DRAM Selection and Configuration for Real-Time Mobile Systems

Manil Dev Gomony*, Christian Weis†, Benny Akesson*, Norbert Wehn†, and Kees Goossens*
*Eindhoven University of Technology, The Netherlands
†University of Kaiserslautern, Germany

***Abstract*—The performance and power consumption of mobile DRAMs (LPDDRs) depend on the configuration of system-level parameters, such as operating frequency, interface width, request size, and memory map. In mobile systems running both real-time and non-real-time applications, the memory configuration must satisfy bandwidth requirements of real-time applications, meet the power consumption budget, and offer the best average-case execution time to the non-real-time applications. There is currently no well-defined methodology for selecting a suitable memory configuration for real-time mobile systems. The worst-case bandwidth, average-case execution time, and power consumption of mobile DRAMs across generations have furthermore not been investigated.**

This paper has two main contributions. 1) We analyze the worst-case bandwidth, average-case execution time, and power consumption of mobile DRAMs across three generations: LPDDR, LPDDR2 and Wide-IO-based 3D-stacked DRAM. 2) Based on our analysis, we propose a methodology for selecting memory configurations in real-time mobile systems. We show that LPDDR (32-bit IO), LPDDR2 (32-bit IO) and 3D-DRAM (128-bit IO) provide worst-case bandwidth up to 0.75 GB/s, 1.6 GB/s and 3.1 GB/s, respectively. We furthermore show for an H.263 decoder that LPDDR2 and 3D-DRAM reduce power consumption with up to 25% and 67%, respectively, compared to LPDDR, and reduce the execution time with up to 18% and 25%.

I. Introduction

Platforms for mobile devices are heterogeneous multi-processor systems that run a large number of concurrent real-time and non-real-time applications. To reduce cost, main memory is shared between applications. This makes it challenging to verify that real-time requirements are satisfied unless the memory controller provides guarantees on worst-case performance [1]. Mobile platforms furthermore have strict power budgets [2] and reducing memory power consumption is identified as an important challenge to satisfy the power constraints of future mobile devices [3].

Off-chip DRAMs are the most commonly used main memory option in multi-processor platforms [2], [4]. DRAM devices targeting mobile devices, such as Low-Power Double Data Rate (LPDDR) and LPDDR2 memories, are already available in the market. LPDDR3, which is the next generation of LPDDR2, is expected to consume more power than previous generations because of its higher operating frequency [5]. However, emerging 3D-stacked DRAM based on the Wide-IO standard [6] is a promising main memory alternative for future mobile devices for three reasons. First, reducing the need of IO drivers and interconnect compared to off-chip memories results in significant power savings [7]. Second, the excellent electrical characteristics of Through Silicon Via (TSV) enable the use of low-power CMOS transceivers, which save up to 98% of interface power [8]. Third, the low area requirement of TSVs makes it possible to have much wider memory interfaces that enable significantly higher bandwidth [9].

978-3-9810801-8-6/DATE12/©2012 EDAA

The DRAMs for mobile devices from the past (LPDDR), present (LPDDR2) and future (3D-DRAMs), come with different operating voltages, frequencies, storage capacities and interface widths. Apart from these, additional system-level parameters, such as memory map and transaction sizes, decide the performance and power consumption of a DRAM device [10]. Identifying the memory configuration suited for a mobile application is challenging [5], because there are many system-level parameters that affect the performance and power consumption, and there is no well-defined methodology for making the right selection for real-time mobile systems. In these systems, the memory configuration must satisfy the worst-case bandwidth requirements of real-time applications, while providing the best average-case performance in terms of execution time and power consumption.

This paper has two main contributions in the context of DRAM selection and configuration for real-time mobile systems. The first contribution is an *analysis of worst-case bandwidth and average-case performance* in terms of execution time and power consumption of mobile DRAMs, both *within and across generations* (LPDDR, LPDDR2 and 3D-DRAM). The second contribution is a *DRAM selection and configuration methodology* to select system-level parameters, such as operating frequency, interface width, request size, and memory map, for real-time mobile systems.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III gives an overview of DRAMs and discusses real-time memory controllers. In Section IV, we introduce the three generations of mobile DRAMs used in our analysis. Section V then presents our worst-case analysis, and Section VI the average-case analysis. Our proposed methodology for DRAM selection and configuration is explained in Section VII, and we conclude our work in Section VIII.

