outthink limits

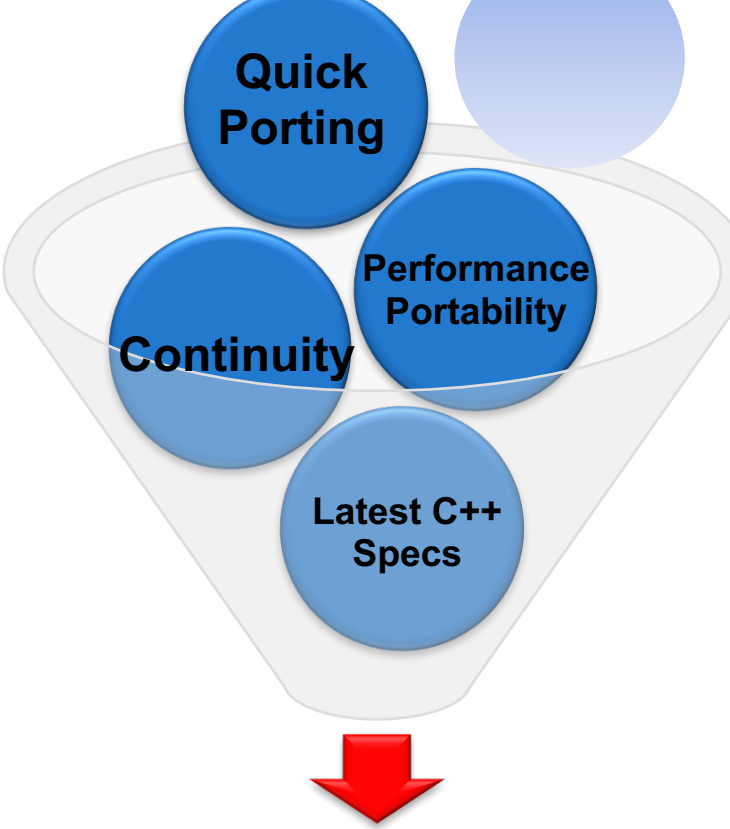
Offloading Support for OpenMP in Clang and LLVM

Carlo Bertolli
Advanced Compiler Technology Team
IBM T.J. Watson Research Center

Team: Samuel F. Antao, Alexey Bataev, Arpith C. Jacob, Gheorghe-Teodor Bercea, Alexandre E. Eichenberger, Georgios Rokos, Matt Martineau, Tian Jin, Guray Ozen, Zehra Sura, Tong Chen, Hyojin Sung, Carlo Bertolli, Kevin O'Brien

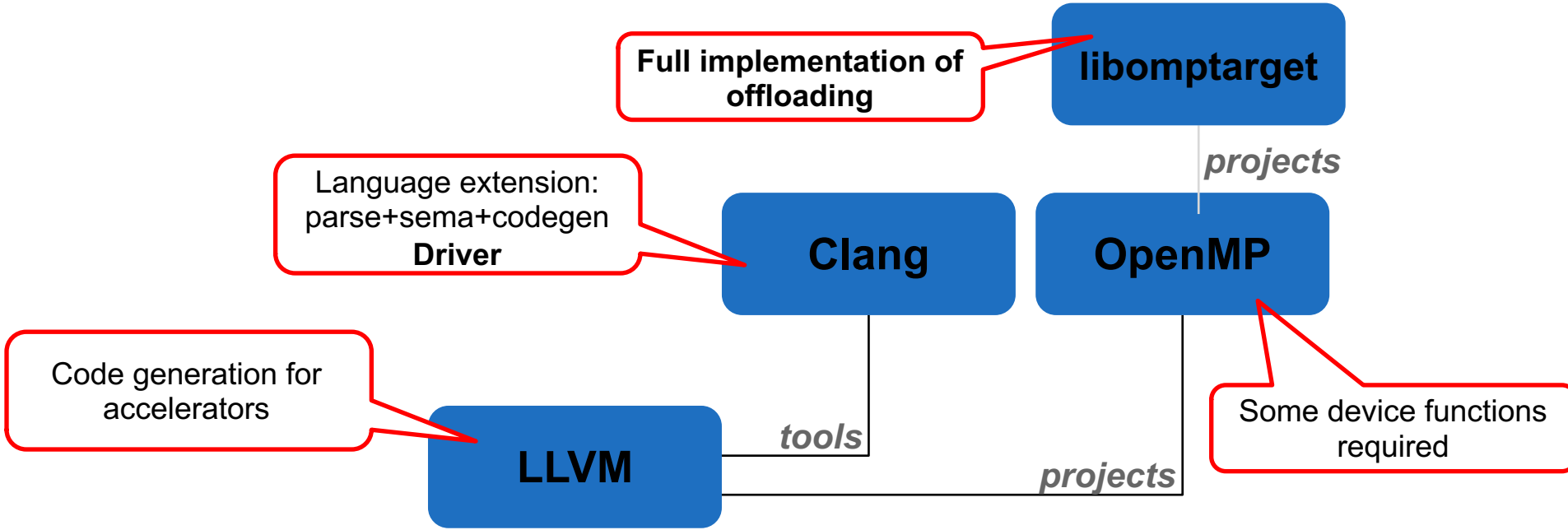
OpenMP in LLVM

What do HPC users need?	OpenMP/LLVM
Large legacy code base • Fortran 66, Fortran 77, C/C++ • MPI, pthreads, MPI+OpenMP, PGAS, etc. • **Quick Porting** is critical • No need to rewrite entire application in new language	**OpenMP allows quick porting of existing code regions to new architectures**
New code development • Latest C/C++ Specifications • Interoperability with "assembly" high performance libraries: CUDA, CUDNN, PETSc • **High-level parallel abstractions** for different patterns (loops, tasks, etc.)	**Clang/LLVM up-to-date with latest specs** **OpenMP has several parallel patterns**
Continuity • Industrial standard • Supported everywhere • Survives project/architecture/fashion/companies/etc. • **Open source**	**OpenMP Supported by: IBM, AMD, Intel, Cray, PGI, Pathscale, etc.**
Performance (Portability) • Should always beat MPI alone • Single version of code: as little device specific code as possible • Run as fast as possible on every host and device	



OpenMP in LLVM

OpenMP Contributions by Community



In this talk:

