IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 12, 2020, accepted October 4, 2020, date of publication October 23, 2020, date of current version November 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3033453

Walter: *W*ide I/O Sca*l*ing of Number of Memory Con*t*roll*e*rs Versus F*r*equency and Voltage

MARIO DONATO MARINO, (Member, IEEE)
School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds LS6 3QS, U.K.
e-mail: m.d.marino@leedsbeckett.ac.uk

ABSTRACT Computational application demands do push the scaling of the number of cores, which themselves further increase the demand for more bandwidth. The use of larger rank widths and/or scaling the number of memory controllers (MCs) is a straightforward way to increase memory bandwidth. Connecting wide ranks and MCs via low-capacitance Through Silicon Vias (TSVs) favors high-bandwidth 3DStacking systems (e.g. Wide I/O). Given that voltage and frequency scaling (VFS) lower power utilization but the use of lower clock frequencies reduces bandwidth, this article proposes *Walter* as a *W*ide I/O technique that trades off sc*al*ing of the number of memory con*t*roll*e*rs (MCs) versus clock f*r*equency and voltage (VFS) to mitigate low bandwidth and improve energy-per-bit usage. Our findings show that *Walter*'s Wide I/O architectural benefits of using a larger number of MCs coupled with wider ranks when combined to VFS are promising: compared to the baseline for a 75% frequency/voltage reduction, MC scalability improved memory bandwidth by 2.4x and energy-per-bit reduced by 20% (most benchmarks for up to 16 MCs). *Walter*'s architectural replacement of ranks set at specification frequencies with ones set at lower frequencies allows temperature reduction thus likely allowing further rank stacking.

INDEX TERMS Bandwidth, controller, memory, wide I/O, 3DStacking.

I. INTRODUCTION

Other than the power wall problem, memory contention due to core scaling demanding for more memory bandwidth typically lowers the overall system performance in the multicore era. Alternatively, in areas such as Data Science, current application trends are likely to further increase the pressure on the memory system. In order to improve memory bandwidth, commonly used memory design mechanisms include memory clock frequency scaling (simply frequency scaling or FS) as well as increasing rank width and/or the number of memory controllers (MCs) [1]. Among these techniques the most used one in the market is (i) memory FS [2]. Very well established and spread Double-data-rate (DDR) family generations strongly rely on FS, which has enabled memory bandwidth to be enlarged by 2x(DDR2), 3x(DDR3), (4x)(DDR4) for CPUs and also for GPUs (5x, DDR5).

The drawbacks of increasing memory bandwidth via (i) FS are the augments in both amount of power spent and larger temperatures. One method to lower power after applying FS is to lower voltages (V, which when combined to FS turns into VFS), for instance as used in low-power DDR (LPDDR) systems. Individually lowering rank clock frequencies (or simply rank frequency) does negatively impact performance and power. Taken into consideration that longer data burst times and MC front engine as well as transaction engine set at lower clock frequencies the amount of transactions queued and the time to prepare/assign them to the ranks are likely to increase. Furthermore, lowering rank clock frequencies is reported [3] to increase read/write energy almost linearly. Moreover, dynamically (D) lowering voltage and scaling frequency (turning into DVFS) to adapt to specific program bandwidth needs is also a widespread technique [3].

Given that memory is one of the most power-hungry parts in current computer systems, an alternative approach to VFS is to employ (ii) wider memories. Wider memories can be implemented via increasing the number of ranks connected to their respective memory controllers (MCs) - in this article simply referred to increasing or scaling the number of MCs/MC counts or MC scalability - and/or using larger-width

The associate editor coordinating the review of this manuscript and approving it for publication was Yue Zhang.

ranks [1], [3]. Either scaling a large number of MCs/ranks or using wider ranks/MCs to achieve a larger bandwidth is limited by the I/O pin problem, which is characterized by pin count and MC count kept at low-magnitude levels due to costs, density per unity of area limits, and large board capacitances [1].

In response to these challenges regarding the scaling of MCs/ranks and wider ranks usage manufacturers have developed High-Bandwidth Memory (HBM), Hybrid Memory Cube (HMC) and Wide I/O memory technologies [4]. HBM, HMC and Wide-I/O are all 3DStacking techniques which approach the I/O pin problem in terms of pin counts, latency, bandwidth, and energy utilization through the use of Through Silicon Vias (TSVs) to connect 3DStacking dies directly to the processor (to guarantee communication on-chip only). It is important to highlight that, as 3DStacking techniques, they do not present I/O pin count limitations. Moreover, the use of TSVs significantly reduces communication delays and processor-to-memory capacitance (e.g. 2pF) [5]).

In particular, these technologies employ ranks in the 128-256-bit range, which is much larger than typical 64-bit DDR-based interfaces, aiming to improve individual rank bandwidth magnitude. Furthermore, by having a stack of multiple dies, each one connected to a different MC, these 3DStacking techniques present a higher number of MC counts/ranks than regular DDR off-chip interfaces. For example, according to JEDEC's report [6], up to 16 MCs/ranks can be used without special cooling techniques in Wide I/O versus 2MCs in DDR/LPDDR typical interfaces.

Comparatively to HBM and HMC, given that Wide I/O is compatible with the widely-used DDR standard, it is likely to be a less expensive technology if adopted in 3Dstacking. Despite not having its production started yet, the very long development time and establishment of the DDR technology alongside different family generations (1x to 5x), in terms of memory protocols and robustness are strong reasons pointing towards the use of this technology. Moreover as previously pointed, it brings the DDR technology to 3DStacking which allows MC/rank scaling and employment of larger rank widths.

The main aim of this article is to tackle the previously mentioned low-bandwidth due to the use of low rank frequencies via MC/rank scaling in 3DStacking using widely available/established market technology (DDR). To the best of our knowledge, this important trade-off frequency/width has not been previously investigated in 3DStacking memory systems yet is DDR-compatible such as Wide I/O. Though similar reasoning was proposed by Olukotun *et al.* [7] report, where multicore memory bandwidth of a larger number of simple low-clock processor cores is higher than beefier high-frequency lower number of processor cores, we propose a novel approach named *Walter* for *W*ide I/O systems, where it trades off the sca*l*ing of the number of memory con*t*roll*e*rs (MCs) and ranks versus lower rank clock f*r*equency and voltage, aiming to leverage the state of the art in memory systems through the following contributions:

- *Walter*'s architectural novelty trades off MC scalability versus lower bandwidth due to (i) individual lower rank frequency magnitudes and (ii) lower rank voltages (iii) and/or when combined with static rank VFS in Wide I/O systems. This static approach is a preliminary investigation aiming to pave a future dynamic (DVFS) mechanism.
- *Walter*'s approach uses larger MC scalability and wider ranks, yet very importantly in combination with a frequency that is significantly lower (e.g. 50 MHz) than the spec Wide I/O *spec*-frequency (typically 200MHz), or to HBM (200MHz) or even to DDR family (666MHz-2800MHz). With the employment of such low-magnitude frequency-range, this design space exploration combines them with proper low voltages to understand their effects on bandwidth-drop, to be approached by *Walter* scaling of the number of MCs. This exploration employs detailed-accurate system simulators [8], [9].
- For the first time, we investigate whether architectural increasing of the number of MCs proposed in *Walter* are likely to compensate processor performance drop due to VFS lowering rank clock frequencies to very low magnitude levels.
- Given that MC scalability demonstrates to significantly improve memory systems bandwidth [1], we further propose to validate MC scalability for different rank clock frequencies and much larger rank widths (Wide I/O).
- Alternatively, to the best of our knowledge, bandwidth, energy and energy-per-bit (further defined) behavioral models are introduced aiming to understand the trade-offs MC scalability versus rank frequency/voltage scaling (VFS) in Wide-I/O systems. Whilst providing trade-offs and insights for designers of future memory solutions, these models aim to assist the design exploration. These models are also compared against validated modeling [10] incorporated in the detailed-accurate simulator [8] experimental results.

The rest of this article is organized as follows: section II introduces Wide I/O systems, the motivation behind the bandwidth and power restrictions whilst comparing advanced memory solutions in terms of approaching MC/rank scaling. Section III introduces the benefits of *Walter* in regards to memory bandwidth, energy and energy-per-bit modeling. Section IV describes the experiments and results that involve bandwidth and energy comparatively to the modeled ones, whilst Section V depicts the related work. Section VI concludes the paper.

II. MOTIVATION AND BACKGROUND

To better understand the motivations of the use of Wide I/O interface rather than other technologies, we first contextualize Wide I/O memories. Subsequently, we illustrate their operation and proceed a comparison between Wide I/O against traditional LPDDR systems. Following that,

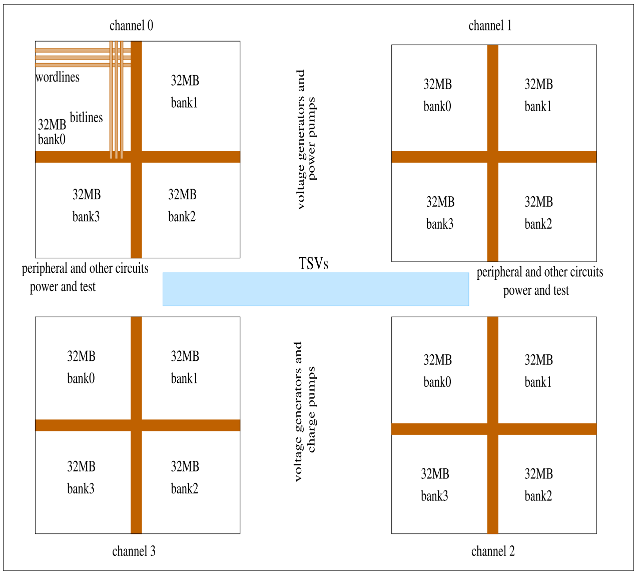

FIGURE 1. 4-stack Wide I/O DRAM example replicated from [5].

we discuss pros/cons of applying memory FS, voltage and MC scaling (increase of the number of MCs) on Wide I/O systems.

A. WIDE I/O AND OPERATION

The general operation of typical memory systems - Wide I/O itself included - is described as follows. In typical DRAMs, sets of banks containing memory elements are organized in arrays of rows and columns. These arrays are hierarchically structured in sub-arrays to explore efficient routes and power consumption reduction. Each sub-array is composed of memory cells which are connected to local wordlines and bitlines.

In Figure 1, one layer of a Wide I/O 3DStacking is exemplified whilst four copies of the same layer represent a complete Wide I/O 3DStacking configuration. In this figure, each quadrant contains memory arrays, bitline/wordline drivers, control logic and sense amplifiers. In addition, TSVs are represented by the light blue area. According to Chandrasekar *et al.* [5], power network, test pads, charge pumps for the high voltage wordline, voltage generators and peripheral circuits are all shared between the channels.

