Reducing DRAM Image Data Access Energy Consumption in Video Processing

Yiran Li and Tong Zhang, *Senior Member, IEEE*

***Abstract*—This paper presents domain-specific techniques to reduce DRAM energy consumption for image data access in video processing. In mobile devices, video processing is one of the most energy-hungry tasks, and DRAM image data access energy consumption becomes increasingly dominant in overall video processing system energy consumption. Hence, it is highly desirable to develop domain-specific techniques that can exploit unique image data access characteristics to improve DRAM energy efficiency. Nevertheless, prior efforts on reducing DRAM energy consumption in video processing pale in comparison with that on reducing video processing logic energy consumption. In this work, we first apply three simple yet effective data manipulation techniques that exploit image data spatial/temporal correlation to reduce DRAM image data access energy consumption, then propose a heterogeneous DRAM architecture that can better adapt to unbalanced image access in most video processing to further improve DRAM energy efficiency. DRAM modeling and power estimation have been carried out to evaluate these domain-specific design techniques, and the results show that they can reduce DRAM energy consumption by up to 92%.**

***Index Terms*—DRAM energy reduction, heterogeneous DRAM architecture, image data access, image frame buffer recompression, video coding.**

I. Introduction

As video processing becomes increasingly indispensable in mobile devices, its energy-efficient implementation is of great practical interest. For typical video processing applications, the image frames are stored in a stand-alone DRAM, which are accessed and processed by a processing logic unit in an SoC. Although the continuous technology scaling helps to reduce energy consumption, DRAM image data access may not benefit as much as the video processing logic operation. This is because, as mobile devices need to support increasingly diverse and more sophisticated functions, the storage capacity of DRAM has to accordingly increase. As a result, DRAM die size may not reduce or even increase in spite of technology scaling down. As we will elaborate later, DRAM access energy consumption is largely dominated by the routing interconnect energy consumption that is proportional to DRAM die size. Therefore, it is reasonable to expect that DRAM access energy consumption will play a more and more important role in determining the overall video processing energy consumption. Taking video decoding as an example, recent work has shown that, an H.264 decoder at 90 nm node only consumes 0.36 pJ per pixel, while the corresponding DRAM access energy consumption is 1.11 pJ per pixel [1]. Nevertheless, most prior work on video decoding mainly focused on reducing the decoder logic power consumption (e.g., see [1] and [2]).

This work is interested in reducing DRAM image data access energy consumption in video processing. Although reducing DRAM energy consumption has been widely studied (e.g., see [3]–[7]), most prior work focused on using DRAM in general purpose computing systems. Very intuitively, once we confine ourselves in certain specific application domain, it is possible to exploit its own unique characteristics to develop domain-specific techniques for reducing DRAM energy consumption. They can complement those general-purpose low-power design techniques to further push DRAM energy efficiency envelope for the specific application domain. Following this intuition, we aim to develop techniques for the domain of video processing, which can exploit image data access characteristics to reduce DRAM power consumption. Leveraging block-based image data access nature of most video processing, several image frame storage schemes [8]–[10] have been proposed to reduce the power consumption overhead incurred by DRAM row activation and bitline precharge. In addition, image frame recompression, in either lossless [11] or lossy [12], [13] manner, has been recently studied with the main objectives of reducing DRAM storage capacity requirement and memory I/O link workload. Clearly, such recompression schemes can also reduce DRAM image data access energy consumption. We note that the concept of data compression can also be applied to reduce the memory I/O link power consumption [14], [15]. As demonstrated later, the data access energy consumption of DRAM with reasonably large capacity (e.g., a few hundred Mb and beyond) is dominated by the on-chip routing, in particular for image data access. We note that many techniques have been proposed for reducing the energy consumption for on-chip address bus and data bus [16]. Nevertheless, although the obvious temporal correlation on address bus can be effectively exploited to reduce energy consumption (e.g., zero-transition encoding [17], working zone method [18], and Gray coding [19]), reducing data bus energy consumption remains much more challenging. Mainly targeting at general purpose computing systems, prior work assumes temporally and spatially random data transmission on data bus (e.g., see [20] and [21]). As a result, existing techniques on reducing data bus energy consumption either have limited effectiveness or suffer from high implementation overhead. Geared to video

Manuscript received June 09, 2011; revised September 06, 2011 and November 09, 2011; accepted November 11, 2011. Date of publication November 22, 2011; date of current version March 21, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yen-Kuang Chen.

The authors are with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: yiran.li@gmail.com; tzhang@ecse.rpi.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2011.2177079

processing applications, we are particularly interested in how to explore image data access characteristics to reduce the energy consumed by DRAM on-chip data bus routing, which nevertheless has not been well studied in the open literature.