- Driver extensions
- libomptarget

OpenMP Offloading

Main application file

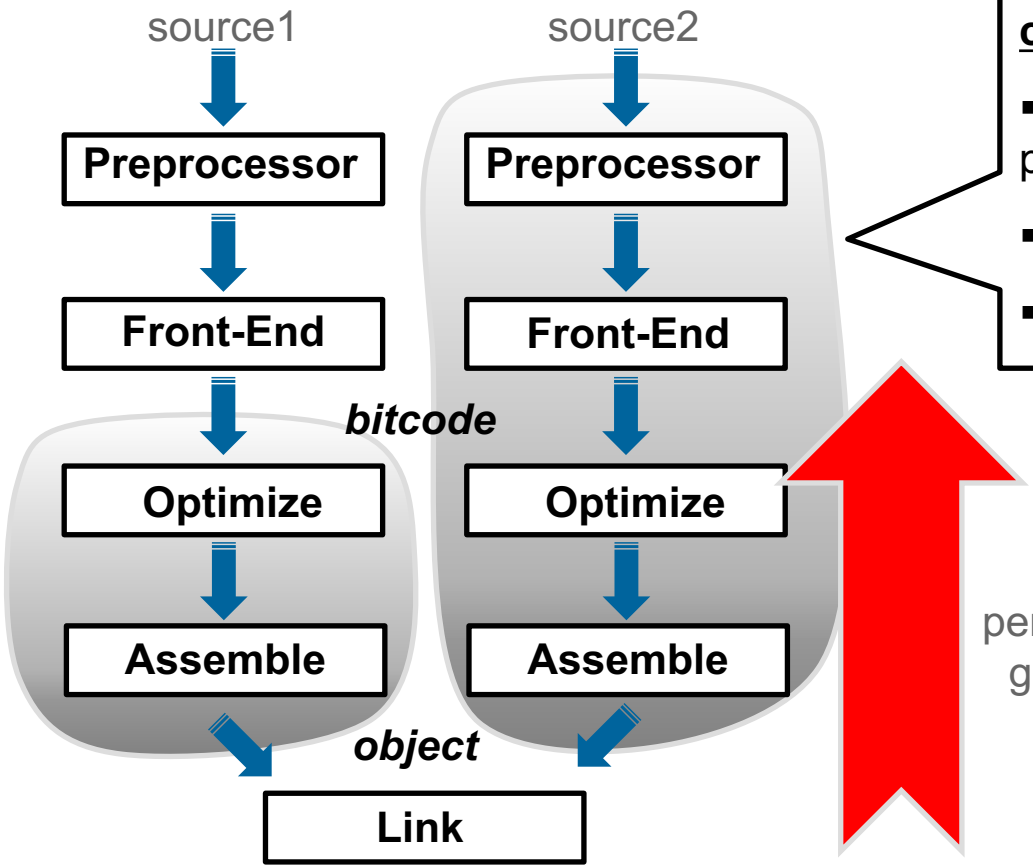
```
#pragma omp declare target
void linSolver(double * A, double * b);
#pragma omp end declare target

int main() {
 // set up A and b and map

  #pragma omp target teams
  {
    linSolver(A, b);
  }
}
```

Library implementation

```
#pragma omp declare target
void linSolver(double * A, double * b)
{
  // .. linear solver implementation
}
#pragma omp end declare target
```

- Same target body can run on host or device
- Different compilation units

Previous Driver Support



Job

clang –cc1 …

- Collect multiple actions that can be performed by a tool
- Define input and output files
- Define parameters

Job creation is performed by visiting graph from root to leaves

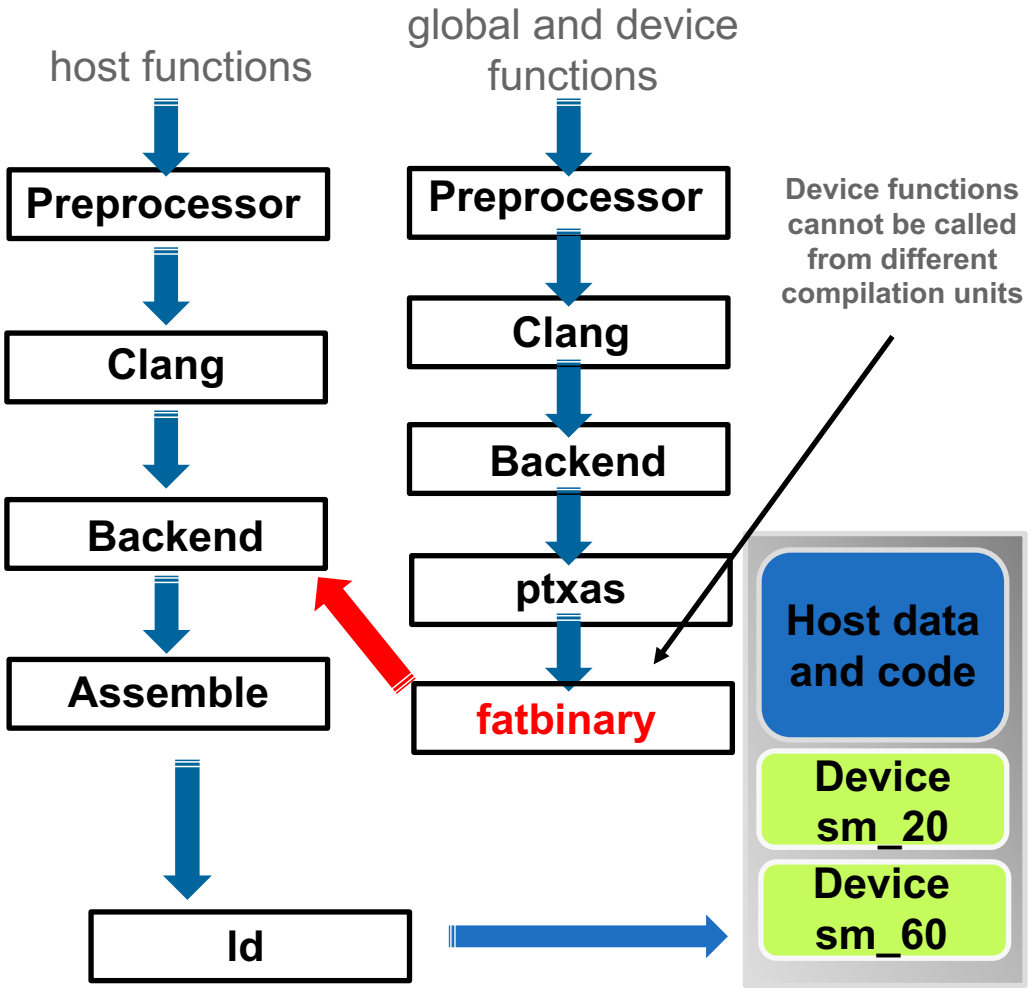
Driver Extensions for CUDA

Support for CUDA extends existing support with

- Host and device actions
- Selection of right toolchain per function based on **host**, **device**, **global** markups
- Linker tool (fatbinary) is used to combine objects for different compute capability
  - Not nvlink
- Host action is used to embed device code into host-produced binary

What is missing for OpenMP

- No relocation of symbols across host/device
- No dependencies between host and device toolchains possible



Driver Extension – Design Properties

Add as few new options as possible to enable offloading

- Avoid option proliferation (e.g. 10 new options for each device type)

Require minimal changes to existing build systems

- No new set of files for offloading

Operate seamlessly on libraries that contain offloading code

- Libraries can contain both host and device code

Support a convenient way to read and modify intermediate files that have a textual format

