بسمه تعالی

درس سیستم‌های نهفته مبتنی بر هسته

تکلیف کامپیوتری 1: یادگیری استفاده از ابزار *GEM5* و بررسی حالت های کاری ***in order*** و ***out of order*** در پردازنده ها

پردیس دانشکده‌های فنی دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

دکتر احمد شعبانی

نیم‌سال دوم سال تحصیلی 1402-1401

نگارش: علی ایمانقلی (ali.imangholi@ut.ac.ir)

Skype: live:.cid.42ec6b61e9878d2b

مقدمه:

هدف از انجام این تمرین کامپیوتری آشنایی با ابزار *GEM5* و بررسی حالت های کاری ***in order*** و ***out of order*** در پردازنده ها می باشد. بدین منظور در گام نخست تمرین ابزار *GEM5* را نصب نمایید و سپس در گام های بعدی مطابق با خواسته های صورت تمرین به بررسی موارد خواسته شده بپردازید.

گام اول:

به منظور نصب ابزار *GEM5* نیازمند است تا سیستم عامل مبتنی بر لینوکس (ترجیحا *Ubuntu*) را بر روی سیستم خود نصب نمایید. لازم به ذکر است که به دلیل حجم محاسباتی بالا **اکیدا** توصیه می شود تا سیستم عاملی مبتنی بر لینوکس موردنظر را به صورت *dual* در کنار دیگر سیستم عامل سیستم خود نصب نمایید و از نصب برروی ماشین مجازی اجتناب نمایید.

پس از نصب سیستم عامل مبتنی بر لینوکس مورد نظر مراحل زیر را به منظور نصب ابزار *GEM5* انجام دهید:

1. در ابتدا با استفاده از دستور زیر ابزار *git* را بر روی سیستم خود نصب نمایید:

```
sudo apt-get install git
```

2. سپس باید فایل ابزار *GEM5* را از رپوزیتوری آن کلون نماییم:

```
git clone https://gem5.googlesource.com/public/gem5
```

3. سپس با استفاده از دستور زیر، نسخه داکر موجود در سیستم را حذف می نماییم تا در ادامه نسخه‌ی موردنظر خود را نصب نماییم:

```
sudo apt-get remove docker docker-engine docker.io
```

4. حال با استفاده از دستور زیر وضعیت سیستم را آپدیت می نماییم:

```
sudo apt-get update
```

5. در ادامه به استفاده از دستور زیر اقدام به نصب *docker.io* می نماییم:

```
sudo apt install docker.io
```

6. سپس با استفاده از دستور زیر داکر را نصب می نماییم:

```
sudo snap install docker
```

7. به منظور بررسی صحت نصب داکر، دستور زیر را اجرا می نماییم. (اگر داکر به درستی نصب شده باشد وروژن داکر موردنظر نمایش داده خواهد شد.)

```
docker –version
```

8. با استفاده از دستور زیر می توان برنامه پیش فرض *Hello World* داکر را اجرا نمود و بدین ترتیب نیز از صحت نصب اطمینان حاصل کرد:

```
sudo docker run hello-world
```

9. حال در این مرحله قصد داریم تا *image* مورد نظر را با استفاده از دستور زیر دریافت نماییم:

```
docker pull <source Dockerfile>
```

به جای عبارت <source Dockerfile> باید یکی از آدرس های زیر را با توجه به ورژن سیستم عامل خود جایگزین نماییم:

Ubuntu 20.04 with all optional dependencies:
gcr.io/gem5-test/ubuntu-20.04_all-dependencies:v21-2

Ubuntu 18.04 with all optional dependencies:
gcr.io/gem5-test/ubuntu-18.04_all-dependencies:v21-2

با استفاده از دستور زیر می توانید ورژن سیستم عامل خود را بیابید:

```
lsb_release –a
```

برای فهم بهتر دستور بالا، به مثال زیر توجه فرمایید:

```
sudo docker pull gcr.io/gem5-test/ubuntu-20.04_all-dependencies:v21-2
```

