如何配置Emacs的C/C++语言LSP

索引的问题

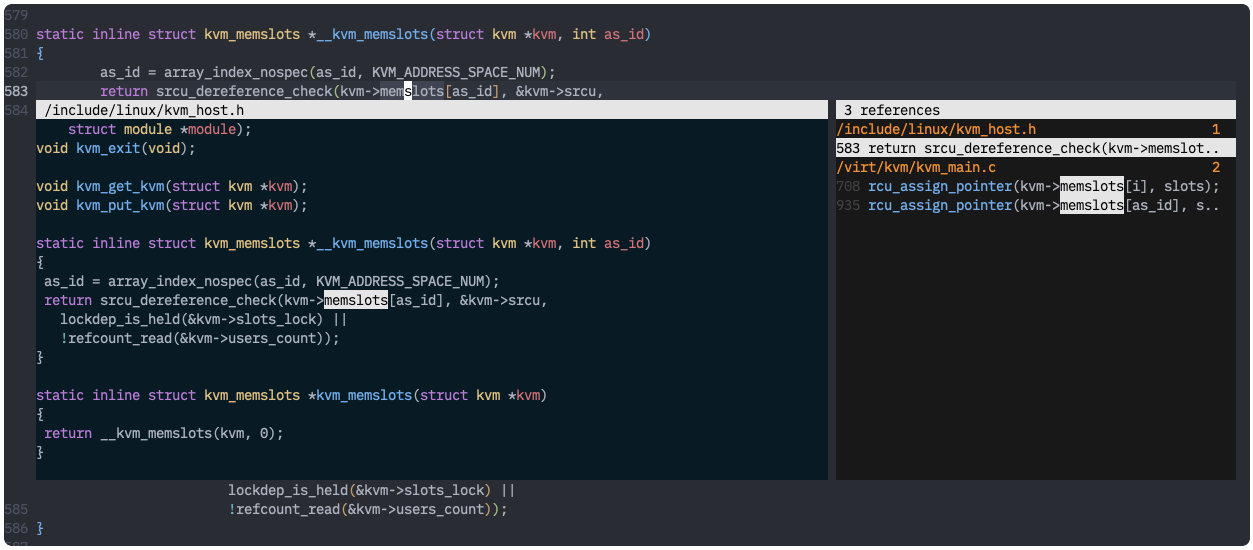
内核代码使用自己的头文件和一套复杂的编译脚本，用ctags/gtags等基于正则方式的工具产生的索引，在使用中跟lsp方式相比有很多不足：

- 没法直接在变量/成员上查看类型定义
- 在变量/成员上查看引用时，所有同名的（不同结构体中的、甚至不同作用域中的）都会显示（比如找一下某个lock域的引用），有时候几乎不具备参考价值
- 代码被各种复杂的宏定义包住，经常要花很大力气才知道哪些代码在起作用（比如preempt_enable的实现）
- 代码有多个arch的版本，每次查看定义或者引用都需要反复选择或者过滤
- 很多看起来像函数的代码其实是宏，如果当成函数会导致后面实际调试中找不到符号
- 没法方便的查看caller/callee hierarchy
- 省略（参考lsp能实现的功能，比如变量更名之类的）