B. WIDE I/O AND LPDDR

Still, Chandrasekar *et al.* report that with the exception of different voltage sources, Wide I/O memory commands - activate, precharge, read, and write - are very similar to DDR/LPDDR ones as follows: (1) to read data from the memory, precharge and activate commands are issued; the former prepares bitlines in terms of voltage settings whilst the latter sets the wordline to a high level and performs the data transfer from the cells to the bitlines. (2) Subsequently, data transferred are stored in the row buffer (sense amplifiers). (3) In addition, using columns to select lines, read commands perform the reading out of the data from the row buffer. (4) Further, data go through datalines and reach the secondary sense amplifiers and I/O ports. (5) At the end of this transfer, wordlines can be switched off, cell capacitors disconnected, and local bitlines submitted to the same cycle which has started with the precharge operation.

Yet, according to Chandrasekar *et al.*, although using DRAM typical circuitry and employing low-capacitance TSV interconnects, rather than typical I/O pin-based circuitry, Wide I/O 3DStacked DRAMs reduce or eliminate some of the circuits in order to reduce power consumption. For example, on-die termination (ODT) is eliminated comparatively to LPDDR memories, whilst replacing delay-locked loop circuit (DLLs) with bus-to-clock programmable delays techniques to keep either power consumption and latencies at lower levels. Still relying on the previous report, several elements with complete voltage (VDDQ) are present in Wide I/O (I/O buffers/drivers, data TSVs and micro-bumps), rather than partially or absently respectively in LDDR2/3 and DDR2/3 memories.

To further exemplify, the report by Khan *et al.* [11] employs TSVs rather than standard packaging interconnection, and shows power saving findings in the range of 75 to 80%, i.e., the I/O read power-per-bit is 0.7mW in Wide I/O rather than 2.3mW in LPDDR2 and 4.6mW in DDR3.

C. FREQUENCY, LARGER RANK WIDTHS, MULTIPLE MCs IN WIDE I/O

Starting from the bandwidth formulation by Marino and Li [1] to initially express bandwidth as a function of width and frequency, we have:

$$bandwidth = width * frequency \tag{1}$$

According to Polka's [12] predictions, current microprocessors range width keeps at low MC count magnitudes - themselves count-restricted due to scalability of pin counts as reported by Marino and Li [1]. Still according to the latter, as a general approximation in current microprocessors, MC counts are proportional to the logarithm of the number of cores (MC counts = *log* core counts), therefore of low degree magnitudes.

Assuming typical DDR context i.e. with low MC counts, FS is a straightforward mechanism to achieve large bandwidth magnitudes, as it can be qualitatively inferred from equation 1. For instance, frequency has been increased by a factor of 10x since the launch of DDR families, and as a consequence, bandwidth has improved by a similar (10x-factor) [13].

The use of larger frequencies causes larger power utilization, which has been approached via low power voltages in LPDDR3/LPDDR4 [2]. Aiming power consumption reduction - given specific circuit explanations previously mentioned, Wide I/O employs lower rank frequencies when compared to typical DDR/LPDDR memories. For example, typical Wide I/O rank frequencies are specified in the range of 200MHz-300MHz [6], while traditional DDR/LPDDR ones in the range of 666MHz to 2800MHz or higher [2].

Given that Wide I/O employs lower rank frequencies to save power, we immediately derive from equation 1 that a larger memory rank width is required in order to achieve a high bandwidth magnitude. This represents a shift of the focus on equation 1, i.e., in typical DDR/LPDDR systems the focus is on the second factor, whilst in Wide I/O is on the first one. As an example of this shift, according to JEDEC [6], Wide I/O is designed for ranks of up to 256-bit wide, which are 8x wider than a typical DDR/LPDDR rank (64 bits).

D. MC SCALABILITY / LARGER NUMBER OF MCs

Assuming memory accesses parallelized/interleaved among its MC counts, since Wide I/O allows the use of a larger number of MCs/ranks, bandwidth is likely to be improved. Very importantly, for different benchmarks and using different DDR rank settings, scalability of MCs connected to DDR ranks is proven to provide up to 8.6x more bandwidth [1], [14], which especially motivates the investigation of the previously mentioned important motivation, i.e., whether the use of higher MC counts can compensate or cancel the effects of lower bandwidths due to lower rank frequency settings (that lowers energy utilization level). Thus, given the similarities with DDR/LPDDR systems, since Wide I/O is designed to have up to 16 MCs [6], Wide I/O systems are likely to present much larger bandwidth magnitudes than the ones in microprocessors with typical DDR/LPDDR systems with 2-4 MCs. Next section proceeds towards the investigation of the bandwidth and energy/energy-per-bit using different rank frequencies (VFS) and number of MCs (memory width, MC counts) in *Walter*.

III. *Walter*: TRADING-OFF NUMBER OF *MCs* AND VFS

In this section we investigate *Walter*'s trade-off, which consists in evaluating rank VFS versus the increase of the number of MCs (or MC counts) and its effects in terms of bandwidth and energy/energy-per-bit aspects.

A. FREQUENCY COMPONENT

From now on, modeling is introduced aiming to understand the previously described trade-off effects. To start, we begin with the definition of maximum rank bandwidth *rankbw* at a general frequency *gfreq* from [1] as:

$$rankbw(gfreq) = rankwidth \,.\, gfreq \tag{2}$$

Subsequently, we adopt the specification rank frequency (*spec*) defined by Micron [2] in equation 2 and rewrite it as:

$$rankbw(spec) = rankwidth \,.\, spec \tag{3}$$

For a general *gfreq* frequency, the total maximum bandwidth *tbw* generated by having a generic *MCcount* number of MCs can be defined as:

$$tbw(gfreq) = MCcount \,.\, rankbw(gfreq) \tag{4}$$

To implement *Walter*'s strategy, *MCbaseline* is defined as a representative of the low-magnitude baseline MC counts (further explained in Section IV). Thus, the total maximum rank bandwidth for *spec* frequency and *MCbaseline* can be similarly defined as:

$$tbw(spec) = MCbaseline \,.\, rankbw(spec) \tag{5}$$

Next, the maximum bandwidth ratio (*Bwr*) between total maximum bandwidth at *gfreq* and *spec* frequencies can be defined as:

$$Bwr = \frac{tbw(gfreq)}{tbw(spec)} \tag{6}$$

Combining the previous equation with equations 3 and 4, *Bwr* can be rewritten as:

$$Bwr = \frac{[MCcount * rankbw(gfreq)]}{[MCbaseline * rankbw(spec)]} \tag{7}$$

or:

$$Bwr = \frac{MCcount}{MCbaseline} \cdot \frac{rankbw(gfreq)}{rankbw(spec)} \tag{8}$$

According to Marino and Li's report [1], upon the presence of higher memory traffic there are several factors that degrade bandwidth (e.g. as larger delays and L2-crossbar contention, etc.), which as the number of MCs increase can further affect performance. Mentioned in the latter report, when using memory-bandwidth benchmarks, these bandwidth restriction factors are approached by using an advanced crossbar design to cope with the high memory traffic due to the employment of a larger number of MCs (32 MCs, when compared to a 2MC-baseline) generating 9x more bandwidth on a 32-core processor. Thus, *Bwr* in equation 8 should be degraded to reflect bandwidth. By observing bandwidth magnitudes on the latter report, a power formulation approximation [15] seems likely, once an asymptotic reduction of the bandwidth is expected as MC counts increase. This likely approximation is further verified (Section IV) but assumed for now by modifying equation 8:

$$Bwrd = k \,.\, \left(\frac{MCcount}{MCbaseline}\right)^{1-degree} \cdot \frac{rankbw(gfreq)}{rankbw(spec)} \tag{9}$$

where *Bwrd* is degraded bandwidth and *k* depends on the previously mentioned degradation factors (crossbar delays and contention, memory traffic, etc.), as well as $1 - degree < 1$, which is able to represent in a power asymptotic manner the bandwidth behavior as MC counts are increased.

Assuming the number of *MC*s constant and *gfreq* being varied, a similar behavior to the previous case is expected. The straightforward consideration would be to have bandwidth reduced in the same proportion as rank frequency (equation 7), which likely follows the asymptotic polynomial behavior previously mentioned and already captured by the $(1-degree < 1)$ exponent.

Though straightforward, a simple analysis of equation 9 is important because it highlights *Walter*'s bandwidth-power trade-offs: with $MCbaseline = 2MCs$, at least $MCcount = 4MCs$ should be used to achieve similar *Bwr* ratio. However to achieve larger bandwidths, the designer should employ more than 4 MCs. If *gfreq* is half or a quarter of *spec* (in case of an aggressive power saving attempt), the designer should double or quadruple the number of MCs. Therefore, *Walter* experimentation (Section IV) should validate that to have *Bwr* larger than 1, more than 8 MCs should be used: actually as further shown, our experiments employ larger *MCcounts* (such as 16 and 32 MCs). Subsequently, power and energy/energy-per-bit aspects are evaluated.

B. POWER COMPONENT IN THE TRADE-OFF

To concentrate on the frequency effects, this section starts with the formulation proposed by Micron [16], which derives the rank power via de-rating (reducing the memory frequency rate) from the specification frequency *spec* to a general *gfreq*:

$$rankpw(gfreq) = \frac{rankpw(spec)\ .\ gfreq}{spec} \tag{10}$$

with *rankpw*(*spec*) representing the rank power configured at the specification frequency *spec*. and *rankpw*(*gfreq*) representing the rank power configured at a general frequency *gfreq*. Considering the above equation, if *gfreq* is lower than *spec*, power *rankpw*(*gfreq*) is also reduced, otherwise if *gfreq* is higher than *spec*, the opposite effect happens.

Now we define *tpw* as the total memory power spent at the set of MCs and ranks available at a general frequency *gfreq*:

$$tpw(gfreq) = trankpw(gfreq) + tMCpw(gfreq) \tag{11}$$

where *trankpw* is the total power spent at the ranks and *tMCpw* which is defined as total the power used at the MCs, which includes the power of the transaction engine (TE), front end engine (FE), and the I/O part (TSVs and circuitry). With the previous formula (11) set at the specification frequency (*spec*) and with *MCbaseline* MCs, *tpw*(*spec*) is derived as:

$$\begin{aligned} tpw(spec) = {} & MCbaseline\ .\ rankpw(spec) \\ & + MCbaseline\ .\ MCpw(spec) \end{aligned} \tag{12}$$

Given the remaining power circuitry elements previously mentioned, this equation represents the trade-off between frequency and width. In order to get lower energy-per-bit levels at higher frequencies, we rewrite the previous equation 12 as:

$$tpw(spec) = MCbaseline\ .\ (rankpw(spec) + MCpw(spec)) \tag{13}$$

Likewise, for a general frequency *gfreq*, we derive:

$$tpw(gfreq) = MCcount\ .\ (rankpw(gfreq) + MCpw(gfreq)) \tag{14}$$

with similar magnitude analysis when it comes to have *gfreq* ≪ *spec*. Total power ratio *tpwr* can be defined as the ratio between total power at *gfreq* (equation 13) and *spec* (equation 14) frequencies:

$$tpwr = \frac{MCcount\ .\ (rankpw(gfreq)+MCpw(gfreq))}{MCbaseline\ .\ (rankpw(spec)+MCpw(spec))} \tag{15}$$

C. VOLTAGE COMPONENT IN THE TRADE-OFF

Assuming power proportional to the square of the voltage [11], previous equation 14 can be approximated to:

$$tpw(Vg) = tpw(Vs)\ .\ \frac{Vg^2}{Vs^2} \tag{16}$$

where *Vg* represents a generic voltage and *Vs* the specification voltage. This voltage aspect will be combined to the energy formulation development at the end of the next section.

