Out Of Process Execution for Clang-Repl

GSOC 2024 Project by Sahil Patidar
Mentors - Vassial Vassilev, Matheus Lzvekov

About me

I’m Sahil Patidar.
Bachelor of Technology (2023), Computer Science and Engineering
at Vindhya Institute of Science and Technology, Indore, India.
Worked on small issues and optimizations in LLVM’s InstCombine.

Introduction to Clang-Repl

Clang-Repl is an interactive C++ interpreter that allows developers to write, compile, and run C++ code in real-time. Built on top of Clang and LLVM's Just-In-Time (JIT) compilation infrastructure, Clang-Repl provides immediate feedback, making it an excellent tool for learning, testing, and rapid prototyping of C++ code. Its interactive nature helps streamline the development process by allowing users to experiment and iterate quickly.

Current State of Clang-Repl

The current design of Clang-Repl runs everything in the same process. This leads to two main problems:

1. High Resource Usage: The in-process model requires a lot of resources, making it difficult to use Clang-Repl on devices with limited power, like Arduino.
2. Stability Issues: If user code crashes, it brings down the entire Clang-Repl process, affecting the stability and user experience.

These issues limit the usability and reliability of Clang-Repl on a wide range of devices.

Project Goals

Out-of-Process Execution: Transition Clang-Repl to an out-of-process execution model, separating the execution of code from the main Clang-Repl infrastructure. This will reduce the resource footprint on the main application and make it suitable for devices with limited resources.

Work to be achieved

- Integrate Out of Process Execution Support for Clang-Repl.
- Orc Runtime Enhancements.

Out Of Process Execution Integration to clang-repl

- We will use llvm-jitlink-executor, which supports TCP and pipe-based communication with the controller process.
- To communicate with llvm-jitlink-executor from clang-repl, we'll introduce two flags: oop-executor and oop-connect.
    - The oop-executor flag will be used to launch the executor, requiring llvm-jitlink-executor path.
    - The oop-connect flag will specify the host and port for the connection.

Once the executor is launched, it will return SimpleRemoteEPC, which facilitates communication with the executor. We will integrate the executor into clang-repl to use SimpleRemoteEPC instead of SelfExecutorProcessControl and use the ORC runtime with the ORC native platform. This setup will allow the executor to execute code on its side and send the results back to the main process.

Orc Runtime Enhancements

The Orc runtime enhancements to prevent reinitializing initializers by implementing dlupdate, which only runs initializers that have been updated recently. This optimizes performance and reduces unnecessary overhead.

Future Scope

- Investigate methods to restart or continue sessions upon a crash.
- Explore existing solutions and best practices in session recovery.
- Design a versatile reliability approach for crash recovery.
- Implement strategies to enhance system robustness and fault tolerance.

Thank you!