DRAMSim2 实验

PB14000556 陈晓彤

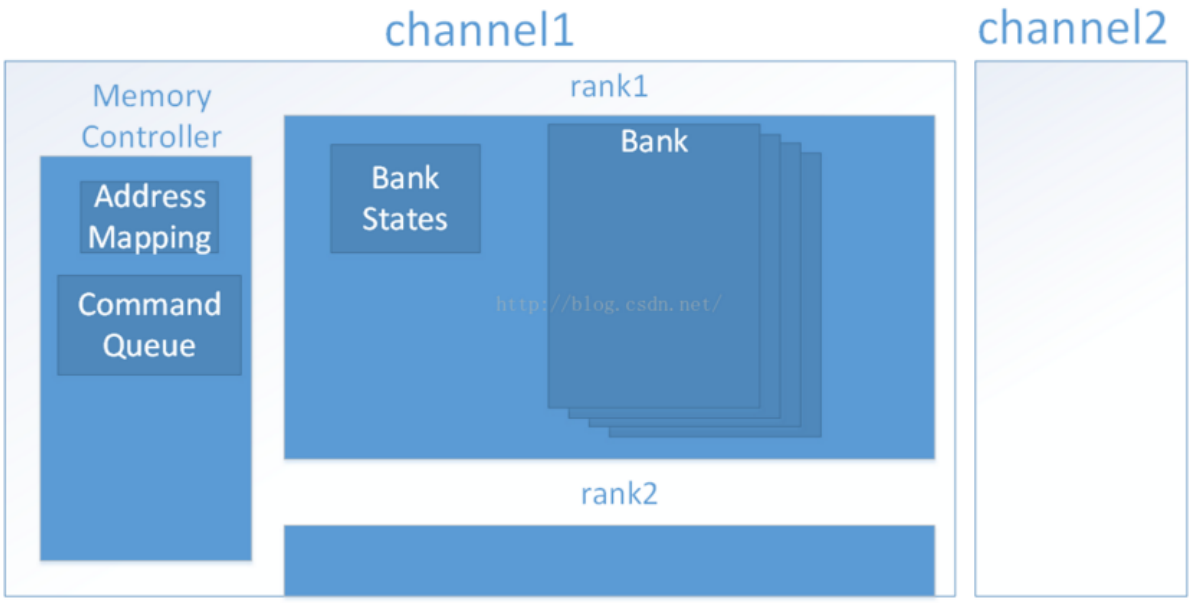
1.DRAMSim2 综述

DRAMSim2 是一个主要模拟 DRAM memory 读写访问延迟和工作能耗的工具，因其模拟结果与实际运行结果非常接近而被科研工作者广泛使用。

2.DRAMSim 架构介绍（参考网页：http://m.blog.csdn.net/article/details?id=51019266）

（一）**DRAMSim2** 逻辑架构：

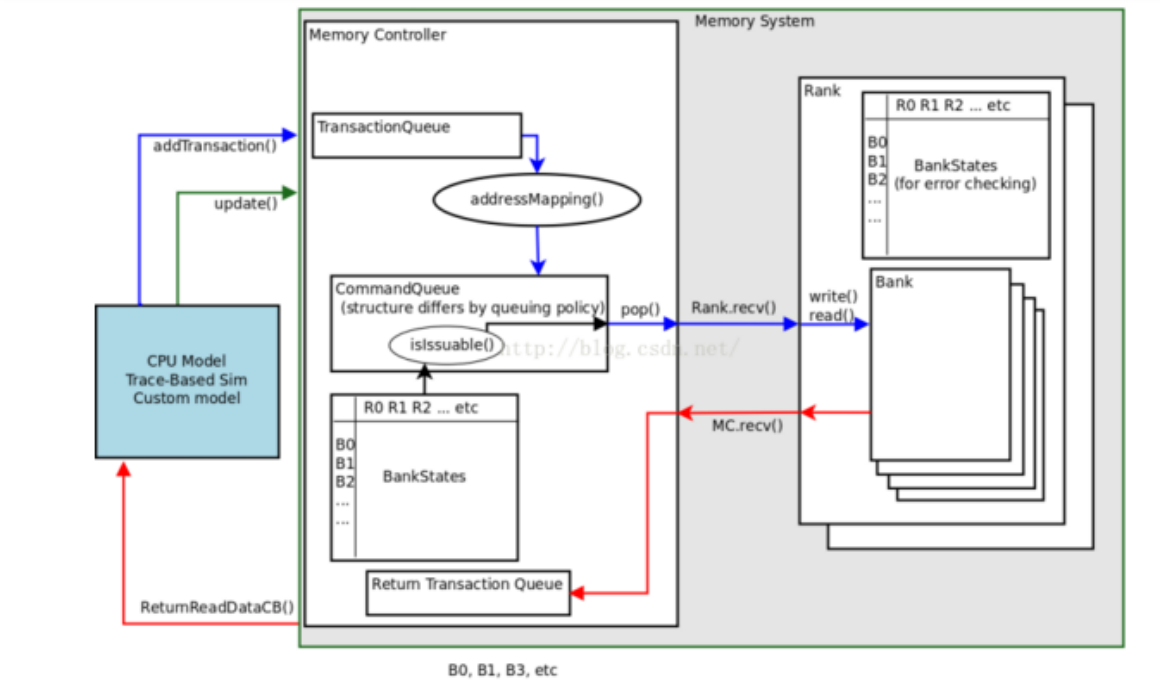
DRAMSim2 的逻辑架构图如图所示：



DRAMSim2 可以模拟多 channel（通道）的内存。概念和分布式存储类似。每一个 channel 内都是一个独立的内存系统。内存系统中的结构我们应该是非常熟悉的：它有一个重要的内存控制器，完成地址映射，记录指令序列，记录各 rank 的状态；同时它分为多个 rank，对于每个 rank：都有一个 bank 控制器，控制着 bank 的状态，访问队列等；同时每个 rank 又有多个 bank，每个 bank 都是地址范围相同的行列矩阵，及我们读写访问的最终地方。

（二）**DRAMSim2** 指令流：

这里借用 DRAMSim2 说明文档中的指令流图：



（三）**DRAMSim2** 代码架构：
下面根据 trace 文件的输入命令过程，跟踪 DRAMSim 的处理流程，来理解 DRAMSim 的代码架构。其中我不可能将源代码都放出来，那样思路太不清晰。更多的是语言来表达，这样就需要大家通过搜索了解关键变量，仔细阅读源码，才能真正理解。
A.TraceBasedSim.cpp：
（1）在其中的 main 函数中，可以找到代码对运行 DRAMSim 时我们输入的指令进行了处理。我们需要找到 traceFile 这个变量，它是一个输入流，绑定了我们要访问的 trace 文件。
（2）在 main 函数中，实例化了一个 MultiChannelMemorySystem 类：memorySystem。这就是整个存储系统。MultiChannelMemorySystem 类是在 MultiChannelMemorySystem.h 中定义的。其中最重要的就是 update 函数。之后我们会详细说明。
（3）在 main 函数中，我们可以找到变量 numCycles。它的意义很容易理解，就是最大运行时钟周期数。当运行时钟周期数超过 numCycles，就会终止运行。
（4）main 函数中：transaction 这个函数也很重要，但很好理解：当我们处理每个请求时，都是把它作为一个 transaction（事务）处理的。而 MultiChannelMemorySystem 类中，定义了 addTransaction 函数，就是处理外部输入的每个事务请求的。详细过程会在之后详细说明。
