-mcpu

A Journey to the Last cycle of Performance

Siddhesh Poyarekar
What are we doing here?

- Appreciate CPU architecture
- Really Appreciate what goes into CPUs
- How compilers deal with the CPU
- Plug my other talks!
Who am I?

- I hack on gcc and glibc
- I lead a team that hacks on gcc and glibc...
  - ... and does microarchitecture optimizations
- I work at Linaro
int main (void) {
    return 0
}
Von Neumann redux - The CPU

FETCH
DECODE
CONTROL UNIT
INTEGER UNIT
FLOATING POINT UNIT
EXECUTE
REGISTERS
rax, rbx, rcx, rdx, r8, r9...
COMMIT
RAM
CPU
Nov Neumann's real face

Image by Hiroshige Goto
The compiler’s view of Von Neumann

- Assumes a safe default
  - For Intel chips, assumes a reasonable older version
- The ISA needs to be accurate, everything else is an approximation
  - Prefetching
  - Caching
  - Scheduling
  - Instruction reordering
  - Speculative Execution
  - Parallelism

Come for my discussion on Hardware Vulnerabilities!
Microarchitectures Rule

- Everyone (including compilers) like to pretend that they don't care
  - But they do! A lot!
- Simplifies the view for higher level programmers
- Helps build complex abstractions
- But we leave a lot of performance on the table
  - Banks care
  - Scientists care
  - Google, Facebook and Government Spy Agencies care
What is a microarchitecture?

- The real logic underneath
  - Does a mov of 128 bytes work as is or break up into 2 instructions moving 64 bytes each?
- Decides what compute units go in and how they interact
  - IBM Power 7 had 4 floating point units while the average Intel processor has just 1
  - Separate load and store units to deal with memory reads or writes
- In order vs out of order
  - Do I want to be Spectre/Meltdown capable?!
- Prefetching Behaviour
  - Hardware vs Software prefetching
  - When user issues a prefetch instruction, do I prefetch or just snigger at their naivety?
- Branch prediction logic
- Register banks (hardware file vs virtual) and Register Renaming
A 30 sec view of a compiler

1. Read Program Code
2. Optimize Program Logic
3. Generate Machine Code
A 60 sec view of a compiler

The front end: A dozen or so programming languages!

The middle end: A few HUNDRED optimization passes!

The back end: Translate code into the machine instructions.

Read Program Code

Optimize Program Logic

Generate Machine Code

Intermediate Representation!
A 90 sec view of gcc

- Read Program Code
- Optimize Program Logic
- Generate Machine Code

gcc/passes.def
gcc/tree-*
...

gcc/c-family/*
gc/cp/*
gcc/fortran/*
...

gcc/config/<arch>/*

GIMPLE
RTL
A 90 sec view of LLVM

- Read Program Code
- Optimize Program Logic
- Generate Machine Code

Clang. It's a separate project!

lib/*

LLVM IR

lib/Target/*
A 120 sec view of a compiler

The front end: A dozen or so programming languages!

The middle end: A few HUNDRED optimization passes!

The back end: Translate code into the machine instructions.

Read Program Code

Optimize Program Logic

Generate Machine Code

Tweak Optimization pass Behaviour to suit the microarchitecture.

Tweak Translation of code to suit the microarchitecture.

Intermediate Representation
A 120 sec view of gcc

The front end: A dozen or so programming languages!

The middle end: A few HUNDRED optimization passes!

The back end: Translate code into the machine instructions.

Read Program Code

Optimize Program Logic

Generate Machine Code

gcc/config/<arch>/*.md

gcc/config/<arch>/*.c

GIMPLE RTL
A 120 sec view of llvm

The front end: A dozen or so programming languages!

The middle end: A few HUNDRED optimization passes!

The back end: Translate code into the machine instructions.

lib/Target/<arch>/*.cpp

gcc/config/<arch>/*.td

GIMPLE RTL
Gcc Example

- Add Instruction patterns in *.md files
  - Conditional instructions like cmov vs branch and set
  - Specific register patterns that may influence a CPU feature
- Add a pipeline description
  - Model the CPU pipeline to influence the scheduler pass (gcc/haifa-sched.c)
- Add tuning structures
  - Specify relative costs of operations (e.g. choosing between addressing modes)
- Add custom passes
  - Do funky things that affect only this microarchitecture
But wait! There’s glibc too!

- String functions rule!
- Can **double** in performance in some cases
- Hand coded assembly implementations
  - Careful selection and ordering of instructions
  - Careful use of register numbers
- Deployed using ifunc mechanism
  - Detect CPU model at runtime
  - Patch the function entry point (PLT. Come to the Toolchain BoF to ask about it!)
- Implementations in sysdeps/<arch>/multiarch/*.S
That’s All Folks!

Come to the Toolchain BoF!

Get this slide deck at: https://siddhesh.in/mcpu.pdf