Aiming to fill this missing link, this paper presents several techniques that can exploit image data access characteristics to reduce the energy consumption induced by DRAM on-chip data routing. In particular, those application-specific characteristics include 1) abundant spatial and temporal image data correlations, and 2) certain image frames (e.g., reference images in video coding) are accessed much more frequently than other image frames, which is referred to as unbalanced image access. First, we propose to use three simple yet effective data manipulation techniques, including *pixel transfer scheduling*, *Gray coding*, and *bit-level interleaving*, to reduce DRAM on-chip routing energy consumption. By exploiting the image data spatial/temporal correlation, the first two techniques aim to reduce the energy consumption induced by routing wire self-transition activities, while the third one aims to reduce the energy consumption induced by routing wire crosstalk coupling. In addition, we discuss the applications of these data manipulation techniques when image frame recompression is used. Secondly, we propose a heterogeneous DRAM architecture design strategy that exploits unbalanced image data access in most video processing to further reduce DRAM image data access energy consumption. The basic idea is simple: We allocate a small hence more energy-efficient memory array to store those hot image data being accessed more frequently, while leave other image data stored in the main memory array. Because of the paramount importance of video processing in mobile devices and the huge market, we argue that such a DRAM architecture customization may be economically justifiable. In addition, besides video processing, many other real-world workloads also exhibit similar unbalanced data access characteristics, which can further enhance the potential benefit of using such heterogeneous DRAM architecture for energy saving.

To evaluate the effectiveness of the proposed data manipulation techniques and the heterogeneous DRAM architecture, we carried out extensive DRAM modeling and power consumption estimations using representative video sequences in the context of video decoding. The popular CACTI tool [22] is used for DRAM modeling and estimating energy consumption induced by DRAM on-chip routing including both wire self-transition and coupling activities. Simulation results show that the three data manipulation techniques together can reduce DRAM access power consumption by up to 40%. When incorporating the video decoding logic power consumption, we estimate that these data manipulation techniques can reduce the overall video decoding system power consumption by up to 30.4%. We also evaluated the effectiveness of the manipulation techniques when image frame recompression is being used, and the results show as high as 80% of power reduction. Regarding the heterogeneous DRAM architecture, we considered different heterogeneity configuration scenarios for a 2 Gb DRAM, and simulation results show that DRAM energy consumption can be reduced by up to 92% when using the proposed data manipulation techniques in a heterogeneous DRAM architecture.

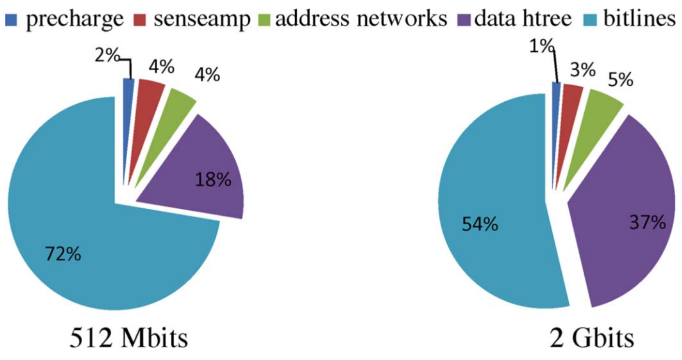


Fig. 1. Estimated breakdown of single-access energy consumption for a 512 Mb and 2 Gb DRAM at 45 nm node using CACTI.

II. DRAM Energy Consumption

DRAM is usually organized with a two-dimensional (2-D) array structure. At the highest level, a data array is composed of multiple identical banks, each of which has its own address and data bus and can be accessed concurrently. Each bank consists of a number of sub-arrays. Each sub-array is a 2-D matrix of memory cells and associated peripheral circuitry, such as row decoders, row drivers, precharge and equalization circuits, sense amplifiers, and output drivers, etc. A row is simply a group of memory cells that are activated in parallel in response to a row activation command. A row is also referred to as a DRAM page, which is cached at the sense amplifiers until a subsequent precharge command is issued. A column of data is the smallest independently addressable unit of memory, and its size is the same as the output data width. In DDRx SDRAM, each column access loads or stores multiple columns of data depending on the burst length. For example, in a DDR3 DRAM device, each memory read command returns 8 columns of data. In the page mode, a row can be kept open while performing multiple reads or writes so that successive reads or writes within the same row do not suffer the delay and energy consumption induced by row activation and precharge. This can significantly increase the system performance when reading or writing bursts of data. On-chip H-tree distribution networks are used to route address and data throughout the DRAM chip.

