 Evaluating novel DRAM architectures for which no datasheets exist, we demonstrate how our tool is able to aid in energy saving compared to standard commodity DDR3 devices. A 16% DRAM performance improvement and 13% DRAM size and the number of banks on performance and power of a system depends on the architecture of the DRAM chip, the design of the memory controller and the access patterns received by the memory controller. Evaluating the impact of DRAM design decisions therefore requires a holistic approach that includes an appropriate model of the DRAM bank, a realistic controller and DRAM power model, and a representative workload which requires a full system simulator, running a complete software stack.

In this paper, we introduce DRAMSpec, an open source high-level DRAM bank modeling tool. As a major contribution, we move the DRAM modeling abstraction level from detailed circuit level to the DRAM bank and by the integration in full system simulators we allow system or processor designers (non-DRAM experts) to tune future DRAM architectures for their target applications and use cases. We demonstrate the merits of DRAMSpec by exploring the influence of DRAM row buffer size and the number of banks on performance and power of a server application (memcached). Our new DRAM design offers a 16% DRAM performance improvement and 13% DRAM energy saving compared to standard commodity DDR3 devices. Additionally, we demonstrate how our tool is able to aid in evaluating novel DRAM architectures for which no datasheets are available.

I. INTRODUCTION

Dynamic random access memories (DRAMs) account for a significant share of any system’s power and energy consumption, be it battery-driven mobile devices or high-performance computing servers [1], [3]. To meet the increasing demands, DRAM architects apply different techniques to provide higher bandwidth and reduce latency [2]. However, the performance improvements come at the cost of larger die sizes, and increases in the overall power consumption of DRAMs. Balancing these trade-offs is therefore key.

DRAM architectures are experts in DRAM trade-offs and the impact of technology scaling on future DRAMs. However, DRAMs are not stand-alone devices, and the performance and power consumption of a DRAM does not only depend on the memory design, but on the workload, the controller, and the overall system configuration. Conversely, system designers understand workloads and their sensitivities, but have limited ability to explore the DRAM design space. Therefore, to evaluate DRAM architectural trade-offs, we need: 1) high-level models of DRAM devices, and 2) these models must be integrated into full-system simulators that run realistic workloads.

For modeling DRAM performance several high-level models of memory controllers are available [18], [21], [22], coupled with DRAM timing. For calculating DRAM power, Micron presented a spreadsheet power calculator for DDR2 and DDR3 chips [6]. DRAMPower [5], an open source DRAM power and energy estimation tool, added some improvements to Micron’s spreadsheet and presented an executable tool for estimating DRAM power. However, these tools rely on DRAM datasheets. These datasheet specifications are pessimistic due to the high process margins added to ensure correct functionality under worst-case conditions and a good-enough yield [1], [7]. Furthermore, by relying only on existing datasheets, exploring future DRAM architectures becomes impossible.

Our proposed high-level DRAM bank modeling tool, called DRAMSpec in the following, generates the key datasheet specifications required by system simulators to evaluate DRAM performance and power consumption. As an introductory example, we consider in Figure 1 two DRAM configurations with different row-buffer sizes. The results show the impact of changing the DRAM row-buffer size on typical DRAM currents (IDD0: activate-precharge current and IDD4R: burst read current). The cost of moving data from the DRAM cells to the DRAM row-buffer, and then writing the data back to the row-buffer, increases for a DRAM with a larger row-buffer size; whereas the cost of reading bursts decreases. Enabling this high-level exploration on its own is not enough though, as the trade-offs have to be explored for realistic workloads.

