUF`** directive.
- The `-i8` option can make programs incompatible with the ACML libraries; use of any INTEGER*8 array size argument can cause failures. Visit developer.amd.com to check for compatible libraries.
- Using `-Mipa=vestigial` in combination with `-Mipa=libopt` with PGCC, you may encounter unresolved references at link time. This problem is due to the erroneous removal of functions by the vestigial sub-option to `-Mipa`. You can work around this problem by listing specific sub-options to `-Mipa`, not including `vestigial`.
- OpenMP programs compiled using `-mp` and run on multiple processors of a SuSE 9.0 system can run very slowly. These same executables deliver the expected performance and speed-up on similar hardware running SuSE 9.1 and above.

4.2. Platform-specific Issues

4.2.1. Linux

The following are known issues on Linux:

- Programs that incorporate object files compiled using `-mcmodel=medium` cannot be statically linked. This is a limitation of the linux86-64 environment, not a limitation of the PGI compilers and tools.

4.2.2. Apple OS X

The following are known issues on Apple OS X:

- The PGI 2015 compilers do not support static linking of binaries. For compatibility with future Apple updates, the compilers only support dynamic linking of binaries.
- Using `-Mprof=func` or `-Mprof=lines` is not supported.

4.2.3. Microsoft Windows

The following are known issues on Windows:

- For the Cygwin `emacs` editor to function properly, you must set the environment variable CYGWIN to the value "tty" before invoking the shell in which `emacs` will run. However, this setting is incompatible with the PGBDG command line interface (`-text`), so you are not able to use `pgdbg -text` in shells using this setting.

  The Cygwin team is working to resolve this issue.
- On Windows, the version of `vi` included in Cygwin can have problems when the SHELL variable is defined to something it does not expect. In this case, the following messages appear when vi is invoked:

  ```
  E79: Cannot expand wildcards Hit ENTER or type command to continue
  ```

  To workaround this problem, set SHELL to refer to a shell in the cygwin bin directory, e.g. `/bin/bash`.
- C++ programs on Win64 that are compiled with the option `-tp x64` fail when using PGI Unified Binaries. The `-tp x64` switch is not yet supported on the Windows platform for C++.
- On Windows, runtime libraries built for debugging (e.g. `msvcrtd` and `libcmtd`) are not included with PGI Workstation. When a program is linked with `-g`, for debugging, the standard non-debug versions of both the PGI runtime libraries and the Microsoft runtime libraries are always used. This limitation does not affect debugging of application code.
- If you are building a 32-bit Windows Fortran program that performs Input/Output (I/O) on a namelist with User-Defined Derived type I/O, then you need to compile and link with the `-Munix` option.

The following are known issues with PGDBG on Windows:

- In PGDBG on the Windows platform, Windows times out `stepi/nexti` operations when single stepping over blocked system calls. For more information on the workaround for this issue, refer to the online FAQs at http://www.pgroup.com/support/tools.htm.

The following are known issues with PGPROF on Windows:

- Do not use `-Mprof` with PGI runtime library DLLs. To build an executable for profiling, use the static libraries.

4.3. PGDBG-related Issues

The following are known issues in PGDBG:

- Before PGDBG can set a breakpoint in code contained in a shared library, `.so` or `.dll`, the shared library must be loaded.
- Debugging of PGI Unified Binaries, that is, 64-bit programs built with more than one `-tp` option, is not fully supported. The names of some subprograms are modified in compilation and PGDBG does not translate these names back to the names used in the application source code. For detailed information on how to debug a PGI Unified Binary, refer to http://www.pgroup.com/support/tools.htm.

4.4. PGPROF-related Issues

The following are known issues in PGPROF:

- Programs compiled and linked for **`gprof`**-style performance profiling using `-pg` can result in segmentation faults on systems running version 2.6.4 Linux kernels.
- Times reported for multi-threaded sample-based profiles, that is, profiling invoked with options `-pg` or `-Mprof=time`, are for the master thread only. To obtain profile data on individual threads, PGI-style instrumentation profiling with `-Mprof={lines | func}` or **`pgcollect`** must be used.