Using the popular memory modeling tool CACTI [22], we estimate the average single memory access energy consumption and its breakdown among different components for a 512 Mb and 2 Gb DRAM chip at 45 nm node, as shown in Fig. 1. The energy consumption of precharge is the energy consumed by precharge drivers, and the energy consumption of address networks consists of that of address routing, decoders and drivers. Bitline energy consumption is used to charge or discharge all the parasitic capacitance connected to bitlines. The total energy for a single access is 1.07 nJ for 512 Mb DRAM and 1.44 nJ for 2 Gb DRAM, respectively. The results clearly show that the overall DRAM access energy consumption is dominated by bitlines and data H-tree routing. In addition, data H-tree routing energy consumption tends to account for a higher percentage as the memory capacity increases, because higher capacity memory comes with a larger chip die size and hence larger aggregated H-tree length.

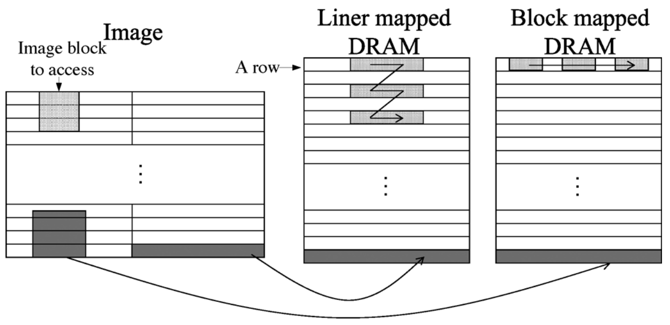


Fig. 2. Illustration of a linear row-by-row translation and block-based mapping from image space to physical memory address space.

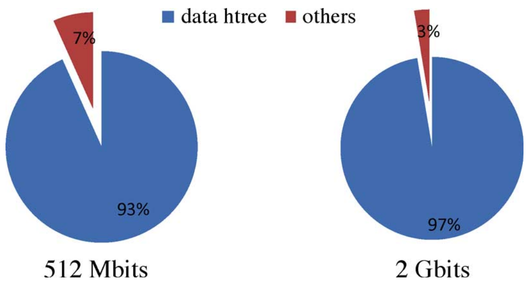


Fig. 3. Estimated breakdown of page-mode energy consumption for a 512 Mb and 2 Gb DRAM at 45 nm node using CACTI.

The above results suggest that it is highly favorable to make as many consecutive reads or writes as possible from the same row before switching to another row in order to reduce the energy consumption. This is usually called page mode in DRAM. Because of the block based nature of image access, if image blocks are linearly row-by-row (or column-by-column) mapped to the physical address space as illustrated in Fig. 2, a significant amount of row activations will occur, which results in high energy consumption. It is very straightforward that, instead of such a linear translated mapping, a block-by-block mapping geared to video processing should be used, i.e., each row stores the pixel data of one or more blocks, and image access mainly incurs page-mode operations in DRAM. As a result, row activations and hence bitline energy consumption will dramatically reduce, and on-chip H-tree data routing energy consumption becomes completely dominant, as shown in Fig. 3.

Therefore, for our interested image access in video processing, it is of paramount importance to reduce the energy consumed by data routing through H-tree in DRAM. The dynamic energy consumption of interconnect wires can be expressed as

$$E = (\alpha_s \cdot (C_s + C_L) + \alpha_c \cdot C_c) \cdot V_{dd}^2 \tag{1}$$

where C_s, C_L, and C_c are wire self-capacitance, load capacitance, and coupling capacitance, respectively; α_s and α_c represent the self-transition and coupling-transition activity factors; and V_{dd} is supply voltage. The self-capacitance consists of both wire to ground capacitance and gate capacitance of repeaters along the wire. The coupling capacitance occurs between adjacent wire segments running in parallel. The coupling transition activity depends on switching activity relation between two adjacent wires, which can be classified into four types as illustrated in Fig. 4. In both type-I and type-III transitions, there is no coupling-induced energy consumption. In type-II and type-IV transitions, the coupling-induced energy consumption is proportional to C_c and $2C_c$, respectively. As DRAM technology continues to scale down, the increase of wire aspect ratio and the decrease of wire pitch result in rapid increase of coupling capacitance, and therefore coupling-induced energy consumption becomes increasingly important.

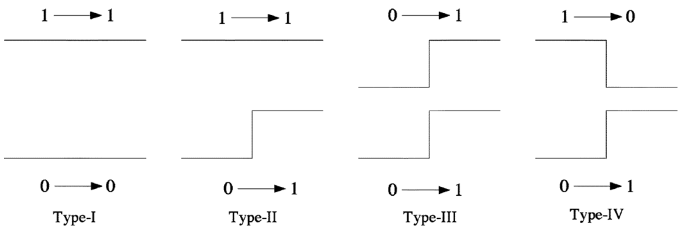


Fig. 4. Signal transition on two adjacent interconnect wires.