D. ENERGY AND ENERGY-PER-BIT

To explore the energy side of the trade-off width (MC scalability) and rank frequency, we start this subsection by developing a formulation that starts by comparing energy levels of higher MC counts to lower ones. Consonantly, the baseline energy (*Enbaseline*) at the *spec* frequency is defined as follows:

$$Enbaseline(spec) = tpw(spec)\ .\ time(spec) \tag{17}$$

where *twp*(*spec*) is the total power spent and *time*(*spec*) the interval to execute the program at *spec* frequency. Further, combining equation 11 with *MCbaseline* MCs, equation (17) turns into:

$$\begin{aligned} Enbaseline(spec) = {} & MCbaseline\ .\ [rankpw(spec) \\ & + MCpw(spec)]\ .\ time(spec) \end{aligned} \tag{18}$$

where *time*(*spec*) is the time to execute the program at *spec* frequency. Likewise, for a general *gfreq* frequency and general *MCcount*, energy at a general frequency *En*(*gfreq*):

$$\begin{aligned} En(gfreq) = {} & MCcount\ .\ [rankpw(gfreq) \\ & + MCpw(gfreq)]\ .\ time(gfreq) \end{aligned} \tag{19}$$

where *time*(*gfreq*) is the interval time to execute the program at *gfreq* frequency.

Now, we define *Enr* as the ratio between baseline energy (*Enbaseline*) at *spec* frequency and energy at *En*(*gfreq*) at *gfreq* frequency:

$$Enr = \frac{Enbaseline(spec)}{En(gfreq)} \tag{20}$$

To finalize, we combine equations 13, 14 and 20:

$$Enr = \frac{[MCcount.(rankpw(gfreq)+MCpw(gfreq))].time(gfreq)}{[MCbaseline.(rankpw(spec)+MCpw(spec)).time(spec)]} \tag{21}$$

which can be rewritten as:

$$Enr = \frac{MCcount}{MCbaseline}\ .\ \frac{[(rankpw(gfreq)+MCpw(gfreq)) * time(gfreq)]}{[(rankpw(spec)+MCpw(spec)).time(spec)]} \tag{22}$$

Previous energy modeling equation 22 does not include the amount of memory bits transferred, which is fundamental on a memory system. To incorporate this, we include the amount of bits transferred as bandwidth per unit of time as in [1]:

$$Enpb(spec) = \frac{tpw(spec)}{tbw(spec)} \tag{23}$$

where *Enpb* represents the energy-per-bit and *tbw* the total bandwidth both at *spec* frequencies. Accordingly, similar formulation can be obtained at *gfreq* frequency. Using previous formulas 7 (bandwidth ratio) and 15 (maximum rank power ratio), energy-per-bit ratio between *gfreq* and *spec* frequencies can be defined as:

$$Enpbr = \frac{Enpb(gfreq)}{Enpb(spec)} \tag{24}$$

or:

$$\begin{aligned} Enpbr &= \frac{MCcount}{MCbaseline} \\ &\quad \cdot \frac{(rankpw(gfreq) + MCpw(gfreq))\ .\ time(gfreq)}{(rankpw(spec) + MCpw(spec))\ .\ time(spec)} \\ &\quad \cdot \frac{MCbaseline\ .\ rankbw(spec)}{MCcount\ .\ rankbw(gfreq)} \end{aligned} \tag{25}$$

which could be simplified to:

$$\begin{aligned} Enpbr = \frac{(rankpw(gfreq) + MCpw(gfreq))\ .\ time(gfreq)}{(rankpw(spec) + MCpw(spec))\ .\ time(spec)} \\ \cdot \frac{rankbw(spec)}{rankbw(gfreq)} \end{aligned} \tag{26}$$

According to the energy-per-bit magnitudes shown in Marino and Li's report [1], energy-per-bit curve behavior typically achieves at an absolute minimum, and later it raises, which we generally interpret as a polynomial or parabola-like shape given it involves several trade-offs, among them power, bandwidth, execution time and different program behaviors. The minimum magnitude observed is a function of the amount of memory transactions per MC, number of MCs and time to process transactions. The further raise happens due to the fact that the number of transactions per MC starts to decrease as the number of MCs is increased.

To facilitate understanding of equation 26, we start by disregarding voltage-related parameters. Subsequently, the following cases are considered in order to intuitively understand the effects of the former formulation and to further assist its validation:

case (i):

$$\begin{aligned} &if\ \ rankpw(gfreq) \gg MCpw(spec)\ \ and \\ &if\ \ rankpw(spec) \gg MCpw(spec) \end{aligned} \tag{27}$$

In this case, with the de-rating derivation developed in equation 10 combined to equation 23, equation 26 can be rewritten as **case (ii)**:

$$\begin{aligned} Enpbr &= \frac{(\frac{rankpw(spec)\ .\ gfreq}{spec} + MCpw(gfreq))\ .\ time(gfreq)}{(rankpw(spec) + MCpw(spec))\ .\ time(spec)} \\ &\quad \cdot \frac{spec}{gfreq} \end{aligned} \tag{28}$$

that could raise the following:

$$\begin{aligned} &if\ \ rankpw(gfreq) \sim MCpw(spec)\ \ then \\ &\quad rankpw(spec) \gg MCpw(spec) \end{aligned} \tag{29}$$

thus, equation 23 is still valid and *Enpbr* shows similar behavior to equation 28.

And **case (ii)**:

$$\begin{aligned} &if\ \ rankpw(spec) \gg MCpw(spec)\ \ then \\ &\quad rankpw(gfreq) \ll MCpw(spec) \end{aligned} \tag{30}$$

where equation 23 is still valid and *Enpbr* produces a similar behavior to one previously observed. If the three following statements are present in a configuration: (i) larger number of MCs versus the baseline ($MCcount \gg MCbaseline$); (ii) some degree of energy efficiency ($Enpbr < 1$); (iii) some of the following conditions whether $gfreq < spec$ or $gfreq \ll spec$; then, if the target is to have $Bwr > 1$, consequently $MCcount/MCbaseline > gfreq/spec$ is likely achievable. For instance, with 8, 16 and 32 MCs and $gfreq/spec = 0.5\ or\ 0.25$, the product of both in equation 28 is larger than 1.

If only 1MC is present, *time*(*gfreq*) will be roughly calculated by $gfreq/spec \cdot time(spec)$ plus some extra-overheads of queuing memory requests at the memory queue for the particular MC. However, since there are several MCs available, $time(gfreq) < time(spec)$ can be a reasonable assumption because memory-bound programs will take advantage of the larger MC-scalability [1], even if ranks are clocked at lower *gfreq* magnitudes comparatively to *spec* ones.

We now incorporate the voltage parameter and the modeling cases are considered:

case(i):

$$\begin{aligned} &if\ \ rankpw(gfreq) \gg MCpw(spec)\ \ and \\ &if\ \ rankpw(spec) \gg MCpw(spec) \end{aligned} \tag{31}$$

In this case, with the de-rating derivation developed in equation 10 combined to equation 23, as well as including voltage by combining equation 23 with previous 13 and 14 ones, we define energy-per-bit ratio *Enpbr* as:

$$Enpbr = \frac{\frac{MCcount}{MCbaseline} \cdot \frac{time(gfreq)}{time(spec)} \cdot g\frac{freq}{spec} \cdot \frac{Vfreq^2}{Vspec^2}}{Bwr} \tag{32}$$

case (ii): if $Vfreq \ll Vspec$ and a low-magnitude *Vfreq* is associated with a low-magnitude *gfreq*, which are cases of a typical VFS combination [11], *MCcount* should be significantly larger than *MCbaseline* targeting a larger *Bwr* whilst aiming to have $Enpbr < 1$ or as lower as possible. For example, if obtaining $Enpbr \sim 1.0\ or\ 1.15$ (energy-per-bit levels than the baseline by 10 and 15%) but with $Bwr > 1$ due to the higher number of MCs, then a combination of the previous may lower energy-per-bit levels.

case (iii): characterized by low-magnitude bandwidths ($Bwr < 1$), due to the adoption of aggressive combinations of voltage and frequency reductions (VFS), bandwidth reduction can still be mitigated by having a larger number of MCs. A similar trend to *Walter*'s bandwidth trade-off but applied to a very different context: scaling processor cores demand more bandwidth was demonstrated by Olukotun *et al.* [7] report, which has shown that having many cores set at lower frequencies produces a larger bandwidth than having one beefier core set at higher ones.

It is important to restate that equation 32 represents *Walter*'s trade-off between frequency and width proposed in this article. To highlight, in order to get lower energy-per-bit levels (*Enpbr* at higher frequencies, the larger power magnitude in *rankpw*(*freq*) due to larger frequencies is likely to compensate the larger bandwidth *rankbw*(*freq*). In this case, lowering rank frequency is likely to provide an energy-per-bit curve that has similar behavioral shape, but energy-per-bit magnitude levels that can vary.

Returning to equation 28, depending on the bandwidth demanded by the application, MC scalability and consequent power increase is likely to compensate the use of higher frequencies. According to Deng *et al.* [3], lowering frequency increases read/write and termination energy almost linearly, and power is not affected. However, memory accesses take longer times directly affecting energy: in equation 21, in longer memory accesses power and bandwidth are kept at the same levels. Furthermore, Deng *et al.* [3] reports that purely lowering rank frequency causes a degradation in performance, but also increases read and write termination energy due to remaining accesses kept for comparatively longer times. However, the combination of lowering rank frequency to higher MC count availability can improve not only energy-per-bit magnitude but also performance. Subsequently, Section IV aims to experiment and measure bandwidth, processor performance and energy/energy-per-bit as well as to validate the previously developed modeling.

IV. EXPERIMENTAL SECTION

In this section, we perform a series of experiments to determine the effects of *Walter*'s trade-off width (number of MCs or *MCcount*s) versus VFS scaling on bandwidth, processor performance and memory energy/energy-per-bit aspects.

