DRAM size independence in single-core processors using gem5

Independencia de la capacidad de la DRAM en procesadores de un núcleo usando gem5

Independência da capacidade da memória DRAM nos processadores de um núcleo utilizando o simulador Gem5

Para citar este artículo / To reference this article / Para citar este artigo: Gallego Garcés, A. y Eslava Garzón, S. (2015). DRAM size independence in single-core processors using gem5. *Ingenio Magno*, *6*(2), 40-47.

Andrés Gallego-Garcés
Facultad de Ingeniería, Grupo de Microelectrónica (GMUN), Universidad Nacional de Colombia
ajgallegog@unal.edu.co

Sebastián Eslava-Garzón
Facultad de Ingeniería, Grupo de Microelectrónica (GMUN), Universidad Nacional de Colombia
jseslavag@unal.edu.co

Fecha de recepción: 20 de Abril de 2015
Fecha de aprobación: 4 de Noviembre de 2015

Abstract

Energy consumption, speed of execution, and integrated circuit area have become important topics in recent years thanks to the growth of the market for mobile devices and the manufacturers of these devices who try to push the limits of their products while maintaining an affordable price. In that race, the constant evaluation of the hierarchy of memory is now a necessary step if we want to improve execution and utilization of devices' resources, because this not only affects the consumption of energy, but also the system capacity and price, known as the bottleneck for instruction execution because each task carried out by the processor must be brought from memory first and later return through it. This document shows how the size of the DRAM does not have a significant effect related to execution benchmarks such as PARSEC 3.0, running on an ARM machine (which in this case is an ARM Cortex-A8). The environment for this simulation is gem5, which is an open source platform for various architectures and is able to change the size of the memory. It is precisely this ability and the ARMv7 architecture model that allows the performance to be related to the memory hierarchy and all other aspects to remain the same within the emulated processor throughout the entire process.

Keywords: cache, dram, gem5, memory hierarchy, parsec.

Resumen

El consumo de energía, la velocidad de ejecución y el área del circuito integrado se han convertido en temas importantes durante los últimos años, gracias al creciente mercado de dispositivos móviles y a sus fabricantes, que tratan de llevar al límite sus productos y mantener un precio accesible. En esa carrera, la evaluación constante de la jerarquía de memorias es ahora un paso necesario si se desea mejorar la ejecución y tener un mejor uso de los limitados recursos del dispositivo, porque esta no solamente afecta el consumo de energía, sino la capacidad del sistema y su precio. Esta, de hecho, es conocida como el "cuello de botella" para la ejecución de instrucciones, porque cada una de las tareas desarrolladas por el procesador tiene que ser traída desde la memoria primero y luego volver a través de ella. Este artículo muestra cómo el tamaño de la DRAM no tiene un impacto significativo cuando se trata de la ejecución de *benchmarks* como PARSEC 3.0, en una máquina ARM (que en este caso es un ARM Cortex-A8). El entorno para esta simulación es Gem5, una plataforma de código abierto para varias arquitecturas, con capacidad de cambiar el tamaño de la memoria. Precisamente, esta capacidad y el modelo para la arquitectura ARMv7 permiten que el desempeño esté relacionado a la jerarquía de memorias y que todos los demás aspectos se mantengan iguales dentro del procesador emulado durante todo el proceso.

Palabras clave: caché, DRAM, gem5, jerarquía de memorias, PARSEC.

Resumo

O consumo de energia, a velocidade de execução e a área do circuito integrado tornaram-se temas importantes nos últimos anos, graças ao crescente mercado dos dispositivos móveis e aos fabricantes dos mesmos que tentam levar os seus produtos para o limite, mantendo um preço acessível. Nesse caminho, a avaliação constante da hierarquia das memorias é agora um passo necessário para melhorar a execução e ter um melhor uso dos recursos limitados do dispositivo, porque a mesma não só afeta o consumo de energia, mas a capacidade do sistema e seu preço, sendo também conhecida como o gargalo da garrafa para a execução de instruções porque cada uma das tarefas desenvolvidas pelo processador tem que ser trazida desde a memória primeiro e logo voltar através dela. O presente artigo mostra como o tamanho da DRAM não tem impacto significativo quando se trata da execução do benchmarks como o PARSEC 3.0, rodando em uma máquina ARM (no caso é um ARM Cortex-A8). O cenário para esta simulação é Gem5, que é uma plataforma aberta para código de múltiplas arquiteturas e tem capacidade de mudar o tamanho da memória. É precisamente esta capacidade e o modelo para a arquitetura ARMv7, o fator que permite que o desempenho esteja relacionado com a hierarquia de memoria e todos os outros aspectos fiquem iguais dentro do processador emulado durante todo o proceso.