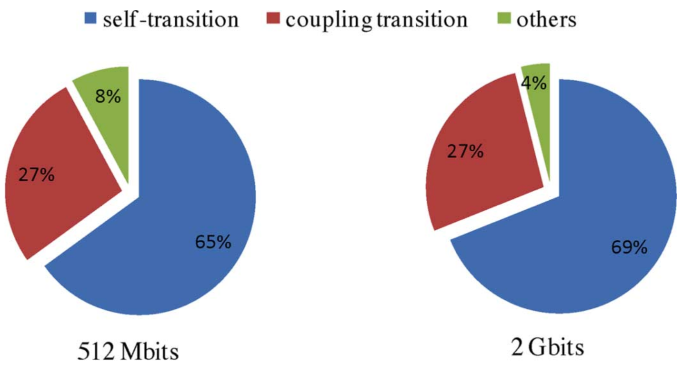


Fig. 5. Estimated energy consumption breakdown with test video sequence "akiyo".

In Figs. 1 and 3, we used the original CACTI model, which assumes both types of transition activities are 50%, to estimate energy breakdown for general cases. However, the self-transition and coupling transition activities are data-dependent, i.e., accessing different images may result in different switch patterns on the data H-tree, thus consumes different amount of energy. Fig. 5 shows how much of total energy self-transition and coupling transition account for with a test video sequence "akiyo". Apparently, the energy consumption due to self-transition and coupling transition activities on data H-tree dominates the total energy consumption.

III. Image-Oriented Data Manipulation to Reduce DRAM Energy Consumption

As pointed out above, on-chip H-tree data routing dominates DRAM energy consumption for image data access. In this section, we present a few simple yet effective data manipulations schemes that aim to exploit the spatial/temporal pixel value correlations in image data to reduce the switching activities on each wire and coupling activities among adjacent wires in on-chip H-tree data routing.

A. Basic Data Manipulation Techniques

In a typical video processing system, image data including both luminance and chrominance components are stored into

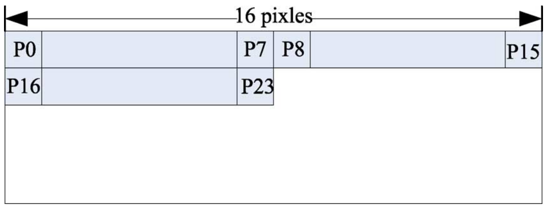


Fig. 6. Illustration of pixel transfer scheduling when transferring a 16×16 image block within a 64-bit-wide DRAM.

and retrieved from DRAM in the unit of blocks. Due to limited internal and external data bus width (e.g., 8-bit chip I/O and 64-bit internal data bus in DDR3 SDRAM [23]), only part of a block is transferred in every clock tick. Thus the temporal correlation of data on data bus is actually the space correlation of pixels in the image. Intuitively, small differences between successive data transferred on the bus more likely incur less transition activities, leading to low dynamic energy consumption. Hence, appropriate scheduling of the pixel transfer may help to reduce the memory access energy consumption.

Following the above intuition, since spatially adjacent pixels tend to have smaller difference, we should try to transfer spatially adjacent pixels successively over the same set of wires. This is referred to as *pixel transfer scheduling*. Without loss of generality, we assume the DRAM has a 64-bit on-chip data bus and the luminance density of each image pixel is represented by 8 bits. Hence, luminance intensities of 8 pixels can be transferred at one time. In conventional design practice, the pixel data is transferred row-by-row within each block. For example, as illustrated in Fig. 6, when fetching a 16×16 image block, pixels P8–P15 are transferred on the data bus right after pixels P0–P7 have been transferred. As a result, the two pixels P0 and P8, which are 7 pixels spatially away from each other, are transferred on the same set of wires over the two successive clock ticks. Following the concept of pixel transfer scheduling, we should put the pixels P16–P23 onto the data bus right after P0–P7 have been transferred, and each pair of vertically adjacent pixels (e.g., P0 and P8) is transferred over the same 8 wires. In this way, we first successively transfer the left half of the 16×16 image block, then transfer its right half in the same order. Clearly, this pixel scheduling is not limited to 64-bit data bus and can be designed for any internal bus width. For example, if the internal data bus is 32 bits, only 4 pixels are transferred at one time, hence we should put P16–P19 on the bus right after P0–P3 have been transferred. In this way, the left-most 1/4 of the block is first accessed and then the next 1/4 block.

Using CIF sample video sequence "akiyo" as a test vehicle, Fig. 7 shows the probability density function of luminance density differences of consecutive pixels transferred on the same set of wires, where the conventional practice and proposed pixel transfer scheduling schemes are used. The results clearly demonstrate that appropriate pixel transfer scheduling can noticeably reduce the consecutive pixel difference.

