

Benefits of upgrading to the latest z/OS XL C/C++ compiler

IBM

Overview

Upgrading to the latest IBM z/OS® XL C/C++ compiler makes good business sense. Upgrading puts new capabilities into the hands of your programmers making them and your business more efficient. New compilers are essential for fully exploiting new hardware but they also help you squeeze more out of your current systems. Every release of the IBM z/OS XL C/C++ compiler introduces a number of new features including increased performance optimizations, additional support for new language specifications, and exploitation of any new hardware and software environments.

IBM develops extensive and mature, industry-leading compilation technology that covers multiple platforms and programming languages. This technology uses a modular development structure to deliver performance optimization and functionality to all supported platforms and languages.

Each compiler release derives from a common code base. Features and performance optimizations are tested in multiple languages on multiple platforms, making source-level portability of applications between platforms easier and providing a reliable development environment. A common compiler architecture provides a cost efficient and scalable development environment that addresses the requirements of a dynamic enterprise.

Users benefit from upgrading to newer releases of the compiler. Every new hardware or z/OS release supported includes new features that the compiler exploits to get more functionality and additional performance benefits. Recompiling with a newer release of the compiler often allows applications to benefit from these new features and get performance gains without source code changes.

IBM compilers are proven technology in scalable development environments and are the compilers of choice for IBM middleware as well as important industry applications and business solutions. New features introduced in the recent releases of the z/OS XL C/C++ compiler and their benefits are outlined in the following sections.

IBM Z exploitation

Used with IBM compilers that are developed to exploit their capabilities, IBM Z servers can deliver unprecedented performance, reliability, and energy efficiency. IBM servers, running applications built with IBM compilers have achieved leading performance on industry benchmarks. The z/OS and hardware exploitation features of the compilers improve programmer productivity by transforming and optimizing code generation and enabling programmers to exploit leading-edge performance of the new hardware without source code changes. The capability of supporting the latest processors, like the IBM z15™ processors, offers greater computational performance and precision for business and financial applications, giving users control over performance versus accuracy trade-offs for floating point calculations. Matching the hardware with compilers that are crafted to fully exploit it ensures that your application benefits from the latest industry-leading hardware advancements.

Feature	Benefits of upgrading		
	XL C/C++ V2.2	XL C/C++ V2.3	XL C/C++ V2.4
New processor exploitation	IBM z3, IBM z 13s	IBM z14™, IBM z14 Model ZR1	IBM z15
New operating system exploitation	z/OS V2.2	z/OS V2.3	z/OS V2.4
z/OS updated architecture level set (ALS)	z10 EC, z10 BC	zEC12, zBC12	
Default ARCH	ARCH(8)	ARCH(10)	ARCH(10)

Performance

The key strength of the IBM XL compiler family is performance. The IBM XL compilers are unmatched in their ability to optimize and tune code for execution on IBM Z. The performance gain from years of compiler optimization experience can be seen in the release-to-release compiler improvements.

The leading-edge compiler optimizations improve the performance of applications running on IBM servers helping maximize the return on hardware investment. This compiler exploits new instructions in the IBM z15. For example, z/OS V2R4 XL C/C++ provides ARCH(13) and TUNE(13) options to help you exploit new instructions that are available on z15 servers. You can use the new TUNE(13) option to generate

code that is optimized for z15 processors. These options are designed to provide better performing applications tuned for the new server. These changes can improve the performance of generated code without the need for changes to the source code. Taking advantage of the new architecture, new vector built-in functions are added for enhanced vector support.

z/OS V2R4 XL C/C++ can reduce CPU usage by an average of 16% and up to 22% on z15, over the same set of key numerically intensive double-precision floating-point applications built by z/OS V2R3 XL C/C++ running on z14.¹

Feature	Benefits of upgrading		
	XL C/C++ V2.2	XL C/C++ V2.3	XL C/C++ V2.4
Processor exploitation	Optimization and tuning for z13 [®] and z13s [®]	Optimization and tuning for z14	Optimization and tuning for z15
Optimizations	<ul style="list-style-type: none"> Exploits the vector facility for z/Architecture[®] Supports IBM Mathematical Acceleration Subsystem (MASS) and Automatically Tuned Linear Algebra Software (ATLAS) libraries for high-performance mathematical computing 	Exploits the vector facility for z/Architecture in the following aspects to gain potential performance improvements: <ul style="list-style-type: none"> Binary floating-point data types Built-in library functions Fixed-point decimal operations 	Exploits the vector enhancement facility 2 and the miscellaneous instruction extensions facility 3.

Compiler option control

The z/OS XL C/C++ compiler provides a rich set of options that enable users to get their solutions up and running quickly.

The options provide flexibility in adapting compiler functionality to the required tasks without requiring source code changes. The options assist programmer productivity and lower maintenance costs by diagnosing potential language semantics adherence while controlling reliable code generation.