II. Related work

A design-space exploration of DRAM system-level parameters is performed in [10] and [11], although the design choices are targeting non-real-time systems. However, some of their conclusions, such as exploiting bank-level parallelism and selecting the correct transaction size to reduce latency, hold true for any system. In contrast, analysis on DRAMs is performed with real-time memory controllers that provide bounds on worst-case bandwidth and latency in [12]–[14]. However, neither of these explores different memory configurations to determine the optimal operating points, nor do they specifically consider memories for mobile devices including 3D-stacked DRAMs.

A design-space exploration of 3D-stacked DRAM architectures with respect to performance, energy, and area efficiency for different memory densities is performed in [15]. In [16] and [17], an average-case analysis of off-chip multi-channel memories is performed to evaluate the performance of multiple memory channels offered by 3D-stacked DRAMs. However, none of these works compare the real-time performance of mobile memories across generations.

[18] compares different memory architectures for mobile devices, and a comparison of parallel interface DRAMs, such as LPDDR2 and Wide-IO, is made with serial interface memories in terms of bandwidth and power consumption. A methodology is proposed in [5] to select the memory configuration for a mobile device based on storage capacity, throughput, latency, power, cost and thermal concerns. However, none of these work consider providing bounds on bandwidth to real-time applications.

To the best of our knowledge, there is no prior work that analyzes worst-case bandwidth and average-case performance in terms of application execution time and power consumption of mobile DRAMs across generations. There is furthermore no methodology to identify and select a memory configuration for mobile systems that satisfies the worst-case bandwidth requirements of real-time applications and the average-case power budget, while providing the best average-case execution time for non-real-time applications.

III. Background

This section presents background information on DRAM and explains the concept of guaranteeing useful bounds on bandwidth using real-time memory controllers.

A. Introduction to DRAM

In a DRAM device, each bit is stored using a single transistor-capacitor pair known as *storage cell* [19]. The storage cells are arranged to form a memory array with a matrix-like structure, as shown in Figure 1a. The intersection of rows and columns, specified by a *row address* and a *column address*, identifies the storage cells inside the memory array. The memory array and a *row buffer* constitute a *bank*. Current DRAM devices contain either 4 or 8 banks that can be accessed concurrently, although they share command, address, and data buses to reduce the number of off-chip pins. During a memory access, the data from storage cells are copied to the row buffer before performing a read/write operation. Data is then transferred over the data bus with a *data rate* of one or two words per clock cycle, depending on if the memory device uses a Single Data Rate (SDR) or a Double Data Rate (DDR). The data rate affects the *peak bandwidth* of the memory, which is defined as the product of its operating frequency, data-rate, and Interface Width (IW).

The memory controller interacts with the DRAM by sending DRAM commands. There are several timing constraints that must be considered while issuing these commands. To understand these timing constraints, an example scenario for a read operation is shown in Figure 1b. The contents of a row inside the memory array is copied to the row buffer by issuing an *activate (ACT)* command. It takes *tRCD* cycles to fetch the data from the storage cells and copy it to the row buffer, which is the minimum time before the *read (RD)* command can be issued. Once the read command is issued, it takes additional *tRL* cycles before the data is available on the data bus, as indicated by D0-D3 for the DDR device in the figure. A read/write command accesses the memory as a burst with a predefined *Burst Length (BL)* (in words). Before another row in the memory array can be read, the existing row must be closed by writing back the contents to the storage cells using a *precharge (PRE)* command. The precharge command can only be issued *tRAS* cycles after the activate command.

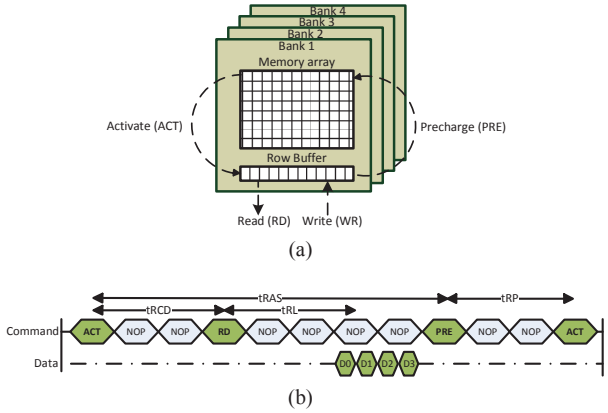

Fig. 1. DRAM architecture and timings

B. Real-time memory controllers

There are several memory controllers, specifically targeting real-time systems [12]–[14]. For such controllers, it is important to bound the *memory efficiency*, which is the fraction of clock cycles with useful data on the data bus. To guarantee high efficiency bounds, these controllers typically employ a *close-page policy*, which means that they close the open row immediately after every memory access to reduce the worst-case overhead of opening another row [1].

