IBM pSeries Compiler Roadmap

Roch Archambault
IBM Toronto Laboratory
archie@ca.ibm.com
Agenda

- The pSeries Compiler Products
- Roadmaps
  - XL Fortran
  - XL C/C++
    - Performance
- Performance Results
- Q&A
The pSeries Compiler Products

- Latest Versions (All POWER5 enabled except Mac OS X)
  - XL C/C++ Enterprise Edition V7.0 for AIX, Linux (includes PPC970 & BG/L)
  - XL Fortran Enterprise Edition V9.1 for AIX, Linux (includes PPC970 & BG/L)
  - XL C/C++ Advanced Edition V6.0 for Mac OS X (PPC970 only)
  - XL Fortran Advanced Edition V8.1 for Mac OS X (PPC970 only)

- Upcoming Versions (All POWER5+ and PPC970 enabled)
  - XL C/C++ Enterprise Edition V8.0 for AIX, Linux (including BG/L)
  - XL Fortran Enterprise Edition V10.1 for AIX, Linux (including BG/L)

All information subject to change without notice
XL Fortran Roadmap: Strategic Priorities

- **Premium Customer Service**
  
  Continue to work closely with key ISVs and customers in scientific and technical computing industries

- **Compliance to Language Standards and Industry Specifications**
  
  OpenMP Fortran API V2.5
  
  Fortran 77, 90 and 95 standards
  
  Emerging 2003 Standard

- **Exploitation of Hardware**
  
  Committed to maximum performance on POWER4, PPC970, POWER5, BG/L and successors
  
  Continue to work very closely with processor design teams
XL Fortran Version 9.1 for AIX

- Fully compliant Fortran 77/90/95 compiler
- Partial Fortran 2003 support
- Many industry extensions
  - 128-bit float, 64-bit integer, Cray pointers, structure record, union map, BYTE, STATIC, AUTOMATIC, asynchronous I/O, SIZE intrinsic
- Optimized OpenMP Fortran V2.0 support
- Symbolic Debugging support (dbx, Totalview)
- Portfolio of Optimizing Transformations
  - Loop optimization for parallelism, locality and instruction scheduling
  - Link-time loop optimization framework under -qipa=level=2 (-O5)
  - Tuned support for all pSeries processors (including POWER5, PPC970 and BG/L)
- Additional directives for user transformations
  - BLOCK_LOOP, STREAM_UNROLL, UNROLL_AND_FUSE
- User-directed hardware prefetching on PWR5 (PROTECTED_STREAM)
- POWER5-specific intrinsics (Popcount, swdiv)
- SMP support up to 64-way (128-way SMT), including thread binding
- MASS and MASSV libraries distributed with the compiler
XL Fortran Version 9.1 for Linux on pSeries

- Based on XL Fortran Version 9.1 for AIX Product
  - Leverage proven industry leading compiler technology
- Fully compliant Fortran 77/90/95 compiler, partial 2003 support
- 32- and 64-bit support
- Optimized OpenMP Fortran V2.0 support
- Symbolic Debugging support with gdb
- Portfolio of Optimizing Transformations (same as XLF V9.1 for AIX)
- Supports SUSE Linux Enterprise Server 9 (SLES 9)
- Supports RedHat Enterprise Linux 3.0 (RHEL 3) and 4.0 (RHEL 4)
- Automatic generation of optimized VMX code (for PPC970)
- Optimized code generation for BlueGene/L
  - Optimization and scheduling for PPC440 processor
  - Automatic generation of SIMD instructions (Double Hummer)
- Alignment analysis and intrinsics for SIMD code generation
Initial rollout of Fortran 2003 (XLF V9.1)

C Interoperability
   BIND attribute and statement
   BIND on derived types, common blocks, and procedures
   -qalign=bindc compiler suboption
PROCEDURE, IMPORT and FLUSH statements
PUBLIC/PRIVATE on derived type components
ISO_FORTRAN_ENV and ISO_C_BINDING intrinsic modules
ASSOCIATE Construct
Command Line Argument and NEW_LINE intrinsics
Intrinsic attribute on USE
IOMSG= specifier
XL Fortran V10.1 And Beyond

- XL Fortran Enterprise Edition V10.1 for AIX (2005)
- XL Fortran Enterprise Edition V10.1 for Linux (2005)
  - POWER5+ support
  - Additional Fortran 2003 features
  - Support for OpenMP API Version 2.5 and beyond
  - Improved performance
  - Improved compilation time and reduced memory requirements at high opt
  - Locality optimization at lower opt levels (-O3)
  - Support for different levels of –qhot optimizations (via suboption)
  - Subset of Level-2 blas routines tuned for PWR4/PWR5 included in the compiler runtime
  - Subset of scalar mass routines included in the compiler runtime
  - Generation of binaries with multiple copies of key routines optimized for different target processors
  - Data structure analysis and reorganization