A more holistic and integrated approach to DRAM design has the potential of investigating new DRAM architectures, which perform better and consume less power for real workloads. This work provides guidance how to trade-off various aspects in DRAM design. For instance, when DRAM standardizations efforts like JEDEC, where DRAM vendors and processor designers are discussing future DRAM generations beyond DDR4 and LPDDR4, our proposed tool is valuable to justify specific design decisions. By integrating DRAMSpec in gem5 [9], we allow system or processor designers to architect DRAM devices and explore the impact of their designs on the system’s performance and power usage. We are not aiming at implementing an exact circuit level model of a DRAM chip, which would require hours of simulation time. Instead, our goal is to have a flexible, fast executing (< 1 second) high-level tool, which generates estimated timing and power values for each DRAM configuration, with an appropriate accuracy to explore the impact on a target system. In this context the paper makes following contributions:
We provide a high-level DRAM bank model that is detailed enough to explore very fast the DRAM design space and key parameters. We have chosen the DRAM bank as abstraction level because it is the base component across all DRAM generations. Our aim is to have a model to estimate very fast with enough accuracy the performance (timing) and power (currents) of different DRAM design options.

- We introduce DRAMSpec, an open-source tool, which generates datasheet timing and current parameters for existing and future DRAM generations.

- We integrate seamlessly DRAMSpec into gem5 [9], which is also possible for other system-level simulators, such as DRAMsim2 [21] and DRAMSys [22]. This allows system engineers to investigate several DRAM architectural parameters; thus, evaluating if a DRAM design fits to the target system.

- We use gem5 to investigate the performance and power impact of changing the DRAM row-buffer size and the number of banks of a DDR3 device while running a memcached application on a server/client system.

- We show how our tool is used to estimate the latency and power for the novel Hybrid Memory Cube (HMC) DRAM architecture [24], [25].

The remaining sections are organized as follows: Section II discusses the related work and states the improvements achieved in our proposed model. Section III explains the key DRAM timing and current parameters. Furthermore, Section IV explains the DRAM bank model used and Section V presents an overview of the structure and input of our DRAMSpec tool. It also explains the integration of DRAMSpec in gem5. Section VI details the validation of the output generated by DRAMSpec. The results of the DRAM design feature exploration in gem5 and the latency and power specifications of a HMC bank model. Finally, we conclude in Section VII.

II. RELATED WORK

When it comes to high-level modeling of DRAMs one of the most cited tools is CACTI-D [10]. It provides not only energy but also some timing parameters and therefore enables estimation of performance and power for future DRAMs. CACTI was originally developed to model SRAMs and then extended to embedded and commodity DRAMs. A drawback of CACTI-D is that the data path is implemented inherently in the code as an H-tree structure. This is not suitable for exploring advanced DRAM architectures or devices [11]. Furthermore, CACTI requires a large amount of input parameters to model a DRAM and most of its output information can only be analyzed and interpreted by DRAM experts. This makes its integration in DRAM controller high-level simulators very challenging.

Vogelsang [3] noticed this lack of flexibility and developed a very detailed DRAM model, which provides the missing flexibility; thus allowing DRAM design explorers to modify several DRAM parameters. Unfortunately, this work only models the current parameters and uses DRAM timing delays from datasheets. Since some of the timing parameters are used in the calculation of the current specifications, the relatively high process margins added to these values lead to pessimistic estimation of the DRAM currents. To the best of our knowledge, this tool is no longer available.

Recently, DArT [11] was presented as a powerful DRAM area, power and timing modeling tool. However, estimates of some of the key timing and current parameters are missing. DArT does not consider refresh delays and currents and does not estimate DRAM background currents. Therefore, exploring DRAM design tradeoffs in a full system simulator is challenging since the simulator requires all the key DRAM parameters as input. To the best of our knowledge, there is also currently no public version available for this tool.

Contrary to prior works our proposed tool based on an abstracted DRAM bank model enables system designers with no DRAM know-how to explore various DRAM key parameters for optimizing their target system.

III. BACKGROUND

In DRAM datasheets, vendors provide timing and current specifications that differ from one DRAM generation to another. The knowledge about these parameters is essential to understand the mechanisms used by DRAMSpec to generate these values. In this section, we do not explain all the DRAM timings and currents modeled in DRAMSpec. However, we briefly explain some of the fundamental specifications. A more detailed explanation of these parameters could be found in [14].