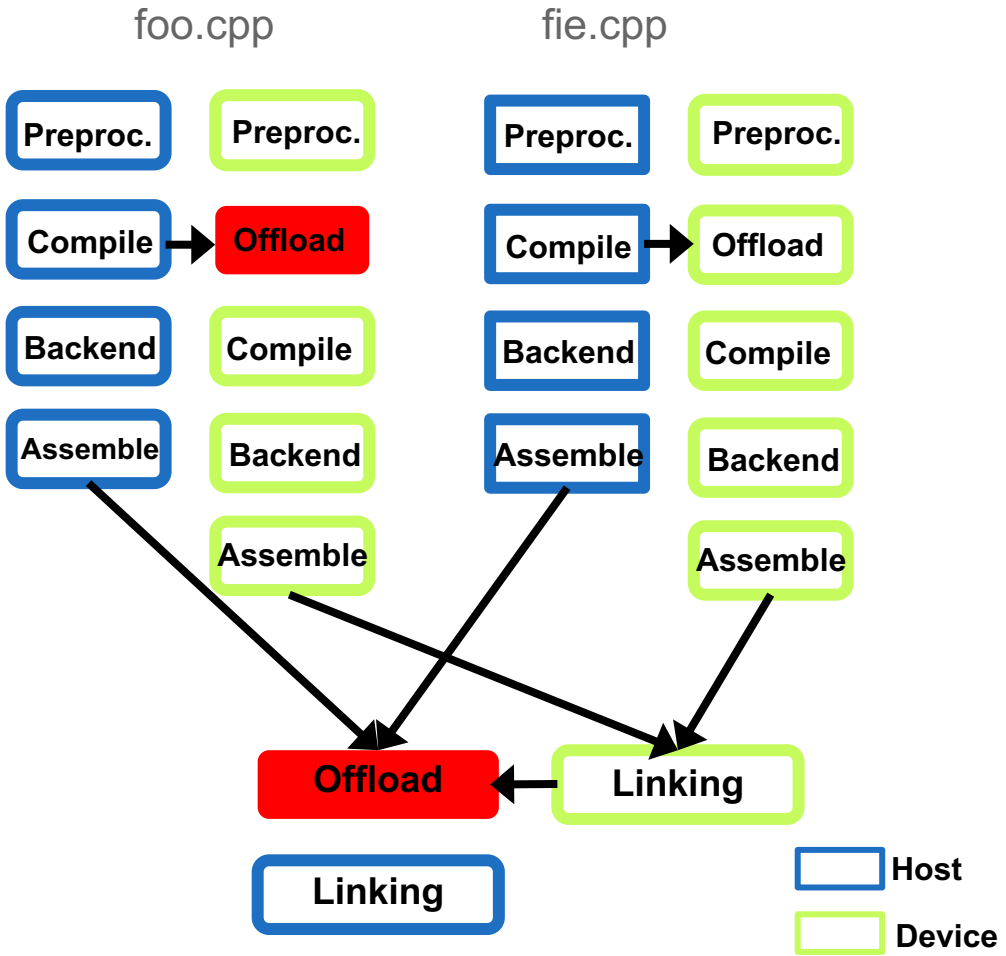
- Bundler tool

Generic Offloading Action

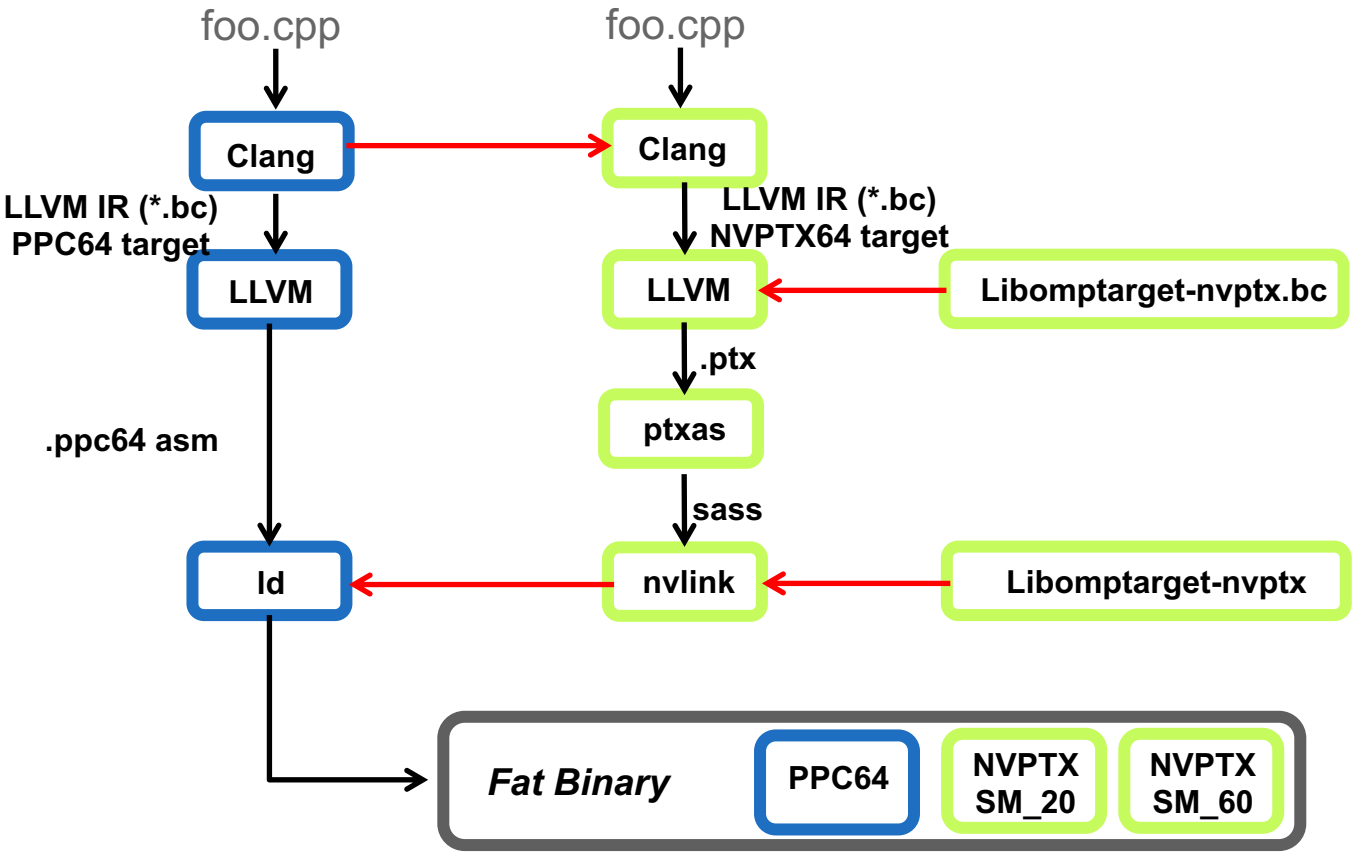
Replaces CUDA's host and device actions
- The offloading kind (e.g. OpenMP, CUDA)
- The toolchain used by the dependencies (e.g. nvptx, amd)
- Device architecture (e.g. sm_60)

Host to device dependency
- The host builds a list of target regions to be compiled for device

