

SIG OpenXLA Community Meeting



August 9, 2022

Agenda

- Introductions (10 min)
- Roadmap updates & discussion (20-25 min)
- SIG collaboration (15 min)
- Next steps (5 min)

Mission

**Open, state-of-art ML
compiler, built collaboratively
with Hardware & Software
communities, using the best of
XLA & MLIR.**



Introductions & housekeeping

OpenXLA Meetings

- Monthly on Google Meet
- Rotating meeting host & scribe
- Proposed agenda shared by host 1 week prior in [community wiki](#) for feedback
- Meeting minutes & slides shared publicly on community wiki within 24 hours
- Meetings should include:
 - Roadmap & development updates
 - Design proposals
 - Community topics

Introductions (<10 minutes!)

- SIG Member Orgs:
 - AMD
 - Apple
 - ARM
 - AWS
 - Google (XLA, TensorFlow, JAX, PyTorch/XLA)
 - Intel
 - Meta
 - NVIDIA
- Any new attendees? What are you looking to focus on?



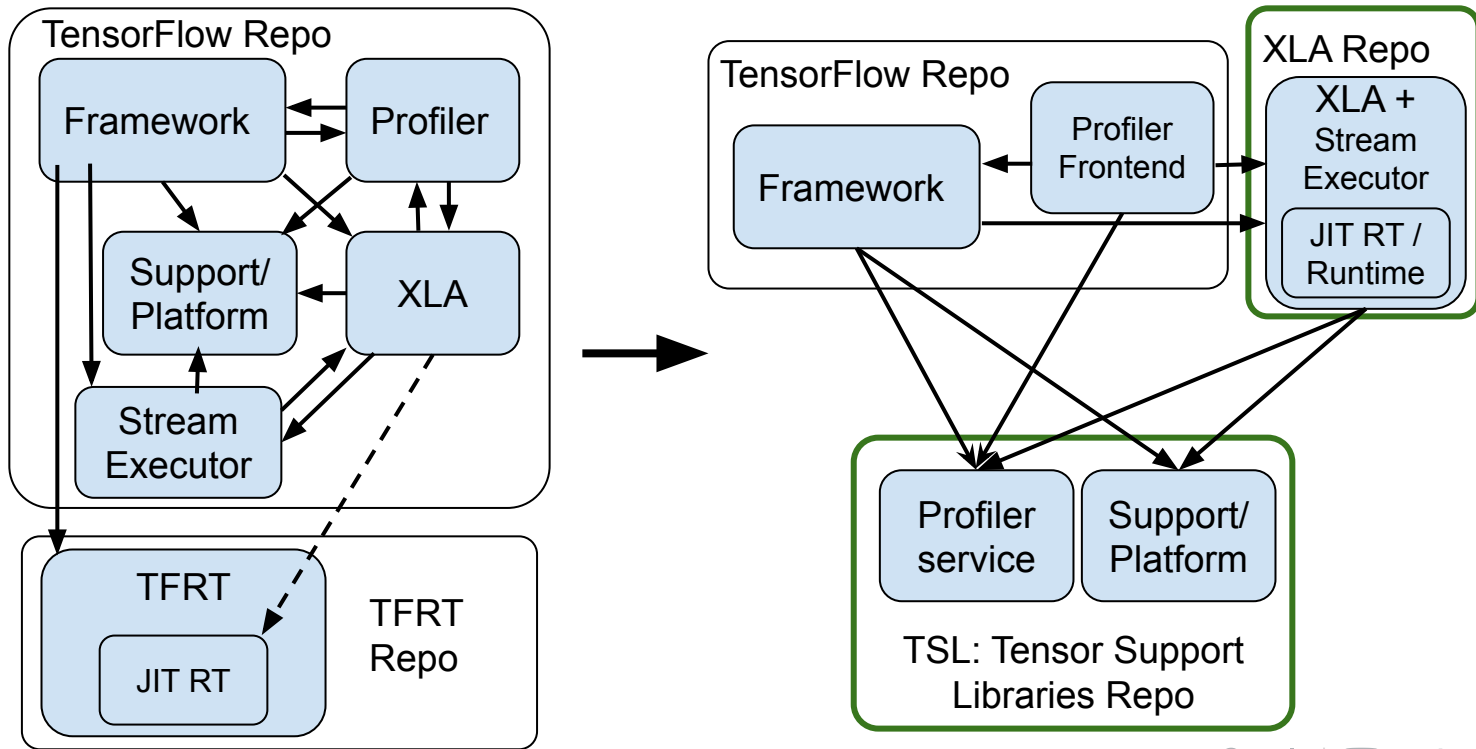
Technical Updates

TensorFlow/XLA Refactoring

Extracting XLA from TensorFlow

A three repos solution

<https://github.com/openxla/xla>



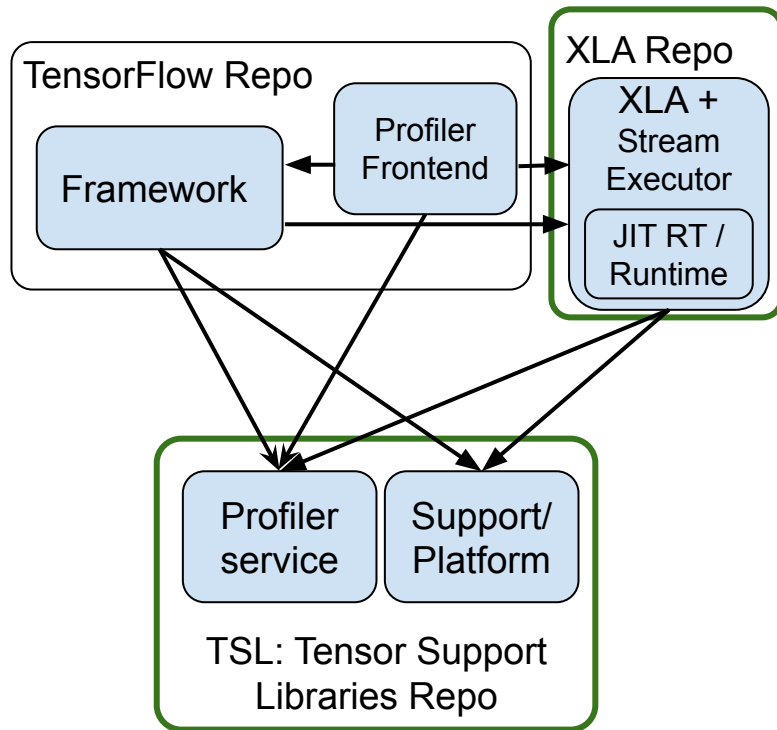
Extracting XLA from TensorFlow

A three repos solution

<https://github.com/openxla/xla>

- TSL (Tensor Support Libraries) is a new repository that will contain TensorFlow platform-independent portability code ([tensorflow/core/lib](#) and [tensorflow/core/platforms](#)) as well as the profiler service. These are the common dependencies between XLA and TensorFlow.
- XLA repo integrates StreamExecutor and JITRT. XLA will depend only on TSL to get platform independent utilities as well as profiler APIs.
- TensorFlow will depend both on XLA and TSL. For now we will vendor the XLA and TSL code into [tensorflow/third_party](#) to avoid cross-repo synchronization. Code depending on TensorFlow, like the TF/XLA bridge, will continue to stay in the TensorFlow repository.

Expected to deliver: 10/2022



OpenXLA Roadmap (Proposed)

What Google plans to contribute as a member of OpenXLA

Google's roadmap contributions subject to change