10. به منظور اجرای دستور این مرحله، ابتدا با استفاده از دستور زیر سیستم را در مد *super user* قرار می دهیم:

```
sudo su
```

حال با استفاده از دستور زیر *image* دریافت شده را *run* می نماییم:

```
docker run -u $UID:$GID --volume <gem5 directory>:/gem5 --rm -it <image>
```

در دستور بالا، به جای عبارت <gem5 directory> آدرس دایرکتوری که *GEM5* در آن قرار دارد را جایگزین می نماییم و به جای عبارت <image> آدرس *image* داکری که با توجه به ورژن سیستم عامل انتخاب کردیم را جایگزین می نماییم.

برای فهم بهتر دستور بالا، به مثال زیر توجه فرمایید:

```
docker run -u $UID:$GID –volume/home/ali/Desktop/CA1-Ali-Imangholi-810197692:/gem5 --rm -it pull gcr.io/gem5-test/ubuntu-20.04_all-dependencies:v21-2
```

11. حال از مد *super user* خارج می شویم.

12. سپس با استفاده از دستور زیر *GEM5* را کامپایل می نماییم:

```
sudo apt install build-essential git m4 scons zlib1g zlib1g-dev \
libprotobuf-dev protobuf-compiler libprotoc-dev libgoogle-perftools-dev \
python3-dev python3-six python-is-python3 libboost-all-dev pkg-config
```

13. در ادامه دوباره مرحله‌ی 12 را تکرار می نماییم، با این تفاوت که قبل از اجرای دستور، سیستم را در مد *superuser* قرار می دهیم:

```
sudo su

sudo apt install build-essential git m4 scons zlib1g zlib1g-dev \
libprotobuf-dev protobuf-compiler libprotoc-dev libgoogle-perftools-dev \
python3-dev python3-six python-is-python3 libboost-all-dev pkg-config
```

گام دوم:

برای انجام این تمرین لازم است تا از میان 3 پردازنده‌ی *X86* و *ARM* و *RISCV* دو پردازنده را به دلخواه انتخاب نمایید و بررسی های مورد نظر را بر روی آن ها انجام دهید. (پردازنده های *X86* و *RISCV* پیشنهاد می شوند.)

سیستم ساخت *GEM5* بر اساس scons (یک سیستم متن باز مبتنی بر پایتون) پیاده سازی شده است و نیاز است که هر کدام از پردازنده های انتخابی توسط دستورات زیر بر اساس scons ، *build* شوند.

1. ابتدا مد سیستم را به حالت *super user* تغییر میدهیم.

```
sudo su
```

2. سپس وارد دایرکتوری *GEM5* می شویم.

3. در ادامه دستور زیر را اجرا می نماییم:

```
scons build/{ISA}/gem5.{variant} -j {cpus}
```

{ISA}: نام پردازنده مورد نظر

{variant}: در این تمرین از مد *opt* استفاده می نماییم.

{cpus}: تعداد *thread* ها که در این تمرین مقدار 1 را برای آن درنظر می گیریم.

Build variant	Optimizations	Run time debugging support
debug		X
opt	X	X
fast	X	

برای فهم بهتر دستور بالا، به مثال زیر توجه فرمایید:

در این مثال *ISA* مربوط به پردازنده‌ی *RISCV* ، *build* شده است:

```
scons build/RISCV/gem.opt –j 1
```

گام سوم:

تا بدین مرحله موفق به نصب شبیه ساز *GEM5* برای پردازنده های انتخابی خود شده اید. حال در ادامه قصد داریم تا با شیوه‌ی اجرای برنامه برروی شبیه ساز *GEM5* آشنا شویم و با اجرای برنامه *Hello World* پیش فرض *GEM5* از صحت عملکرد شبیه ساز اطمینان حاصل نماییم.

