گزارش پروژهٔ معماری کامپیوتر

بخش دوم – کار با gem5

اعضای گروه:

یاشار ظروفچی بنیسی **۹۷۱۰۶۱۱۹**

کسری عبدالله سروی **۹۷۱۰۶۱۲۱**

آتوسا چگینی **۹۷۱۰۶۲۵۱**

بهار ۹۹

این پروژه را میتوان به بخش های زیر تقسیم کرد :

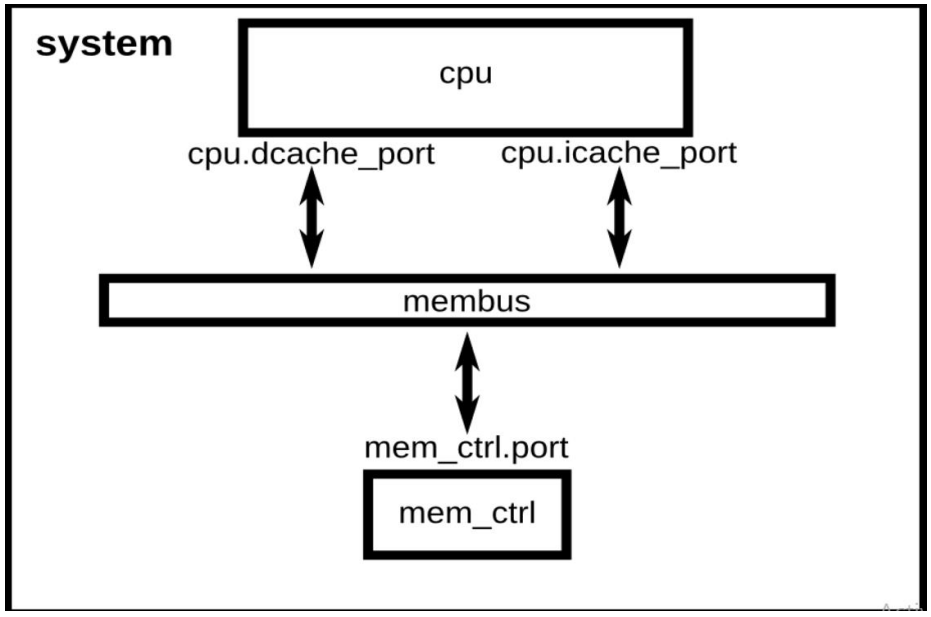
۱) اجرای برنامه قبل از اضافه شدن دستور
۲) اجرای برنامه بعد از اضافه شدن دستور

برای اجرای برنامه ها با استفاده از X86 ابتدا باید فایل gem5.opt را به کمک scons ایجاد کنیم ؛ سپس باید یک فایل config برای اجرای برنامه ها به کمک این فایل gem5.opt بنویسیم که از آنجایی که user gem5 interface با python است این فایل تنظیمات به زبان python نوشته میشود.

در این مرحله ما دو نوع فایل config داریم ۱) بدون L2cache ۲) با L2cache ، به این دلیل که میخواهیم تاثیر اضافه کردن cache لایه ی دوم در اجرای برنامه ها را بررسی کنیم.

فایل تنظیمات اول به نام simple.py نوشته شده است .(در zip قرار دارد.) که در آن حافظه ی نهان نداریم و نکته مهم در این فایل این است که در آن خط system.cache_line_size = '64' را داریم که cache_line_size کل سیستم تعیین میشود که یا ۳۲ یا ۶۴ است

