Compiler Project Readme

Atharv Rajendra Jagtap (2101088)
Gaurav Kumar Patel (2101075)
Kushagra Patel (2101109)

Date: 04/03/2024

Problem Statement:

Design an LLVM pass to partition C++ code into regions based on power consumption, where regions are functions of the program. Also, print the ID of the desired function.

1 LLVM Overview

LLVM (Low Level Virtual Machine) is a compiler infrastructure project that provides a set of modular and reusable compiler and tool-chain technologies. It is widely used in both industry and academia.

1.1 Components of LLVM

- Front-end for different programming languages
- Mid-level Intermediate Representation (IR)
- Optimization passes
- Code generators
- Debuggers
- Just-In-Time (JIT) compiler

1.2 Features of LLVM

- Modular and reusable components
- Support for various programming languages: C, C++, Objective-C, Swift, Rust, etc.
- Flexibility to target a wide range of hardware platforms
- Extensibility for custom tool-chains and optimizations

2 Project Details

2.1 Assumptions

a) Energy consumption is measured using the Energy-interference-free Debugger (EDB) with voltage watchpoints.

b) We already have assumed (approximated with respect to time required for each instruction) power values for operations like add, sub, etc.

2.2 Actual Methodology

- **Energy Measurement Setup:** Utilize EDB connected to the capacitor on the target device to record capacitor voltage at watchpoints.
- **Voltage Watchpoints Placement:** Place voltage watchpoints in the code to mark start (Vfrom) and end (Vto) of sections to measure energy consumption.
- **Energy Calculation:** Calculate energy consumed between watchpoints using the formula $E = \frac{1}{2}C(V_{from}^2 - V_{to}^2)$.
- **Effective Capacity Calculation:** Compute effective capacity using Von and Voff voltages for accurate energy calculations.
- **Measurement Precision:** Ensure precise voltage measurements using EDB for accurate energy consumption calculations.
- **Power Calculation:** Calculate average power by dividing energy by execution time for accurate power representation.

2.3 Reference paper

You can access the research paper related to this project by following this link:

https://dl.acm.org/doi/pdf/10.1145/3178372.3179525

3 Steps to Install and Run LLVM

3.1 Install LLVM

If you are using a Linux-based operating system, you can install LLVM and Clang using the following commands:

```
sudo apt-get install llvm clang
```

3.2 Verify Installation

After installation, verify that LLVM and Clang are correctly installed by checking their versions:

```
llvm-config --version
clang --version
```

3.3 Running LLVM Commands

Once LLVM and Clang are installed, you can use the following commands to compile and run LLVM programs:

```
clang++ -S -emit-llvm test.cpp -o test.ll
clang++ -shared -o printInst.so printInst.cpp `llvm-config --cxxflags --ldflags --libs` -fl
opt -load ./printInst.so -printInst -enable-new-pm=0 < test.ll > test-inst.ll
```

These commands demonstrate compiling C++ code to LLVM IR, creating a shared object file, and running an LLVM pass on the LLVM IR file.