8/9/2022

Performance & Scalability

2022

2023

Goals

- Deliver out-of-the-box performance that meets current user needs
- Build MLIR-based infrastructure to scale with future needs

- Best-in-class performance for training and inference as measured by throughput, net latency, compile time, and peak memory
- Unlock new use cases for ML developers

Performance

- Improved fusion heuristics and use of shared memory on GPU
- Tile-and-fuse for cwise and reductions
- GPU codegen has SOTA performance for SoftMax
- Dynamism, sparsity, and quantization in StableHLO
- NVIDIA-driven cuDNN Graph API integration

- Unbounded dynamism in XLA CPU/GPU
- Sparsity and quantization in XLA GPU/CPU
- Improved codegen performance for convolutions and Matmul
- Tuning for vectorization
- Sophisticated fusion heuristics

Scalability

- Enable multi-host horizontal scaling unlocking ultra-large model training on GPU

Usability

2022

2023

Goals

- Up-level OSS developer productivity by investing in tooling, processes, and documentation

- 75% CSAT on testing, tooling, and metrics
- Unlock improvements to debuggability via MLIR adoption across compiler stack

Developer Experience

- StableHLO source-of-truth and development in GitHub
- Spec, test suite, and reference implementation for StableHLO v1.0
- Serialization, versioning, and backwards compatibility in StableHLO
- Well-documented compilation stack and capabilities

- OpenXLA has end-to-end build with partner integration

Debuggability

- Metrics dashboard v1.0

- Testing and metrics for XLA CPU/GPU to:
 - Triage complex workloads
 - Triage workload sensitivity to numerics
- Automatic MLIR reproducer generation
- Adopt and develop MLIR testing & debugging facilities

Hardware & Framework Optionality

2022

2023

Goals

- Build MLIR-based abstractions that unlock device optionality
- Provide a landing pad for all frameworks

- Increase velocity of hardware device and feature adoption
- Improve interoperability of models across frameworks, compilers, and hardware