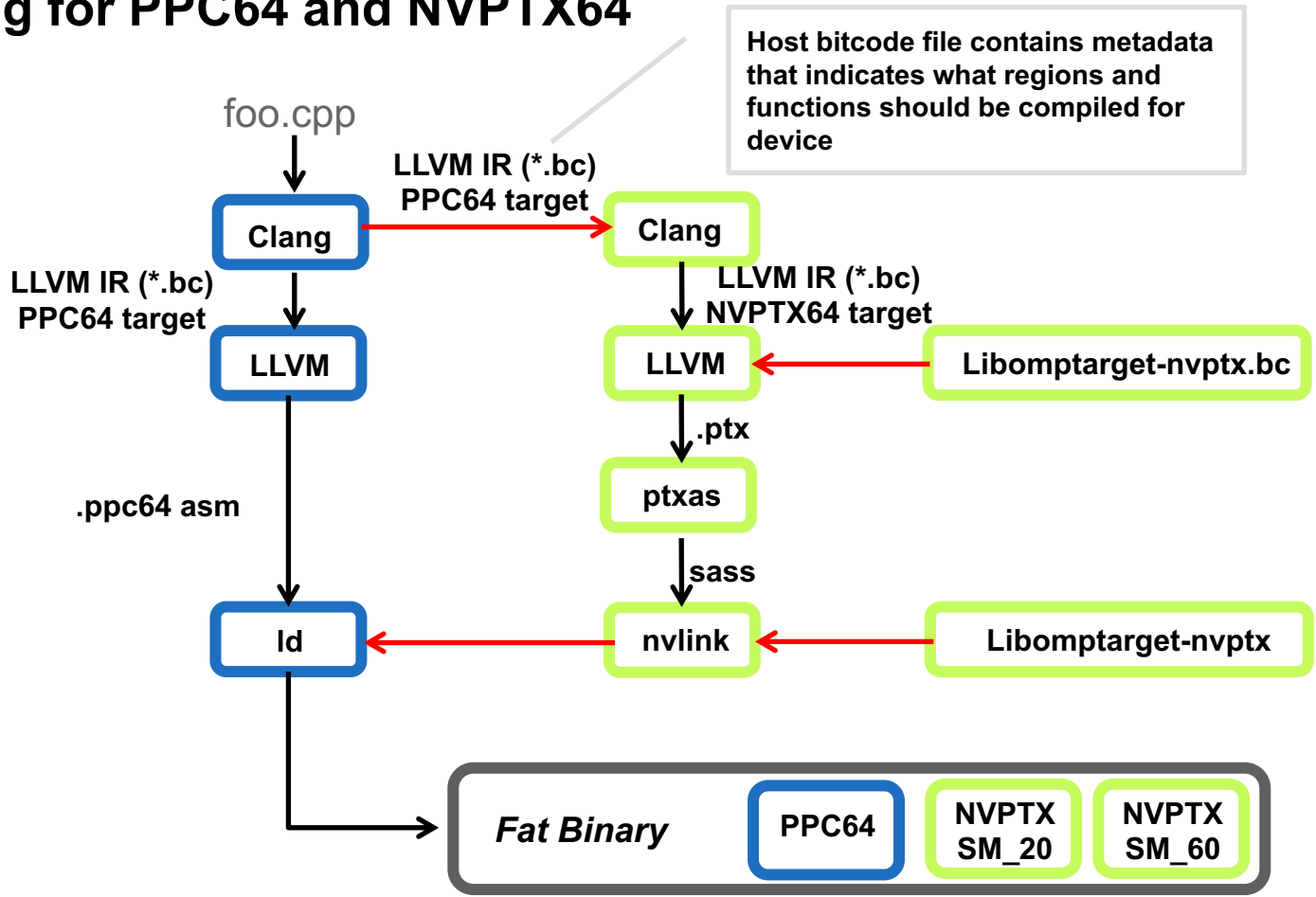
Device to host dependency
- Bundling of object code in single binary