All information subject to change without notice
Prioritized rollout of Fortran 2003 (XLF V10.1)

1. OO extensions
2. ENUM/ENUMERATOR
3. Derived type I/O
4. Asynchronous I/O additions
5. Allow character on MAX/MIN family of intrinsics
6. Procedure pointers
7. I/O specifiers (DECIMAL=, SIGN=, ROUND=, and PAD=)
8. Complex literal
9. Derived type parameters
10. Abstract interface
11. Enhance structure constructors
12. Enhanced array constructors
13. Pointer assignment enhancement (specify bounds)
14. Support for International Usage

All information subject to change without notice
C/C++ Roadmap: Strategic Priorities

- Premium Customer Service
- Compliance to Language Standards and Industry Specifications
  - ANSI / ISO C and C++ Standards
  - Support for OpenMP API Version 2.5 and beyond
- Exploitation of Hardware
  - Committed to maximum performance on POWER4, PPC970, POWER5, BG/L and successors
  - Continue to work very closely with processor design teams
- Exploitation of OS and Middleware
  - Synergies with operating system and middleware ISVs (performance, specialized function)
  - Committed to AIX Linux affinity strategy and to Linux on pSeries
- Reduced Emphasis on proprietary Tooling
  - Affinity with GNU toolchain
XL C/C++ Version 7.0 for AIX

- Fully compliant ISO C 1999 and ISO C++ 1998 standards
  - Supports some C99 features under C++
- Partial GNU C/C++ language and options compatibility
  - Computed gotos, VLA, partial inline asm support, gcc attributes
- Optimized OpenMP C/C++ API V2.0 support
- C++ specific optimizations for exception-handling, templates, etc...
- Tuned automatic parallelization for C/C++
- Symbolic Debugging Support
  - TotalView, IBM Distributed Debugger and dbx/pdbx
  - Full support for debugging of OpenMP programs
  - Partial support for debugging of optimized code
  - Runtime memory debug support
- Portfolio of optimizing transformations
  - Loop optimization for parallelism, locality and instruction scheduling
  - Link-time loop optimization framework under -qipa=level=2 (-O5)
  - Tuned support for all pSeries processors (including POWER5, PPC970 and BG/L)
XL C/C++ Version 7.0 for AIX (continued)

- New built-in functions for POWER5 exploitation
  - Population count, protected_stream, swdiv
- Scalability of whole-program analysis (-qipa) due to the use of 64-bit compiler components
- New PDF tooling for examination of collected PDF data and weighted merge of profile data
- SMP support up to 64-way (128-way SMT), including thread binding
- MASS and MASSV libraries distributed with the compiler
- More info:
  - www.ibm.com/software/awdtools/vacpp
XL C/C++ Version 7.0 for Linux on pSeries

- Based on XL C/C++ Version 7.0 for AIX
  - Leverage proven industry leading compiler technology
- Fully compliant ISO C 1999 and ISO C++ 1998 standards
- Further GNU C/C++ language and options compatibility
- 32- and 64-bit support
- Automatic Parallelization for C/C++
- Optimized OpenMP C/C++ API V2.0 support
- Portfolio of Optimizing Transformations (same as C/C++ V7.0 for AIX)
- Supports SUSE Linux Enterprise Server 9 (SLES 9)
- Supports RedHat Enterprise Linux 3.0 (RHEL 3) and 4.0 (RHEL 4)
- Gcc-compatible options processing (gxic and gxic)
- Gcc-compatible vector extensions for VMX (for PPC970)
- Automatic generation of optimized VMX code (for PPC970)
- Optimized code generation for BlueGene/L
  - Optimization and scheduling for PPC440 processor
  - Automatic generation of SIMD instructions (Double Hummer)
- Alignment analysis and intrinsics for SIMD code generation
XL C/C++ V8.0 And Beyond


    ISO C/C++ 200x subset
  - POWER5+ support
  - OpenMP C/C++ API V2.5
  - Further GNU C/C++ compatibility features
  - Improved performance
  - Parallelization of link step optimizer (-qipa=threads)
  - Support for UPC language (AIX only)

All information subject to change without notice
Tentative GNU C/C++ Compatibility Enhancements

- Global register variables
- Nested functions
- Naming types
- Conditionals with omitted operands
- Labeled elements (C only)
- Case ranges (C only)
- Cast to union (C only)
- Incomplete enums
- Function names as strings
- Additional asm support