Memory accesses by real-time memory controllers can be characterized by three parameters: BL, *Banks Interleaved (BI)*, and *Burst Count (BC)*. These are collectively referred to as the *memory map* [20] and determine the physical location of data in the memory array. BI specifies the number of banks over which the data is interleaved and BC specifies the number of bursts per bank [21]. These parameters determine the *access granularity* (AG) of the memory controller. This is the granularity of data that the memory controller uses to access the memory device, which is computed according to $AG = BI \cdot BC \cdot BL \cdot IW$ bytes. The choice of memory map is done at design time and determines the memory efficiency that is guaranteed for a given mix of request sizes [20].

IV. Mobile DRAM generations

Having provided background on DRAM memories and real-time memory controllers, this section proceeds by presenting the configuration space of the three generations of mobile DRAMs considered in our analysis.

Low-power DRAMs differ from standard DRAM in the initialization sequence, input/output circuitry and clocking [22]. Table I shows an overview of the mobile memories LPDDR, LPDDR2 and Wide-IO based 3D-DRAM based on the JEDEC specifications [6], [23], [24]. From LPDDR to LPDDR2,

TABLE I
OVERVIEW OF MOBILE DRAM GENERATIONS

Memory	*Operating Speed (MHz)*	*Capacity*	*Operating Voltage (V)*	*IO width*	*Burst Lengths*	*Banks*	*Data rate*
LPDDR	100 - 200	64 Mb to 2 Gb	1.8	16,32	2,4,8,16	4	DDR
LPDDR2	100 - 533	64 Mb to 2 Gb	1.2	16,32	2,4,8,16	8	DDR
3D-DRAM	100 - 360	256 Mb to 2 Gb	1.2	64,128	1,2,4	4,8	SDR, DDR

operating frequency has been increased to provide higher bandwidth, and supply voltage has been reduced for lower power consumption. The JEDEC Wide-IO draft specification states that a 3D-DRAM device consists of 4 independent channels, each having interface width of 128 bits [6] and operates at 1.2 V. The speeds of 3D-DRAMs in Table I are selected based on the chosen technology and architecture using the 3D-DRAM generator presented in [15].

The investigated 3D-DRAMs are closely aligned to the new Wide-IO DRAM JEDEC standard. Figure 2 depicts the two 3D-DRAM (Wide-IO) architectures used. [15] explored an optimized 3D-DRAM that consists of 8 layers (tiers), corresponding to the number of banks. The two architectures differ in the tile size used by the 3D-DRAM macro block to compose a bank. For bank architecture type (a), the tile size is 64 Mb and for type (b) 128 Mb.

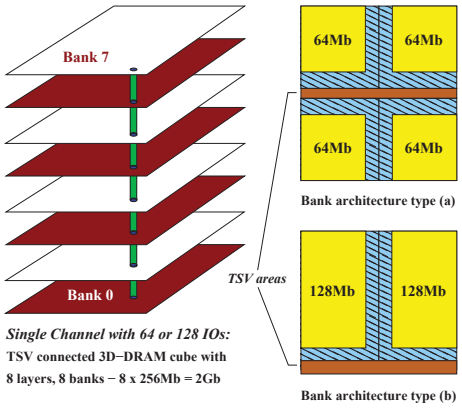

Fig. 2. 3D-DRAM architectures

Architecture option (a) is very similar to the published Wide-IO device of Samsung [25]. The areas with tile size numbers in Figure 2 show the DRAM cell arrays, the areas with stripes are occupied by column, row and control circuits, as well as all other peripheral circuits, and the remaining areas are reserved for the vertical TSV connections as indicated in the figure.