Palavras Chaves: cache, dram, gem5, hierarquia de momórias, parsec.

1. Introduction

In modern embedded systems, resources like power, size and capacity have always been limited in order to make devices portable and fully functional. Evolution of in-order and Out-of-Order microprocessors execution is now limited by memory hierarchy organization used to provide instructions and data, a reason to focus on memory design as well (Harris & Harris, 2007).

A good way to improve performance is to have a bigger capacity in memories like *DRAM* and cache (Hennessy y Patterson, 2009), but this impacts price and power consumption too (Hennessy & Patterson, 2012). This trade-off is making designers (Cadence, 2013a, 2013b) to test the whole system by simulating its *ISA* and some I/O devices. This full-system-simulation, which is cheaper and faster than a real implementation, has been usually compared to real hardware in order to have a validated framework. Based on recent software capable of modeling CPUs, memory and I/O devices, and also simulating an operating system, *PARSEC 3.0* benchmarks were run to have memory hierarchy tested using *gem5* because are commonly used and open source.

In this document, an embedded system is going to be simulated changing *DRAM* size within *Gem5*. The architecture under test is ARMv7 core, used in processors like CortexA8, widely used in mobile devices and having a configurable memory hierarchy as well. *DRAM* size will be changed in order to have a better understanding about the impact while running 8 PARSEC 3.0 benchmarks. (Hennessy, 2007; Hennessy & Patterson, 2009).

A number of optimizations to the on-chip memory hierarchy in modern processors have become one of the most important resources that need to be managed efficiently, however, system main memory has a complex architecture in order to increase its bandwidth according to CPU's and cache's (Jacob & Wang, 2007). This architecture includes a DRAM controller, which allows communication with different cache levels as many cores request information (Leupers & Temam, 2007; Augustine et al., 2012).

A complex DRAM controller described in (Hansson et al., 2014) is included in the gem5 simulator, and that is an advantage to our simulation as we analyze the memory size impact. While some optimizations are related to low level cell architecture for main memory construction (Fong & Roy, 2013; Augustine et al., 2012), the memory controller and size in general are also described working together to increase bandwidth and reduce power consumption (Jacob & Wang, 2007;

Cadence, 2013a; ARM, 2009). Here, we will use the most recent gem5 validations so we can have the most accurate results like (Endo, Couroussé & Charles, 2014; Cadence, 2013b).

A. Software

As we have mentioned before, up growing technologies always require testing and development, and this is the main reason to use full-system simulators. Time-to-market is improved as we make changes in modular platforms for computer system architecture research, and gem5 is one of the newest in this group that can achieve pervasive object orientation, multiple interchangeable CPU models, event-driven memory system, multiple ISA support, full-system capability and a few more features.

Research in computer architecture using *gem5* are usually related but not limited to software validation against real hardware, memory hierarchy, virtualized systems or mathematical modeling. They all rely on software's accuracy.

Ubuntu 14.04 was used as host, and all simulations were run through the system terminal. This was an advantage as simulations scripts for a large command list were used.

B. Single-core processors

Although multi-core processors is the main market nowadays according to mobile devices and personal computers high demand, there are still a lot of single core processors used in eReaders, Digital TV, home gateways, netbooks, braking systems, printing and some smart phones. An example of this is ARM, whose microprocessor technology has sold more than 50 billion products around the world. In this particular case, a single core processor will be tested in order to evaluate its performance against DRAM size, and results will lead us to a better understanding of its impact in performance so we can have better choices when designing new single core devices. As mentioned before, this could help manufacturers and users save money, space and power consumption without performance reduction (Wulf, 1995).

