

GEM5 与 NVMain 混合编译

张丘洋

2020 年 10 月 19 日

1 前言

NVMain 代码已好几年没有更新,其使用的 GEM5 接口早已被弃用,使用现在的 GEM5 代码无法进行混合编译。我在混合编译过程中遇到了超多的问题,主要是 python 版本(NVMain 必须使用 python2)问题、GEM5 版本较高问题以及代码 bug 问题。这里记录一下成功混合编译的方法以供参考。

2 (可选) 使用阿里云服务器

由于 NVMain 只能够使用 python2,在 GEM5 官网上描述最好在 Ubuntu 18.04 上使用 python2,所以以下都是在 **Ubuntu 18.04** 上进行配置的。为了方便我直接在阿里云上开了一个性能较高的临时服务器进行实验。

i 你也可以在虚拟机或本机上实验。用临时服务器的好处是随意折腾,不怕破坏本地环境,且可以租一个性能高的服务器编译 GEM5 减少编译时间。

推荐使用**抢占式实例**付费方式阿里云服务器,建议选用计算型 8 核 CPU

i 抢占式付费和按量付费必须要预充 100 元,不过使用完可以提现。不同地域、同一地域不同可用区之间价钱差别较大,注意挑选价格较低的(我所选的为 0.29 每小时)。

镜像选用 Ubuntu 18.04。

网络选择按量计费并将带宽峰值调到 50MB 以上。

密钥对填入本机密钥(或后期设置密码)。

可以设置服务器自动释放时间以免忘记释放。

之后可以在控制台得到服务器的公网 ip 并使用 `ssh root@xxx.xxx.xxx.xxx` 登录服务器。

❗ 做完实验别忘记释放服务器避免额外支出

3 编译环境配置

进入后首先执行 `apt update` 更新本地索引。之后运行以下命令安装 GEM5 依赖。¹

```
1 apt install build-essential git m4 scons zlib1g zlib1g-dev libprotobuf-dev
  ↳ protobuf-compiler libprotoc-dev libgoogle-perftools-dev python-dev
  ↳ python-six python libboost-all-dev pkg-config
```

`apt install` 命令第一次运行可能会出错，若出错可多执行几次。

以上命令来源于 GEM5 官网，由于我们使用的 GEM5 版本较老，该版本还需要安装 `swig`：

```
1 apt install swig
```

安装 `unzip` 来压缩 `zip` 文件：

```
1 apt install unzip
```

由于 GEM5 一直在更新，目前 GEM5 官网的版本都无法与 NVMain 中使用的 GEM5 接口相匹配。好在 github 上有一个仓库包含了较老的 GEM5 代码以及 NVMain 代码：<https://github.com/cyjseagull/gem5-nvmain-hybrid-simulator>

除了 GEM5 代码以及 NVMain 代码之外，运行 GEM5 全系统模拟还需要对应的 linux 镜像等文件，运行 PARSEC 负载需要有包含 PARSEC 的镜像文件，这些文件都可以在 GEM5 官网以及实验室网站上获取，为了方便这里打了一个包供大家下载。

在服务器上运行以下命令获取编译和运行所需的文件（该轻量级服务器带宽峰值较低，请耐心等待）：

```
1 wget http://jianyue.tech/gem5-nvmain.tar.xz
```

获取后使用 `sha1sum` 命令对其进行校验：

```
1 sha1sum gem5-nvmain.tar.xz
```

正确的校验和应为：`ba1d7fd74e281744e2a4188008a1257d501ed320`

¹https://www.gem5.org/documentation/general_docs/building

该压缩包文件包含了三个压缩文件，运行以下命令对其解压：

```
1 tar xJf gem5-nvmain.tar.xz
2 unzip gem5-nvmain-hybrid-simulator-master.zip
3 tar xJf fs-image.tar.xz
4 tar xJf parsec.tar.xz
```

由于 NVMain 只能够使用 python2 进行编译，保险起见将 `/usr/bin/scons` 文件的第一行改为：

```
1 #!/usr/bin/python2
```

4 混合编译

进入 GEM5 所在的目录：

```
1 cd gem5-nvmain-hybrid-simulator-master/gem5-stable/
```

首先按照 github 仓库的说明运行 patch 命令：

```
1 patch -p1 < final_patch/nvmain-classic-gem5-9850
```

此时我们就可以进行编译，不过我在编译的时候出现了一些问题，这些问题及其解决办法可以在下一章中找到，你可以先编译，遇到问题了再去解决然后重新编译，也可以先把所有可能出现的问题解决了再进行编译。使用如下命令进行编译：

```
1 scons EXTRAS=../nvmain build/X86/gem5.opt -j $((`nproc`*2))
```

5 编译过程中可能出现的问题及其解决办法

问题描述：

OSError: [Errno 2] No such file or directory:

```
File "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/SConstruct",
↳ line 1328:
```

```
    SConscript('src/SConscript', variant_dir = variant_path, exports =
↳ 'env')
```

```
File "/usr/lib/scons/SCons/Script/SConscript.py", line 614:
```

```
    return method(*args, **kw)
```