Since different effects are expected, as a general approach the next steps are followed:

1) gem5 simulator [8] is composed of several subsystems (parts), each one responsible for simulating a different system of the architecture. For example, in gem5 there are processor, cache and memory subsystems among others. Each gem5 simulation will correspond to a different *Walter* configuration. We use (gem5) processor subsystem, to emulate the multicore system that is going to run memory-bound applications (further discussed) that will generate (gem5) cache transactions.
2) We employ a multilevel cache subsystem with parameters set with Cacti [17] cache simulator outputs, which are based on real cache parameters (e.g. associativity, hit latency time, etc further specified and discussed in next subsection). When receiving these previously mentioned cache transactions, the cache subsystem will generate memory transactions that will be captured by the memory subsystem.
3) To mimic each *Walter* configuration, the memory subsystem is configured with several numbers of MCs attached to ranks with different rank frequencies (*freq*) and voltages (*Vfreq*, *Vspec*) besides the standard specification ones (*spec*, *gfreq*) and Wide I/O memory settings. Following that, the memory subsystem responds to the previously stated memory requests generated by the cache subsystem.
4) Whilst gem5 memory power modeling is incorporated in gem5 memory subsystem, gem5 power modeling is validated and based on real components [10].
5) Furthermore, through using processor subsystem outputs beforehand, we derive processor power measurements of the applications using McPaT [18] simulator.
6) As formerly stated, to be able to understand the separate effects of rank voltage (V) clock frequency scaling (FS) as well as when both elements are combined (VFS), we perform a design space exploration with different rank VFS combinations.
7) Using the memory power measurements combined to McPaT processor power measurements previously stated as inputs to 3d-ICE [9] simulator, we determine temperature distribution when evaluating a large number of MCs combined with different VFS settings.

Subsequently, we concentrate on the methodology discussion settings for the simulation tools previously described aiming to reflect the trade-off frequency/voltage versus bandwidth and processor performance investigated here.

A. METHODOLOGY, BASELINE, BANDWIDTH AND PERFORMANCE

To evaluate *Walter* a clustered microprocessor architecture with 32 cores is selected in order to have enough memory pressure and demonstrate that higher bandwidth magnitudes can be achieved when the number of MCs is scaled up (to *MCcount*). Furthermore, to ensure higher memory pressure, OOO-processors (based on Alpha, 4-wide issue) have been employed with a shared L2 to reflect typical processor configurations. Furthermore, a banked-scalable L2 miss status holding register (MSHR) structure is assumed with enough MSHRs to allow misses to be directed to a scalable number of *MCs* that take care of memory requests [1].

To reflect low capacitance between memory and processor when adopting TSVs (low intercommunication delays), all following ranks have low delay settings based on Chandrasekar *et al.* methodological considerations [5]. In regards to the ranks themselves, to the best of our knowledge since Wide I/O components are not on the market, nonetheless we assume four versions: (i) starting with very conservative parameters using timings and voltage parameters derived from DDR memories with reduced delays based on Chandrasekar *et al.* observations [5] to address lower capacitancies, defined as the baseline version (further details discussed); (ii) applying 50% clock frequency reduction (100 MHz) together with 30% voltage reduction assumptions based on Deng *et al.* scaling assumptions [3] but with proportional time adjustments (further detailed); (iii) regression extrapolation for 75MHz and (iv) aggressive settings, by applying successive 50% rank frequency reduction (50MHz, which accounts for 25% of the baseline frequency) combined with 30% VFS on top of (b) configuration. To summarize former observations, configuration (a) (baseline) presents 200MHz (period of 5ns, 200MT/s), (b) 100MHz (period 10ns, 100MT/s), (c) 75MHz and (period 13.25ns, 75MT/s) and (d) 50MHz (period 20ns, 50MT/s).

Restating with a focus on specific parameters, timing parameters typically follow DDR settings with the appropriate timing reduction/increase proportional to the frequency decrease when VFS is applied. Furthermore, the protocol guidelines are followed: *tburst* is assumed as a 4-clock-period BL4 single data ram (SDL) device, *twtr* greater than 2 clocks, *trtw* of 2 clocks as well as *tcs* and *trrd* as 2 clocks. Moreover, when selecting rank voltages, the first pair (i) or baseline voltages (at 200MHz) are assumed as $VDD = 1.8V$ and $VDD2 = 1.2V$, whilst for (ii) (100MHz) $VDD = 1.26V$ and $VDD2 = 0.84V$, for (iii) (75MHz) $VDD = 1.07V$ and $VDD2 = 0.72V$, as well as for (iv) (50MHz), $VDD = 0.882V$ and $VDD2 = 0.588V$. Some low-magnitude timings were adopted to reflect low-magnitude delay settings due to TSVs low-capacitance. To summarize, configurations (i, ii, iii and iv) settings are all illustrated in Table 1a.

To highlight, configuration (i) is selected as the baseline for all measurements: ranks set at 200MHz (200MT/s) and with 1.8V and 1.2V as input voltages. Importantly, we set our baseline with 2 MCs. That said, this amount is the minimum memory degree level of parallelism at rank level which is assumed on a system that runs memory-bound applications in a scientific/financial/commercial scenario. We further highlight that:

- baseline has the lowest magnitude in terms of number of MCs so that higher $MCcounts$ can be observed and their performance can be compared relatively to it. Our goal is to evaluate the effect of the increase of the number of MCs (or $MCcounts$) and VFS settings on each configuration.
- Since our focus is on the memory system, the energy measured only includes memory parts related to it. To the best of our knowledge, employing significant lower frequencies (100MHz, 75MHz and 50MHz) than the Wide I/O standard baseline (200MHz) allows us to comparatively observe memory power and energy to the latter. We compare the experimented energy effects of the scalability combined to VFS against the formulation modeling previously developed (Section III).

TABLE 1. Top: (a) Parameters of the modeled architecture; bottom: (b) benchmarks configuration.

Core, 3.0 GHz, OOO-Core, 4-wide issue,22nm,turnament branch predictor
L1 cache: 32kB dcache + 32 kB icache; associativity = 2 MSHR = 16, latency = 2 cycles
L2 cache: 32MB; associativity = 8, MSHR = 20/core; latency = 4 cycles
crossbar: latency = 1 cycle [1], freq=2x
MC: 1 to 32 MCs; 1 MC/core, 3.0GHz, on-chip buffer size = 64/MC, open page mode
Memory, Wide I/O, 1 rank/MC rank: 528 bits, 1GB, 4 banks, 16384 rows, 1024 columns

Configuration (a)
data rate: 200MHz(200MT/s),trcd=tcl=trp=18ns,tras=42ns,twr=15ns, trtp=20ns
tburst=20ns,trfc=210ns,tREFI=3.9us,trtw=tcs=trrd=10ns,taxw=50ns,actlim=2
IDD0=8mA,IDD02=60mA,IDD2N=0.8mA,IDD2N2=26mA
IDD3N=2mA,IDD3N2=34mA,IDD4W=2mA,IDD4W2=190mA
IDD4R=2mA,IDD4R2=230mA,IDD5=28mA,IDD52=150mA,IDD2P1=0.8mA
IDD3P1=1.4mA,IDD3P12=11mA,IDD2P1=0.8mA,IDD2P12=1.8mA
IDD6=0.5mA,IDD62=1.8mA,VDD=1.8V,VDD2=1.2V

Configuration (b)
data rate: 100MHz(100MT/s),trcd=tcl=trp=54ns,tras=128ns,twr=45ns,trtp=53ns
tburst=53ns,trfc=210ns,tREFI=3.9us,trtw=tcs=trrd=20ns,taxw=50ns,actlim=2
IDD0=5.6mA,IDD02=42mA,IDD2N=0.56mA,IDD2N2=18.2mA
IDD3N=1.4mA,IDD3N2=23.8mA,IDD4W=1.4mA,IDD4W2=133mA
IDD4R=1.4mA,IDD4R2=161mA,IDD5=19.6mA,IDD52=10.5mA
IDD3P1=0.98mA,IDD3P12=7.7mA,IDD2P1=0.56mA,IDD2P12=1.26mA
IDD6=0.35mA,IDD62=1.26mA,VDD=1.26V,VDD2=0.84V

Configuration (c)
data rate: 75MHz(75MHz),trcd=tcl=trp=36ns,tras=84ns,twr=30ns,trtp=40ns
tburst=40ns,trfc=210ns,tREFI=3.9us,trtw=tcs=trrd=26.5ns,taxw=50ns,actlim=2
IDD0=4.75mA,IDD02=35mA,IDD2N=0.475mA,IDD2N2=15.5mA
IDD3N=1.2mA,IDD3N2=20.05mA,IDD4W=1.2mA,IDD4W2=112mA
IDD4R=1.2mA,IDD4R2=136mA,IDD5=17.2mA,IDD52=9.1mA
IDD3P1=0.85mA,IDD3P12=6.55mA,IDD2P1=0.475mA,IDD2P12=1.07mA
IDD6=0.30mA,IDD62=1.07mA,VDD=1.07V,VDD2=0.72V

Configuration (d)
data rate: 50MT/s(50MHz),trcd=tcl=trp=72ns,tras=168ns,twr=60ns, trtp=40ns
tburst=80ns,trfc=210ns,tREFI=3.9us,trtw=tcs=trrd=40ns,taxw=50ns,actlim=2
IDD0=3.92mA,IDD02=29.4mA,IDD2N=0.392mA,IDD2N2=12.64mA
IDD3N=0.98mA,IDD3N2=16.7mA,IDD4W=0.98mA,IDD4W2=91.1mA
IDD4R=0.98mA,IDD4R2=112.7mA,IDD5=13.72mA,IDD52=7.35mA
IDD3P1=0.686mA,IDD3P12=5.39mA,IDD2P1=0.392mA,IDD2P12=0.882mA
IDD6=0.245mA,IDD62=0.882mA,VDD=0.882V,VDD2=0.588V

Benchmark	Input Size	read : write	MPKI
CG:Conjugate Grad(NPB)	ClassA,3iter	76:1	16.9
FT: Fourier Transform(NPB)	ClassW,3iter	1.3:1	6.8
MG:Multigrid(NPB)	ClassA,3iter	76:1	16.9
SP:Scalar Pentadiag(NPB)	ClassA,2iter	1.9:1	11.1
Hotspot(Rodinia)	6000x6000,3iter	2.5:1	12.5
Add,Copy Scale Triad(STREAM)	32Mdoubles;6iter 80Mdoubles;6iter 64Mdoubles;6iter	2.54:1	54.3
pChase	64MB/thread,3iter,random	158:1	116.7

In all experiments, to avoid likely locality benefits, generated memory addresses are homogeneously distributed via cache-address interleaving along the available MCs. Furthermore, open page mode optimized for sequential access has been adopted aiming to focus on the benefits of MC parallelism rather than bank (intra-rank) parallelism by having a larger availability of *MCs*. To model memory contention, each MC has queues (FIFO) to store read/write memory requests, as well as duration and occupation of the banks.