Table II summarizes the generated 3D-DRAM configurations. Typical performance (timing), area and technology data are shown. In order to comply with JEDEC, the supply voltage was set to $V_{DD} = 1.2\ V$ for all configurations. The footprint for a single channel is calculated by $A_{footprint} = Area/8$, because all 3D-DRAM cubes are composed of 8 layers. In contrast to [8], we use a TSV diameter value of 8 μm and 16 μm pitch. This diameter and pitch are a good compromise between reported yield and density. The TSV capacitance evaluates to 94 fF and TSV resistance to 23 $m\Omega$. Overall six different 2 Gb 3D-DRAM DDR configurations are created.

TABLE II
2 GB 3D-DRAM DDR CONFIGURATIONS

#	*Arch. type*	*# of IOs*	*Techn.*	*Cell size (F^2)*	*Area [mm^2]*	*Freq [MHz]*	*tRAS [ns]*
1	(b)	64	58 nm	6	80.3	100.0	31.9
2	(b)	64	46 nm	6	59.7	133.3	26.4
3	(a)	64	45 nm	4	57.6	360.0	26.0
4	(b)	128	65 nm	6	110.2	100.0	27.6
5	(b)	128	58 nm	6	80.5	133.3	32.0
6	(a)	128	45 nm	4	57.7	360.0	26.0

V. WORST-CASE ANALYSIS

This section presents the preconditions for our worst-case analysis and our methodology for worst-case bandwidth computation. We show the worst-case bandwidth results of mobile DRAMs across generations and analyze the impact of operating frequency, interface width, and request size on the results.

A. Preconditions

The different memories and the BL's selected for our analysis are given in Table III. The numbers shown along with the memory names represent doubled operating frequency (because of DDR) and interface width, respectively. The memory capacity of is chosen to be 2 Gb because it is available in all generations. To study the trends in operating speeds, we choose the slowest and fastest 2 Gb devices specified by Micron for each generation. To evaluate the performance impact of the memory interface width, we consider 16, 32, 64, and 128-bit devices. Note that 64-bit interface devices are not included in the Wide-IO specifications, but have been generated using the 3D-DRAM generator in [15] to understand the impact of interface width on performance and power consumption. The BL is reduced as the interface width increases to keep the same access granularity for all devices.

TABLE III
MEMORY TYPES

Memory types	*BL*
LPDDR-266-x16,416-x16; LPDDR2-667-x16,1066-x16	16
LPDDR-266-x32,416-x32; LPDDR2-667-x32,1066-x32	8
3D-DRAM-200-x64, 266-x64, 720-x64	4
3D-DRAM-200-x128, 266-x128, 720-x128	2

Although our Wide-IO-based 3D-DRAM provides four independent channels, our analysis considers only a single channel for a valid comparison to LPDDR/LPDDR2 devices. For a useful comparison, all devices (LPDDR, LPDDR2 and 3D-DRAM) operate in Double Data Rate (DDR) mode. Since LPDDR has only 4 banks, the maximum number of banks interleaved in other memories is also limited to 4 for a useful comparison. The memory command timings for LPDDR and LPDDR2 are based on Micron specifications [26] and [27], respectively. The memory timings of the 3D-DRAMs are generated using the 3D-DRAM generator model [15].

Our evaluation uses a real-time memory controller [12]. This controller dynamically schedules pre-computed schedules of SDRAM commands called *memory patterns*. Memory efficiency is bounded by determining and analyzing the worst-case combination of memory patterns, as explained in [21]. The guaranteed bandwidth is then determined by the product of memory efficiency and the peak bandwidth of the memory.

B. Worst-case bandwidth analysis

For the design-space exploration of memory maps, we compute the worst-case bandwidth for different combinations of BI and BC (BL is fixed as shown in Table III) and for different request sizes (32 B, 64 B, 128 B, and 256 B). The different request sizes are analyzed to understand the impact of *data efficiency* on bandwidth. Data efficiency is the fraction of fetched data that contains requested data, which is determined by the relation between request sizes and the access granularity of the memory controller. Due to lack of space, we do not show all results from the design-space exploration, but our investigations have shown that the maximum worst-case bandwidth is achieved by adhering to the following three rules for the selection of memory map. The rules should be respected in the order they are presented.

1) *The access granularity of the memory maps must be at most equal to the request size*, so that we achieve 100% data efficiency.
2) *Interleave data over the maximum number of banks.* This is because efficient pipelining of memory commands across different banks, i.e. bank-level parallelism, improves overall efficiency by hiding some of the DRAM timing constraints.
3) *Maximize BC* to amortize remaining overhead over a larger access granularity.