Option category	Benefits of upgrading		
	XL C/C++ V2.2	XL C/C++ V2.3	XL C/C++ V2.4
Optimization and tuning: allow users to control the optimization and tuning process, which can improve the performance of applications at run time	<ul style="list-style-type: none"> Option to enable the compiler to generate code, when possible, using the SIMD instructions enabled under the vector facility for z/Architecture 	<ul style="list-style-type: none"> Options to optimize and tune your applications to exploit the instructions on z14 servers 	<ul style="list-style-type: none"> Options to optimize and tune your applications to exploit the instructions on z15 servers
Language element control: specify the characteristics of the source code, to enforce or relax language restrictions, and enable or disable language extensions	<ul style="list-style-type: none"> Suboption to control whether the <code>nullptr</code> feature is enabled 		

¹ These performance improvements are based on internal IBM lab measurements of double-precision scalar functions from the z/OS V2R4 MASS libraries for 31-bit XPLINK and 64-bit XPLINK addressing modes. The functions were compiled with z/OS V2R4 XL C/C++ with the ARCH(13), TUNE(13), and OPT(3) options, and z/OS V2R3 XL C/C++ with the ARCH(12), TUNE(12), and OPT(3) options. The ARCH(13) measurements were run on a z/OS V2R3 dedicated LPAR with 1 CP and 16GB central storage on an IBM z15 system. The ARCH(12) measurements were run on a z/OS V2R3 dedicated LPAR with 1 CP and 16GB central storage on an IBM z14 system. The "up to" figures are an average of the top 5 CPU reductions. Performance results for specific applications will vary, depending on the source code, the compiler options specified, and other factors.

Option category	Benefits of upgrading		
	XL C/C++ V2.2	XL C/C++ V2.3	XL C/C++ V2.4
Error checking and debugging: allow users to detect and correct problems in the source code	<ul style="list-style-type: none"> Option to control whether a null pointer check is performed on the pointer that is returned by an invocation of the throwing versions of operator new and operator new [] 	<ul style="list-style-type: none"> New STACKPROTECT option provides protection against malicious code or programming errors that overwrite or corrupt the stack New suboptions of INFO that control whether to emit warnings for procedures that are not protected against stack corruption A new DEBUG (NOFILE) suboption allows you to place the debugging information in the object file 	
Listings, messages, and compiler information: allow users control over the listing file, as well as how and when to display compiler messages.		<ul style="list-style-type: none"> New suboptions of AGGREGATE that allow you to specify whether to list the structure member offsets in decimal format or hexadecimal format 	

Debug capability

z/OS XL C/C++ helps increase programmer productivity and lower maintenance costs by providing information consumable by standard symbolic debugging tools, including IBM Application Delivery Foundation for z/OS and dbx. The user benefits from a familiar development environment using debugging tools of choice with increased proficiency and productivity, debugging source and some optimized code.

z/OS XL C/C++ supports the DWARF industry standard format for debugging information. The z/OS XL C/C++ compiler generates debugging information in both DWARF format and the legacy ISD format for compatibility. z/OS XL C/C++ also supports IBM Developer for z/OS, which enables you to examine, monitor, and control the execution of C, C++, COBOL, and PL/I programs.

Benefits of upgrading	
XL C/C++ V2.2	XL C/C++ V2.3
<ul style="list-style-type: none"> Non-XPLINK CDA runtime library for use with non-XPLINK calling code More complete debugging information by including empty header file names in the mdbg file 	<ul style="list-style-type: none"> Debugging information can be put in the GOFF NOLOAD classes in the object file instead of a separate debug side file

System programming support

The z/OS XL C/C++ compiler provides a feature to support system programming capabilities. This Metal C facility allows you to use C in place of assembler language for system program development. It is capable of generating optimized assembler code, which can take experienced mainframe assembler programmers a relatively long time to develop. Metal C code is highly portable amongst IBM Z[®] servers. To move from one platform to another, all you need is to recompile Metal C source to target the new platform. This requires no coding effort and therefore significantly reduces cost, risk and time to market.

System programming	Benefits of upgrading	
	XL C/C++ V2.2	XL C/C++ V2.3
Metal C	<ul style="list-style-type: none"> Allows requesting user fields of a specific size to be reserved on the stack 	<ul style="list-style-type: none"> Sets two new flags in the prefix data block to indicate the presence of two new optional fields: one contains the offset of the end of current CSECT and the other one contains the offset of the debug data block A new debug data block is added for each CSECT, which can be used to check whether the debug side file matches the object file Allows you to declare a function pointer with the new <code>__fdptr</code> keyword so that this function pointer points to a Metal C function descriptor

September 2019

References in this document to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

IBM®, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

© Copyright International Business Machines Corporation 2019.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.