（5）最后我们就要了解 main 函数中最重要的一个函数:update。它是 MultiChannelMemorySystem 类中的一个子函数。如果仔细看代码，会发现每个时钟周期都会执行一次 update，所以我们可以确定，这就是内存中每个周期更新状态的关键函数。
B.MultiChannelMemorySystem.cpp：
其实 MultiChannelMemorySystem 类并不是重点。因为我们通常设置的 channel 为 1.而在其中我们可以找到 channel 变量。可以发现 channel 为 MemorySystem 类的实例化。MemorySystem 类是在 MemorySystem.h 中定义的。所以对照逻辑架构，发现一个 channel 其实是一个 MemorySystem 类。同样 MultiChannelMemorySystem 类中的 addTransaction 函数和 update 函数其实都只是调用了对应的 MemorySystem 类中的 addTransaction 函数和 update 函数。
C.MemorySystem.cpp：
逻辑架构中已经说过每个 channel 中有两部分：MemoryController 和 rank。同样 MemorySystem 中也分别调用了两部分：MemoryController 类和 rank 类。(a)对于 addTransaction 函数，因为它是添加一个新的事务，并要分配给对应的 rank，属于 MemoryController 的任务，所以调用了 MemoryController 类中的 addTransaction 函数。(b)而 update 函数则要将 MemoryController 和 rank 都更新一个时钟周期，所以对应的调用了 MemoryController 和每个 rank 的 update 函数。
D.MemoryController.cpp：
这里到达了 addTransaction 的底层，它将事务放入了 transactionQueue（事务队列）中；对于 update 同样到了真正有“实际意义”的 update 函数中 DRAMSim 之所以可以准确模拟出延迟功耗，是因为针对每个时钟周期，都在这里找到了非常准确的 update，所以这里是整个代码的核心部分。
E.Rank.cpp：
rank 中的 update 函数主要是记录更新 rank 的工作状态
F.AddressMapping.cpp：
定义了 addressMapping 函数，主要功能是将输入指令的地址进行处理，确定它属于哪个 channel，rank，bank，row，col