For different request sizes, we select the optimal BI and BC configuration for the highest guaranteed bandwidth based on our memory map selection rules, as shown in Table IV. The worst-case bandwidth for each memory type with different operating frequencies and interface widths for different request sizes is shown in Figure 3. As an example with a request size of 256 B, LPDDR, LPDDR2 and 3D-DRAM guarantee 0.75 GB/s, 1.6 GB/s and 3.1 GB/s of worst-case bandwidth, respectively.

TABLE IV
OPTIMAL MEMORY MAP CONFIGURATIONS

Request size (Bytes)	*Memory map*	
	BC	BI
32	1	1
64	1	2
128	1	4
256	2	4

When selecting a memory device, it is important to know that the internal architecture of the DRAMs across generations remain unchanged due to the fact that DRAM vendors focus on minimizing the cost per bit. For this reason, the inherent circuit delays ("overhead") are constant (in nanoseconds) across memory generations. Hence, the worst-case bandwidth of mobile DRAMs does not depend on memory generation, but on the operating frequency and interface width. In the next section, we discuss the impact of operating frequency and interface width on the guaranteed worst-case bandwidth across mobile DRAM generations.

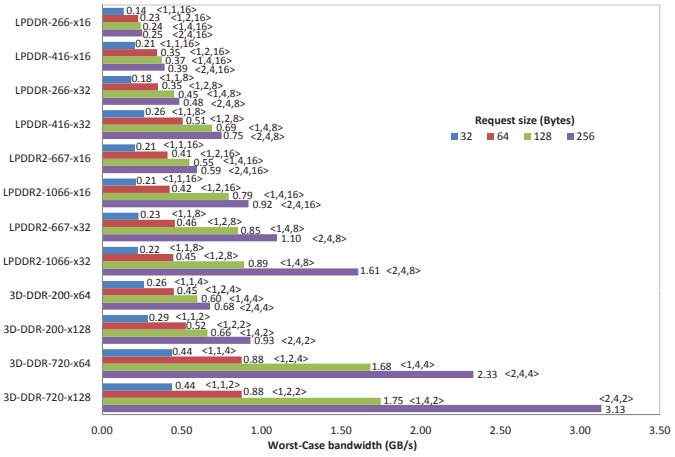

Fig. 3. Worst-case bandwidth of mobile DRAM generations for different request sizes and corresponding <BC,BI,BL> configurations

1) Impact of operating frequency: Process technology has improved across memory generations, which has enabled higher operating speeds. However, as the operating frequency increases, the overhead in terms of clock cycles grows [21], reducing memory efficiency. Figure 4a shows the worst-case bandwidth results of 16-bit LPDDR (133 MHz and 200 MHz) and LPDDR2 (333 MHz and 533 MHz) devices, normalized to LPDDR 133 MHz. The figure shows that the difference in worst-case bandwidth between the fastest and slowest memories increases with the request size when the interface width is constant. This happens because the operating frequency not only increases the peak bandwidth, but also the overhead cycles. This results in reducing memory efficiency, unless requests are large enough to support a larger access granularity that amortizes the overhead. Similar trends are observed across different interface widths, and memory generations.

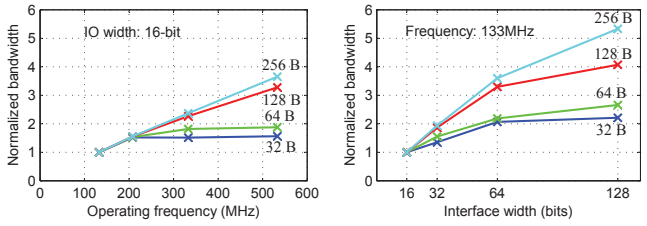

Fig. 4. Effect on worst-case bandwidth of operating frequency (left) and interface width (right), for different request sizes

2) Impact of interface width: Figure 4b shows the worst-case bandwidth results at a constant operating frequency of 133 MHz with interface widths of 16 and 32 bits for LPDDR and 64 and 128 bits for 3D-DRAM. The results are normalized to that of an LPDDR 16-bit device. The figure shows that the worst-case bandwidth improves with increasing interface width. However, the efficiency of wider interfaces is also limited by overhead, but the bandwidth increases at a higher rate when interface width is increased compared to when increasing operating frequency. This is because *the number*

of overhead cycles remains constant when interface width is increased, whereas it increases with frequency. For example with a request size of 32B, when the operating frequency increases from 208 MHz to 533 MHz, worst-case bandwidth remains constant, as shown in Figure 4a. However, when the interface width is increased from 16 bits to 32 bits, the worst-case bandwidth increases by 26%.