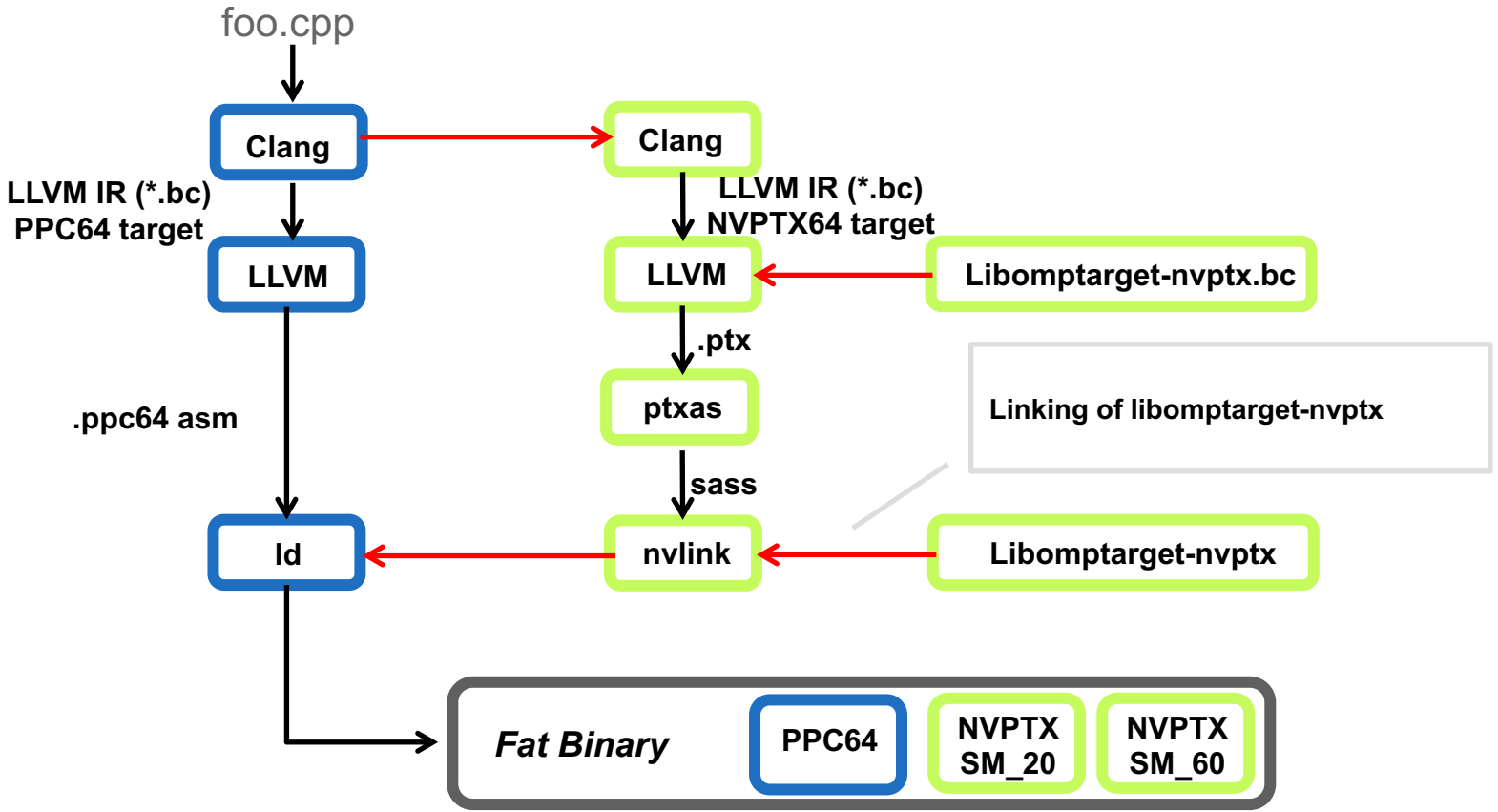
Offloading for PPC64 and NVPTX64



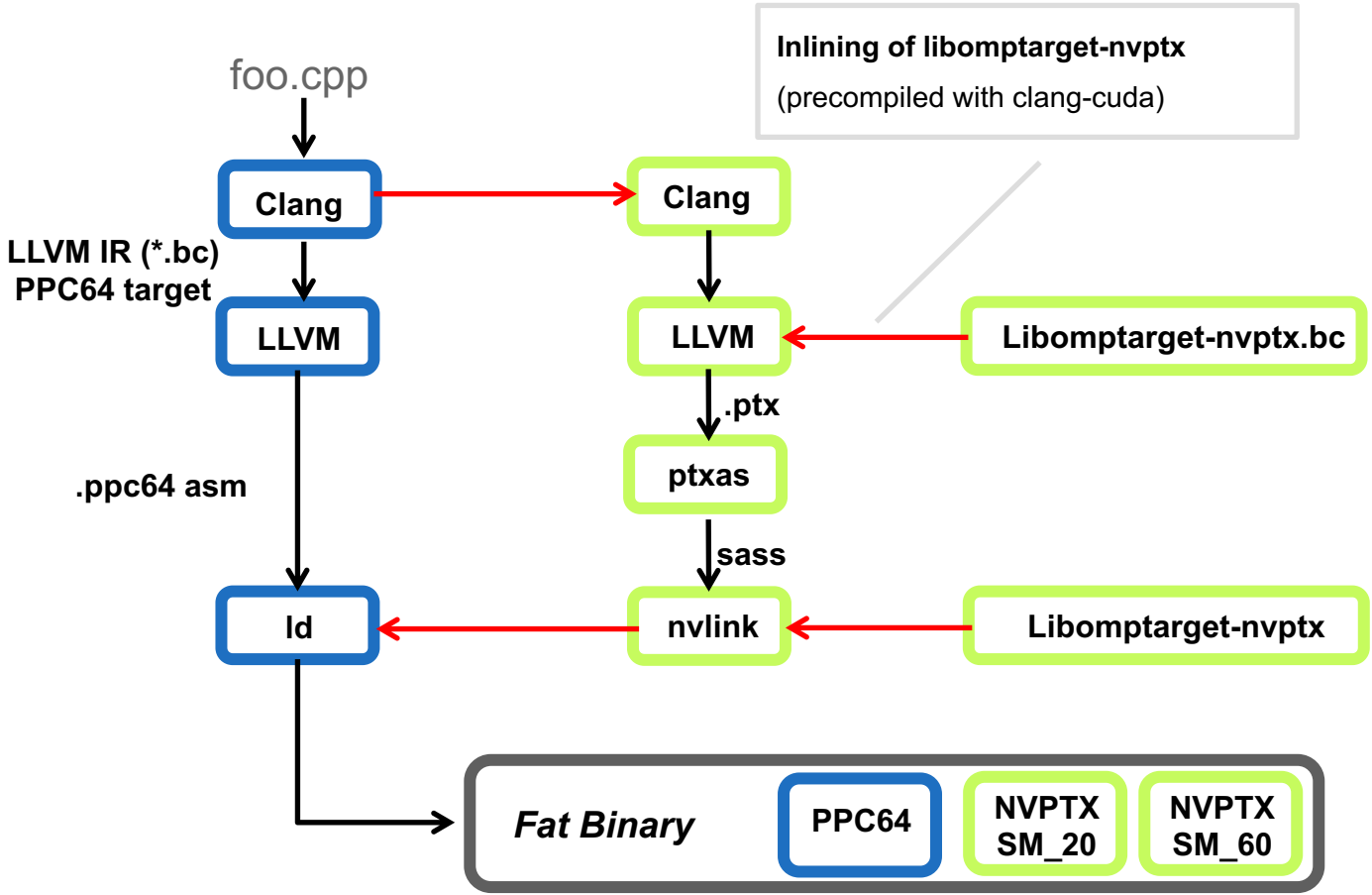
Offloading for PPC64 and NVPTX64



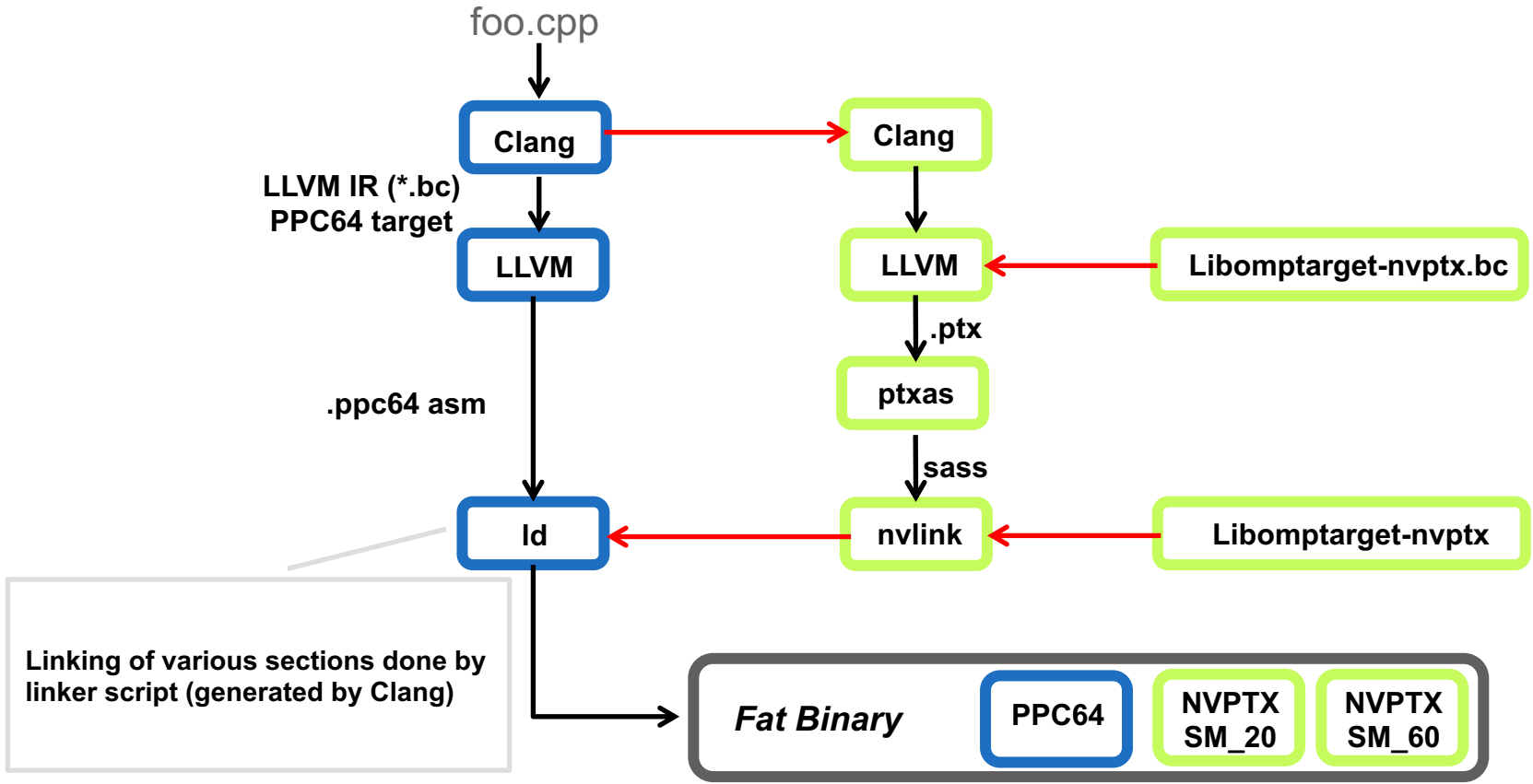
Offloading for PPC64 and NVPTX64