A. DRAM Timing Parameters

A DRAM device responds to the commands received from the memory controller. The memory controller has to keep track of the DRAM states and issue commands at the right point in time [13]. The required timings are available in datasheets provided by the DRAM vendors and mainly depend on the architecture and technology of the designed chip. Table I lists and explains some of the main timing specifications. These times correspond to physical delays of wires, drivers etc. and are constraints which should be considered for the design of a memory controller. In Figure 2, a model of the DRAM array-structure (DRAM Bank) is presented. The DRAM cell is the smallest component of a DRAM and consists of a transistor and a capacitor, the storage element. When accessing a DRAM, a row of cells, the DRAM page, is activated and its content is moved to the row-buffer. After the time \( t_{RCD} \) (row to column

<table>
<thead>
<tr>
<th>Timing</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_{RCD} )</td>
<td>Row to Column Delay. The time interval between row access and data ready at sense amplifiers.</td>
</tr>
<tr>
<td>( t_{RAS} )</td>
<td>Row Access Strobe. The time interval between row access command and data restoration in a DRAM array. A DRAM bank cannot be precharged until at least ( t_{RAS} ) time after the previous bank activation.</td>
</tr>
<tr>
<td>( t_{RP} )</td>
<td>Row Precharge. The time interval that it takes for a DRAM array to be precharged for another row access</td>
</tr>
<tr>
<td>( t_{RC} )</td>
<td>Row Cycle. The time interval between accesses to different rows in a bank. ( t_{RC} = t_{RAS} + t_{RP} )</td>
</tr>
<tr>
<td>( t_{CCD} )</td>
<td>Column-to-Column Delay. The minimum column command timing, determined by internal burst(prefetch) length. Multiple internal bursts are used to form longer burst for column reads.</td>
</tr>
<tr>
<td>( t_{RFC} )</td>
<td>Refresh Cycle time. The time interval between Refresh and Activation commands.</td>
</tr>
</tbody>
</table>
delay), the data in the row-buffer becomes ready and a read or write access could take place. $I_{\text{BAS}}$ includes $t_{\text{RC}}$ and an additional time constraint for which the data should remain in the row-buffer before it is written back ($t_{\text{RP}}$ - read to precharge delay plus margin). Furthermore, $I_{\text{RP}}$ is the time required to write the content of the row-buffer back to the DRAM cells. As mentioned previously, DRAM vendors add relatively high process margins to these DRAM timings.

### B. DRAM Current Parameters

In addition to the timing parameters, DRAM datasheets define several currents, used in the power consumption calculation. The most important power specifications are briefly explained in Table II. $I_{\text{D0}}$ is the average current consumed for activating a DRAM row, moving the data in the activated row to the row-buffer and writing the data back from the row-buffer to the DRAM row. Similar to DRAM timings, vendors tend to scale up these specifications due to the existing large process margins [1].

### IV. DRAM Bank Model

DRAM generations from DDR2 through WIDE I/O share a similar bank structure. One of the key contributions of this work is implementing a high-level DRAM bank model that provides high enough accuracy to evaluate DRAM design trade-offs and to perform fast design space explorations. Our bank model is similar to the ones presented in [3], [11]. However, since our goal is to create a high-level approximate model of the DRAM bank and not an exact circuit level model, we do not provide information about the transistor’s technology, such as threshold voltage, mobility, oxide thickness, etc.; instead we define the DRAM cell as a black-box with resistance and capacitance values in both bitline and wordline directions, as shown in Figure 2. This is key, as it enables non-experienced DRAM users to easily understand the model and use the tool in their research. DRAMSpec considers the shared bank structure as its highest level of abstraction allowing users to generate current and timing parameters for many of the existing DRAMs and future generations. Having a bank model enables the estimation of most of the essential timing and current specifications; however, timing specifications for switching between ranks, bank-groups and low-power timings and their respective current parameters are left for future work. Figure 2 shows the bank model implemented in DRAMSpec.