以上是C语言，如果是C++代码那tags的问题就更多了

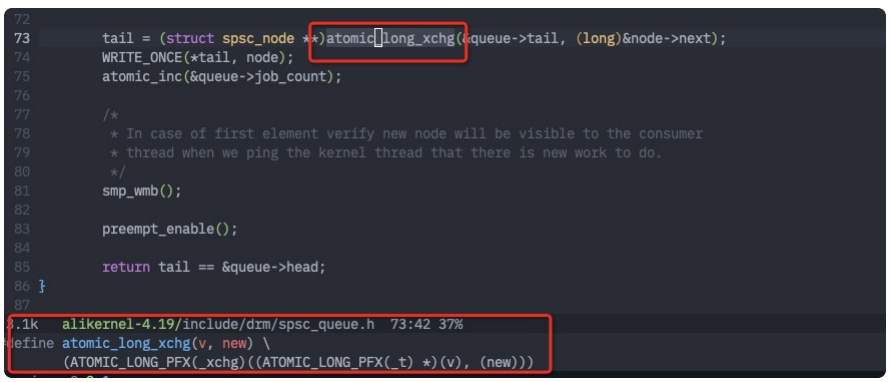
最终效果（LSP）

- 直接查看变量或者结构体成员引用/定义（快捷键：M-?/M-.）



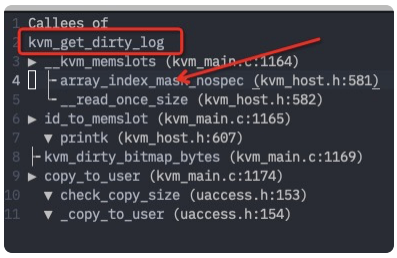
- 查看定义时，自动跳转至实际起作用的代码段，不起作用的代码灰色显示（preemt_disable函数）

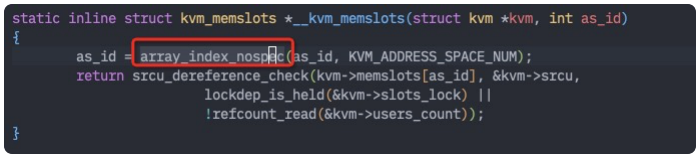
- 光标停留在符号上，直接显示符号信息，比如这里atomic_long_xchg能看到是个宏

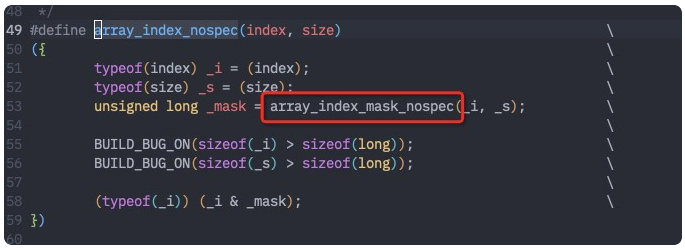


- 查看被调用栈（caller hierarchy）（快捷键：C-c s l 然后选择c）

- 查看调用链（callee hierarchy）（快捷键：C-u C-c s l 然后选择c）
  这里能看到__kvm_memslots调用的array_index_nospec其实是一个宏，直接展开成里面包住的函数了







- 更多的有待使用中探索

准备工作

- Emacs 27 或者更高版本（建议打开jansson支持，加速lsp协议传输）
- 升级gcc到9.2.1，不然ccls没法编译
- 安装llvm和clang
  sudo yum -b current install llvm11 clang11 clang11-devel llvm11-devel llvm11-static
- 安装高版本cmake（> 3.8.0）
  https://github.com/Kitware/CMake/releases

编译ccls

- 参考：https://github.com/MaskRay/ccls/wiki/Build

Bash 复制代码

```
git clone --depth=1 --recursive https://github.com/MaskRay/ccls
cd ccls
cmake -H. -BRelease -DCMAKE_BUILD_TYPE=Release
cmake --build Release

# 把 Release/ccls 放到 PATH 中
```

生成编译信息

- 参考：https://github.com/MaskRay/ccls/wiki/Example-Projects

Bash 复制代码

```
# 进入内核代码目录
mkdir out
cp configs/xxxxxx.config ./out/.config

make O=./out CC=/usr/bin/clang olddefconfig
make O=./out CC=/usr/bin/clang bzImage # generate .<target>.cmd files

# 新的内核直接用LLVM=1编译
# make O=./out LLVM=1 bzImage

# modules 建议只编译需要看的
make O=./out CC=/usr/bin/clang SUBDIRS=arch/x86/kvm

# 如果有objtool报错：Segmentation fault
# 在对应的目录Makefile里加入：
# OBJECT_FILES_NON_STANDARD := y
# 或者
# OBJECT_FILES_NON_STANDARD_foo.o := y

# 这是 upstream kernel 里的 scripts/clang-tools 中的脚本
gen_compile_commands.py -d ./out  # generate compile_commands.json
```

安装插件

参考：https://github.com/MaskRay/ccls/wiki/lsp-mode

直接git pull更新.emacs.d配置即可