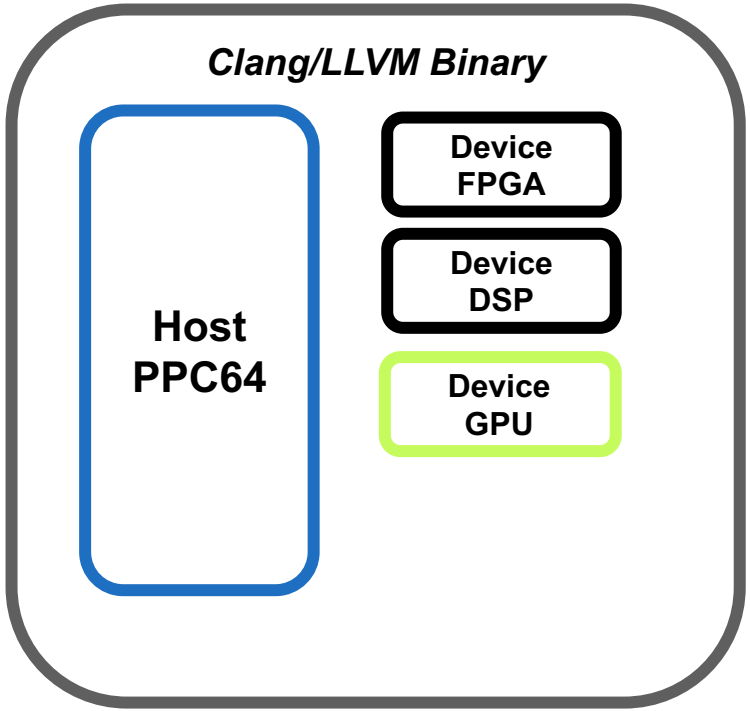
Offloading for PPC64 and NVPTX64



Offloading for PPC64 and NVPTX64



Fat Binaries and Libomptarget

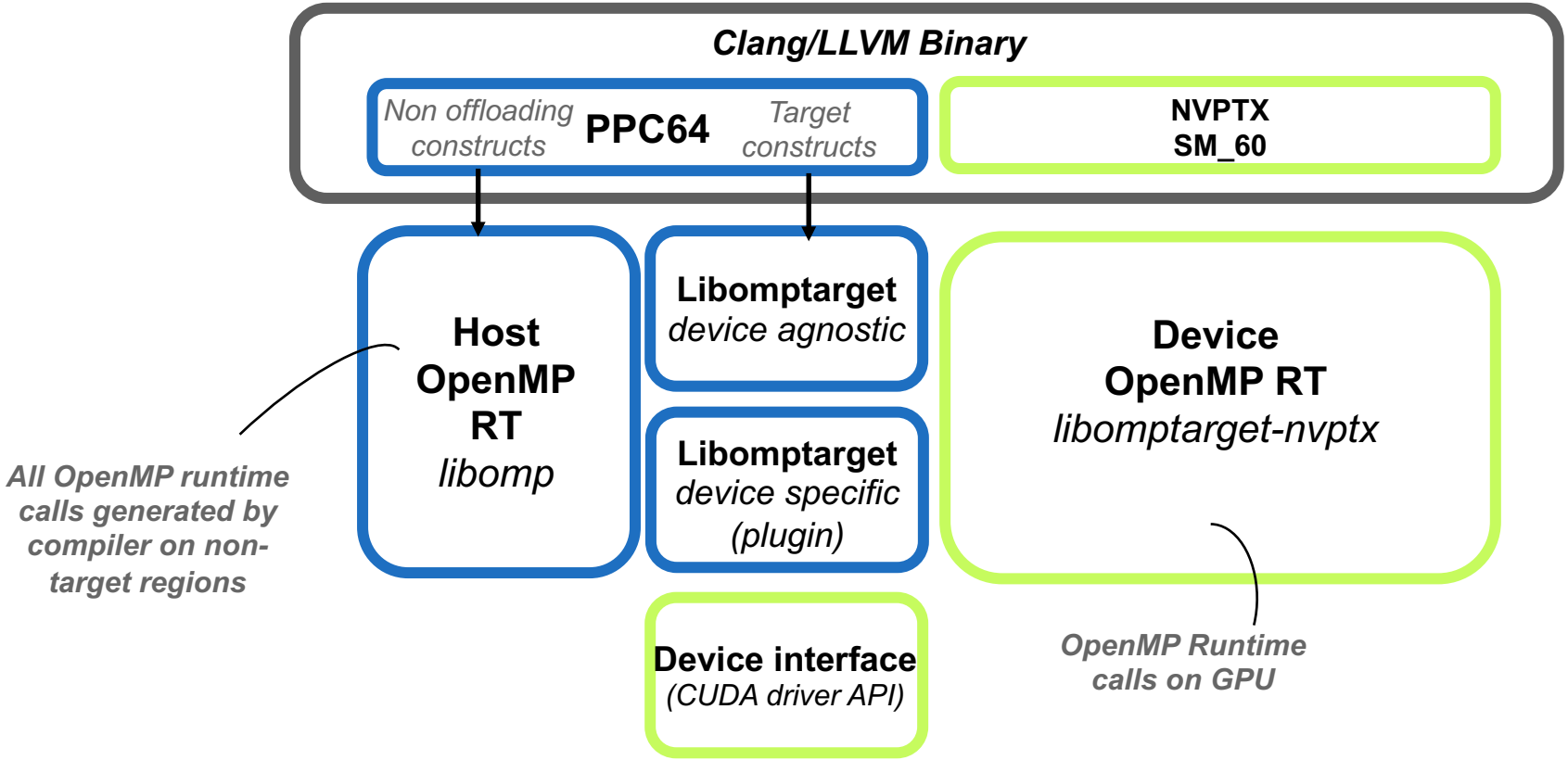


A single binary for multiple devices

- FPGA, DSP accelerator, GPUs, etc.