باید توجه داشت که در این تمرین از فاز کاری *SE* ابزار *GEM5* استفاده می‌نماییم؛ فلذا در هنگام اجرای ابزار باید از اسکریپت *configs/example/se.py* به جای *configs/example/fs.py* استفاده نمایید.

مقایسه‌ی میان فاز کاری *SE* و *FS* :

- **Full system (FS)**
  - For booting operating systems
  - Models bare hardware, including devices
  - Interrupts, exceptions, privileged instructions, fault handlers
  - Simulated UART output
  - Simulated frame buffer output
- **Syscall emulation (SE)**
  - For running individual applications, or set of applications on MP
  - Models user-visible ISA plus common system calls
  - System calls emulated, typically by calling host OS
  - Simplified address translation model, no scheduling

به منظور اجرای برنامه محک بر روی شبیه ساز *GEM5* از قالب دستوری زیر استفاده می نماییم:

3. سپس با استفاده برنامه محک را بر روی شبیه ساز اجرا می نماییم:

build/{ISA}/gem5.{variant} {simulation script} {file} [gem5 options] [script options]

{ISA}: نام پردازنده مورد نظر

{variant}: در این تمرین از مد *opt* استفاده می نماییم.

{simulation script}: آدرس اسکریپت مورد نیاز برای انجام سیمولیشن در فاز کاری *SE*. (*configs/example/se.py*)

{file}: برنامه محکی که قصد اجرای آن روی شبیه ساز را داریم. بدین منظور توسط سینتکس زیر آدرس فایل مورد نظر را برای دستور مشخص می نماییم:

--cmd file address

[gem5 options]: برخی از پرکاربرد ترین *option* ها به شرح زیر می باشند:

--cpu-type= توسط این *option* نوع cpu برای اجرا مشخص می شود. به طور مثال *O3CPU*، *AtomicSimpleCPU*، *TimingSimpleCPU* و... . این *option* به صورت پیش فرض در حالت *Atomic* قرار دارد و در این حالت شبیه ساز زمان بندی را انجام نمی دهد.

--l2cache با تنظیم این *option* از کش *level 2* استفاده خواهیم نمود.

[script options]: اگر برنامه محک نیازمند دریافت آرگومان ورودی باشد می توان توسط این *option* آرگومان ورودی را برای آن ارسال نمود. بدین منظور باید آرگومان ورودی را درون “ ” قرار داده و توسط سیتنکس زیر به دستور کلی اضافه نماییم.

-- options “OPTIONS” و یا -o “OPTIONS”

با استفاده از دستور زیر می توان باقی پارامتر ها و *option* مورد نیاز را مشاهده نمود. (این دستور برای پردازنده *X86* می باشد، برای باقی پردازنده ها نیز می توان با تغییر جزئی در دستور زیر از قسمت *help* آن استفاده نمود.)

```
build/X86/gem5.opt configs/example/se.py –help
```

حال که با قالب کلی دستور بالا آشنا شدید، به منظور فهم دقیق تر، مثال زیر را که برنامه محک فرضی *queens* با آرگومان ورودی 16 بر روی پردازنده *ARM* اجرا می نماید درنظر بگیرید:

1. ابتدا مد سیستم به حالت *super user* تغییر داده شده است:

```
sudo su
```

2. سپس وارد دایرکتوری *GEM5* شده است.

3. در ادامه دستور زیر را اجرا شده است:

```
build/ARM/gem5.opt configs/example/se.py --cmd queens –o 16
```

حال در ادامه به منظور بررسی صحت ابزار *GEM5* و آشنایی شما با نحوه‌ی اجرای برنامه بر روی این ابزار، برنامه *Hello World* پیش فرض *GEM5* را برروی **هر دو پردازنده انتخابی خود** در حالات ذکر شده اجرا بنمایید و نتایج را در گزارش کار ثبت نمایید: (برای هر دو حالت زیر کش سطح 1، *data* با سایز 64 کیلوبایت (*l1d_size=64kB*) و کش سطح 1، *instruction* با سایز 16 کیلوبایت (*l1i_size=16kB*) درنظر بگیرید.)