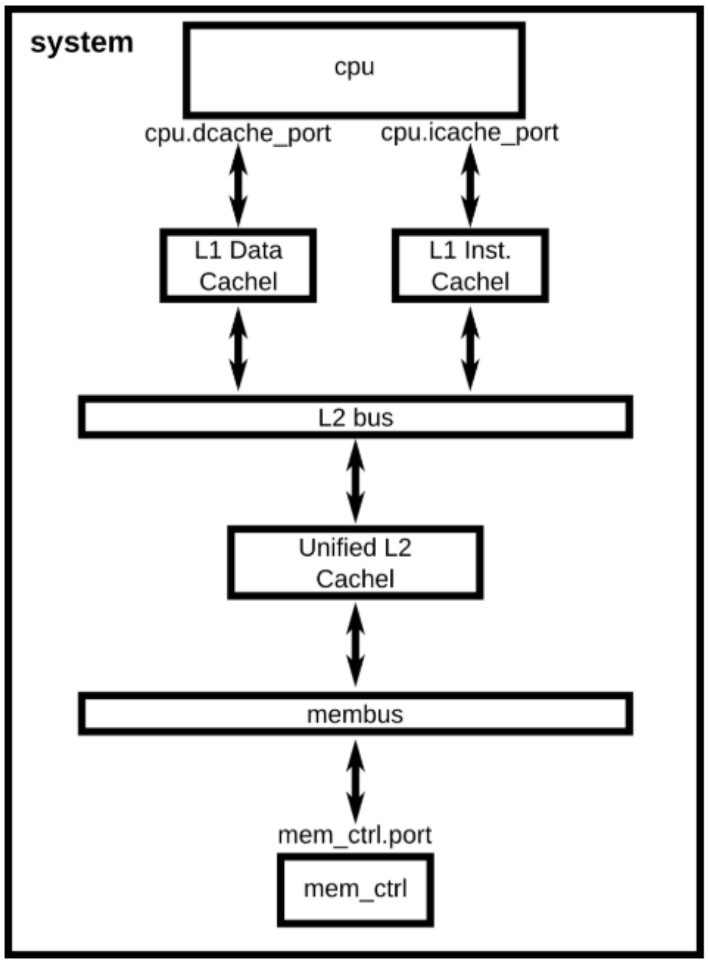
طراحی simple.py به شکل زیر است :



فایل تنظیمات دوم در دو قسمت Caches.py و two_level.py نوشته شده است (هر دو در zip قرار دارند.) در Caches.py تنها ۴ کلاس L1Cache و L1Icache و L1dcache و L2Cache قرار دارند و در هر کلاس پارامتر های ان حافظه ی نهان تنظیم شده اند (مانند assoc , tag_latency , mshrs , response_latency و ...) هم چنین با توجه به صورت پروژه سایز حافظه ی نهان داده 32KB و

سایز حافظه ی نهان دستور 64Kb و سایز حافظه ی نهان لایه ی دوم 1024KB است . به علاوه همان طور که در اسلاید های درس هم داشتیم نیاز به تنظیک کردن تعداد write_buffer ها هم داریم ؛ همچنین مقدار associativity حافظه نهان لایه ی دوم نیز در کلاس مربوط به خودش ست میشود.

در فایل two_level.py هم کدل کلی طراحی را مشخص میکنیم که به شکل زیر است :



توجه کنید که در هر دو حالت ۱ و ۲ تعدادی از موارد مشترک هستند :

```
system.clk_domain.clock = '1GHz'
system.mem_mode = 'timing'
system.mem_ctrl = DDR3_1600_8x8()
```

توضیح کلی simple.py :

ابتدا سیستمی که میخواهیم با آن کار کنیم را میسازیم ؛

ویژگی های clock ان را مشخص میکنیم (مانند فرکانس کاری و دامنه ی ولتاژ کلاک سیستم تعریف شده)

نوع cpu سیستم را از TimingSimpleCPU() میگذاریم .و بازه ی آدرس حافظه ی اصلی و نوع آن (Timing) را مشخص میکنیم .

Bus حافظه ی اصلی را ایجاد میکنیم. (از نوع xbar() است.)

پورت های cpu را (icache و dcache) را به bus وصل میکنیم.(چون در این حالت هیچ حافظه ی نهانی نداریم مستقیما این پورت ها را به memory bus وصل میکنیم .)

مکانیزم interrupt پردازنده را تعریف میکنیم .

یک کنترل کننده ی memory ایجاد میکنیم .

حال باید یک simObject دیگر به نام process ایجاد کنیم و command آن را برابر با ادرس برنامه ای که میخواهیم آن را اجرا کنیم قرار میدهیم .(در این جا آدرس sha256 را میدهیم .)

در آخر یک Root object ایجاد میکنیم و سیستم را instantiate میکنیم .و در نهایت سیستم را simulate میکنیم .