All information subject to change without notice
Performance Improvements Delivered in 2003

- Included in XLF V8.1.1 release
- OpenMP
  - Much faster barrier implementation (over 5x faster on 32-way p690)
  - Much faster uncontended atomic (over 4x faster on 32-way p690)
  - Improved parallel region startup (25%)
- POWER4
  - Expanded scope and precision of instruction scheduling
  - More precise loop unrolling for pipelining and parallelization of reductions
  - Use of dcbz to optimize store streams
  - Loop unrolling for prefetch stream utilization
- Loop Optimization
  - Aggressive loop fusion to exploit data reuse
  - Index set splitting and gather/scatter to move branches out of loops
  - Loop peeling for automatic parallelization
  - Improved loop interchange for automatic parallelization
  - Strip-mining of vectorized loops
  - Vector calls to specialized routines for POWER4 (vsqrt, vrsqrt, vtan, vlog, vexp)
Performance Items Delivered in 2004

- Included in XLF V9.1 and XL C/C++ V7 (AIX, Linux)
- POWER5
  - Modified scheduling machine model
  - Usage of improved prefetch facilities
  - Usage of new instructions
- PPC970 & BG/L
  - Automatic generation of SIMD code (VMX and double-hummer)
  - Interprocedural pointer alignment propagation
- OpenMP and Automatic Parallelization
  - Tuned support for 64-way SMP
  - Continued improvements in overhead reduction
- Intrinsic functions (Fortran Only)
  - MATMUL, TRANSFER, INDEX, TRANSPOSE
Performance Items Delivered in 2004

- Included in XLF V9.1 and XL C/C++ V7 (AIX, Linux)
- Tuning assists:
  - BLOCK_LOOP and LOOID directives to specify which set of loops to tile, interchange or strip-mine
  - NOVECTOR and NOSIMD directive to tell compiler not to vectorize or simdize loop
  - Built-in functions for generating software divides (full double precision on POWER5)
  - Thread binding (set via startproc and stride in XLSMPOPTS env variable)
  - Environment variable (set via XLSMPOPTS intrithds env variable) to control number of threads used by MATMUL and RANDOM_NUMBER
  - View and manipulate information gathered by profile directed feedback (-qpdf1/-qpdf2) via showpdf and mergepdf tools
  - Prefetch directives for new stream prefetch control on POWER5
Performance Items Delivered in 2004

- Included in XLF V9.1 and XL C/C++ V7 (AIX, Linux)
- **Loop Optimizations**:
  - Modulo scheduling of loops which contain branches
  - Further improvements to loop fusion for data reuse (e.g. loop alignment)
  - Perform vectorization on Linux
  - Enhancement of vectorization (additional functions, loop versioning, vector merging)
  - Tiling for BLAS-like and streaming loop nests
  - Predictive Commoning (common subexpression elimination across loop iterations)
  - Improved data dependence analysis
  - Improved induction-variable and reduction recognition
  - Automatic generation of software divides on POWER5
  - Automatic generation of new stream prefetch instructions on POWER5
Performance Items Delivered in 2004

- **Included in XLF V9.1 and XL C/C++ V7 (AIX, Linux)**
- **Other Optimizations**:
  - Interprocedural Code Motion
  - Interprocedural Strength Reduction
  - Interprocedural Register Allocation
  - Split array of structures into multiple arrays for better exploitation of hardware streams and smaller d-cache footprint
  - Use Profile Directed Feedback (PDF) information to:
    - Specialize calls to malloc/calloc to use pools of small objects
    - Specialize memcpy and memset with small lengths
    - Specialize integer divide and modulo
    - Superblock formation for better instruction scheduling
Expected Performance Improvements in 2005

- **Loop Optimizations:**
  - Sparse Vectorization
    - Provide tuned versions on some BLAS routines, and automatically use the ESSL if available at runtime (even if not available during link-time)
  - Further tuning of automatic parallelization thresholds
  - Improved synchronization overhead for OpenMP and automatic parallelization
  - Improved dependence analysis, including runtime checks
  - Array Data flow analysis for array privatization
  - Automatic array padding

- **PDF Optimizations:**
  - Outline “cold” fields of data structures for smaller d-cache footprint
  - Discover semi-constant arguments to a function and create specialized versions when profitable

All information subject to change without notice
Documentation

- An information center containing the documentation for the current versions of the AIX compilers (both Fortran and C/C++) is available at: http://publib.boulder.ibm.com/infocenter/comphelp/index.jsp

- Similar documentation for the current version of the Linux compilers (both Fortran and C/C++) is available at: http://publib.boulder.ibm.com/infocenter/lnxpcomp/index.jsp

- This information center contains all the html documentation shipped with the compilers. It is completely searchable.