3.DRAMSim2 示例实验
下载 DRAMSim2 压缩包并解压，make 后出现 DRAMSim2 文件目录

```
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master$ ls
addgpl.sh             BusPacket.h          ini                          MultiChannelMemorySystem.deppo  SimulatorObject.dep
AddressMapping.cpp    BusPacket.o          IniReader.cpp                MultiChannelMemorySystem.h      SimulatorObject.deppo
AddressMapping.dep    Callback.h           IniReader.dep                MultiChannelMemorySystem.o      SimulatorObject.h
AddressMapping.deppo  ClockDomain.cpp      IniReader.deppo              PrintMacros.cpp                 SimulatorObject.o
AddressMapping.h      ClockDomain.dep      IniReader.h                  PrintMacros.dep                 SystemConfiguration.h
AddressMapping.o      ClockDomain.deppo    IniReader.o                  PrintMacros.deppo               system.ini.example
Bank.cpp              ClockDomain.h        Makefile                     PrintMacros.h                   TraceBasedSim.cpp
Bank.dep              ClockDomain.o        MemoryController.cpp         PrintMacros.o                   TraceBasedSim.dep
Bank.deppo            CommandQueue.cpp     MemoryController.dep         Rank.cpp                        TraceBasedSim.o
Bank.h                CommandQueue.dep     MemoryController.deppo       Rank.dep                        traces
Bank.o                CommandQueue.deppo   MemoryController.h           Rank.deppo                      Transaction.cpp
BankState.cpp         CommandQueue.h       MemoryController.o           Rank.h                          Transaction.dep
BankState.dep         CommandQueue.o       MemorySystem.cpp             Rank.o                          Transaction.deppo
BankState.deppo       comparison_gen.py    MemorySystem.dep             README                          Transaction.h
BankState.h           CSVWriter.h          MemorySystem.deppo           README.pdf                      Transaction.o
BankState.o           docs                 MemorySystem.h               README.tex
BusPacket.cpp         DRAMSim              MemorySystem.o               README.txt
BusPacket.dep         DRAMSim.h            MultiChannelMemorySystem.cpp results
BusPacket.deppo       example_app          MultiChannelMemorySystem.dep SimulatorObject.cpp
```

设置好路径
cd traces
./traceParse.py k6_aoe_02_short.trc.gz
用初始化好的配置文件进行模拟
cd ..