توضیح two_level.py :

این فایل تنظیمات مثل simple.py است و تنها تفاوت این است که بعد از ایجاد cpu ابتدا برای سیستم حافظه ی نهان لایه ی اول ایجاد میکنیم که خود از icache و dcache ایجاد شده است.و این بار به جای وصل کردن پورت های icache و dcache پردازنده به mem bus آن ها را به خود cache ها وصا میکنیم .و حالا یک bus برای L2 ایجاد میکنیم و سمت cache memory های لایه ی اول را به آن وصل میکنیم .حالا یک L2cache ایجاد میکنیم .سمت cpu آن را به bus L2 وصل میکنیم و سمت دیگر ان را به mem bus وصل میکنیم .بقیه ی موارد کاملا مشابه simple.py است .

چون برنامه ی sha256 را باید با x86 اجرا کنیم پس ابتدا باید c++ code آن را که دارای فایل main باشد در داخل یک فایل c++ ذخیره کنیم .(این پیاده سازی را از سایت http://www.zedwood.com/article/cpp-sha256-function برداشتم .) این پیاده سازی را در tests/test-progs/hello/bin/x86/linux/ ذخیره کرده و سپس به

کمک دستور g++ -static sha256 –o sha فایل shaرا در همین جا ایجاد میکنیم . و آدرس این فایل که در بالا نوشته شده است همان چیزی است که باید در فایل های تنظیمات به process.cmd بدهیم تا sha256 اجرا شود.

نکته ی حائز اهمیت ایجاد simObject از sha256 است :

که در دایرکتوری src/learning_gem5/part2 یک فایل sha256Object.py میسازیم که در آن یک کلاس ساخته که از simObject ارث میبرد و type و cxx_header را در آن به شکل زیر مشخص میکنیم :

Type = ‘sha256Object’

Cxx_header = “learning_gem5/sha256.hh”

سپس در همین دایرکتوری فایل sha256_object.cc و sha256_object.hh را ایجاد میکنیم .

در sha256_object.hh کد زیر را مینویسیم :

```
#ifndef __LEARNING_GEM5_SHA256_OBJECT_HH__
#define __LEARNING_GEM5_SHA256_OBJECT_HH__

#include "params/Sha256Object.hh"
#include "sim/sim_object.hh"
Class sha256Object : public simObject{
   Public:
      Sha256Object(sha256ObjectParams *p)
#endif // __LEARNING_GEM5_HELLO_OBJECT_HH__
```

و سپس در فایل sha256_object.cc کد c++ به علاوه ی یک constructor برای sha256Object میسازیم .

حال در SConstruct ۳ خط زیر را اضافه میکنیم :

Import(‘*’)

SimObject(‘sha256Object.py’)

Source(‘sha256_object.cc’)

در مرحله ی بعد یک بار دیگر از scons build/X86/gem5.opt استفاده میکنیم .

و در فایل های تنظیمات root.sha256 = sha256Object() میگذاریم و در اخر برنامه را run میکنیم .

(البته ما این کار را فقط به این دلیل که یک simObject را ساخته باشیم انجام دادیم و برنامه را با آن run کردیم اما میتوانستیم مانند آن چه در ابتدای صفحه ی ۴ نوشتیم برنامه را run کنیم .

تا این جا تمام موارد لازم برای اجرای برنامه قبل از اضافه کردن دستور جدید را بیان کردیم .

حال نتایج اجراهای مختلف را در زیر نشان میدهیم :

حالت اول) دستور اضافه نشده است و حافظه ی نهان نداریم (simple.py) و cache_line_size = 32 است :

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8088280000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 1.txt قرار داده شده است .

حالت دوم) دستور اضافه نشده است و حافظه ی نهان نداریم (simple.py) و cache_line_size = 64 است .

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8088280000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 2.txt قرار داده شده است .

حالت سوم)دستور اضافه نشده است . حافظه ی نهان دو سطحی داریم و cache_line_size = 32 و associativity_L2 = 8

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8364195000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 3.txt قرار داده شده است .

حالت چهارم)دستور اضافه نشده است . حافظه ی نهان دو سطحی داریم و cache_line_size = 32 و associativity_L2 =16

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8364195000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 4.txt قرار داده شده است .