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 551:
    return _SConscript(self.fs, *files, **subst_kw)
File "/usr/lib/scons/SCons/Script/SConscript.py", line 256:
    call_stack[-1].globals)
File
↳  "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/build/X86/SConscript",
↳  line 358:
    SConscript(joinpath(root, 'SConscript'), variant_dir=build_dir)
File "/usr/lib/scons/SCons/Script/SConscript.py", line 614:
    return method(*args, **kw)
File "/usr/lib/scons/SCons/Script/SConscript.py", line 551:
    return _SConscript(self.fs, *files, **subst_kw)
File "/usr/lib/scons/SCons/Script/SConscript.py", line 256:
    call_stack[-1].globals)
File
↳  "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/build/X86/nvmain/
↳  SConscript", line 61:
    stdout=subprocess.PIPE, stderr = open(os.devnull, 'w'))
File "/usr/lib/python2.7/subprocess.py", line 394:
    errread, errwrite)
File "/usr/lib/python2.7/subprocess.py", line 1047:
    raise child_exception

```

解决办法:

该问题是由于hg命令缺少导致,由于我们不需要用到hg命令,所以可以直接将../nvmain/SConscript 文件中的 56-73 行删掉:

```

1  # Attempt to check the revision number of gem5
2  print "Checking gem5 revision number...",
3
4  # Use 'hg log' to get qparent revision number information
5  proc = subprocess.Popen([HG_COMMAND, 'log', '--template', '{rev}\n', '-r',
    ↳ 'qparent'],
6      stdout=subprocess.PIPE, stderr = open(os.devnull, 'w'))
7  proc.wait()
8
9  # If the return code is not successful, assume no patches are applied

```

```

10 gem5_rv = 0
11 if proc.returncode == 0:
12     gem5_rv = int(proc.communicate()[0].rstrip())
13     print gem5_rv
14 else:
15     print "N/A"
16
17 gem5_rv_define = '-DNVM_GEM5_RV=' + str(gem5_rv)
18 env.Append(CCFLAGS=gem5_rv_define)

```

问题描述:

AttributeError: Flag NVMain already specified:

File "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/SConstruct",
↳ line 1328:

```

SConscript('src/SConscript', variant_dir = variant_path, exports =
↳ 'env')

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 614:

```

return method(*args, **kw)

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 551:

```

return _SConscript(self.fs, *files, **subst_kw)

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 256:

```

call_stack[-1].globals)

```

File

↳ "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/build/X86/SConscript",
↳ line 358:

```

SConscript(joinpath(root, 'SConscript'), variant_dir=build_dir)

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 614:

```

return method(*args, **kw)

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 551:

```

return _SConscript(self.fs, *files, **subst_kw)

```

File "/usr/lib/scons/SCons/Script/SConscript.py", line 256:

```

call_stack[-1].globals)

```

File

↳ "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/build/X86/nvmain/
↳ Simulators/gem5/SConscript", line 40:

```
    DebugFlag('NVMain')
File "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/build/X86/
↳ SConscript", line 297:
    raise AttributeError, "Flag %s already specified" % name
```

解决办法:

似乎是因为该仓库的 GEM5 代码对 NVMain 进行了部分适配（可能是 patch 命令引起的）使得某个注册函数重复调用。

将 ../nvmain/Simulators/gem5/SConscript 第 40 行: DebugFlag('NVMain') 注释掉即可。

问题描述:

```
In file included from build/X86/dev/etherint.hh:41:0,
                  from build/X86/dev/etherint.cc:32:
build/X86/dev/etherpkt.hh:72:24: error: 'template<class> class
↳ std::auto_ptr' is deprecated [-Werror=deprecated-declarations]
```

解决办法:

该问题是因为 C++ 编译时启用了 -Werror 选项，将所有的 Warning 视为了 Error。

将文件 src/SConstruct 中的第 162 行中的 Werror=True 改为 Werror=False，第 946 行注释掉 error_env.Append(CCFLAGS='-Werror')。

问题描述:

```
build/X86/nvmain/MemControl/L0-Cache/L0-Cache.cpp: In member function
↳ 'virtual void NVM::L0_Cache::RestoreCheckpoint(std::__cxx11::string)':
build/X86/nvmain/MemControl/L0-Cache/L0-Cache.cpp:568:84: error: no match
↳ for 'operator<<' (operand types are 'std::basic_ostream<char>' and
↳ 'std::stringstream {aka std::__cxx11::basic_stringstream<char>}')
    std::cout << "L0_Cache: Warning: Could not open checkpoint
↳ file: " << cpt_file << "!" << std::endl;
```

解决办法:

该问题是因为某个类没有定义 << 操作符（可能跟 C++ 版本有关?），然而这个输出命令无关紧要，我们直接将其注释掉即可：将文件 ../nvmain/MemControl/L0-Cache/L0-Cache.cpp 中第 568 行注释掉。