Information on the exact modeling of a DRAM bank is limited due to its proprietary nature. Our high-level model is much faster than a very exact circuit level simulation, which requires at least a three orders of magnitude longer run-time than our proposed tool. However, as we shall see in Section VI, it is suitable to study the influence of different DRAM design parameters on DRAM timings and currents. A DRAM bank is made of two half-banks with the row logic fitted in-between. Figure 4 shows a detailed view of the DRAM half-bank model. The column logic is placed at the bottom of the DRAM bank [3], [4]. Each half-bank consists of a group of memory blocks. A memory block is a group of sub-arrays placed in the horizontal and vertical directions. A typical memory sub-array consists of 256K cells connecting up to 512 cells per local bitline and per local wordline. The lowest level of abstraction in our model is a DRAM cell. This is modeled as a black-box with resistance and capacitance values in both word-line and bit-line directions. For different technologies, we offer estimated values of the resistance and capacitance parameters [19], [20], [23].

### V. DRAMSpec Overview

The following section explains the structure and the different modules of the DRAMSpec tool. The tool considers two JSON files as inputs. The technology input file defines the technology features of the designed DRAM chip, whereas, the architecture input file defines the structure and the “high-level” design specifications of the DRAM chip. DRAMSpec consists of four different modules: 1) a DRAM bank model, 2)
a timing parameters calculator, 3) a current (power) specifications calculator and 4) an area calculator, as seen in Figure 3. In addition to the output printed on the terminal, DRAMSpec generates two JSON files: a DRAM timing parameters file and a DRAM current parameters file.

A. Timing Calculator

For modeling timing delays a Resistance-Capacitance (RC) model is used. However, since a lumped model is pessimistic in estimating resistive wire delays, a distributed model is used for calculating wire delays [17]. For each timing delay, we sum up all the partial delays contributing in the delay, similar to the approach done in [11]. Figure 4 illustrates the strategy used to estimate $t_{RCD}$ timing parameter. $t_{RCD}$ is the time required to activate a chain of sub-arrays forming the DRAM row and then moving the data from the sub-arrays to the bitline sense amplifiers. It is the time required to open a DRAM page. $t_{RCD}$ is calculated by summing up four partial delays, as shown in Figure 4. The first delay (1) is the time required by the master wordline driver (in Row logic) to drive the master word-line which activates a chain of sub-arrays. The second delay (2) is the delay is the time needed by the local wordline drivers of the sub-array to activate a row of DRAM cells. Moreover, the third delay (3) occurs when the data is moved from the DRAM cells to the bitline sense amplifiers through the local bitlines. Finally, the last partial delay (4) is the internal delay required by the bitline sense amplifiers. All the other DRAM timings are estimated similarly by summing up the partial delays.

B. Current Calculator

Similar to the timing delays estimation, capacitances are summed up from the interconnect wires driven in the different DRAM operations. The total capacitance multiplied with the driving voltage results in the total charge. Therefore, currents are calculated through a Capacitance-Voltage (CV) model. The driving voltage results in the total charge. Therefore, currents are calculated in a similar way.

C. Tool Input

DRAMSpec requires two sets of inputs, a technology input file, with 29 parameters, and an architecture input file, with 13 parameters.

1) Technology Input: As mentioned before, the lowest level of abstraction for DRAMSpec’s bank model is the DRAM cell. This is modeled as a black-box with capacitance and resistance values in both the word-line and bit-line directions. These parameters are set in the technology input. In addition to the DRAM cell resistance and capacitance values, the technology input also defines wire resistance and capacitance values, DRAM cell dimensions, DRAM subarray dimensions, drivers resistances and supply voltages. The defined parameters are sufficient to estimate RC delays and calculate charges. Moreover, DRAMSpec offers non-expert users different technology inputs estimates: a 50nm technology input, a 20nm, 30nm and 40nm with buried wordline technology input based on information found in [19], [20], [23]. This enables most of the users to use these files as they are and spare the time required to research and understand DRAM bank implementations. However, experienced users are advised to configure the technology inputs relying on their knowledge and inputs from DRAM designers or the International Technology Road-map of Semiconductors (ITRS) [20].