4.5. CUDA Toolkit Issues

Targeting a CUDA Toolkit Version

- The CUDA 6.5 Toolkit is set as the default in PGI 15.9. To use the CUDA 6.5 Toolkit, first download the CUDA 6.5 driver from NVIDIA at www.nvidia.com/cuda.
- You can compile with the CUDA 7.0 Toolkit either by adding the option `-ta=tesla:cuda7.0` to the command line or by adding `set DEFCUDAVERSION=7.0` to the `siterc` file.
- `pgaccelinfo` prints the driver version as the first line of output. For a 6.5 driver, it prints:

```
CUDA Driver Version 6050
```

CUDA 6.5 Toolkit Limitations

- Scientific computing libraries including cuBLAS, cuSPARSE, cuFFT and cuRAND are not available for Linux 32-bit.
- Support for CUPTI, and therefore PGI_ACC_TIME, is not provided in 32-bits.

- The CUDA 6.5 Toolkit is not supported on Linux 32-bit systems with GNU version 4.9 and up.
- The CUDA 6.5 Toolkit is not supported on Windows Server 2012 although it is supported on Windows Server 2012 R2.
- The CUDA 6.5 Toolkit is not available for OS X 32-bit.

CUDA 7.0 Toolkit Limitations

- The CUDA 7.0 Toolkit is not available for Linux 32-bit.
- The CUDA 7.0 Toolkit is not available for OS X 32-bit.
- The CUDA 7.0 Toolkit is not available for Windows 32-bit nor is it supported on Windows Server 2012; it is, however, supported on Windows Server 2012 R2.

4.6. OpenACC Issues

This section includes known limitations in PGI's support for OpenACC directives.

PGI plans to support these features in a future release, though separate compilation and extern variables for Radeon will be deferred until OpenCL 2.0 is released.

ACC routine directive limitations

- The `routine` directive has limited support on AMD Radeon. Separate compilation is not supported on Radeon, and selecting `-ta=radeon` disables `rdc` for `-ta=tesla`.
- The `bind` clause on the `routine` directive is not supported.
- Reductions in gang-scheduled loops in procedures with `acc routine gang` are not supported. This is being added as a restriction in the next OpenACC specification.
- Fortran assumed-shape arguments are not yet supported.

Clause Support Limitations

- Not all clauses are supported after the `device_type` clause.

OpenMP Limitations

- There are known problems when using OpenACC for GPU accelerators inside of OpenMP parallel regions. This problem will be fixed in the near future.

4.7. Corrections

A number of problems are corrected in this release. Refer to www.pgroup.com/support/release_tprs.htm for a complete and up-to-date table of technical problem reports, TPRs, fixed in recent releases of the PGI compilers and tools. This table contains a summary description of each problem as well as the version in which it was fixed.

Chapter 5. CONTACT INFORMATION

You can contact PGI at:

20400 NW Amberwood Drive Suite 100
Beaverton, OR 97006

Or electronically using any of the following means:

Fax: +1-503-682-2637
Sales: sales@pgroup.com
Support: trs@pgroup.com
WWW: http://www.pgroup.com

The PGI User Forum is monitored by members of the PGI engineering and support teams as well as other PGI customers. The forum newsgroups may contain answers to commonly asked questions. Log in to the PGI website to access the forum:

http://www.pgroup.com/userforum/index.php

Many questions and problems can be resolved by following instructions and the information available at our frequently asked questions (FAQ) site:

http://www.pgroup.com/support/faq.htm

All technical support is by e-mail or submissions using an online form at:

http://www.pgroup.com/support

Phone support is not currently available.

PGI documentation is available at http://www.pgroup.com/resources/docs.htm or in your local copy of the documentation in the release directory `doc/index.htm`.

PGI®