2. Software resources

Next we will mention the tools used to simulate ARM cortex A8 single core processor.

A. Gem5

In order to use *gem5* software, it must be downloaded, and compiled using the built in *scons software* for certain architecture. The software version used was stable_2014_08_26, it was compiled for ARM architecture and used through the system terminal (Nathan et al., 2011).

In our simulation, a single ARMv7 core and a Versatile Express board were used as processor and machine type. The simulator was tuned so it would behave more like a Cortex A8 processor by changing the ROB value of an O3 model. After running some test and comparing it to the results shown in (Endo, Couroussé & Charles, 2014), where we based some of the changes to the O3 model, we decided that the size of ROB could be 512.

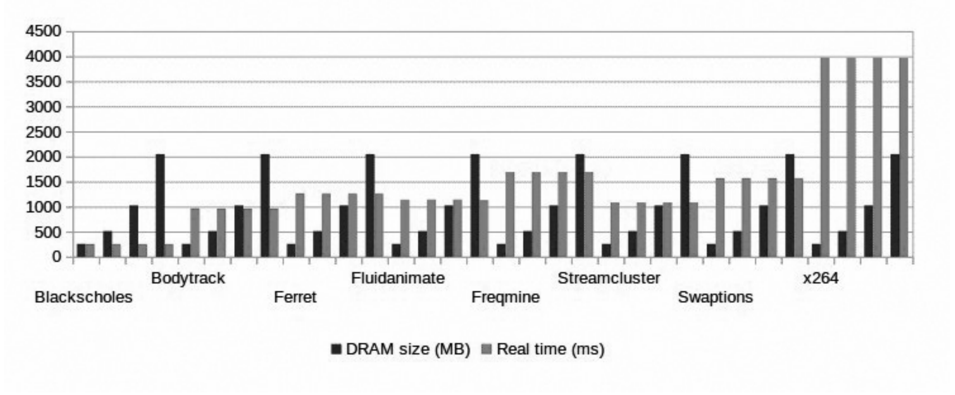
B. Host machine

The host machine was a 4th generation intel core i7 along with 8 GB RAM running Ubuntu 14.04. python 2.7 interpreter and C compilers were included within the operating system, which are necessary to run *gem5*.

C. Operating system

In order to have a full system simulation, we must use an image of an operating system to be run within the platform. We used the operating system Ubuntu natty from linaro, and some files within the image were modified or deleted so it could be managed by *gem5*. The files deleted or modified were the ones related to the system startup, as it is not the same as in a real embedded system.

Figure 1. Real time results for PARSEC benchmarks with different DRAM size and simsmall workload



A. Benchmarks

In order to continue using open source software, PARSEC benchmarking suite was used throughout the process. They had to be compiled using QEMU, which is an computer architectural simulator that has no cycle accurate results, but allowed us to cross-compile the benchmarks for our main platform.

A test using QEMU was also run to test the benchmarks. The objective was to save time as *gem5* simulations could take several hours, and this is also the reason why we worked with *simsmall* workloads for each benchmarks in the beginning.

Some of the benchmarks like Canneal or Facesim did not have support for ARM or would take several days to be simulated. The following PARSEC benchmarks were selected:

- Blackscholes
- Bodytrack
- Ferret
- Fluidanimate
- Freqmine
- Streamcluster
- Swaptions
- x264

3. Simulation

We used gem5 as our simulator, because it integrates single and multi-core processors with a configurable memory hierarchy within different architectures like ARM and x86. We began with a 256MB DRAM, and then increase it size to 512MB, 1024MB and 2047MB. Those values were chosen according to the most common values for main memory in cell phones and tablets using single core ARM processors.