6 运行过程中可能出现的问题及其解决办法

在编译好后可以先运行一下 SE 测试：

```
1 ./build/X86/gem5.opt configs/example/se.py -c
  ↪ tests/test-progs/hello/bin/x86/linux/hello --mem-type=NVMainMemory
  ↪ --nvmain-config=../nvmain/Config/PCM_ISSCC_2012_4GB.config
```

运行后可能出现以下问题：

问题描述：

Traceback (most recent call last):

File "<string>", line 1, in <module>

File "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/src/python/m5/
↪ main.py", line 388, in main

exec filecode in scope

File "configs/example/se.py", line 113, in <module>

Options.addCommonOptions(parser)

File

↪ "/root/gem5-nvmain-hybrid-simulator-master/gem5-stable/configs/common/
↪ Options.py", line 235, in addCommonOptions

parser.add_option(arg, type="string", default="NULL", help="Set NVMain
↪ configuration value for a parameter")

File "/usr/lib/python2.7/optparse.py", line 1021, in add_option

self._check_conflict(option)

File "/usr/lib/python2.7/optparse.py", line 996, in _check_conflict

option)

optparse.OptionConflictError: option

↪ --nvmain-config=../nvmain/Config/PCM_ISSCC_2012_4GB.config: conflicting

↪ option string(s):

↪ --nvmain-config=../nvmain/Config/PCM_ISSCC_2012_4GB.config

解决办法：

该问题是因为以 --nvmain- 开头的命令行参数被处理了多次，解决办法是将 configs/common/Options.py 文件中第 59–62 行注释掉：

```

1 for arg in sys.argv:
2     if arg[:9] == "--nvmain-":
3         parser.add_option(arg, type="string", default="NULL",
4                             help="Set NVMain configuration value for a parameter")

```

7 全系统模拟 (FS) 的配置

GEM5 的全系统模拟需要 linux 镜像作为系统内核，还需要虚拟磁盘文件作为系统硬盘及分区。

在下载压缩包包含了 fs-image 文件夹，里面是 GEM5 全系统模拟所需要的文件。这些文件及说明如下：

```

fs-image/
├── binaries/
│   ├── x86_64-vmlinux-2.6.22.9 ..... GEM5 官网镜像
│   └── x86_64-vmlinux-2.6.28.4-smp ..... 包含 PARSEC 的镜像
└── disks/
    ├── linux-bigswap2.img ..... 用作 linux swap 分区的虚拟磁盘文件
    ├── linux-x86.img ..... GEM5 官网的虚拟磁盘文件
    └── x86root-parsec.img ..... 包含 PARSEC 的虚拟磁盘文件

```

GEM5 使用环境变量 M5_PATH 来记录全系统模拟使用的文件文件夹。


编辑 ~/.bashrc 文件，在文件末尾追加一行：

```

1 export M5_PATH=/root/fs-image/

```

之后运行 source ~/.bashrc 命令更新。

 这里的 /root/fs-image/ 请根据实际情况修改对应的路径！

之后可以运行以下命令检查是否可以运行 FS 模拟：

```

1 ./build/X86/gem5.opt configs/example/fs.py --kernel=x86_64-vmlinux-2.6.22.9
   ↪ --disk-image=linux-x86.img

```

如果正常会输出以下内容：


```
1 gem5 Simulator System.  http://gem5.org
2 gem5 is copyrighted software; use the --copyright option for details.
3
4 gem5 compiled Oct 19 2020 20:17:59
5 gem5 started Oct 19 2020 22:16:50
6 gem5 executing on iZbp1915fj4otwjs6951bhZ
7 command line: ./build/X86/gem5.opt configs/example/fs.py
   ↪ --kernel=/root/fs-image/binaries/x86_64-vmlinux-2.6.22.9
   ↪ --disk-image=linux-x86.img
8 Global frequency set at 1000000000000 ticks per second
9 info: kernel located at: /root/fs-image/binaries/x86_64-vmlinux-2.6.22.9
10      0: rtc: Real-time clock set to Sun Jan  1 00:00:00 2012
11 Listening for com_1 connection on port 3456
```

若正常可按 Ctrl-C 结束。

这里第11行:Listening for com_1 connection on port 3456 说明模拟运行的 linux 系统输出可以通过 3456 端口查看。

使用 screen 来启用多个窗口，使其中一个窗口运行全系统模拟，另一个窗口连接到对应端口查看输出。


首先安装 screen：

```
1 apt install screen
```

之后运行命令 screen。

在新的窗口中重新输入之前的 FS 命令，之后按 Ctrl-a-d（先按 Ctrl-a，在按 d）暂时退出窗口（命令仍在后台执行），在 gem5-nvmain-hybrid-simulator-master/gem5-stable 目录下运行以下命令连接到系统输出：

```
1 ./util/term/m5term 127.0.0.1 3456
```

 具体的端口号请根据实际情况进行修改！

8 参考网址

1. https://www.gem5.org/documentation/general_docs/building
2. <https://github.com/cyjseagull/gem5-nvmain-hybrid-simulator>
3. <https://blog.shunzi.tech/post/gem5-and-nvmain/>
4. http://www.cs.utexas.edu/~parsec_m5/