GCC 7 Support Preview

JT Plotzke
z/TPF Development

Disclaimer

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

Contents

Contents

Discontinuing GCC 4.1 support

Previously announced statement of deprecation:

GNU C Compiler Collection (GCC) 4.1

Currently, z/TPF supports both the GCC compilers 4.1 and 4.6. Clients are required to move to GCC 4.6 as new z/TPF capabilities, such as MongoDB, can only be compiled with GCC 4.6. Support for GCC 4.1 will be discontinued on June 30, 2018. z/TPF Object Code Only (OCO) code created after June 30, 2018, will be compiled with GCC 4.6. To receive compiler or C/C++ language support, client applications must be built by using GCC 4.6 or another compiler supported by z/TPF.

Discontinuing GCC 4.1 support

GNU C Compiler Collection (GCC) 4.1

Currently, z/TPF supports both the GCC compilers 4.1 and 4.6. Clients are required to move to GCC 4.6 as new z/TPF capabilities, such as MongoDB, can only be compiled with GCC 4.6. Support for GCC 4.1 will be discontinued on June 30, 2018. z/TPF Object Code Only (OCO) code created after June 30, 2018, will be compiled with GCC 4.6. To receive compiler or C/C++ language support, client applications must be built by using GCC 4.6 or another compiler supported by z/TPF.

z/TPF deliverable (APAR)

- All OCO libraries rebuilt using GCC 4.6
  - GCC 4.6 will be used for all OCO updates going forward.
- MakeTPF tools updated to no longer allow GCC 4.1
- Some .mak file updates to remove any GCC 4.1 guards
- No additional rebuild requirements beyond OCO updates
- Target release early 3Q 2018
  - GCC 7 support will be a separate deliverable with a target release in late 3Q 2018.

Discontinuing GCC 4.1 support

GNU C Compiler Collection (GCC) 4.1

Currently, z/TPF supports both the GCC compilers 4.1 and 4.6. Clients are required to move to GCC 4.6 as new z/TPF capabilities, such as MongoDB, can only be compiled with GCC 4.6. Support for GCC 4.1 will be discontinued on June 30, 2018. z/TPF Object Code Only (OCO) code created after June 30, 2018, will be compiled with GCC 4.6. To receive compiler or C/C++ language support, client applications must be built by using GCC 4.6 or another compiler supported by z/TPF.

Customer migration to GCC 4.6

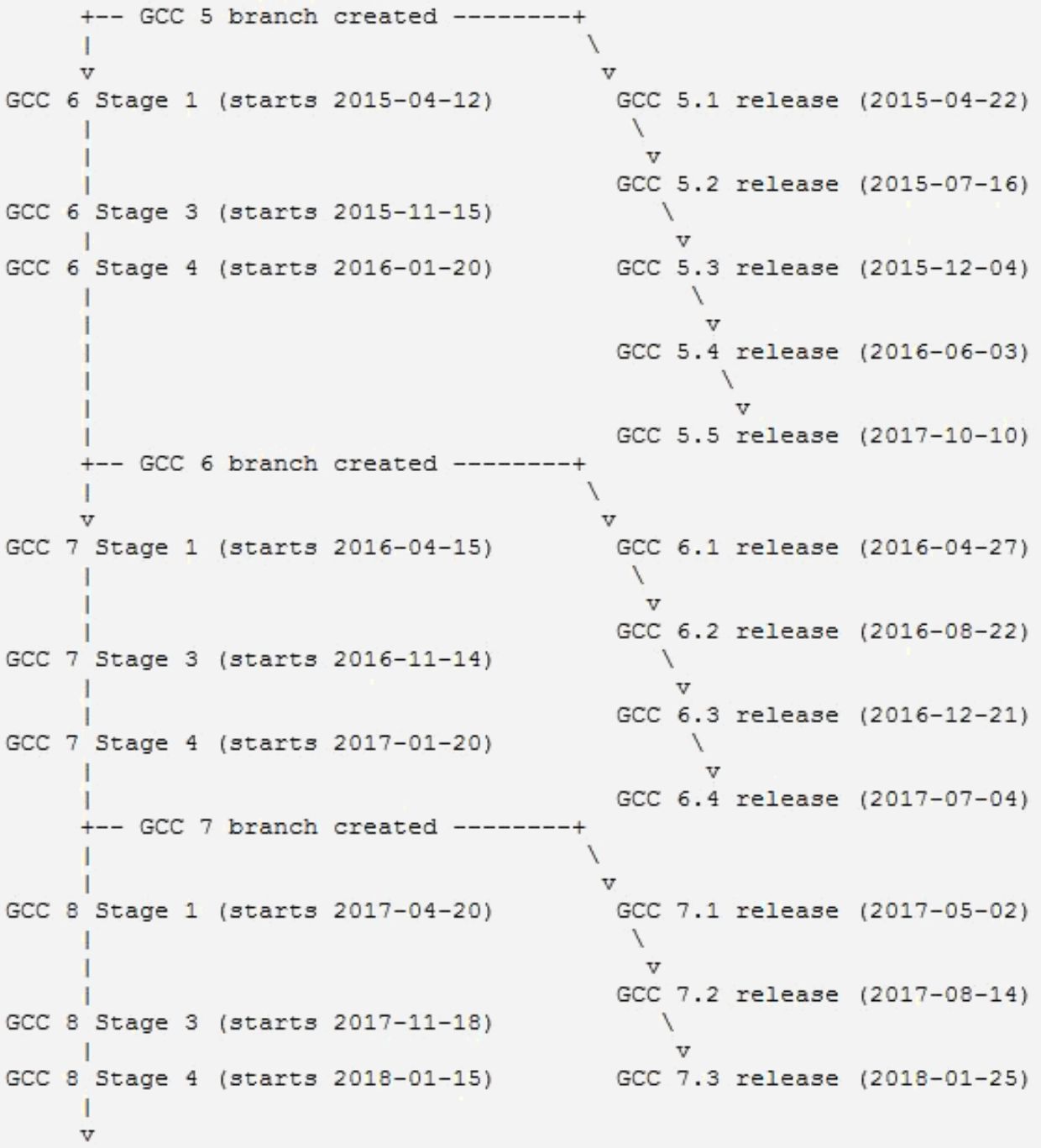
- Can be done at any time, do not need to wait for GCC 4.1 discontinuation APAR
- Obtain the GCC 4.6 cross-compiler (and support contract)
- Rebuild and reload minimum set of necessary z/TPF and opensource components using GCC 4.6
  - This includes archives (libgcc.a, libtpf_eh.a), CPP1HDRS46, and the libstdc++ library itself (CPP1).
- Rebuild and reload all z/TPF product code using GCC 4.6
- Rebuild and reload all C/C++ application code using GCC 4.6
  - Initial rebuild request is a **recommendation only**. You do not need to immediately rebuild all product or application code with GCC 4.6. However, if you encounter a compiler-related problem after June 30, you will be asked to do so for the related modules and verify the problem still exists.