Hardware
Optionality

- Improved layering to enable reuse and easier re-targeting
- Common runtime for CPU and GPU
- Enable easier addition of floating point formats
 - Support for bfloat16 today
- Feature parity between MHLO and HLO
- Investigate StableHLO on edge/mobile

- Investigate and enable support for different device runtimes
- Enable StableHLO on edge/mobile in TensorFlow Lite stack

Framework
Optionality

- Launch StableHLO v1.0
 - Compatible with TensorFlow, JAX, PyTorch/XLA, Torch-MLIR,¹ XLA, and IREE
- Explore alignment of StableHLO with other opsets (e.g. TOSA, ONNX, MIL)
- Improved interoperability between compiled and non-compiled code

- Further adoption of StableHLO across Google-supported systems

¹ Thank you to Torch-MLIR contributors, especially Alibaba and ByteDance.

2024 and beyond

Goals

- Cultivate 3P ecosystem of layers, plug-ins, and back-ends around XLA
- Expand framework and (edge/mobile) hardware optionality
- *"Train anywhere, serve anywhere"*
- Deliver further performance improvements for customers

Initiatives

- AOT machine learning compiler for 3P edge/mobile ecosystem
 - Exploring opportunities to achieve this in 2023
- Consolidate framework to StableHLO translation
- Enable more device-specific optimization via extension mechanism
- Automated heterogenous execution of models across CPU, GPU and accelerators
- Use ML to improve/replace heuristics in ML compilers

StableHLO

openxla/stablehlo

- [The repository](#) has been created! 🚀
- GitHub-first development
 - GitHub is the source of truth
 - Development via GitHub pull requests
 - Testing via GitHub actions
 - Planning via GitHub issues
 - Discussions via GitHub discussions / Discord (TBA)

In the short term

- Bootstrap the repository
 - Fork MHLO* from MLIR-HLO and call it StableHLO
 - Move CHLO from MLIR-HLO
 - Pull request: [openxla/stablehlo#1](https://github.com/openxla/stablehlo/pull/1)
- Vendor the repository into MLIR-HLO
 - Regularly integrate changes from the repository into MLIR-HLO
 - Introduce a conversion from StableHLO to MHLO

* Except for the 8 MHLO ops which are private to XLA

In the longer term

- **Workstream #1: Stable version of HLO/MHLO**
Specification, test suite, reference implementation - ETA: H2 2022
- **Workstream #2: Evolution beyond what's currently in HLO/MHLO**
Ongoing work on dynamism, sparsity, quantization and extensibility - ETA: H2 2022
- **Workstream #3: Adoption of StableHLO**
Support for ML frameworks (TensorFlow, JAX, PyTorch/XLA) and ML compilers (XLA and IREE) - ETA: H2 2022

Community discussions

- [Coordinate LLVM commits for different project](#)
 - Significant need in compatibility guarantees for dialects like MHLO
 - We can already start providing guarantees for StableHLO
 - In the longer term, also leverage serialization & versioning RFCs in MLIR upstream
- [\[RFC\] Proposal for a high-level ML dialect in MLIR](#)
 - "We believe that we need a dialect that is at a higher-level of abstraction than, say, Linalg"
 - StableHLO / MHLO are pretty related work
 - OpenXLA's collaboration & governance model are a great fit for these conversations!



SIG Collaboration

Collaboration channels

Channel	Content	Access	Archive
OpenXLA GitHub Org	Code, Design proposals, PRs, Issues, Roadmaps	Public	n/a
Community repo	Governance, meetings, code of conduct	Public	Public
GitHub Discussions (Community)	Meta discussions on openxla/community repo	Public	Public
GitHub Discussions (Technical)	Technical discussions on individual repos - stablehlo, xla	Public	Public
Discord Server	Sync discussions	Open invites	Archived chats
SIG Meetings	Monthly live meetings	Public	Public agenda, slides, meeting minutes
<i>SIG Google Drive</i>	<i>Shared docs, decks</i>	<i>Read-only to non members</i>	<i>Indefinite</i>

Thoughts on meetings/workstreams

- **Meeting cadence & scope**
 - Monthly high-level alignment(?) meeting
 - Optional every other week technical deep dive
- **Technical workstreams / working groups.** Initial ideas:
 - Fusion
 - StableHLO
 - DevInfra (testing, sec, release, etc)
- **Non-technical workstreams / working groups.** Initial ideas:
 - Community outreach (events, blog, etc)
 - Documentation

Open questions

- **Membership**
 - Public membership expectations
 - MEMBERS.md file
 - GitHub Org membership / teams
- **Proposal process**
 - PR-based (eg TensorFlow, Swift)?
 - Docs-based



Next steps

Next steps

- Find standing monthly meeting slot(s)
- Kick off technical workstreams/focus groups
- Establish proposal / RFC process
- Draft & discuss next version of project governance
- Set up openxla/xla repo - CI/CD, issue labels, etc