From the above analysis, we derive the following design rule to make a choice on operating frequency and interface width: *For a given request size, first select the widest interface possible such that access granularity is at most equal to the request size, then select the highest operating frequency.* Note that this design rule does not consider cost since DRAM prices are volatile.

VI. Average-case analysis

This section evaluates the average-case performance in terms of application execution time and power consumption of mobile DRAMs across three generations. We start this section by introducing our simulation setup and power estimation models. Finally, we discuss the average-case performance results.

A. Simulation setup

A cycle-accurate transaction-level trace is generated by running an H.263 video decoder application on a *SimpleScalar* ARM simulator [28]. To evaluate performance and power with different request sizes, application traces are generated with cache-line sizes of 32 B, 64 B, 128 B, and 256 B, respectively. The traces are replayed by a SystemC traffic generator attached to a cycle-accurate real-time DRAM memory controller [12].

We use two power models to estimate the power based on the commands sent to the DRAM device. Our first power model is a command-based power calculator [29] that estimates power based on the IDD values provided in the memory data sheets. We use this model to estimate power consumption of LPDDR and LPDDR2 devices [26], [27]. Since IDD values for the 3D-DRAMs are not available, we use our second power model that estimates power using a cycle-accurate SystemC DRAM model running the workload. To ensure that the values delivered by the two power models are comparable, the second model is correlated to the first for LPDDR/2 memories. We achieved less than 1% error between the models, as shown by the correlation results in Figure 5. We used three different workloads for this correlation (high - HL, medium - ML and low - LL).

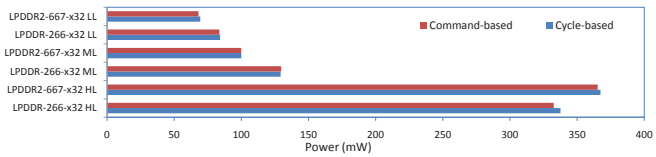

Fig. 5. Power model comparison for different workloads

B. Average-case performance results

We evaluated the average-case performance using the memory map configurations from our worst-case analysis in Table IV. Figure 6 shows the execution time of the trace vs. power consumption for different memories and request sizes, normalized to LPDDR-266-x16.

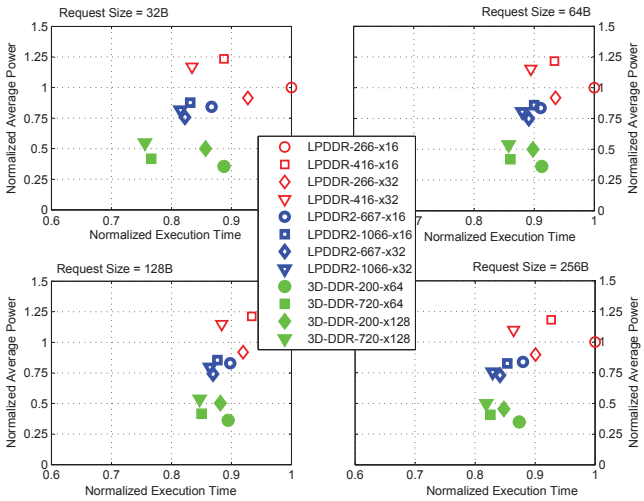

Fig. 6. Execution time vs. Power normalized to LPDDR-266-x16

The performance and power consumption trends across memory generations are similar with different request sizes. In general, increasing the operating frequency or interface width reduces the execution time because of the improvement in bandwidth. The difference in power consumption across memory generations is more significant than within due to different supply voltages and process technologies. In contrast, differences within generations are only because of different operating frequencies. Compared to LPDDR, LPDDR2 has up to 25% lower power consumption because of its lower operating voltage and 3D-DRAMs provide up to 67% power savings. 3D-DRAMs have 40% lower power consumption than LPDDR2, although they have similar operating voltages because of the low-power 3D architecture [15].