2) Architecture Input: This input allows DRAMSpec users to configure multiple designs of DRAM chips and explore the influence of several DRAM design parameters on the performance and power of DRAMs. For example, a DRAM designer could design two DRAM chips with different row-buffer size using DRAMSpec. One chip has a larger row-buffer than the other chip with the same density (e.g. 2Gb). The timing and current parameters generated by DRAMSpec for the different DRAM designs are passed with the different design parameters to gem5’s DRAM controller model [18]. After running a gem5 simulation for the two different DRAM configurations, the performance and power output is analyzed and decisions can be made on, which DRAM design fits best the designed system for certain type of workloads.

D. Integration of DRAMSpec in gem5

Figure 5 depicts the integration of DRAMSpec in gem5. For a system explorer it is straight forward to design a DRAM chip with DRAMSpec. Gem5 fetches in addition to the timing and current parameters generated by DRAMSpec, the architectural features of the designed chip, such as the $t_{RC}$ (row-cycle time). All the other currents are calculated in a similar way.
Today’s DRAM technologies suffer from high process variations, which rapidly increase with process scaling. In order to ensure a high enough yield in volume production, vendors have to add large process margins to the datasheet current and timing values. Therefore, we expect the timing and current parameters generated by DRAMSpec to be less than datasheet values. Figure 6 and Figure 7 show our generated parameters compared to datasheet values from different DRAM vendors for 1Gb DDR3 and 2Gb DDR3 chips respectively. As expected, the current parameters from Samsung and Micron are clearly larger than the generated results due to the high process margins. DAuT [11] observes large deviations as well. However, values from SK Hynix are smaller than our values. This difference could be due to the power domain splitting (between VDD and VDDQ) used by SK Hynix, as it is partially possible to shift more I/O related circuits to the VDDQ (for I/O) voltage domain instead [12]. Obviously, this reduces the current consumption in the VDD domain. As we do not aim at reproducing DRAM datasheet values, these deviations do not limit the ability of exploring DRAM designs in a full system simulator. Our tool reflects the impact of DRAM design features, for example the increase in $I_{DD0}$ current while increasing DRAM row-buffer size as shown in Figure 1 and the increase in $I_{DD4R}$ while increasing I/O pins as shown in Figures 6 and 7.

B. DDR3: DRAMSpec versus Real Measurements

In order to verify the results of DRAMSpec, we compare key currents of a DDR3 model to datasheet values, measurements and NGSPICE model values presented in [8]. The NGSPICE model also presented in [1] considers worst-case currents without process margins. The measurements are done on a 512MB DDR3 module mounted on a Xilinx ML605 board as stated in [8]. Each DIMM has 4 x16 DDR3-1066 MICRON devices. Using DRAMSpec, we model the 1Gb x16 DDR3-1066 device which has 8 banks, a burst length equal 8 and a row buffer size of 2KB. Similar to the DDR3 SPICE model, we assume a 4x nm technology node. We compare the results of 3 key current parameters against the measurements on real hardware. As seen in Figure 8, compared to the measured currents, the error margins of the currents $I_{DD0}$, $I_{DD4R}$ and $I_{DD5}$ are much less than the margins of the NGSPICE and datasheet values. Although our tool mainly aims at visualizing the relative impact of DRAM architectural parameters on the latency and power specifications, the lower error margins verify our realistic model.

C. Improving DDR3 Design for Memcached Application

Using DRAMSpec as part of gem5, we design an experiment to explore the impact of DRAM design decisions on a server workload. We evaluate the impact of varying the row-buffer size and the number of banks of a DDR3 DRAM on performance and DRAM energy consumption. A realistic application in the server world is a server/client system where the client sends requests to the server and the server responds. As an example for such applications, this experiment considers memcached-Twitter with a server configuration in gem5. The system configured in gem5 is made up of two systems. One system represents the client system sending requests to the server system which in term processes the requests and responds back to the client. Since the experiment performed in this work explores the DRAM energy for the server system, Table III provides the configuration parameters for this system. The server system configured in gem5 is a made up of a single core 2GHz out-of-order CPU with three levels of caches. The first level of cache consists of a 32KB data cache and a 32KB number of banks, row-buffer size, and more. Using the memory controller model implemented in gem5, system explorers have the ability to analyze the performance of the designed chips while running certain types of workloads. The current integration of DRAMPower [5], a DRAM power calculating tool, in gem5 enables system explorers to also analyze the power consumption of the modeled DRAM chip.