الف) *cpu-type* از نوع *TimingSimpleCPU*

ب) *cpu-type* از نوع *O3CPU* باشد.

مثال از نحوه‌ی اجرای برنامه‌ی *Hello World* بر روی پردازنده *ARM*

```
sudo su
```

وارد دایرکتوری *GEM5* می شویم

```
build/ARM/gem5.opt configs/example/se.py --cmd=tests/test-progs/hello/bin/arm/linux/hello --
cpu-type=TimingSimpleCPU --l1d_size=64kB --l1i_size=16kB –caches
```

گام چهارم:

حال در ادامه قصد داریم تا برنامه محک (*Quicksort.c*) که به همراه صورت پروژه ارسال شده است را بر روی پردازنده های انتخابی خود در حالت های مختلف اجرا نماییم و نتایج حاصل را بررسی و مقایسه نماییم اما با توجه به اینکه در این تمرین از مد کاری *SE* ابزار *GEM5* استفاده می نماییم، ابتدا نیاز است تا برنامه محک را بر اساس نوع پردازنده ای که انتخاب نموده اید کامپایل نمایید تا در ادامه بتوانید آن را بر روی شبیه ساز اجرا نمایید.

بدین منظور باید بر اساس پردازنده انتخابی، کامپایلر متناظر با آن را بر روی سیستم خود نصب نمایید:

به عنوان نمونه اگر پردازنده انتخابی شما *RISCV* باشد، با دستور زیر می توانید کامپایلر مورد نظر را نصب نمایید:

```
sudo apt-get install g++-riscv64-linux-gnu
```

به طریق مشابه و با سرچ در اینترنت می توانید روش نصب کامپایلر دیگر پردازنده ها را بیابید و آن ها را نصب بفرمایید.

پس از نصب کامپایلر های مورد نظر، مطابق دستورات زیر فایل برنامه محک را کامپایل می نماییم:

پردازنده با معماری X86:

```
gcc -static -o Quicksort.x86 Quicksort.c
```

پردازنده با معماری *RISCV*:

```
riscv64-linux-gnu-gcc -o Quicksort Quicksort.c -static
```

پردازنده با معماری *ARM*:

```
arm-linux-gnueabi-gcc -o Quicksort Quicksort.c -static
```

❋ حتما در دستورات بالا از فلگ *static*- استفاده نمایید.

گام پنجم:

حال فایل کامپایل شده برنامه محک را **برای هر یک از پردازنده های انتخابی خود** در حالات زیر اجرا نمایید و نتایج حاصل را بررسی و مقایسه نمایید. (برای هر دو حالت زیر کش سطح 1 داده با سایز 64 کیلوبایت (*l1d_size=64kB*) و کش سطح 1، *instruction* با سایز 16 کیلوبایت (*l1i_size=16kB*) درنظر بگیرید.)

راهنمایی: برنامه محک تمرین (*Quicksort.c*) نیاز به ارسال آرگومان ورودی ندارد.

* لازم به ذکر است که باید نتایج حاصل از بخش (الف) و (ب) را با یکدیگر مقایسه نمایید و به علاوه نتایج حاصل از هر یک از اجرا های بخش (ب) را نیز با یکدیگر مقایسه نمایید.

الف) برنامه محک را در حالت *In Order* اجرا نمایید.

ب) برنامه محک را در حالت *Out of Order (O3CPU)* و با توجه به پارامترهای ذکر شده در جدول زیر اجرا نمایید:

	اجرای اول	اجرای دوم	اجرای سوم
issueWidth	4	8	12
fetchWidth	8	8	12
DispatchWidth	4	8	12
decodeToFetchDelay	1	1	1
numIQEntries	32	64	128
numROBEntries	192, 256	192,256	192, 256

* فایل تنظیمات پارامتر های مورد استفاده در هر یک از مدل *In Order* و *Out of Order* تحت عنوان فایلی با همان نام مدل و پسوند *.py*. در مسیر برنامه شبیه ساز وجود دارد. به طور مثال به منظور تنظیم پارامتر های مدل *Out of Order* باید فایل *gem5/x/cpu/o3/o3CPU.py* را ویرایش نمایید. (برای مدل *In Order* در فولدر *cpu* برخی نسخه ها *inorder* و برخی دیگر *minor* این مدل را توصیف می نماید.)