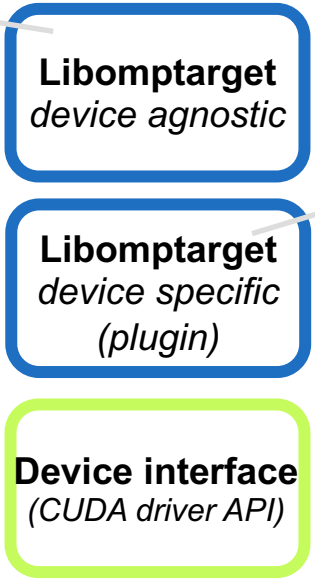
No need to recompile source if device section is already present

OpenMP Runtimes



Libomptarget Offloading Support

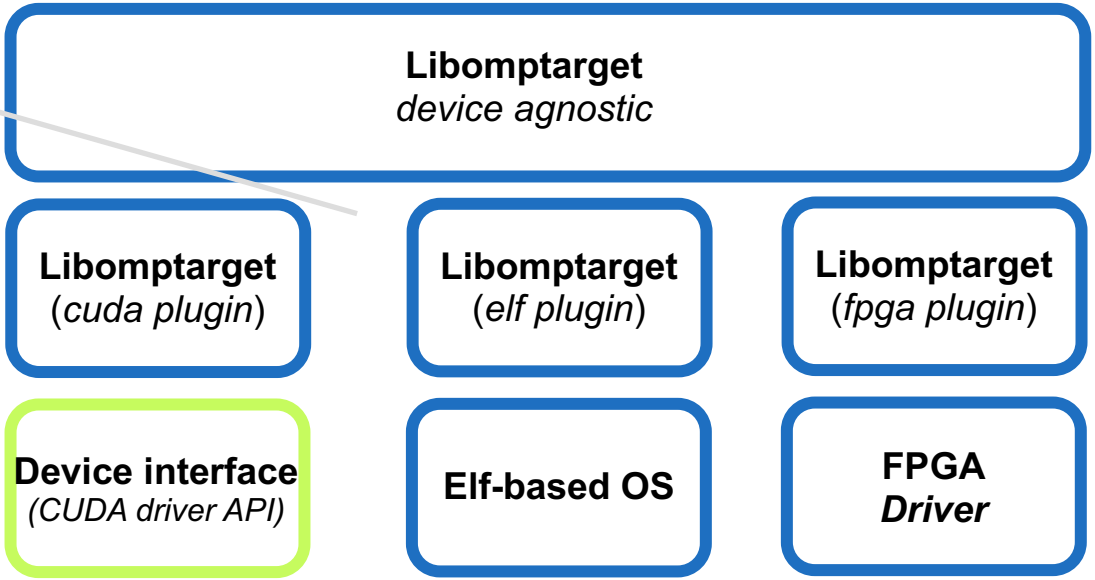
- Implement calls from generated code for offloading
  - map data, start target region, etc.
- **Not target a device type**
  - only device IDs



- Implement calls from libomptarget above it
  - map data, start target region, etc.
- **For a specific device type**
- Implementation is done by using device specific infterface

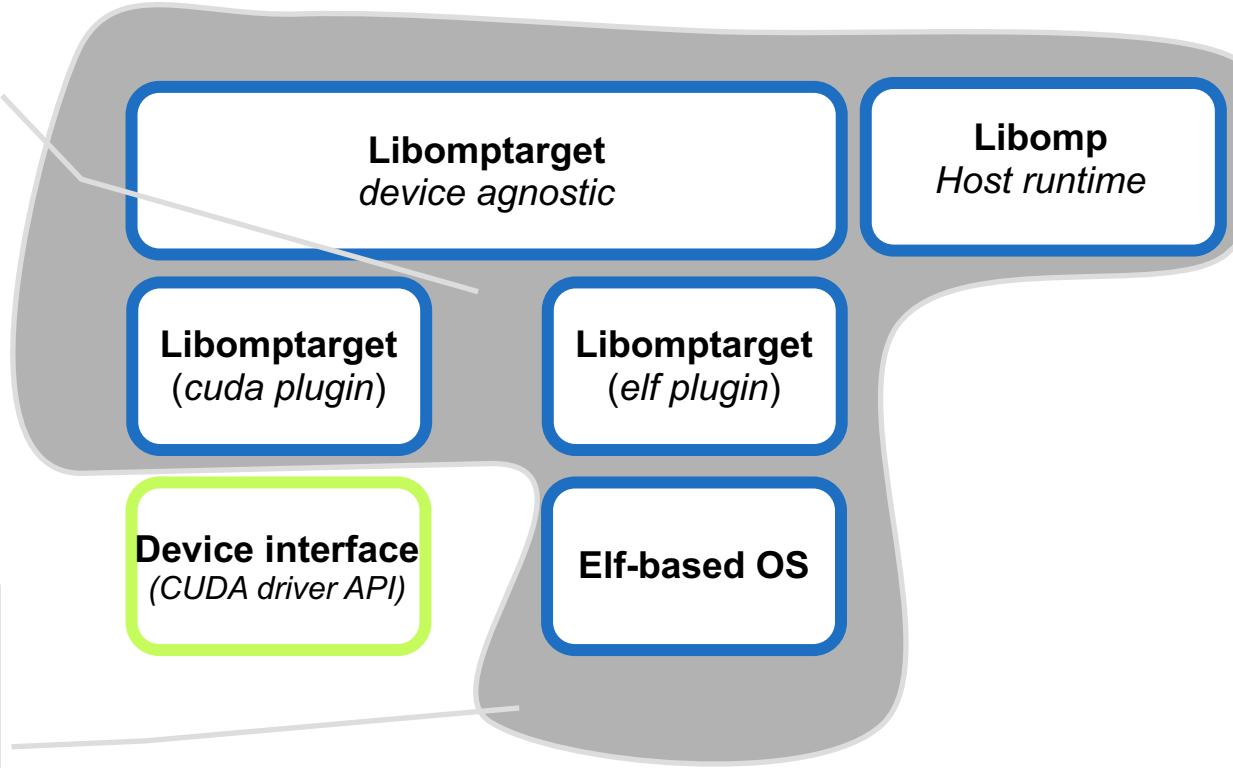
Libomptarget Offloading Support – Multiple Device Types

- Each device type requires to implement a **plugin**



Libomptarget Offloading Support – Multiple Device Types

- Each device type requires to implement a **plugin**
- **This design does not prevent optimized implementation of libomptarget where we have a single library**



- Optimized libomp for GPU device
- **Full inlining**

Libomptarget Binary Registration

Required to allow host to offload to device

- Device code may be produced by a device-specific toolchain
- Can be loaded dynamically: based on device availability

Some devices require special binary registration support

- NVIDIA GPUs

The compiler generates a CUBIN object section with

- All code (kernels and device functions)
- All static data

Libomptarget asks CUDA driver to load CUBIN

- cuModuleLoadDataEx
- Requires start memory address of CUBIN
- Libomptarget retrieves the symbol address from a host/device table
- Host/device table is populated by Clang, linker, and loader

Patch Status

Full driver implementation is available in trunk

- Thanks to everybody that made this happen!

Libomptarget is provided as three patches

- Host agnostic, plugins for ELF and CUDA, device library for NVPTX
- We are answering comments made by community (thanks!)

Code generation patches

- In preparation
- Working with Clang community to prepare code

Optimizations

- Several optimizations are already standard implementation in our internal repo
- Will have to come after full-fledged implementation

What is still missing – a lot!