As any small change in the cache parameters could have a considerable impact in performance and power consumption (ARM, 2013), then we decided to have only a basic default cache in *gem5* for the processor and no hard disc emulated at all. L1 data and instruction cache are 32KB, 2-way set associative, no prefetch and have a one clock hit latency or response latency. On the other hand, L2 cache is 1MB, 16-way set associative, simple 8 degree stride prefetch and has a response, or hit latency, of 12 clock cycles (ARM, 2009). Only changing DRAM size will allow performance to be more related to it as this memory hierarchy behaves like a bottleneck for instruction execution. The PARSEC benchmarks were cross compiled using qemu and ARM toolchain within the emulator as we have mentioned before.

A script was used to run each benchmark after 10 s (wait for the system to calm down), reset the statistics output to avoid wrong data from booting, and call to exit immediately after the benchmark finish. There was no need to have a time command as PARSEC console output can be read easily, taking the real measurements as our results.

4. Results

Table I describes the real time results with 256MB for DRAM size, and figure. 1 then shows the results of the simulations for the PARSEC benchmarks evaluated changing main memory size as described in section 3. Even though the simulation model can be tuned for emulating certain hardware, only DRAM size was changed before running a new simulation for the same benchmark. Cache, CPU model, I/O devices, host machine, run script and disk image remain the same throughout the process.

Although DRAM size changed, and simulation parameters were tuned for a single core processor, the biggest difference between the results for the same benchmark is less than 2%. For this reason, PARSEC workloads were change from *simsmall* to *simlarge* to see if a bigger file and time of execution would increase the difference in time while running the same benchmark with different memory size. This results are shown in figure. 2, while table II shows the results for a 256MB memory.

Table 1. Real time results with 256MB DRAM size and PARSEC simsmall workload

PARSEC benchmark	Real time (ms)
Blackscholes	249
Bodytrack	957
Ferret	1259
Fluidanimate	1138
Freqmine	1691
Streamcluster	1082
Swaptions	1563
x264	3969

Table 2. Real time results with 256MB DRAM size and PARSEC simlarge workload

PARSEC benchmark	Real time (ms)
Blackscholes	3932
Bodytrack	9713
Ferret	15759
Fluidanimate	7683
Freqmine	23671
Streamcluster	20937
Swaptions	24929
x264	88581

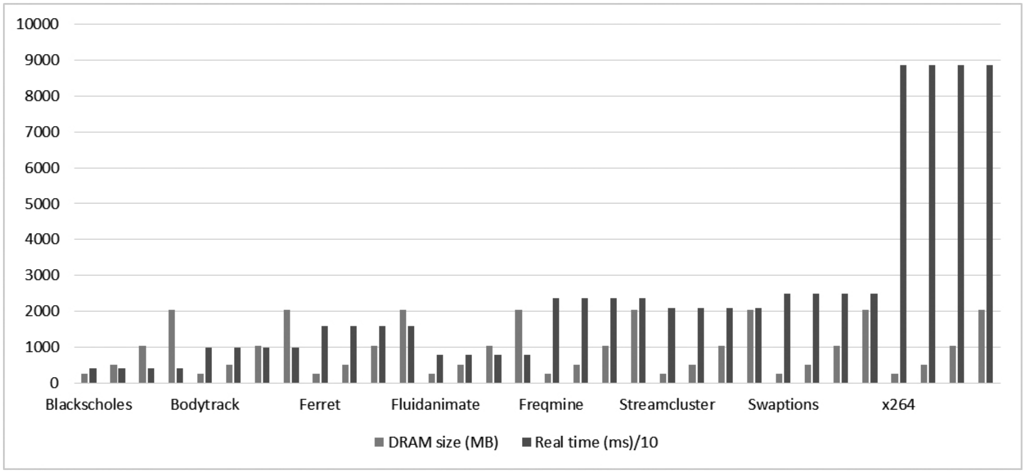


Figure 2. Real time results for PARSEC benchmarks with different DRAM size and simlarge workload

5. Conclusion

We changed the DRAM size from 256MB to 2047MB gradually within a full system simulation in order to model its impact on time when executing PARSEC benchmarks. Although we chose the most common values for main memory in mobile devices with single core processors, execution time changed less than 2\% in the worst case. A bigger main memory for single cores processors could have a negative result in power consumption, price and size, because one single task executed within an operating system in an embedded device has the same results with 256MB and 2047MB.