* تذکر: پس از هر بار اعمال تغییرات در فایل تنظیمات و پس از اجرای برنامه محک بر روی شبیه ساز، فایل *cofig.ini* موجود در فولدر *m5out* که شبیه ساز تولید کرده است را بررسی نمایید تا از صحت اعمال مقادیر موردنظرتان برای هر پارامتر اطمینان حاصل بفرمایید.

اگر مقادیر موردنظری که برای پارامتر ها تنظیم نموده بودید در فایل *cofig.ini* اعمال نشده بود، نیاز است تا به کمک دستور زیر متغیر محلی به ترمینال خود اضافه نمایید:

```
sudo su
export M5_OVERRIDE_PY_SOURCE=true
```

توجه شود که پس از هر بار باز کردن ترمینال جدید نیاز است تا دو مرتبه دستور بالا اجرا شود تا متغیر محلی مورد نظر به ترمینال اضافه شود و تغییرات پارامتر های اعمالی به درستی در هنگام شبیه سازی اعمال شود.
به منظور مشاهده خروجی شبیه سازی، نیاز است تا فایل *stats* تولیید شده توسط شبیه ساز را بررسی نمایید. (در بررسی و مقایسه های خود موارد مهمی همانند توان مصرفی، زمان اجرا و... را مقایسه و گزارش نمایید.)

شکل زیر بخشی از خروجی شبیه ساز برای یک اجرای فرضی را به عنوان مثال نشان می دهد:

```
[/work/gem5] cat m5out/stats.txt

---------- Begin Simulation Statistics ----------
sim_seconds                                 0.001687                       # Number of seconds simulated
sim_ticks                                1686872500                       # Number of ticks simulated
final_tick                               1686872500                       # Number of ticks from beginning of simulation
sim_freq                                 1000000000000                     # Frequency of simulated ticks
host_inst_rate                                 103418                       # Simulator instruction rate (inst/s)
host_op_rate                                   111421                       # Simulator op (including micro ops) rate(op/s)
...
system.physmem.bytes_read                       43968                       # Number of bytes read from this memory
system.physmem.bytes_written                        0                       # Number of bytes written to this memory
...
system.cpu.numCycles                          4076245                       # number of cpu cycles simulated
system.cpu.committedInsts                     2763927                       # Number of instructions committed
system.cpu.committedOps                       2977829                       # Number of ops (including micro ops) committed
...
system.cpu.commit.branchMispredicts             93499                       # The number of times a branch was mispredicted
system.cpu.cpi                               1.220635                       # CPI: Cycles Per Instruction
...
```

نکات قابل توجه:

* انجام تمرین به صورت **فردی** می باشد.
* نمره دهی در این تمرین بر اساس گزارش کار شما صورت می گیرد، بنابراین لطفا گزارش کار خود را به صورت کامل و دقیق نگارش نمایید. (در گزارش کار خود تمامی مراحل انجام تمرین را با قرار دادن تصاویر مناسب و توضیحات کافی شرح دهید.)
* فایل های خروجی شبیه سازی را به تفکیک (برنامه اجرایی - پردازنده انتخابی - تنظیمات) برای هر اجرا در پوشه های جداگانه و نام گذاری مشخص به همراه گزارش کار خود ارسال نمایید.

نمره دهی هر بخش:
گام اول: 10 درصد
گام دوم: 10 درصد
گام سوم: 20 درصد
گام چهارم:10 درصد
گام پنجم: 50 درصد

تاریخ تحویل: ساعت 23:59 روز 1402/1/14

موفق و سلامت باشید