Nevertheless, we note that small difference between two pixels does not necessarily result in small number of bits changing in binary representation, i.e., the Hamming distance between two binary numbers is not necessarily proportional to the deference between the two values they represent. Hence,

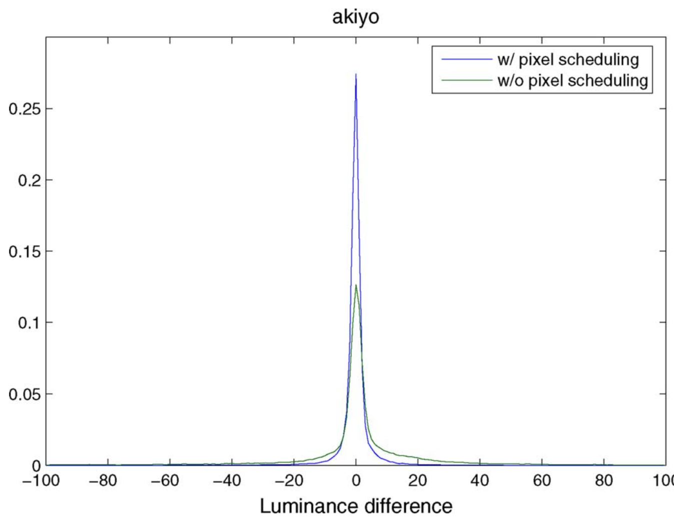


Fig. 7. Probability density function of the luminance density difference of consecutive pixels transferred on the same set of wires.

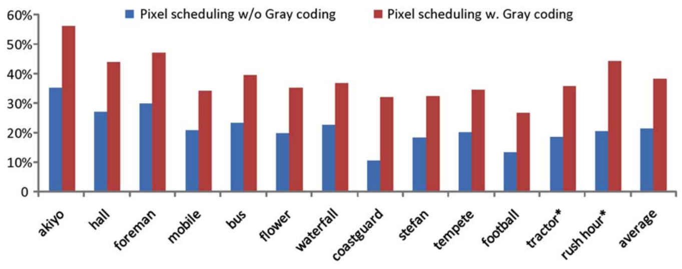


Fig. 8. Self-transition activity reduction when using pixel scheduling with and without Gray coding.

we propose to further use Gray coding to represent the pixel value in order to ensure small number of bit changes when pixel values are close. Let $B_{n-1} = b_{n-1} \ldots b_1 b_0$ be the n-bit binary code, and $G_{n-1} = g_{n-1} \cdots g_1 g_0$ be its associated Gray code, where $b_i, g_i \in \{0, 1\}$. The conversion between binary and gray code can be easily implemented in digital logic, i.e., to convert binary to Gray, we have $g_{n-1} = b_{n-1}$ and $g_i = b_{i+1} \oplus b_i, i = n-2, \ldots, 0$; to convert from Gray to binary, we have $b_{n-1} = g_{n-1}$ and $b_i = b_{i+1} \oplus g_i, i = n-2, \ldots, 0$.

This Gray coding scheme can be combined with the above pixel transfer scheduling scheme to reduce the self-transition activities on DRAM on-chip H-tree routing wires. Fig. 8 shows the self-transition activity reduction when using pixel scheduling with and without Gray coding, compared with conventional intra-block row-by-row image data transfer. All the video sequences, except the 1080p HD video sequence "tractor" and "rush hour", are in CIF format. The results show that on average pixel transfer scheduling itself reduces the self-transition activity by 21.5%, and the reduction can increase to 38.3% when combining pixel transfer scheduling with Gray coding. These pixel-level manipulation schemes are more effective for those video sequences that have less spatial high frequency information, such as "akiyo".

The above two data manipulation schemes aim to reduce the self-transition activities on each H-tree data routing wire. As pointed out in Section II, inter-wire coupling capacitance may

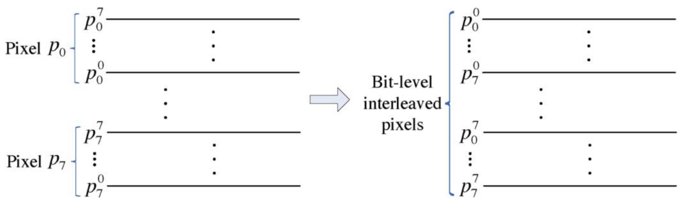


Fig. 9. Illustration of using bit-level interleaving to increase the correlations among bits transferred over adjacent wires in order to reduce coupling-induced energy consumption.