- Please send any comments or suggestions on this information center or about the existing C, C++ or Fortran documentation shipped with the products to compinfo@ca.ibm.com.
Software Divide Improvements On POWER5

![Software Divide Instrinsics](chart)

- **SP SWDIV**
- **DP SWDIV**
- **DP SWDIV -qnostrict**
- **SP SWDIV_NOCHK**
- **DP SWDIV_NOCHK**
- **DP SWDIV_NOCHK -qnostrict**
SPEC FP Base Improvements From Compiler On POWER4

SPECFP Benchmarks

- 168.wupwise
- 173.aplu
- 179.art
- 188.ammp
- 200.sixtrack
- 171.swim
- 177.mesa
- 183.equake
- 189.lucas
- 301.apsi
- 172.mgrid
- 178.galgel
- 187.facercc
- 191.fma3d
- SPEC FP + 18%
SPEC FP Base Improvements From Compiler On POWER5

![SPEC FP Benchmarks Improvement Chart]

- SPEC FP + 20%
- 168.wupwise
- 173.applu
- 188.ammp
- 189.lucas
- 191.fma3d
- 171.swim
- 177.mesa
- 183.equake
- 301.apsi
- 172.mgrid
- 178.galgel
- 187.facerec
SPEC FP Base Auto-Parallelization (2 CPUs, POWER4)

SPECFP Benchmarks

- 168.wupwise
- 173.applu
- 179.art
- 188.ammp
- 200.sixtrack
- 171.swim
- 177.mesa
- 183.equake
- 189.lucas
- 301.apsi
- 172.mgrid
- 178.galgel
- 187.facerec
- 191.fma3d
- SPEC FP + 8%
SPEC FP Peak Differences Between IT2 (1.5 GHZ) and POWER5

-50 -30 -10 10 30 50 70 90

P5 / IT2 % difference

1.65 GHZ

168.wupwise 173.applu 179.art 188.ammp 200.sixtrack
171.swim 177.mesa 183.equake 189.lucas 301.apsi
172.mgrid 178.galgel 187.facerec 191.fma3d SPECFP Peak

1.9 GHZ
SPECOMP Base Improvements From Compiler On POWER4 (32-way)

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>310.wupwise_m</td>
<td>30%</td>
</tr>
<tr>
<td>312.swim_m</td>
<td>15%</td>
</tr>
<tr>
<td>314.mgrid_m</td>
<td>10%</td>
</tr>
<tr>
<td>316.applu_m</td>
<td>5%</td>
</tr>
<tr>
<td>318.galgel_m</td>
<td>2%</td>
</tr>
<tr>
<td>320.equate_m</td>
<td>1%</td>
</tr>
<tr>
<td>324.apsi_m</td>
<td>0%</td>
</tr>
<tr>
<td>326.gafort_m</td>
<td>0%</td>
</tr>
<tr>
<td>328.fma3d_m</td>
<td>0%</td>
</tr>
<tr>
<td>330.art_m</td>
<td>0%</td>
</tr>
<tr>
<td>332.ammp_m</td>
<td>0%</td>
</tr>
<tr>
<td>SPECOMP + 8%</td>
<td>0%</td>
</tr>
</tbody>
</table>
SPECOMP Scalability On POWER4 (16 vs 32 CPUs)
SPEC OMPM2001 Base Versus Competition

<table>
<thead>
<tr>
<th>System</th>
<th>SPECMARK Thousands</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM 32x1.7</td>
<td></td>
</tr>
<tr>
<td>HP-I 32x1.5</td>
<td></td>
</tr>
<tr>
<td>SGI-I 32x1.5</td>
<td></td>
</tr>
<tr>
<td>HP-A 64x1.15</td>
<td></td>
</tr>
<tr>
<td>HP-I 64x1.5</td>
<td></td>
</tr>
<tr>
<td>SGI-I 64x1.5</td>
<td></td>
</tr>
<tr>
<td>FUJ 64x1.3</td>
<td></td>
</tr>
<tr>
<td>IBM P5 570 16*1.9</td>
<td></td>
</tr>
<tr>
<td>IBM P5 595 64*1.9</td>
<td></td>
</tr>
</tbody>
</table>
Himeno Benchmark Improvement on POWER4

Small Data Problem – Measured in MFLOPS on 1.1GHZ p690
64-way EPCC results on AIX 5.3 p5 595 system (1.65 GHz)

Time in micro-seconds - lower is better

All information subject to change without notice