VI. RESULTS

A. DDR3: DRAMSpec versus Datasheet

Figures 6 and 7.
instruction cache. The second level and the third level are a 256KB and 1MB cache respectively. The DRAM configured is a single channel single rank DDR3 with four devices per rank. This provides 2GB of main memory. The experiments performed here evaluates three different cases for a base x16 DDR3-1066 model with 3x technology node: 1) Decreasing the row-buffer size from 1KB to 512B, 2) Increasing the number of banks from 8 to 16 and 3) Case 1 and Case 2 combined together. Before exploring the impact of these changes on the performance and energy consumption of a workload, an analysis of the effect of each change on the latency and power parameters of the DDR3 DRAM is provided.

1) Case 1: Reducing the Row-Buffer Size (DDR3r): The first part of the performed experiments focuses on the impact of reducing the row-buffer size from 1KB to 512B in the base model (DDR3) to 512B in the new model (DDR3r) on the latency and power parameters of the modeled DDR3 chip. Using DRAM-Spec, the relative variation in these parameters is captured. This is shown in Figure 9, where the activation energy drops by 50%. Although, the refresh energy per command decreases; in total this energy remains the same because the DRAM should be refreshed more frequently. Finally, the read and write energy mainly depends on the bank dimensions; thus their values remain constant since these dimensions do not vary for the newly modeled chip.

2) Case 2: Increasing the Number of Banks (DDR3p): Another experiment is done to explore the impact of increasing the number of banks on the DRAM latency and power parameters. The number of banks is increased from 8 in the base model (DDR3) to 16 in the new model (DDR3p). Increasing the number of banks while keeping the memory size constant reduces the area and the dimensions of each bank. The resulting bank offers 20% reduction in the bank width and 30% reduction in the bank height according to the simulation results. The core frequency could be viewed as the speed for moving data from the DRAM cells to an I/O buffer located at the bottom of the DRAM bank. Therefore, as the bank height decreases, the latency towards the I/O buffer drops which results in a higher core frequency. Therefore, the newly modeled bank offers an increase in the interface frequency, a multiple of the core frequency, from 800MHz to 1066MHz. Figure 10(a) shows the impact of this design parameter on the latency specifications of the modeled DRAM. The size of the newly modeled bank is half the size of the base DDR3 bank; therefore, since the row-buffer size (number of columns) remains unchanged, the number of rows is reduced by half. This implies that the refresh rate drops by 50% which can be observed for t\textsubscript{REFI} (Refresh-Interval). Unfortunately, t\textsubscript{RFC}, the time required by each refresh command to complete, increases by approximately 50%. As the height of the bank is reduced, the read latency (t\textsubscript{RL}) drops by 17% and the core frequency increases by 33%. Figure 10(b) presents the effect of the change done in this experiment on the power specifications of the modeled DRAM. As the row-buffer size

![](image)
remains unchanged, no variation in the activation/precharge energy is observed. Although the refresh energy per command is doubled, the total refresh energy remains constant due to half the number of issued refresh commands. Finally, the read/write energy mainly depends on the length of the DRAM bank and is therefore reduced by 38%.

3) Case 3: Reducing the Row-Buffer Size and Increasing the Number of Banks (DDR3rp): The last case combines the changes done for case 1 and 2 in one experiment. Figure 11(a) shows the effect of the changes on the DRAM latency parameters. The refresh rate of the new bank model (DDR3rp) is the same as the one of the base model (DDR3). Although the number of rows is doubled (case 1), the bank size is also reduced by half due to the doubled number of banks (case 2). This leads to no variation in the number of rows. However, an increase in the number of banks results in a longer refresh time per refresh command ($t_{RFC}$) as seen in Figure 11(a). Similar to results of case 2, the read latency drops by 13% and the core frequency rises. The new model also runs at frequency of 1066MHz.