Contents

GCC 7

Why upgrade to GCC 7?

- Last release of the 4.1 branch (4.1.2) was 11 years ago
- Last release of the 4.6 branch (4.6.4) was 5 years ago
- 6 newer branches, or release series, since GCC 4.6
- GCC 7 is current and should remain open for 2 more releases
  - Current = ability to support the latest standards, architecture levels, and optimizations
  - Open = ability to commit z/TPF opensource mods into mainline
- GCC 7 support in addition to continued z/TPF support of GCC 4.6
  - The GCC 4.6 branch being closed for support is not the same as z/TPF no longer supporting the compiler.



GCC Feature Additions

GCC 4.6
(2011/03/25 – 2013/04/12)

C11 experimental support
C++11 experimental support

Max z10 instruction set support

GCC 5
(2015/04/22 – 2017/10/10)

C11 full support (default)
C++11 full support
C++14 full experimental support

Max z13 instruction set support

General optimizer improvements

std::string now uses small string optimization instead of copy-on-write reference counting

std::list::size() now O(1)

GCC 6
(2016/04/27 – est. 2018)

C11 full support (default)
C++11 full support
C++14 full support (default)
C++17 experimental support

Max z13 instruction set support

General optimizer improvements (in addition to GCC 5)

New -Wmisleading-indentation warning, among others

GCC 7
(2017/05/02 – est. 2019)

C11 full support (default)
C++11 full support
C++14 full support (default)
C++17 full experimental support

z14 instruction set support

General optimizer improvements (in addition to GCC 6)

New builtin vector instructions via -mzvector option

Contents

New C++ standards

New supported C++ standards:

- Full C++11
  - GCC 4.6 only has experimental, “nearly all” C++11 support: https://gcc.gnu.org/gcc-4.6/cxx0x_status.html

C++ standards available for future z/TPF support (may require infrastructure updates):

- Full C++14
  - Minor standard update
  - <shared_mutex>, type deduction, [[deprecated]] attribute
- Full experimental C++17 (eventual full, non-experimental)
  - Major standard update
  - Parallel algorithm execution, shared_mutex class, shared_ptr with an array, <filesystem>, Boost extensions (any, optional)

C++14/17 <shared_mutex> example

C++14 adds the <shared_mutex> header

– Specifically std::shared_lock and std::shared_timed_mutex

C++17 adds std::shared_mutex

– Member functions:

  • Exclusive locking: lock, try_lock, unlock (same as std::mutex)

  • Shared locking: lock_shared, try_lock_shared, unlock_shared

– Would still obtain an exclusive lock for writing, but can now obtain multiple shared locks for reading simultaneously

```
#include <shared_mutex>

std::shared_mutex lock;
unsigned int avail_seats;  //thread safe counter

unsigned int read(){
    lock.lock_shared();
    unsigned int rc = avail_seats;
    lock.unlock_shared();
    return rc;
}

bool reserve(){
    lock.lock();
    bool rc = false;
    if (avail_seats > 0){
        --avail_seats;
        rc = true;
    }
    lock.unlock();
    return rc;
}
```


Contents

New Z instruction sets

Compiler options:

- -march=*cpu-type*: generate code that runs on *cpu-type*
- -mtune=*cpu-type*: tune to *cpu-type* everything applicable about the generated code, except for the ABI and instruction set

Supported -march= and -mtune values:

- Same as GCC 4.6: z900, z990, z9-109, z9-ec, z10
- Included in GCC 7: z196, zEC12, z13, and z14

z/TPF currently uses -march=z10 -mtune=z9-109

- -mtune=z9-109 is due to an optimization issue with our current pairing of GCC and Glibc versions
- GCC 7 support plan to ship with -march=z10 -mtune=z13
  - Will test -march=z13 and -march=z14 and relay any concerns

New GCC 7 optimizations

Previous z10 optimization error involving memory copy and comparison functions

- memcpy, memcmp, strcpy, strcmp
- Required -mtune=z9-109 for GCC 4.6 to revert back to builtin compiler functions instead of Glibc versions of the functions
- Issue will be addressed via a Glibc update, which will allow for tuning to more recent architecture levels

GCC 7 performance testing will be done

- No specific expectations, but will share the results

Thank you

JT Plotzke
z/TPF Development
—
jtplotzk@us.ibm.com
ibm.com

Trademarks

Notes

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.