LPDDR-266-x32 has the same execution time as LPDDR-416-x16 with a request size of 64 B, but has 25% lower power consumption. This is because the power consumption reduces with frequency, while the wider interface compensates for the loss in performance. Wider interfaces with lower operating speeds hence give better performance and a relatively lower power consumption than faster memories with narrow interfaces, which is aligned with our guidelines for the selection of operating frequency and interface width, previously explained in Section V-B.

In LPDDR/2, the power consumption reduces by 10% when the interface width is doubled from 16 to 32 bits, whereas 3D-DRAMs with 128-bit interfaces consume up to 25% more power than the 64-bit interface devices. This is because the 64-bit IO 3D-DRAM device is optimized for wiring resources to keep the area and cost low. When the interface width is increased to 128-bit, the need for additional wiring resources increases the IR drop significantly. As the LPDDR/2 devices are originally designed for a 32-bit interface, doubling the interface width from 16 to 32 bits has no impact on IR drop in the wiring resources. However, 3D-DRAMs with 128-bit IO provide significant power savings compared to LPDDR/2 devices.

VII. Selection and configuration methodology

In this section, we introduce a five-step methodology for selecting and configuring DRAM system-level parameters in real-time mobile systems. An overview of the methodology is shown in Figure 7.

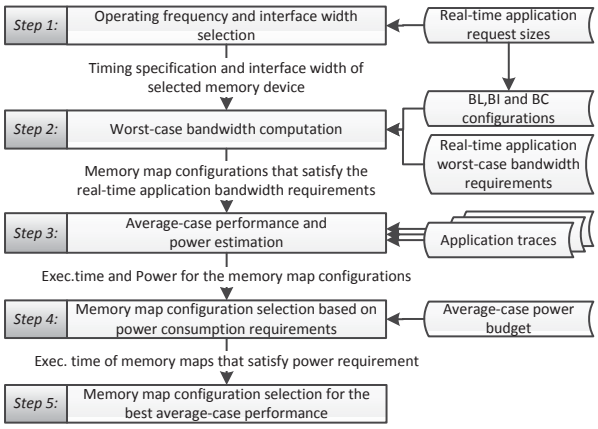

Fig. 7. Proposed five-step methodology

Given a set of memory devices, the first step is to select the operating frequency and interface width. This is done using our design rule from Section V-B stating that for a given request size, first choose the maximum possible interface width with access granularity less than or equal to the request size, and then choose the highest operating frequency.

The second step is to identify the memory map configurations that satisfy the real-time application bandwidth requirements. We determine the memory maps using our memory map selection criteria from Section V-B, which is to maximize the bank interleaving provided that the access granularity is less than or equal to the request size. The timing specification of the selected memory device is used to compute the guaranteed worst-case bandwidth for the different memory map configurations. From the worst-case bandwidth results, the memory map configurations are selected that satisfy the bandwidth requirements of the real-time applications.

In the third step, the average-case performance of the memory configuration is evaluated in terms of execution time and power consumption in a setup consisting of a real-time memory controller, executing traces of all applications together. The fourth step refines the memory map selection by removing the ones that do not that satisfy the power consumption requirement. Finally, in the last step, the memory map configuration that gives the best average-case performance based on power vs. execution time trade-off is selected.

VIII. Conclusions

Selecting and configuring DRAM for real-time mobile systems is challenging, since there are many system-level parameters that determine the performance and power consumption. The selected DRAM configuration must furthermore provide guarantees on worst-case bandwidth to the real-time applications, satisfy the average-case power demands, and minimize the average-case execution time of non-real-time applications.

In this work, we performed a worst-case and average-case analysis across and within three generations of mobile DRAM. In terms of worst-case bandwidth, LPDDR (32-bit IO), LPDDR2 (32-bit IO), and 3D-DRAM (128-bit IO), can guarantee up to 0.75 GB/s, 1.6 GB/s and 3.1 GB/s, respectively. In average-case for an H.263 video decoder, LPDDR2 and 3D-DRAM reduce power by 25% and 67%, and reduce the execution time by 18% and 25%, respectively, compared to LPDDR. We also identified that the Wide-IO based 3D-DRAMs with 64-bit IO provide up to 25% lower power consumption compared to the devices with 128-bit IO, specified in the standard. We proposed a methodology to make a selection of DRAM system-level parameters, such as operating frequency, interface width, request size, and memory map for real-time mobile systems.