Being a 3D-stacking technique, Wide I/O already has a wider rank size (256 bits) than a typical DDR interface (64 bits). Very importantly, in this evaluation we further assume future Wide I/O development and adopt a 512-bit interface (larger than the current 256-bit Wide I/O standard width) and the number of MCs (*MCcounts*) can be varied up to 32 MCs (larger than the current 16-MC Wide I/O standard range). Given that we arbitrarily selected a 32-core processor, core:MC proportion is varied from the baseline configuration 32:2 up to 32:32 (32 cores, 32 MCs) via having different gem5 simulations with different numbers of MCs represented by different *MCcounts* in order to understand bandwidth effects as the the number of MCs increases.

We have adopted high-bandwidth low-latency crossbar settings from the report by Marino and Li [1] in order to prevent larger interconnection delays to masking memory settings. Moreover, Cacti [17] has been used to obtain cache latencies to set gem5 cache subsystem. A synopsis of the formerly setting parameters and discussion used in this environment is in Table 1a.

In order to have the memory system with different number of MCs responding to different request rates, program behaviors and memory amount, benchmarks have been selected according to Loh's [19] report focusing on memory-bound benchmarks with a high number of misses per kiloinstructions (MPKI). Benchmarks selection includes (i) STREAM [20] suite to evaluate bandwidth, represented by its four sub-benchmarks (Copy, Add, Scale, and Triad); (ii) pChase [21] designed to evaluate bandwidth and latency, set up with pointer-chase sequences randomly accessed; (iii) Hotspot from Rodinia [22]; (iv) Conjugate Gradient (CG), Scalar Pentadiagonal (SP) and Fourier Transform (FT), from NPB High Performance challenge [23]. Table 1b includes a synopsis of the benchmarks, input sizes, read-to-write rate, and L2 MPKI obtained. In addition, all benchmark parallel regions of interest are executed until completion. Moreover, aiming to have behavior variation, selected benchmark input sizes (120MB to 1.8GB) are large enough to perform the design space exploration of the different memory configurations whilst results are calculated using harmonic average. We now turn to the bandwidth experimentation.

B. BANDWIDTH

Bandwidth experiments varying the number of MCs and VFS are presented in Figure 2a. For all benchmarks the increase on the number of MCs does improve bandwidth by up to 7.92x. Results also show that despite Wide I/O individual rank width (512 bits) larger than typical dual inline memory module (DIMM, 64 bits, DDR), bandwidth improves significantly by increasing the number of MCs. This can also be confirmed by the average number of simultaneous memory transactions shown in Figure 4c (and further analysed in subsection IV-F) being lower for 32/16 MCs when compared to 2/4/8 MCs. Highest magnitudes were obtained for stream-pattern ones (Add, Copy, Scale and Triad), followed by FT (NPB, scientific Fourier Transform), pChase (notable result considering it implement random memory accesses), other NPB ones (MG and SP), next Hotspot and, to finalize, CG (NPB suite).

Additionally, both for any benchmark and any rank clock frequency, bandwidths also follow the same behavior, i.e., they increase with the number of MCs. Moreover, in Figure 2a lower rank clock frequencies produced lower bandwidths which follow the previously modeled behaviors developed in Section III, and further discussed on next Subsection. Still in Figure 2a, comparing to the baseline (200MHz), around 50% frequency (100MHz) reduction lowers bandwidth proportionally for Stream (Add, Copy, Scale, Triad), pChase and Hotspot but interestingly, less for the NPB benchmarks, which would potentially allow the latter ones to be executed at lower frequency settings (obviously depending on their energy-per-bit consumption usage, to be discussed in Subsection IV-C). Importantly, with a 75% frequency reduction (50MHz), MC scalability allowed (Add) up to 240% more bandwidth compared to the baseline, which is a very important finding despite the aggressive rank frequency reduction settings. Similarly and generally, for the other frequencies and benchmarks, bandwidth drop due to lower frequencies is compensated by increasing the number of MCs.

1) BANDWIDTH AND VALIDATION OF BANDWIDTH MODELING

Without loss of generality, by arbitrarily selecting Add and MG benchmarks, we employ the bandwidth magnitudes obtained in Figure 2a as inputs to a power regression[1] in order to determine the experimental bandwidth behavior as a function of the number of MCs and rank clock frequency. Arbitrarily, by selecting $freq = 100MHz$ and the Add benchmark, the formerly mentioned regression produced the following approximation:

$$Bwm(MCr) = 1.095494671 \,.\, \frac{MCcount}{MCbaseline}^{0.7936559687} \qquad (33)$$

where *Bwm* is the measured bandwidth. This regression presented the lowest residual sum of squares and/or the maximum approximation error for *Bwm* at around 5% for Add and 8.6% for MG. Before continuing, we compare equations 37/38 to the modeled equation 8 developed in Section III and re-listed below:

$$Bwr = k \,.\, (\frac{MCcount}{MCbaseline})^{1-degree} \,.\, \frac{rankbw(freq)}{rankbw(spec)} \qquad (34)$$

[1] to highlight, power regression refers to a type of Mathematical regression and not the to the dissipation power.

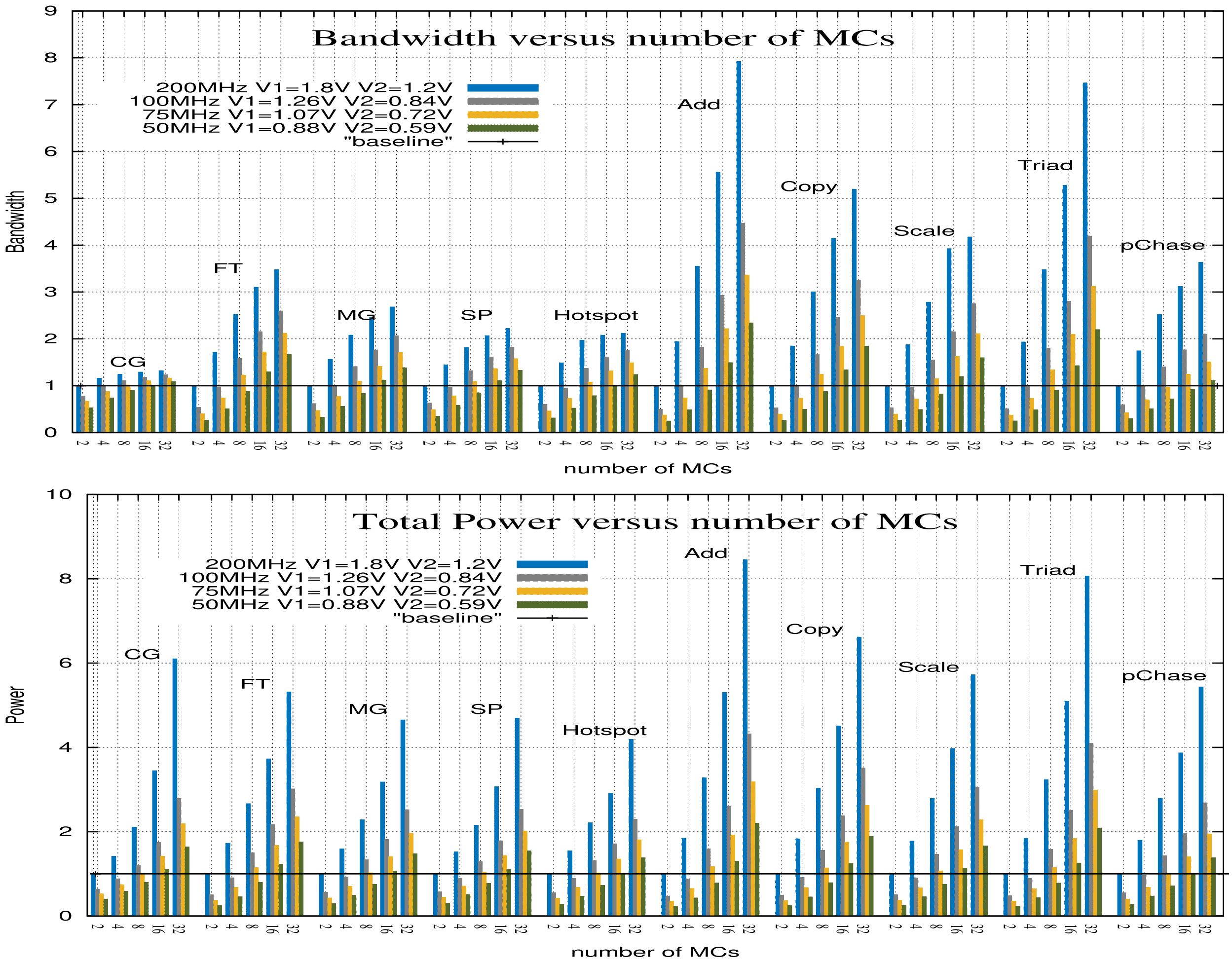

FIGURE 2. **a. Top: bandwidth versus number of MCs; b. bottom: power versus number of MCs.**

To quickly check the formerly equations, we assume that the formerly mentioned factors deteriorate bandwidth (crossbar contention, cache misses, etc.) by 10%-15% (*degree* variable range). Further, we assume *spec* frequency is fixed to 100MHz, $MCcount = 8$ and baseline is ($MCbaseline = 2$,$spec = 100MHz$). Using Figure 2 for Add, we roughly obtain $rankbw(100MHz)/rankbw(100MHz) = 1$, thus for $degree = 15\%$, we have $Bwr = k\,.(8/2)^{1-0.15}\,.1 = k\,.3.249$, and for $degree = 10\%$, $Bwr = k\ .\ (8/2)^{1-0.9}\ .\ 1 = k\ .\ 3.49$. Comparing the previous modeled value with the experimental formula 33 and same settings, we obtain: $Bwm(4) = 1.095494671\ .\ (8/2)^{0.7936559687} = 3.2918$. Therefore, for $k = 1$ or $k = 1.1$, modeled and experimental are close within a margin of around 10% error.

Returning to the comparison against the obtained experimental bandwidth result, by arbitrarily selecting 8 MCs ($MCcount = MCbaseline = 8MCs$) and $freq = 100MHz$ and $spec = 200MHz$ and using Figure 2 for MG, we roughly obtain $rankbw(100MHz)/rankbw(200MHz) = 0.679$. Thus, for $degree = 15\%$, $Bwr = k\ .\ (8/8)^{1-0.15}\ .\ 0.679 = k\ .\ 0.679$ and for $degree = 10\%$, $Bwr = k\ .\ (8/8)^{1-0.10}\ .\ 0.679 = k\ .\ 0.679$. Comparing the previous modeled parameter value with the experimental formula 33 and same settings, we obtain: $Bwm(1) = 1.095494671\ .\ (1/1)^{0.7936559687} = 1.095$. Comparing the latter result with the modeled (equation 9) for k between 1.5 and 1.6, modeled and experimental magnitudes are around within 10% error-margin. Application read-write ratio and contention delays due to higher traffic are among the reasons to have such k magnitudes.