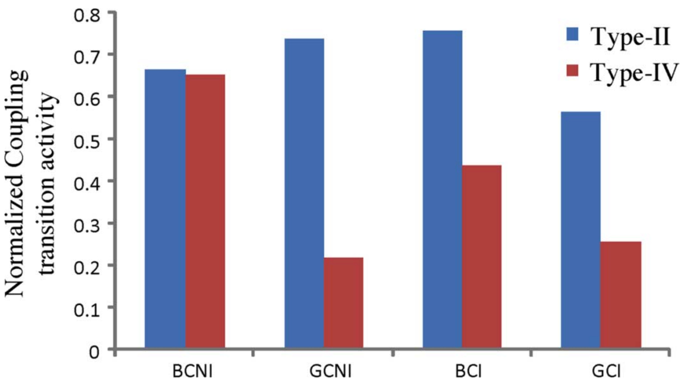


Fig. 10. Normalized coupling transition activity when using binary coding without interleaving (BCNI), Gray coding without interleaving (GCNI), binary coding with interleaving (BCI), and Gray coding with interleaving (GCI). Pixel transfer scheduling is used in all the four scenarios.

account for a noticeable percentage of on-chip data routing energy consumption, especially as the wire pitch reduces and aspect ratio increases with technology scaling down. Hence, reducing the coupling-transition activity can also noticeably reduce DRAM image data access energy consumption. Intuitively, if bits transferred on adjacent wires have a larger correlation, the associated coupling-transition activity tends to be relatively small. Hence, the objective is to increase the correlations of bits transferred on adjacent wires for image data access. In the straightforward design practice, the bits of a pixel are grouped together and transferred on a set of adjacent wires. As a result, bits on adjacent wires are less correlated since the bits representing the same pixel should not be highly correlated to each other. Clearly, bits at the same position in spatially adjacent pixels tend to have more correlations, especially bits at higher position (i.e., those more significant bits). Therefore, we propose to interleave the pixels at the bit-level so that adjacent wires carry the same-position bits, which can increase the bit correlations between adjacent wires and hence reduce the coupling-induced energy consumption. This is referred to as *bit-level interleaved pixel transfer*. Assume the DRAM on-chip H-tree data routing has a width of 64 bits and we transfer eight 8-bit pixels at one time. Let p_i^j represent the jth bit of the ith pixel p_i, where $0 \leq i, j \leq 7$. As illustrated in Fig. 9, instead of the straightforward pixel-by-pixel mapping, we can interleave all the eight pixels at the bit-level to increase the correlations among bits transferred over adjacent routing wires.

The bit-level interleaving scheme can be directly combined with the above pixel-level data manipulation schemes (i.e., pixel transfer scheduling and Gray coding). Using the most straightforward design practice as the baseline scenario (i.e., pixels are transferred in a row-by-row without using any data manipulation techniques above), Fig. 10 shows the reduction of coupling transition activities of the type-II and type-IV transitions as described in Section II, where the CIF sample video sequence “akiyo” is used. All the four scenarios in Fig. 10 use pixel transfer scheduling, where BCNI and BCI represent the scenarios of using binary coding without and with bit-level interleaving, respectively, and GCNI and GCI represent the scenarios of using Gray coding without and with bit-level interleaving, respectively. The results suggest that we can largely reduce the coupling-induced energy consumption by using such bit-level interleaving scheme. Finally, we note that this bit-level interleaving technique essentially aims to reduce the total coupling transition activities and does not necessarily guarantee to always reduce both type-II and type-IV transitions, as shown in Fig. 10. The exact changes (either increase or reduction) of each individual type of transition are data dependent and could largely vary from one video sequence to another.

Finally, we note that the above presented data manipulation techniques can be realized within the video processing engine (either a software running on a processor or a dedicated hardware core), which is independent on specific video processing algorithms and completely transparent to SoC DRAM controller and DRAM chips.

B. Impact of Image Frame Recompression

Prior work [12], [24]–[26] proposed to apply image frame recompression to reduce the required DRAM storage capacity, chip-to-chip link bandwidth, and DRAM access power consumption in video processing systems. The image frame is compressed before being stored in external memory and read back in compressed form before being used for display, coding, or other purposes. Image frame recompression algorithms are typically transform-based, e.g., discrete cosine transform (DCT) and Hadamard transform, and can be either lossless or lossy. Compared with lossless image recompression, lossy image recompression can achieve much better efficiency and realize predictable compression ratio. Hence, we are particularly interested in the use of the above presented data manipulation techniques when lossy image recompression is being employed.

In this work, we consider the scenario where lossy image recompression employs 8-point modified Hadamard transform (MHT) [26] followed by simple shift-based quantization. Among the output of MHT, the DC coefficient is kept un-quantized since it conveys the most important information of the original eight pixels, and the other 7 higher frequency coefficients are quantized through simple binary right shift. Following [12], we consider four sets of quantization shift parameters listed in Table I. As we increase the number of right shift bits, the corresponding higher frequency coefficient is scaled down more significantly and hence a larger compression ratio can be realized at the cost of bigger loss of image quality. When the quantization parameter set QP is 0, the MHT output is not quantized at all.