./DRAMSim -t traces/k6_aoe_02_short.trc -s system.ini.example -d ini/DDR3_micron_64M_8B_x4_sg15.ini -c 10000

```
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master$ cd traces
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master/traces$ ./traceParse.py k6_aoe_02_short.trc.gz
Unzipping gz trace... OK
Parsing  k6 trace ...
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master/traces$ cd ..
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master$ ./DRAMSim -t traces/k6_aoe_02_short.trc -s system.ini.example -d ini/DDR3_micron_64M_8B_x4_sg15.ini
-c 10000
== Loading trace file 'traces/k6_aoe_02_short.trc' ==
== Loading device model file 'ini/DDR3_micron_64M_8B_x4_sg15.ini' ==
== Loading system model file 'system.ini.example' ==
===== MemorySystem 0 =====
WARNING: Cannot create memory system with 2048MB, defaulting to minimum size of 4096MB
CH. 0 TOTAL_STORAGE : 4096MB | 1 Ranks | 16 Devices per rank
writing vis file to results/k6_aoe_02_short.trc/DDR3_micron_64M_8B_x4_sg15/4GB.1Ch.1R.scheme2.open_page.32TQ.32CQ.RtB.pRank.vis
DRAMSim2 Clock Frequency =666666666Hz, CPU Clock Frequency=666666666Hz
 ================================================== 
 ============== Printing Statistics [id:0]==============
   Total Return Transactions : 0 (0 bytes) aggregate average bandwidth 0.000GB/s
      -Rank   0 :
        -Reads  : 0 (0 bytes)
        -Writes : 0 (0 bytes)
        -Bandwidth / Latency  (Bank 0): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 1): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 2): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 3): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 4): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 5): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 6): 0.000 GB/s              -nan ns
        -Bandwidth / Latency  (Bank 7): 0.000 GB/s              -nan ns
 == Power Data for Rank        0
   Average Power (watts)     : 0.000
     -Background (watts)     : 0.000
     -Act/Pre    (watts)     : 0.000
     -Burst      (watts)     : 0.000
     -Refresh    (watts)     : 0.000

 == Pending Transactions : 0 (0)==
==== Channel [0] ====
 ================================================== 
 ============== Printing Statistics [id:0]==============
   Total Return Transactions : 522 (33408 bytes) aggregate average bandwidth 2.074GB/s
      -Rank   0 :
        -Reads  : 488 (31232 bytes)
        -Writes : 34 (2176 bytes)
        -Bandwidth / Latency  (Bank 0): 0.207 GB/s              138.202 ns
        -Bandwidth / Latency  (Bank 1): 0.143 GB/s              107.542 ns
```

```
        -Writes : 34 (2176 bytes)
        -Bandwidth / Latency  (Bank 0): 0.207 GB/s              138.202 ns
        -Bandwidth / Latency  (Bank 1): 0.143 GB/s              107.542 ns
        -Bandwidth / Latency  (Bank 2): 0.302 GB/s              163.600 ns
        -Bandwidth / Latency  (Bank 3): 0.191 GB/s              128.438 ns
        -Bandwidth / Latency  (Bank 4): 0.389 GB/s              154.533 ns
        -Bandwidth / Latency  (Bank 5): 0.318 GB/s              174.312 ns
        -Bandwidth / Latency  (Bank 6): 0.318 GB/s              143.586 ns
        -Bandwidth / Latency  (Bank 7): 0.207 GB/s              140.971 ns
 == Power Data for Rank        0
   Average Power (watts)     : 1.785
     -Background (watts)     : 0.840
     -Act/Pre    (watts)     : 0.180
     -Burst      (watts)     : 0.710
     -Refresh    (watts)     : 0.055
 ---  Latency list (25)
       [lat] : #
       [20-29] : 44
       [30-39] : 60
       [40-49] : 31
       [50-59] : 30
       [60-69] : 39
       [70-79] : 18
       [80-89] : 28
       [90-99] : 24
       [100-109] : 15
       [110-119] : 25
       [120-129] : 30
       [130-139] : 22
       [140-149] : 21
       [150-159] : 14
       [160-169] : 22
       [170-179] : 14
       [180-189] : 10
       [190-199] : 8
       [200-209] : 14
       [210-219] : 4
       [220-229] : 4
       [230-239] : 5
       [240-249] : 3
       [270-279] : 2
       [280-289] : 1

 == Pending Transactions : 2 (10000)==
//// Channel [0] ////
chenxiaotong@chenxiaotong-Lenovo-G50-70:~/download/DRAMSim2-master$
```