Other configurations with Add or any other benchmarks can be similarly compared by selecting different number of MCs ($MCcount$), frequencies ($freq$) as well as appropriate *degree* and k. We conclude that even with 10% to 15% error margin, this modeling is still of remarkable note given its simplistic theoretical modeling assumptions. Furthermore, as equation 8 previously showed and also confirmed in Figure 2a, bandwidth achievements can be observed typically when more than 4 MCs and lower frequencies are set (all comparisons to the baseline version, 2 MCs, 200MHz). We now switch to rank energy and energy-per-bit analyses.

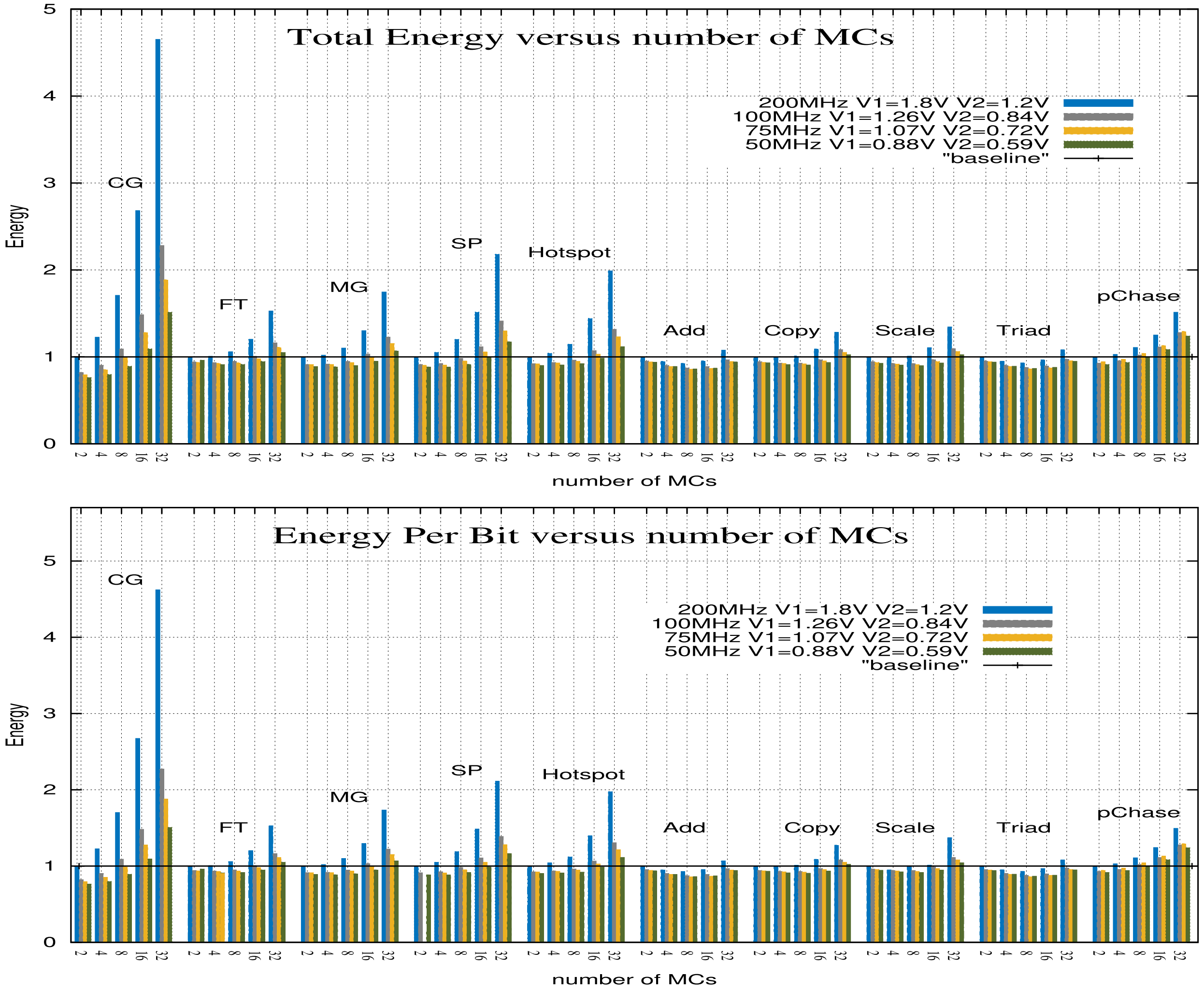

FIGURE 3. a. Top: memory energy versus number of MCs/VFS; b. bottom: memory energy-per-bit versus number of MCs/VFS.

C. TOTAL MEMORY POWER, ENERGY AND ENERGY-PER-BIT

The following analyses aim to identify and compare baseline total energy with total energy at different frequencies, and for different number of MCs.

Total memory power, which includes dynamic and static power spent by all ranks plus MC circuitry, is shown in Figure 2b. By solely focusing on the frequency aspect, our findings show that lower frequencies proportionally lower power magnitudes to some degree. Using similar regression technique we conclude that previously developed formula 13 is within 20%-range error margin when compared to equation 32, and above around 10%-15% margin-range which is still acceptable given the simple memory maker [2] circuitry initial hypothesis (it does not include important degrading factors such as contention, read-write ratio, etc).

In regards to the versions with lower rank frequencies (100MHz, 75MHz and 50MHz) when compared to the baseline (200MHz), they need longer time to perform memory operations (due to the low-magnitude clock frequency versions stated previously stated in Section III). However, given that more memory parallelism through the presence of more MCs allows programs to use more memory bandwidth and, as a consequence lower execution times, effects can be easily derived from the former analyses (from equation 17 on). Furthermore, as shown in Figure 3a, each individual benchmark and its several configurations use similar energy levels, which is an interesting fact since it happens in stream-based memory patterns (Stream suite), scientific benchmarks (NPB), and also in random memory address accesses (pChase). Moreover, larger memory energy utilization happens for CG, SP and Hotspot, whilst the lowest ones appear in Add and Triad.

Changing our focus to energy-per-bit magnitudes, our experiments in Figure 3b show that, for the same rank clock frequency, energy-per-bit levels increase as we increase the number of MCs except for the Stream-suite ones, which roughly present similar energy-per-bit usage for up to 16 MCs. This energy-per-bit behaviour increases and follows our prediction model beforehand in Section III which can similarly be justified by the larger levels of memory bandwidth achieved, as well as also formerly highlighted in Figure 2a and in Section IV-B1 analysis.

Importantly, we can observe in Figure 3b that for different benchmarks several versions clocked at lower frequencies but with a larger number of MCs present lower/equivalent energy-per-bit magnitude levels than/to versions clocked at higher frequencies but with a lower number of MCs. This is a very interesting finding as having more MCs enlarges bandwidth and lowers energy utilization, likely to compensate lower rank frequency usage. For instance, for the Add benchmark, with 16 MCs at 100MHz has around 3% lower memory-energy-per-bit than 2/4 MCs version at 200MHz however the former has 270-550% more bandwidth, which clearly shows how more MC availability is beneficial. For Add, for a 75% frequency reduction, we observe that energy-per-bit can be reduced by up to 20% (for up to 8-to-16 MCs) compared to the baseline, which confirms that MC scalability can keep low energy-per-bit magnitudes. For MG, the configuration with 8 MCs and 100MHz uses less energy-per-bit than the one with 2 MCs at 200MHz, however the former offers around 50% more bandwidth. Many other cases that follow the same behavior can be found (Figure 3b) on the other benchmarks, thus generalizing this important finding.

We turn to the comparison between modeled and experimented energy-per-bit. Using the obtained energy measurements in Figure 3a and the regression tool [15] set with polynomial regression, we obtain the formulation 35 and 36. By arbitrarily selecting Add benchmark without loss of generality, the following equation is obtained:

$$\begin{aligned}&Enpbrm(MC, f = 100MHz)\\&\quad = 4.8496868810^{-6} \,.\, MC^4\\&\qquad - 3.121810826 \,.\, 10^{-4} . MC^3 + 6.750203993 \,.10^{-3} . MC^2\\&\qquad - 5.769511607 \,.\, 10^{-2} \,.\, MC + 1.04413427\end{aligned} \tag{35}$$

where *Enpbrm* is the measured energy-per-bit. Similarly, for an arbitrary selection of MG, we obtain:

$$\begin{aligned}&Enpbm(MC, f = 75MHz)\\&\quad = 1.081141493 \,.\, 10^{-6} \,.\, MC^4\\&\qquad - 7.077427455 . 10^{-5} . MC^3 + 1.605063368 \,.\, 10^{-3} . MC^2\\&\qquad - 7.105937504 \,.\, 10^{-3} \,.\, MC + 0.9194575175\end{aligned} \tag{36}$$

For the selected cases (Add, MG) polynomial regression resulted in a perfect match, i.e. null residual sum of squares. Furthermore, as formerly discussed, equations 35 and 36 follow a polynomial behavior with expected minimum energy-per-bit levels happening for a 4-16MC range. In regards to VFS, lower frequencies/voltage configurations lowered energy/energy-per-bit levels proportionally.

We now switch to the modeled energy. Figure 4a illustrates the obtained experimental results which we do confirm to contain the previously described behaviors discussed when modeling total energy (equation 22): (1) Energy levels achieve an absolute minimum level for 8 MCs, which we believe is the best configuration to balance the number of memory requests generated by the 32-core and memory configuration with the parameters set at table 1b. Importantly, this follows and validates the absolute minimum polynomial behavior previously described. (2) Furthermore, energy behavior matches formerly stated modeled predictions: polynomial-shape with the absolute minimum residual sum of square error. Interestingly, previously mentioned power regression formulas 35 and 36 present a reasonable low residual sum of square errors (around 5% for Add and 8.6% for MG), and likely validate the expected power behavior given the hypotheses and straightforward modeling.

We now turn to the validation of energy-per-bit modeling. Using similar polynomial power regression technique [15] for Add, the following equation is obtained:

$$\begin{aligned}&Enpbitmeasured(MC, f = 100MHz)\\&\quad = 1.276362971 * 10^{-4} . MC^4 - 7.893950521 . 10^{-3} . MC^3\\&\qquad + 0.156852934 \,.\, MC^2 - 1.230403345 \,.\, MC\\&\qquad + 3.804224378\end{aligned} \tag{37}$$

Similarly, for MG:

$$\begin{aligned}&Enpbitmeasured(MC, f = 75MHz)\\&\quad = 9.823015253 \,.\, 10^{-5}\\&\qquad .\, MC^4 - 6.06293285 \,.\, 10^{-}3 . MC^3 + 1.201593385 . 10^{-1}\\&\qquad .\, MC^2 - 9.400415298 \,.\, 10^{-1} \,.\, MC + 3.383097486\end{aligned} \tag{38}$$

For either equations 37 and 38, polynomial regression matches the polynomial-shape behavior for the energy-per-bit parameter, which includes the previously described absolute minimum (polynomial behavior). Furthermore, this regression presents a null residual sum of square errors matching the previously developed polynomial behavior which likely validates our modeling. We now turn to the instruction per cycle analysis (IPC).