TABLE I
SHIFT-BASED QUANTIZATION PARAMETERS

QP	Number of right shift bits						
	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	0	0	0	0	0
1	1	1	2	1	2	2	3
2	2	2	3	2	3	3	4
3	3	3	4	3	4	4	5

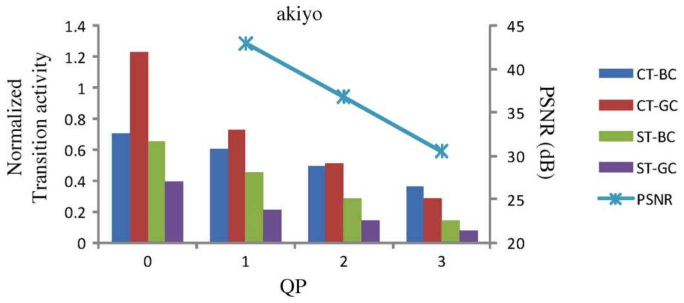


Fig. 11. Normalized coupling and self-transition activities under different QPs. CT-BC: coupling activity with binary coding; CT-GC: coupling activity with Gray coding; ST-BC: self-transition activity with binary coding; ST-GC: self-transition activity with Gray coding. All the four scenarios use pixel transfer scheduling to reduce self-transition activity.

To minimize the computational complexity, image recompression carries out the transform in one dimension. Since transform inherently de-correlates the pixels, the eight output coefficients of 1-D transform are much less correlated. As a result, the bit-level interleaving is no longer effective to reduce the coupling-induced energy consumption. On the other hand, spatially adjacent groups of 8-point transform output still have strong correlations. Therefore, we can still apply pixel transfer scheduling and Gray coding to reduce the self-transition activity. Using the CIF sample video sequence "akiyo" as a test vehicle, Fig. 11 shows the normalized coupling and self-transition activities under different QPs when using pixel scheduling with either binary coding or Gray coding. It also shows the corresponding PSNR (peak signal-to-noise ratio) since different QPs result in different image quality. Although Gray coding can always lead to less self-transition activity, it may not necessarily ensure less coupling activity over binary coding. In fact, the results show that Gray coding tends to result in higher coupling activity. Therefore, Gray coding is not always preferred over binary coding in this context, especially when coupling capacitance accounts for a large portion of total wire capacitance.

IV. HETEROGENEOUS DRAM ARCHITECTURE FOR VIDEO PROCESSING

Image data access in video processing can be unbalanced. For example, in video coding, the reference image frames are read from DRAM multiple times, while the other frames may be read only once. Intuitively, it is favorable if those *hot* data can be accessed with a relatively low energy consumption. However, present commodity DRAM employs a fully balanced on-chip H-tree routing strategy to ensure the uniform access to any memory sub-arrays in terms of latency and energy. The mismatch between unbalanced image data access and balanced memory access cost directly motivates us to investigate the potential of re-architecting DRAM for reducing image data access energy consumption.

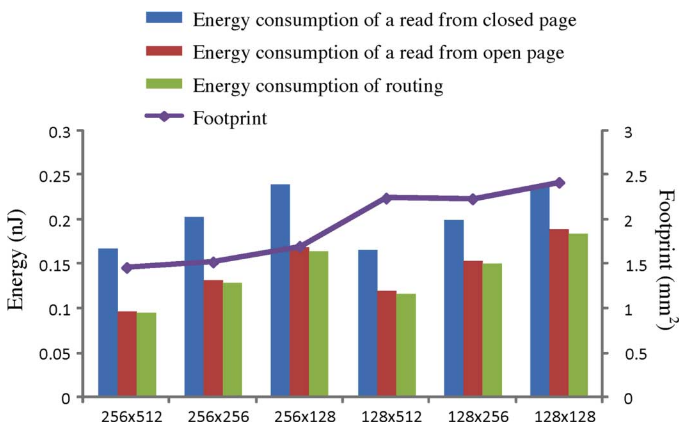


Fig. 12. Energy consumption versus memory footprint of a 32 Mb memory with different sub-array sizes.

A. Basic Design Strategy

Assuming it is feasible to customize DRAM architecture for video processing due to its huge commercial market, we propose a heterogeneous DRAM architecture geared to video processing. The key is to allocate a small portion of memory sub-arrays optimized for low-power data access to store those hot image data, which is referred to as *hot data zone*. This leads to a heterogeneous DRAM architecture consisting of a small hot data zone optimized for low power data access and main memory array optimized for storage density. As long as the hot data zone has a reasonably small storage capacity, such a heterogeneous DRAM architecture will not incur significant bit cost penalty over its conventional homogeneous counterpart.