Code generation for GPUs not yet available

- We are preparing patches
- Challenging to extract patches from full-fledged implementation of OpenMP 4.5. on GPUs
- Several design choices need to be discussed with community

Code generation for GPU is significantly different from CPU

- Due to SIMT GPU model
- No thread “spawning” or “recruiting”
- Cannot hide everything in runtime

Data sharing

- A single thread (team master) may need to share data with all other threads within its team
- Compiler needs to identify variables to be shared
- Requires multiple passes

How to play with full-fledged OpenMP 4.5. Implementation for GPU

Full-fledged implementation is available on Github

- https://github.com/clang-ykt
- Give it a try and report bugs

This will be discontinued as soon as upstreamed

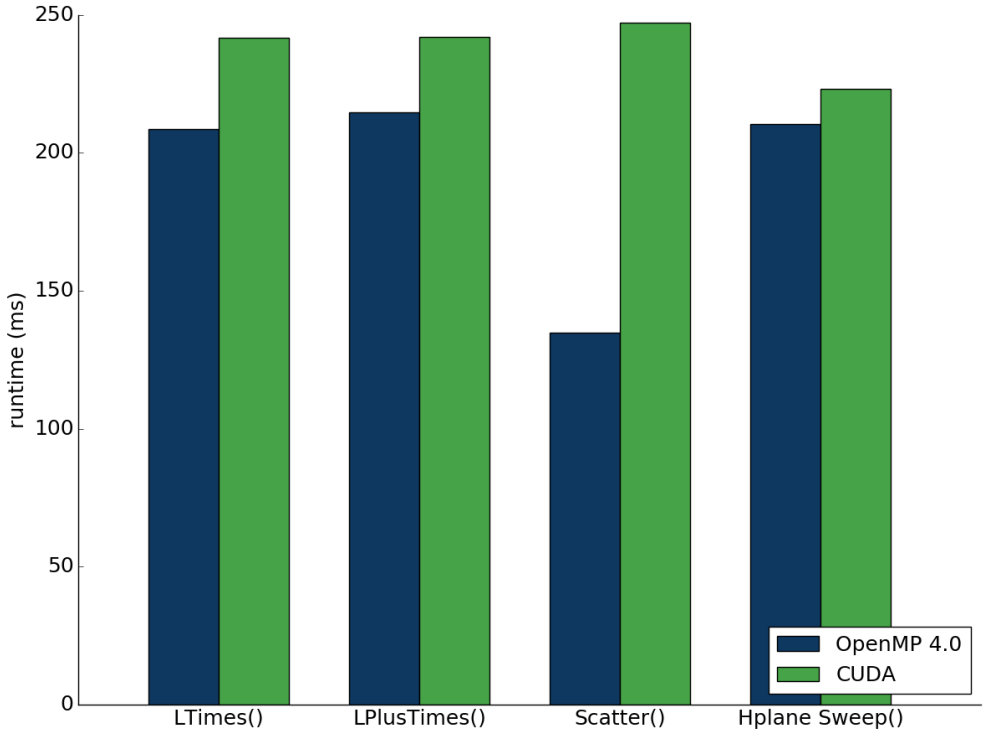
- Only meant as a collaboration hub before upstreamed
- Only expect bug fixes on this version
- New features will be done directly in trunk

Kripke Runtimes OpenMP vs CUDA

Porting to OpenMP and CUDA started at the same time

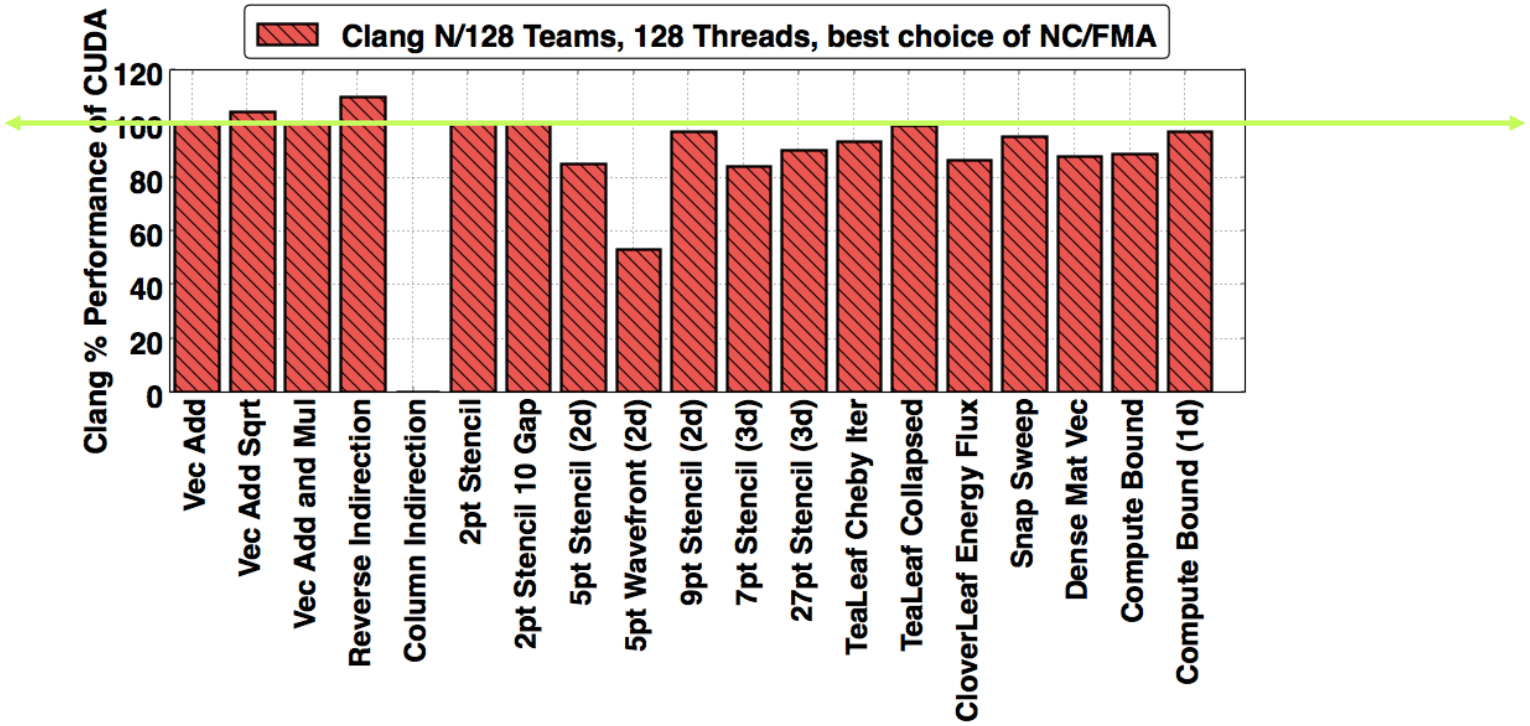
- OpenMP version with collapse
  - Complex code synthesis
  - Hard to reproduce in CUDA
- CUDA version uses multiple block dimensions
- Eventually CUDA catches up, after some debugging

OpenPower P8 and K40m NVIDIA GPU



David Appelhans. **Performance Portability Experience with LLVM, OpenMP 4, and Kripke**. DOE Centers of Excellence Performance Portability Meeting

Performance of Mantevo



OpenPower P8 and K40m NVIDIA GPU

Matt Martineau et al. **Performance Analysis and Optimization of Clang's OpenMP 4.5 GPU Support**. PMBS16.

Thank you!

ibm.com/systems/hpc

Legal notices

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectually property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER OR IMPLIED. IBM LY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, ed or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 1 0504- 785
U.S.A.

Information and 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.

Special notices

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquires, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

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

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.