D. PERFORMANCE: INSTRUCTIONS PER CYCLE (IPC)

Figure 4a illustrates the IPC results and shows that performance (IPC) magnitudes improve as the number of MCs is increased. The maximum IPC achieved (9.56 compared to the baseline) was obtained for Add with 32MCs at 200MHz whilst other benchmarks such as Copy, Scale, Triad and pChase also present notable improvements (respectively 7.6, 6.7, 3.9 and 3.8). Moreover, IPC magnitudes proportionally reduce as rank frequencies are reduced (increase of execution times). However, the increase of the number of MCs has similar effects to bandwidth, i.e., it compensates IPC drop due to VFS.

E. TEMPERATURE

In this subsection temperature experiments are performed in *Walter* via scaling ranks/MCs under different

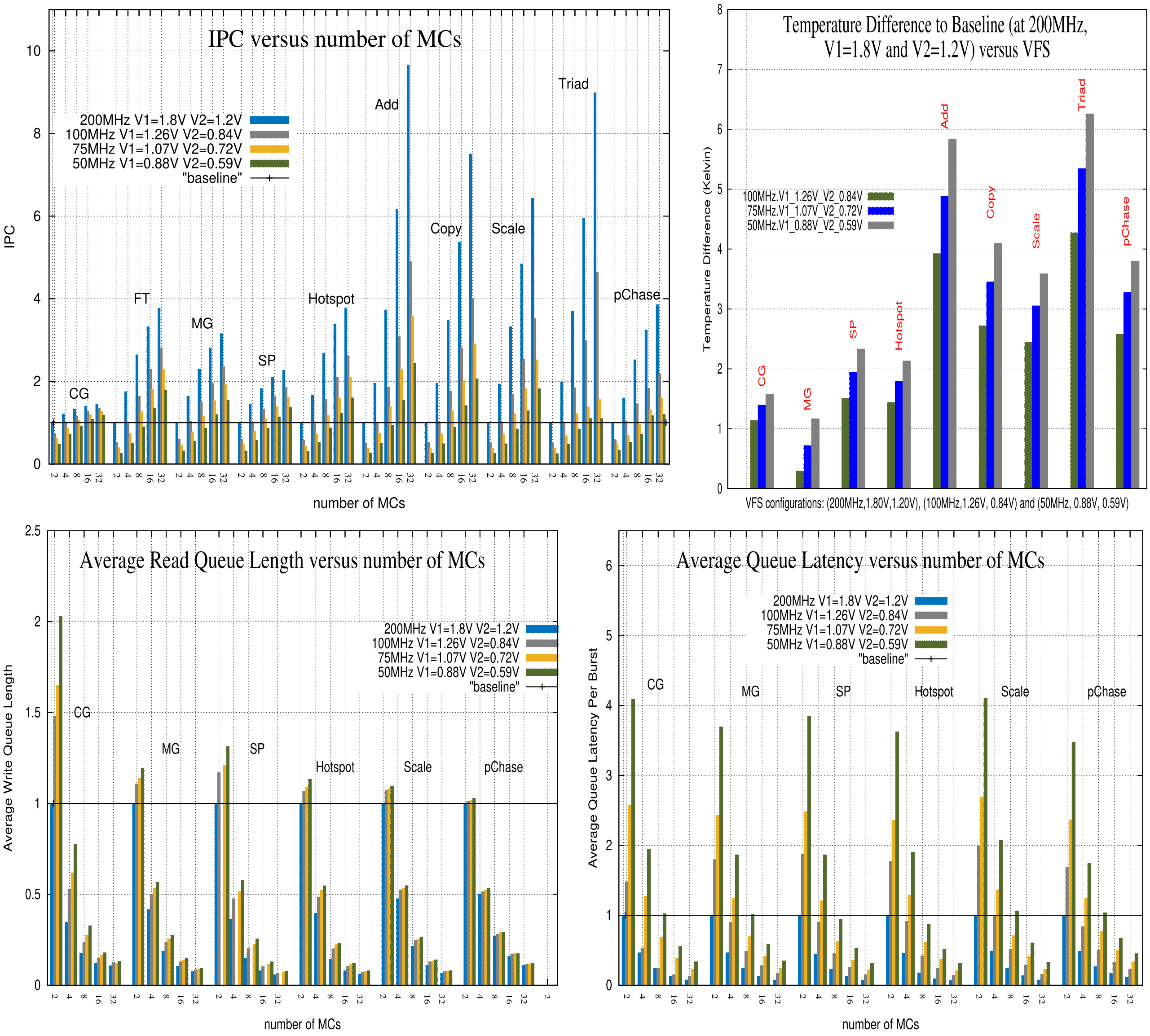

FIGURE 4. Top left a: IPC versus number of MCs/VFS; top right b: temperature versus VFS; bottom left c: average read transaction queue length; bottom right d: average latency per burst.

VFS conditions. Each Wide I/O rank is placed on a different stack and the number of ranks is scaled to 32 in order to match the maximum number of MCs. In addition, the following settings are assumed: (i) 256 $u\text{m}^2$ for rank area based on 3DStacking rank dimensions [19] once 3DStacking is an on-package/on-die technology; (ii) initial rank temperatures set to the same temperature magnitude of the L2 caches (assumed as 60 degree Celsius). Restating, rank power magnitudes are extracted from gem5 memory simulator statistic outputs whilst processor power magnitudes are obtained from McPaT statistic outputs.

Using the previously mentioned settings in 3d-ICE simulator [9], for each benchmark we perform different experiments with different rank and processor power measurements aiming to obtain different temperature distributions, restricted to 200MHz (Wide I/O standard). The results of these experiments are shown in Figure 4b and demonstrate that by reducing rank frequency by 75% (200MHz up to 50MHz) there is a temperature reduction of about 6.25 Kelvin, which is an important result. Further, this result indicates that even including 32 extra ranks (total of 64), *spec* temperature could not be matched, which clearly shows that power reduction could allow to fit 64 ranks which, if combined to the presence of counterpart MC, are likely to further improve bandwidth and performance.

F. LATENCY AND TRANSACTION QUEUE SIZE

In order to understand *Walter* trade-offs (number of MCs and VFS) in a complementary way, for some benchmarks we also illustrate the average MC read transaction queue size (Figure 4c) and the average transaction latency (Figure 4d). For these parameters our findings are similar to the ones obtained in previous report [1]: more MCs lowers the number of memory transactions per queue, which increases memory parallelism and lowers the number of simultaneous transactions. Furthermore, for a 75% rank clock frequency reduction, an increase in the transaction queue size of up to 100% is present, which is due to the use of lower rank clock frequencies (higher magnitude timing settings). Likewise, given the lower rank frequencies, more available MCs lowers the average transaction latency and proportionally increases latencies (4x roughly for 50MHz, i.e., 25% of the 200MHz-*spec* Wide I/O frequency).

G. SENSITIVE ANALYSIS

We analyse the gains from MC/rank scalability first, and next we approach VFS. The bandwidth obtained for different numbers of MCs (MCcounts) for different benchmarks shows that this technique is valid for a broad range of memory-bound benchmarks with very different memory behaviours. Moreover, compared to Marino and Li's report [1] which has employed traditional 64-bit DDR ranks, our bandwidth findings demonstrate that scalability of MCs/ranks is also beneficial for much wider ranks (512 bits, 8x larger). Following that, on the architectural experiments when ranks at *spec* temperature were replaced by ranks at lower *gfreq* frequency (after VFS), having demonstrated that a larger number of ranks can be fit without significant temperature increase, future transistor technology improvements are likely to benefit from lower voltages, thus allowing further power and temperature reduction in *Walter*.

In Ramon's [1] report, scalability of MC/ranks is demonstrated for a 16MB-private-L2 cache configuration versus a 32MB-shared-L2 in this report, therefore likely generalizing its benefits to different cache types and sizes. Furthermore, the previous report also demonstrates that scalability of MCs can be applied to different (than 32) number of cores. Moreover, Wide I/O MC scalability does provide bandwidth growth for different memory settings and number of MCs/ranks, with which also confirms previous report conclusions [1].

In regards to the generality of the approach, given the benefits of MC scalability were also verified for other types of DDR memories [1] and VFS [3], *Walter* results further restates the benefits in either. Interestingly, as previously discussed, some benchmarks (e.g. FT) indicate that lower than obtained energy-per-bit levels could be achieved via having more than 32 MCs, which is planned for future work. In regards to VFS, whilst lower rank frequencies and voltages lower rank power, memory energy-per-bit magnitudes are kept at similar levels until around 8 MCs, and after that they typically increase to up to around 10%-15% more than baseline energy (2 MCs, 200MHz). In terms of bandwidth, lower rank frequencies proportionally lowered bandwidths, however when combined to larger number of MCs/ranks available, total bandwidth is improved, which was demonstrated for several benchmarks and configurations (different number of MCs and rank frequencies).

V. RELATED WORK

Olukotun *et al.* [7] introduced the idea of single-chip multiprocessor or multicores and demonstrated that having a higher number of smaller/simpler in-order cores set at smaller frequencies can achieve better bandwidths than a beefier (larger) out-of-order (OoO) core set at a higher frequency. This work has employed a larger number of MCs/ranks set at lower frequencies aiming to achieve larger bandwidths (rather than a low number of ranks set at higher frequency) comparatively to the previous study [7], which employs smaller (simpler) cores set at lower frequencies to achieve larger bandwidths.

In Memscale [3] co-design for typical DDR servers, memory bandwidth dynamically (DVFS) changes according to the application bandwidth utilization whilst the number of MCs is kept at lower magnitudes. In *Walter*, we perform a static VFS approach in 3DStacking systems with a significant larger number of MCs and wider ranks than used in Memscale to determine bandwidth, energy/energy-per-bit and temperature trade-offs.

Multiscale [24] operating system application further advances DVFS Memscale in the direction of multiple MCs, whilst using an algorithm which selects a frequency that reduces total system energy given user-specific application performance constraints. Different from Multiscale, *Walter* is planned to be employed on the MC side, by employing MC scalability to improve bandwidth drop due to static VFS whilst reducing energy-per-bit levels.

CoScale [25] technique relies on execution profiling of each processor core via performance counter monitoring, focusing on memory performance and power consumption. It employs a set of possible selectable frequency settings to minimize total energy consumption within performance constraints, whilst saving a significant amount of energy. CoScale could be coupled to *Walter* in order to dynamically change the number of MCs aiming to save energy-per-bit whilst improving bandwidth.