حالت پنجم)دستور اضافه نشده است . حافظه ی نهان دو سطحی داریم و cache_line_size =64 و associativity_L2 =8

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8276280000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 5.txt قرار داده شده است .

حالت ششم)دستور اضافه نشده است . حافظه ی نهان دو سطحی داریم و cache_line_size =64 و associativity_L2 =16

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
sha256('grape'):0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8276280000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 6.txt قرار داده شده است .

نتایج :

منطقا در دو حالت اول چون با simple.py برنامه را run میکنیم و در این دو حالت هیچ حافظه ی نهانی نداریم پس تغییر سایز cache_line نباید تغییری در مدت زمان اجرای برنامه بگذارد .

در ۴ حالت اخر حافظه ی نهان داریم که به طور کلی زمان اجرای برنامه را زیاد کرده است ! (تنها دلیلی که به ذهن من میرسد این است که احتمالا تاخیر های decode و bus ها بیشتر شده است و به همین دلیل زمان اجرای برنامه بیشتر شده است.)

و در اجراها متوجه میشویم که با تغییر associativity حافظه ی نهان لایه ی دوم تغییری در زمان اجرای برنامه ایجاد نمیشود .

در حالی که اگر cache_line_size از ۳۲ به ۶۴ تغییر کند زمان اجزا از ۸۳۶۴۱۹۵۰۰۰ به ۸۲۷۶۲۸۰۰۰۰ تغییر میکند که بهبود 1% در اجرای برنامه را نشان میدهد .

حال دستور جدید را به isa اضافه میکنیم .

با توجه به سودو کد sha256 در https://en.wikipedia.org/wiki/SHA-2 هر کدام از دستورات موجود در این سودو کد را میتوان به عنوان pseudo_instruction به x86 اضافه کرد برای مثال دستور (a & b) ^ (!a & c) را اضافه میکنیم :

ابتدا به ./src/arch/x86/isa/decoder/two_byte_opcodes.isa رفته و میبینیم که ۴ آپکد رزرو شده برای دستوراتی که میخواهیم به x86 اضافه کنیم وجود دارد ما آپکد x56 را انتخاب میکنیم و کد زیر را اضافه میکنیم :

```
0 x56: mynewop({{
        Rax = PseudoInst::mynewop(xc->tcBase(), Rdi, Rsi,Rdx);
}}, IsNonSpeculative);
```

در مرحله ی دوم به ./src/sim/pseudo_inst.cc رفته و function زیر را اضافه میکنیم :

```
uint64_t
```

```
mynewop(ThreadContext *tc, uint64_t arg1, uint64_t arg2, uint64_t arg3)
{
   if (!FullSystem) {
      panicFsOnlyPseudoInst("mynewop");
      return 0;
}
   return (arg1 && arg2) ^ (~ arg1 && arg3);
}
```

در مرحله ی سوم در همان دایرکتوری به pseudo_inst.hh رفته و

```
uint64_t mynewop(ThreadContext *tc, uint64_t arg1, uint64_t arg2, uint64_t arg3);
```

را اضافه کرده و در خط های پایینتر کد در کنار بقیه دستور ها به جای M5OP_RESERVED2 مینویسیم :

```
    case M5OP_MYNEWOP:
        result = invokeSimcall<ABI>(tc, mynewop);
        return true;
```

در مرحله ی چهارم به ./include/gem5 رفته و در فایل m5ops.h مینویسیم :

```
uint64_t m5_mynewop(uint64_t arg1, uint64_t arg2, uint64_t arg3);
```

در مرحله ی بعد به include/gem5/asm/generic رفته و در فایل m5ops.h به جای M5OP_RESERVED2 مینویسیم :

```
#define M5OP_MYNEWOP          0x56 // Reserved for user
```

و در قسمت پایین تر آن هم اضافه میکنیم :

```
MSOP(m5_mynewop, M5OP_MYNEWOP)                \
```

و به این ترتیب دستور جدید به x86 اضافه میشود الان تنها کافی است بار دیگر با gem5.opt scons را build کنیم .