The reason why this is happening could be the size of modern applications, since most of them are usually designed to run along with others of the same kind. Memory usage then must be low. However, we leave this conclusion for future work.

Large DRAM memories are necessary to execute multiple tasks at the same time, like in modern multicore processors like ARM Cortex A15. Its impact using this method has been proposed as future work, however, embedded systems for small and single tasks could be improved by managing energy and price consumption efficiently using a small main memory size. On the other hand, when a memory system is being tested using this kind of simulators and workloads, designers using benchmarks with the same memory size as *simsmall* or *simlarge* from PARSEC should be aware that DRAM size has no impact in execution time. Price and area could be reduced significantly for single-core processors executing single tasks.

6. Future work

Here we simulated a single core execution for each benchmark at a time, however, microprocessors market has been evolving in the last years to multi-core processors and a more detailed DRAM controller with different cache levels in order to improve performance. In other words, simulations need to be repeated for a different number of cores and cache levels in order to compare the impact of main memory size at execution. Also, the biggest workload for PARSEC *native* is, which could take days to throw results but could be able to show some difference while changing main memory size.

References

Ltd. ARM (2009). *Amba LPDDR2 dynamic memory controller DMC-342 technical reference manual.* Tech.

Rep.Retrieved from http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0436a/index.html

ARM (2013). *Cortex A8 Technical Reference Manual. Revision: R2p1.* http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0344i/index.html (2013).

C. Augustine, C., X. Mojumder, X, H. Fong, H, S. Choday, S., P. Park, P. and & K. Roy, K. (2012). STT-MRAMs for future universal memories: perspective and prospective. *Proc. of 2012 28th Int. Conf. on Microelectronics*, 349-355.

Inc. Cadence Design Systems (2013a). *Cadence design ip: Wide-i/o controller.* Tech. Rep.

Inc. Cadence Design Systems (2013b). *Sources of error in full-system simulation. Advanced Computer Architecture Laboratory*, Michigan: University of Michigan, Advanced Computer Architecture Laboratory..

Fernando Endo, F., Damien Couroussé, D. & and Charles, H.P.. Henri-Pierre (2014). Microarchitectural simulation of in-order and out-of-order arm microprocessors with gem5. *International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV).*

Xuanyao Fong, X. and& Kaushik Roy, K. (2013). *Low-power robust complementary polarizer STT-MRAM (CPSTT) for on-chip caches.* West Lafayette: Purdue University.

Hansson, A., Agarwal, N., Kolli, A., Wenisch, T. & and Udipi, A. (2014). Simulating dram controllers for future system architecture exploration. *Proceedings of the International Symposium on Performance Analysis of Systems and Software (ISPASS).*

David Money Harris, D. and& Sarah L. Harris, S. (2007).. *Digital Design and Computer Architecture.* California: Elsevier., inc.[1], 2007.

John Hennessy, H. and& David. Patterson, D. . Computer Organization and Design (2009). *The hardware software interface* (4th ed.). California. Elsevier inc.

Hennessy, H. & Patterson, D. (2012). *Computer architecture: a quantitative approach* (2012). (5th ed.). John Hennessy and David. Patterson. California:. Elsevier inc.

Jacob, B. and D. Wang, D. (2007). *Memory Systems: Cache, DRAM, Disk.* (2007). B. California:. Morgan Kaufmann Publishers Inc.

Jain, P. rachi and& Wadhawan, J. anakrani (2014). Desing and comparative analysis of SRAM cell structures using 0.5 nm technology. *International Journal of Computer Applications, 87*(3)..

Rainer Leupers, R. and& Olivier Temam, O. (2007). *.Processor and system-on- chip simulation. (2007).* USA. Springer.

Nathan, B. et al. (2011). The gem5 Simulator. (2011). Recuperado de http://research.cs.wisc.edu/multifacet/papers/can11_gem5.pdfACM SIGARCH Computar Architecture News.

Wulf, M. (1995). cKee. Hitting the memory wall: implications of the obvious. (1995). USA. *Computer Architecture News*, *23*, 20-24.