Snatch [26] focuses on reducing the number of power and ground pins, which represent the majority in a 3DStacking system. Via diverting between processor and memory power delivery network as well as using a bidirectional on-chip voltage regulator, Snatch avoids throttling performance when higher processor and memory power requirements are demanded. Orthogonally, *Walter*'s VFS in combination with a larger number of MCs could be used to trigger Snatch mechanism to further reduce memory power utilization.

Syscale [27] is a domain power management technique targeting energy efficiency of mobile system on chips (SoCs). Its algorithm predicts bandwidth and latency performance according to the application bandwidth demands and different DVFS domains (e.g. processor, memory, etc) whilst trying to minimize DVFS latency overheads. Instead, *Walter* is designed for the typical memory domain and uses larger number of MCs to improve performance via static VFS.

HMC [28] memory technology employs 128-256bits ranks placed on memory dies, whilst serial/deserial 10-Gbit/s-I/O-links are used to communicate processor with the ranks. *Walter* employs 512-bit ranks and uses Wide I/O memory technology (which follows DDR family). Alternatively, the report by Jian *et al.* [29] investigates the power bottlenecks of memory network systems (e.g. HMC) and shows that I/O links are the most power consuming ones. Furthermore, by applying rapid on-off and DVFS, Jian *et al.* reports a significant reduction of I/O power. Whilst the former is focused on I/O links, *Walter* is focused on scalability of MCs to further reduce energy-per-bit utilization. Furthermore, despite a different memory domain (HMC) than the one used in *Walter* (Wide I/O), I/O power techniques investigated by Jian *et al.* could be complementary combined to *Walter*'s VFS to further reduce power consumption.

The report by Xie *et al.* [30] dynamically partitions its memory banks according to thread utilization profiling. Jantz *et al.* [31] software scheduling allows OS-to-applications interaction to determine their dynamic memory footprint utilization. Xie's and Janz' techniques can be orthogonally applied to *Walter*, which does not employ any memory scheduling technique.

Whilst Ausavarungnirun *et al.* [32] report employs a MC management technique that groups memory requests according to row-buffer locality first, where inter-application and FIFO scheduling can be applied, Kayiran *et al.* [33] manage to alleviate graphics processing units (GPU) contention for shared resources. Either formerly mentioned techniques could be orthogonally applied to *Walter*.

In *Ramon*, Marino and Li [1] demonstrate the scalability of MCs/ranks for CPUs and GPUs via the creation of reconfigurable regions with CPUs, GPUs or a combination of both to allow different bandwidth allocations. In a multiple-application scenario, *Ramon* regions could be created and used with *Walter* VFS mechanisms to allow a combination of a different number of MCs with different rank settings adapting to different application bandwidth and energy requirements.

VI. CONCLUSION AND FUTURE WORKS

This investigation advances the state of art in Wide I/O systems by evaluating the benefits of MC scaling combined to VFS scaling. Contrary to the likely statement that lower clock frequencies increase energy-per-bit levels, if configurations are set with rank frequencies but combined with a higher number of MCs, our findings show that there are remarkably many of them presenting higher bandwidth magnitudes and lower energy-per-bit levels when compared to other ones set with a lower number of MCs and higher frequencies. Furthermore, due to the employment of pairs of MCs and ranks with VFS, lower temperatures than the standard Wide I/O clock frequency are achieved, thus allowing further 3D-stacking scaling.

We have demonstrated that using a simplistic energy modeling based on memory maker circuitry [2], it is possible to achieve a bandwidth and energy/energy-per-bit parameters estimation within a 10%-to-20% margin error range (or even lower if degrading factors are included) when compared to the validated modeling [10] incorporated in the used detailed-accurate simulator [8].

As future endeavours, *Walter*'s VFS static design space exploration paves the way for a full dynamic (DVFS) approach. Moreover, a general power-saving strategy should not only consider Wide I/O but also other 3D-stacking systems (e.g. HMC [28]) as well as memory traffic patterns appearing in Big Data applications.

ACKNOWLEDGMENT

The author would like to thank Maria A. G. Marino and anonymous reviewers for their precious feedbacks.

REFERENCES

[1] M. D. Marino and K.-C. Li, "RAMON: Region-aware memory controller," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 4, pp. 697–710, Apr. 2018.

[2] *Micron Manufactures DRAM Components and Modules and NAND Flash*. Accessed: Jan. 6, 2020. [Online]. Available: http://www.micron.com/.

[3] Q. Deng, "Memscale: Active low-power modes for main memory," in *Proc. 16th ASPLOS*, New York, NY, USA, 2011, pp. 225–238.

[4] S. Ghose, T. Li, N. Hajinazar, D. Senol Cali, and O. Mutlu, "Understanding the interactions of workloads and DRAM types: A comprehensive experimental study," 2019, *arXiv:1902.07609*. [Online]. Available: http://arxiv.org/abs/1902.07609

[5] K. Chandrasekar, C. Weis, B. Akesson, N. Wehn, and K. Goossens, "System and circuit level power modeling of energy-efficient 3D-stacked wide I/O DRAMs," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2013, pp. 236–241.

[6] *JEDEC Global Standards for the Microelectronics Industry*, Standard JESD229-2, Sep. 2014. Accessed: Jul. 2020. [Online]. Available: https://www.jedec.org/standards-documents/docs/jesd229-2

[7] K. Olukotun, B. A. Nayfeh, L. Hammond, and K. Wilson, "The case for a single-chip multiprocessor," in *Proc. ASPLOS*, 1996, pp. 2–11.

[8] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, May 2011.

[9] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza, "3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2010, pp. 463–470.

[10] K. Chandrasekar, B. Akesson, and K. Goossens, "Improved power modeling of DDR SDRAMs," in *Proc. 14th Euromicro Conf. Digit. Syst. Design*, Aug. 2011, pp. 99–108.

[11] N. H. Khan, S. M. Alam, and S. Hassoun, "Power delivery design for 3-D ICs using different through-silicon via (TSV) technologies," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 4, pp. 647–658, Apr. 2011.

[12] *Moore's Law, 40 Years and Counting*. Accessed: Mar. 4, 2020. [Online]. Available: http://download.intel.com/technology/silicon/Interpack

[13] LPDDR4 Moves Mobile. *Mobile Forum 2013, presented by Daniel Skinner*. Accessed: Feb. 14, 2020. [Online]. Available: http://www.jedec.org/sites/../D_Skinner_Mobile_Forum_May_2013_0.pdf

[14] K. Therdsteerasukdi, "The DIMM tree architecture: A high bandwidth and scalable memory system," in *Proc. IEEE 29th Int. Conf. Comput. Design (ICCD)*, Oct. 2011, pp. 388–395.

[15] *Xuru's Website*. Accessed: May 1, 2020. [Online]. Available: http://www.xuru.org/Index.asp.

[16] *Calculating Memory System Power for DDR3 Introduction*. Accessed: Sep. 5, 2020. [Online]. Available: http://www.micron.com/

[17] *CACTI 5.1*. Accessed: Apr. 21, 2020. [Online]. Available: http://www.hpl.hp.com/techreports/2008/HPL200820.html

[18] S. Li, "McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarchitecture*, New York, NY, USA, 2009, pp. 469–480.

[19] G. H. Loh, "3D-stacked memory architectures for multi-core processors," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2008, pp. 453–464.

[20] J. D. McCalpin, "Memory bandwidth and machine balance in current high performance computers," in *Proc. IEEE TCCA Newslett.*, Dec. 1995, pp. 19–25.

[21] *The pChase Memory Benchmark Page*. Accessed: Apr. 2, 2020. [Online]. Available: http://pchase.org/.

[22] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in *Proc. IEEE Int. Symp. Workload Characterization (IISWC)*, Oct. 2009, pp. 44–54.

[23] *NAS Parallel Benchmarks*. Accessed: Mar. 20, 2020. [Online]. Available: http://www.nas.nasa.gov/Resources/Software/npb.html/

[24] Q. Deng, D. Meisner, and A. Bhattacharjee, "MultiScale: Memory system DVFS with multiple memory controllers," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design*, Jul. 2012, pp. 297–302.

[25] Q. Deng, D. Meisner, A. Bhattacharjee, T. F. Wenisch, and R. Bianchini, "CoScale: Coordinating CPU and memory system DVFS in server systems," in *Proc. 45th Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2012, pp. 143–154.

[26] D. Skarlatos, R. Thomas, A. Agrawal, S. Qin, R. Pilawa-Podgurski, U. R. Karpuzcu, R. Teodorescu, N. S. Kim, and J. Torrellas, "Snatch: Opportunistically reassigning power allocation between processor and memory in 3D stacks," in *Proc. 49th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2016, pp. 1–12.

[27] J. Haj-Yahya, M. Alser, J. Kim, A. G. Yaglikci, N. Vijaykumar, E. Rotem, and O. Mutlu, "SysScale: Exploiting multi-domain dynamic voltage and frequency scaling for energy efficient mobile processors," in *Proc. ACM/IEEE 47th Annu. Int. Symp. Comput. Archit. (ISCA)*, May 2020.

[28] *Hybrid Memory Cube Specification 1.0*. Accessed: Mar. 8, 2020. [Online]. Available: http://www.hybridmemorycube.org/

[29] X. Jian, P. K. Hanumolu, and R. Kumar, "Understanding and optimizing power consumption in memory networks," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2017, pp. 229–240.

[30] M. Xie, D. Tong, K. Huang, and X. Cheng, "Improving system throughput and fairness simultaneously in shared memory CMP systems via dynamic bank partitioning," in *Proc. IEEE 20th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2014, pp. 344–355.

[31] M. R. Jantz, C. Strickland, K. Kumar, M. Dimitrov, and K. A. Doshi, "A framework for application guidance in virtual memory systems," in *Proc. 9th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ. (VEE)*, 2013, pp. 344–355.

[32] R. Ausavarungnirun, K. K.-W. Chang, L. Subramanian, G. H. Loh, and O. Mutlu, "Staged memory scheduling: Achieving high performance and scalability in heterogeneous systems," in *Proc. 39th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2012, pp. 416–427.

[33] O. Kayiran, N. C. Nachiappan, A. Jog, R. Ausavarungnirun, M. T. Kandemir, G. H. Loh, O. Mutlu, and C. R. Das, "Managing GPU concurrency in heterogeneous architectures," in *Proc. 47th Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2014, pp. 114–126.

MARIO DONATO MARINO (Member, IEEE) is currently a Senior Lecturer with Leeds Beckett University. He has worked in several institutions such as the University of Sao Paulo and The University of Texas at Austin. He has coauthored articles in computer architecture, high-performance, and parallel/distributed computing. He is a member of ACM. He has received the International Top Conference Best Paper Award. He has been serving as a PC Member of international conferences/workshops and a Reviewer of renovated journals. He has been serving as an Associate Editor for IEEE Access and IJES (Inderscience) since 2015.