Figure 11(b) shows the power specifications of the new DDR3 model compared to the ones of the base model. The activation energy decreases by almost 50% of its base value due to the reduction of the row buffer size and the read/write energy are reduced by 38% due to the smaller bank size. As expected, the refresh energy per command is unchanged.

D. Exploring Energy and Bandwidth of Case 1 to 3 for a Twitter Server System running Memcached

Figure 12 shows the impact of the different configurations on bandwidth and average memory access latency of the analyzed memcached-twitter workload. As seen in Figure 12(a), the modeled workload has a low memory intensity since the DRAM bus utilization is around 10% for all configurations. However, the memory access latency, how much the CPU has to wait for a DRAM access, is an essential performance metric in realistic systems. Reducing the row buffer size impacts the latency negatively as more activations will be required (more misses); however, the effect is negligible (an increase of only 6% as seen in Figure 12(b)). In this context, considering a single core setup is sensible since using multi-cores leads to even less locality, thus worsening the latency for a given row-buffer size. Additionally, increasing the number of banks provides more parallelism and thus enhances performance. This could be observed through a 20% and 16% drop in the average memory access latency for DDR3p and DDR3rp respectively (Figure 12(b)).

Moreover, the DRAM energy per bit drops from 24 pJ/Bit to about 16 pJ/Bit for the proposed DDR3rp. The drop in the read energy contributes significantly to the energy saving as seen in Figure 13. The decrease in the activation energy due to the smaller row buffer size offers an additional saving as seen for cases DDR3r and DDR3rp.

![Figure 11](image1.png)  ![Figure 12](image2.png)  ![Figure 13](image3.png)
However, since we are considering a full-system simulation, it makes more sense to analyze the DRAM energy cost of each server response as the possible negative impact on the system performance could be captured. DDR3rp provides here also the highest saving (13% compared to the base DDR3 energy consumption). In this experiment, our optimized DDR3 design with an increase in the number of banks and a reduction in the row buffer size offers for our target server application a better performance and energy solution. Unfortunately, a higher number of banks has a negative effect on the area of the DRAM chip which in term means that the cost of the chip will rise. This could be an obstacle for obtaining optimized DRAM solutions.

E. Modeling a HMC Bank with DRAMSpec

One of the essential merits of DRAMSpec is that it offers an estimation of DRAM datasheet specifications prior to their release by DRAM vendors. Since our tool level of abstraction is a DRAM bank, it makes no difference if we are modeling 2D or 3D DRAM. As HMC is gaining more importance, we use DRAMSpec to model a HMC device. For modeling HMC, we use the architectural information found in [24]. A HMC consists of 4 stacked memory die layers on top of a logic base. Each layer is composed of 16 memory vaults with 2 banks per vault. With a bank capacity of 4MB, the device capacity sums up to 4Gb. Moreover, the device which runs at 1.25 GHz has a burst length of 8 and a x32 I/O interface with a relatively small row-buffer size (256 Bytes). We use this information to model a HMC layer with DRAMSpec. To show the merits of having a HMC design, we compare the key DRAM core energies to the energies of a 1Gb x16 DDR3-1066 DRAMSpec model with a row-buffer size of 2KB. The results shown in Figure 14 show a 90% saving in the activation energy due to the smaller row-buffer size and a 50% saving in the per bit read/write DRAM core energy due to the smaller bank size and the larger number of interface pins.

VII. CONCLUSION

Whether researching mobile or server systems, considering DRAM is essential in determining the system’s power and performance. In addition to a realistic DRAM controller model, and a realistic workload, which in turn requires a full system simulator that runs a complete software stack, a high level model of the DRAM bank is required and crucial for future system architecture explorations. In this work, we introduce DRAMSpec, an open-source DRAM bank modeling tool. The integration of our tool in a full-system simulator, gem5, allows system designers to easily explore and study DRAM design requirements for their target applications and use-cases. The experiment done in this work varies essential DRAM design features and aims at motivating potential users of our proposed tool to start researching future DRAM designs.

In our future research, we will vary more DRAM design features to see a more clear influence of DRAM design on the power and performance of the system. We will also include GPU traffic and perform more DRAM design exploration studies on various memory-intensive workloads.

REFERENCES