按照介绍，修改初始化文件 system.ini.example

```
// COPY THIS FILE AND MODIFY IT TO SUIT YOUR NEEDS

NUM_CHANS=1                                                     ; number of *logically independent* channels (i.e. each with a separate memory controller); shou
ld be a power of 2
JEDEC_DATA_BUS_BITS=64                          ; Always 64 for DDRx; if you want multiple *ganged* channels, set this to N*64
TRANS_QUEUE_DEPTH=32                                    ; transaction queue, i.e., CPU-level commands such as:  READ 0xbeef
CMD_QUEUE_DEPTH=32                                      ; command queue, i.e., DRAM-level commands such as: CAS 544, RAS 4
EPOCH_LENGTH=100000                                     ; length of an epoch in cycles (granularity of simulation)
ROW_BUFFER_POLICY=open_page             ; close_page or open_page
ADDRESS_MAPPING_SCHEME=scheme2  ;valid schemes 1-7; For multiple independent channels, use scheme7 since it has the most parallelism
SCHEDULING_POLICY=rank_then_bank_round_robin  ; bank_then_rank_round_robin or rank_then_bank_round_robin
QUEUING_STRUCTURE=per_rank                      ;per_rank or per_rank_per_bank

;for true/false, please use all lowercase
DEBUG_TRANS_Q=false
DEBUG_CMD_Q=false
DEBUG_ADDR_MAP=false
DEBUG_BUS=false
DEBUG_BANKSTATE=false
DEBUG_BANKS=false
DEBUG_POWER=false
VIS_FILE_OUTPUT=true

USE_LOW_POWER=true                                      ; go into low power mode when idle?
VERIFICATION_OUTPUT=false                       ; should be false for normal operation
TOTAL_ROW_ACCESSES=4    ;                               maximum number of open page requests to send to the same row before forcing a row close (to prevent starvation)
~
~
```

此为未修改的文件
尝试把一些 debug 选项改写为 true，可以看到每步循环结果，具体步骤太多，在此仅截取一个片段

```
== Printing bank states (According to MC)
[idle] [idle] [idle] [idle] [1013] [idle] [idle] [idle]

== Printing Per Rank Queue
 = Rank 0  size : 3
    0]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    1]BP [ACT] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    2]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
 ++ Adding IDD3N to total energy [from rank 0]
== Printing transaction queue
== Printing bank states (According to MC)
[idle] [idle] [idle] [idle] [1013] [idle] [idle] [idle]

== Printing Per Rank Queue
 = Rank 0  size : 3
    0]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    1]BP [ACT] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    2]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
 ++ Adding IDD3N to total energy [from rank 0]
== Printing transaction queue
== Printing bank states (According to MC)
[idle] [idle] [idle] [idle] [1013] [idle] [idle] [idle]

== Printing Per Rank Queue
 = Rank 0  size : 3
    0]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    1]BP [ACT] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
    2]BP [READ] pa[0x7ea7b00] r[0] b[4] row[1013] col[61]
==== Channel [0] ====
 ======================================================
 ============== Printing Statistics [id:0]==============
   Total Return Transactions : 522 (33408 bytes) aggregate average bandwidth 2.074GB/s
      -Rank   0 :
        -Reads  : 488 (31232 bytes)
        -Writes : 34 (2176 bytes)
        -Bandwidth / Latency  (Bank 0): 0.207 GB/s              138.202 ns
        -Bandwidth / Latency  (Bank 1): 0.143 GB/s              107.542 ns
        -Bandwidth / Latency  (Bank 2): 0.302 GB/s              163.600 ns
        -Bandwidth / Latency  (Bank 3): 0.191 GB/s              128.438 ns
        -Bandwidth / Latency  (Bank 4): 0.389 GB/s              154.533 ns
        -Bandwidth / Latency  (Bank 5): 0.318 GB/s              174.312 ns
        -Bandwidth / Latency  (Bank 6): 0.318 GB/s              143.586 ns
        -Bandwidth / Latency  (Bank 7): 0.207 GB/s              140.971 ns
 == Power Data for Rank        0
   Average Power (watts)     : 1.785
     -Background (watts)     : 0.840
```