IX. Acknowledgment

This work was supported in part by Agentschap NL (an agency of the Dutch Ministry of Economical Affairs), as part of the EUREKA/CATRENE/COBRA project under contract CA104.

References

[1] B. Akesson *et al.*, "Memory Controllers for High-Performance and Real-Time MPSoCs," in *Proc. CODES+ISSS*, 2011.
[2] C. van Berkel, "Multi-core for mobile phones," in *Proc. DATE*, 2009.
[3] "International Technology Roadmap for Semiconductors (ITRS) - System Drivers," 2009, http://www.itrs.net/reports.html.
[4] P. Kollig *et al.*, "Heterogeneous Multi-Core Platform for Consumer Multimedia Applications," in *Proc. DATE*, 2009.
[5] A. B. Kahng and V. Srinivas, "Mobile System Considerations for SDRAM Interface Trends," in *Proc. SLIP*, 2011.
[6] "JEDEC Draft Wide IO Specification," 2010.
[7] Y. Xie and G. H. Loh, "3D Stacked Microprocessor: Are We There Yet?" *IEEE Micro*, vol. 30, no. 3, 2010.
[8] M. Facchini *et al.*, "System-level power/performance evaluation of 3D stacked DRAMs for mobile applications," in *Proc. DATE*, 2009.
[9] D. H. Woo *et al.*, "An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth," in *Proc. HPCA*, 2010.
[10] B. Jacob, "A Case for Studying DRAM Issues at the System Level," *IEEE Micro*, 2003.
[11] V. Cuppu and B. Jacob, "Concurrency, Latency, or System Overhead: Which Has the Largest Impact on Uniprocessor DRAM-System Performance?" *SIGARCH Comput. Archit. News*, vol. 29, no. 2, 2001.
[12] B. Akesson *et al.*, "Predator: A predictable SDRAM memory controller," in *Proc. CODES+ISSS*, 2007.
[13] M. Paolieri *et al.*, "An Analyzable Memory Controller for Hard Real-Time CMPs," *Embedded Systems Letters, IEEE*, vol. 1, no. 4, 2009.
[14] J. Reineke *et al.*, "PRET DRAM Controller: Bank Privatization for Predictability and Temporal Isolation," in *Proc. CODES+ISSS*, 2011.
[15] C. Weis *et al.*, "Design Space Exploration of 3D-stacked DRAMs," in *Proc. DATE*, 2011.
[16] E. Aho *et al.*, "A case for multi-channel memories in video recording," in *Proc. DATE*, 2009.
[17] J. Nikara *et al.*, "Performance analysis of multi-channel memories in mobile devices," in *Proc. SOC*, 2009.
[18] Y. Choi *et al.*, "Future evolution of memory subsystem in mobile applications," in *Proc. IMW*, 2010.
[19] B. Jacob *et al.*, *Memory systems: cache, DRAM, disk*. Morgan Kaufmann, 2007.
[20] S. Goossens *et al.*, "Memory-Map Selection for Firm Real-Time SDRAM Controllers," in *Proc. DATE*, 2012.
[21] B. Akesson *et al.*, "Classification and Analysis of Predictable Memory Patterns," in *Proc. RTCSA*, 2010.
[22] Micron, "Low-Power Versus Standard DDR SDRAM Introduction," Micron Technology, Inc, Tech. Rep. TN-46-15, 2007.
[23] "JEDEC Low Power Double Data Rate (LPDDR) SDRAM Standard," Feb 2009.
[24] "JEDEC Low Power Double Data Rate (LPDDR2) SDRAM Standard," Dec 2010.
[25] J.-S. Kim *et al.*, "A 1.2V 12.8GB/s 2Gb mobile Wide-I/O DRAM with 4x128 I/Os using TSV-based stacking," in *Proc. ISSCC*, 2011.
[26] Micron, "Mobile Low-Power DDR SDRAM specifications," www.micron.com, 2009.
[27] ——, "Mobile LPDDR2 SDRAM specifications," www.micron.com, 2010.
[28] "http://www.simplescalar.com."
[29] K. Chandrasekar *et al.*, "Improved Power Modeling of DDR SDRAMs," in *Proc. DSD*, 2011.