و حالا مانند مرحله ی اول (که دستور اضافه نکرده بودیم) با استفاده از فایل های تنظیمات برنامه sha را run کنیم .

نتایج اجراهای مختلف بعد از اضافه کردن دستور :

حالت هفتم) دستور اضافه شده است و حافظه ی نهان نداریم (simple.py) و cache_line_size = 32 است :

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8243878000 because exiting with last active thread context
```


فایل خروجی stats.txt این اجرا در zip با نام 7.txt قرار داده شده است .

که نسبت به حالت مشابه قبل از اضافه شدن دستور(حالت اول) 1.8% کند تر اجرا شده است !

حالت هشتم) دستور اضافه شده است و حافظه ی نهان نداریم (simple.py) و cache_line_size = 64 است :

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8243878000 because exiting with last active thread context
```


فایل خروجی stats.txt این اجرا در zip با نام 8.txt قرار داده شده است .

همان طور که راجع به حالت ۱ و ۲ گفتیم این جا هم تفاوتی بین حالت ۷ و ۸ نیست چون اصلا cache نداریم که تغییر سایز آن در اجرای برنامه تاثیر بگذارد.

در این جا هم 1.8% برنامه کند تر اجرا شده است .

حالت نهم)دستور اضافه شده است . حافظه ی نهان دو سطحی داریم و cache_line_size =32 و associativity_L2 =8

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8514851000 because exiting with last active thread context
```


فایل خروجی stats.txt این اجرا در zip با نام 9.txt قرار داده شده است .

نسبت به حالت مشابه قبل از اضافه شدن دستور(حالت سوم) 1.7% کند تر شده است !

حالت دهم)دستور اضافه شده است . حافظه ی نهان دو سطحی داریم و cache_line_size =32 و associativity_L2 =16

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8514851000 because exiting with last active thread context
```


فایل خروجی stats.txt این اجرا در zip با نام 10.txt قرار داده شده است .

چون تغییر associativity_L2 زمان اجرای برنامه را تغییر نمیدهد پس در این جا هم نسبت به حالت مشابه قبل از اضافه کردن دستور (حالت چهارم) برنامه 1.7% کند تر اجرا شده است .

حالت یازدهم)دستور اضافه شده است . حافظه ی نهان دو سطحی داریم و cache_line_size =64 و associativity_L2 =8

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8428285000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 11.txt قرار داده شده است .

چون cache_line_size دو برابر شده است پس زمان اجرای برنامه کاهش پیدا کرده (نسبت به حالت ۱۰) اما به طور کلی نسبت به حالت مشابه قبل از اضافه شدن دستور (حالت ۵) 1.8% کند تر اجرا میشود !

حالت یازدهم)دستور اضافه شده است . حافظه ی نهان دو سطحی داریم و cache_line_size =64 و associativity_L2 =16

```
Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (
512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
info: Entering event queue @ 0.  Starting simulation...
info: Increasing stack size by one page.
warn: readlink() called on '/proc/self/exe' may yield unexpected results in various
settings.
      Returning '/home/atoosa/gem5/tests/test-progs/hello/bin/x86/linux/sha'
warn: ignoring syscall mprotect(...)
0f78fcc486f5315418fbf095e71c0675ee07d318e5ac4d150050cd8e57966496
Exiting @ tick 8428285000 because exiting with last active thread context
```

فایل خروجی stats.txt این اجرا در zip با نام 12.txt قرار داده شده است .

چون تغییر associativity_L2 زمان اجرای برنامه را تغییر نمیدهد پس در این جا هم نسبت به حالت مشابه قبل از اضافه کردن دستور (حالت ششم) برنامه 1.8% کند تر اجرا شده است .

*** دلیل کند تر شدن اجرای برنامه بعد از اضافه کردن دستور را میتوان به موارد زیر مرتبط دانست :

۱) این که gcc (g++) به صورت اختصاصی برای دستورات خود isa است و با اضافه شدن دستور جدید از آن استفاده میکند ولی optimization های رایج را نمیتواند برای آن انجام دهد .

۲) چون دستور pseudo_instruction است هر چقدر هم بهینه باشد باز هم به چند دستور کوچک تقسیم میشود و زمان اجرا زیاد میشود