4 Output

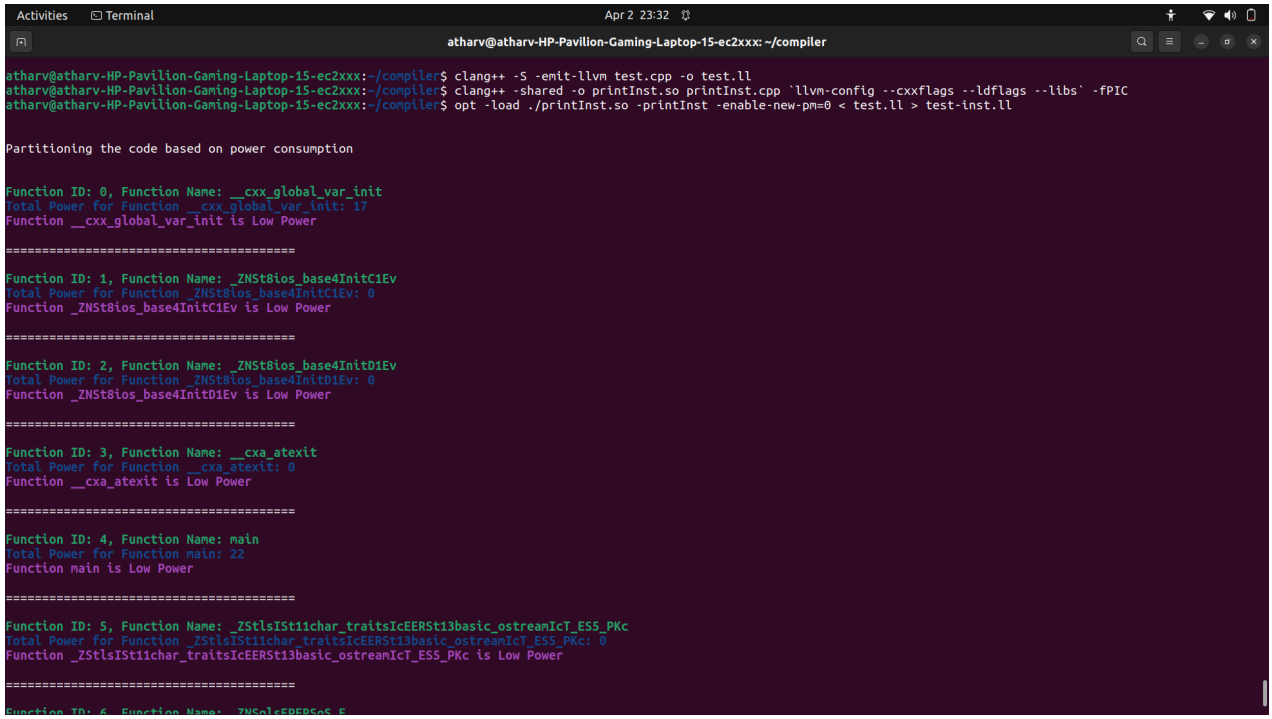

Figure 1: First Image

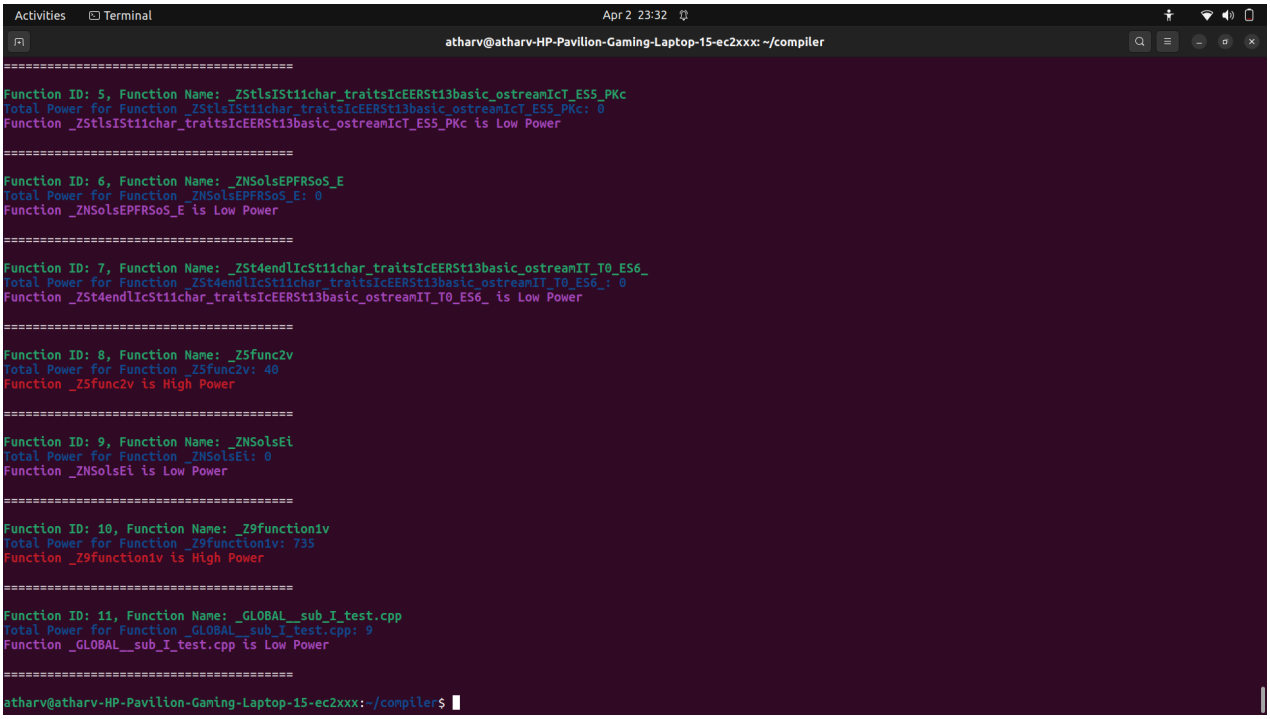

Figure 2: Second Image

5 Conclusion

By implementing the described methodology and using LLVM infrastructure, we can determine the power consumption of specific operations . This approach is beneficial for fine-grained energy profiling and optimization in embedded systems.