As pointed out above, image data access should employ page mode DRAM operation, in which the on-chip H-tree data routing dominates the energy consumption as demonstrated in Fig. 3. Therefore, the routing of the hot data zone should be separate from the main memory array, and the hot data zone should be the closest to the memory chip I/O. Moreover, the size of memory sub-array in the hot data zone should be carefully chosen. We use CACTI to estimate the energy consumption and footprint of a 32 Mb memory with different sub-array sizes under either open page or closed page mode, as shown in Fig. 12. The energy consumption per access from closed page with 128×512 sub-arrays is lower than with 256×512 sub-arrays, because longer bitlines are precharged in the latter case. Nevertheless, with respect to the energy consumption per access from open page, which is dominated by on-chip routing energy consumption, the design with 128×512 sub-arrays incurs higher energy consumption than the one with 256×512 sub-arrays, because too small sub-array incurs higher routing energy consumption. For image data access where page mode is heavily used as pointed out before, the one with the minimal energy consumption for open page should be used to implement the hot data zone.

In order to obviate any impact on memory I/O interface and addressing, the small hot data zone should have the same number of banks as the main memory array, and its address space is fixed so that the DRAM controller can easily determine whether the request is for the hot data zone or main memory array using address masking. Due to smaller sub-array sizes employed by the hot data zone, some internal operations can be configured differently from the main memory array. For example, small sub-array size corresponds to a higher storage capacitance versus bitline capacitance ratio, leading to a longer retention time. Hence, the hot data zone can have a longer refresh period. Furthermore, we can adopt the multiple supply voltage technique that has been widely used in logic circuit design, i.e., we still use high supply voltage in the main memory array to maintain low random access latency, while use relatively low supply voltage on H-tree in the small hot data zone to further reduce data access energy consumption. Since most image data access is from open page in video processing, the hot data zone is small and the routing from hot data zone to I/O is short, the small increase in latency due to low supply voltage may not be an issue.

B. Impact on Video Processing Algorithm Design

In the above proposed heterogeneous DRAM architecture, the capacity of the hot data zone is determined in the design time and remains fixed in the run time. Nevertheless, various video processing applications may have largely different amount of hot image data, and the hot data zone may not be able to store all the hot data for certain video processing applications. Since not all the hot data are frequently accessed at the same time, we can use run-time data swap between the hot data zone and main memory array when necessary. Since such DRAM internal data swap incurs energy consumption and latency overhead, we should minimize the occurrence of data swap activity, for which the video processing algorithm may need to be appropriately modified.

In this work, we use 1080p HDTV video decoding as a case study to further demonstrate the above discussion. To cope with 1080p HDTV video decoding, we need 24 Mb memory to store both luminance and chrominance data of one reference frame. Since multiple reference frame motion estimation (MRF-ME) is widely used for HD video coding, the hot reference frame data may occupy around 100 Mb memory, which can be much larger than the storage capacity of the hot data zone. As a result, we should rely on run-time data swap to move hot reference image data being accessed into the hot data zone. As illustrated in Fig. 13, we assume m reference frames $\{R_1 \ldots R_m\}$ are used to predicate the successive n frames $\{P_1 \ldots P_n\}$. In current design practice, only after the decoder finishes the motion compensation for one frame P_i, it will begin to process the next frame P_{i+1}. If we keep the same frame-by-frame processing flow when using the proposed heterogeneous DRAM, each reference frame has to be moved into and out from the hot data zone once for each frame P_i, leading to large energy consumption and latency overhead. To address this issue, we can change the processing flow as follows: We move the same region of all the reference frames into the hot data zone (e.g., the shaded region in all the m reference frames as illustrated in Fig. 13), and they are used to carry out motion compensation for the same region of all the n frames $P_1 \ldots P_n$ (e.g., the macroblocks in the

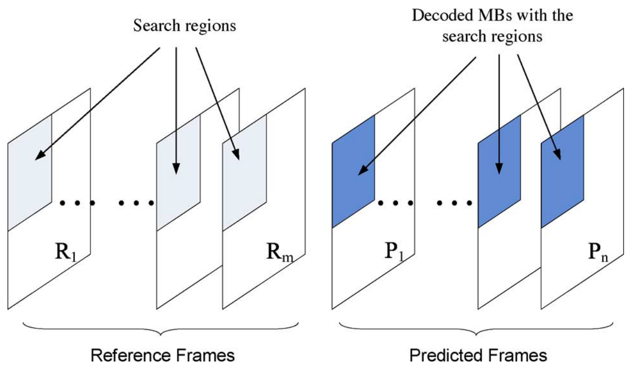


Fig. 13. Illustration of decoding sequence with limited capacity of *hot* data arrays.

shaded region in the n frames as illustrated in Fig. 13). Therefore, the motion compensation for all the n frames are carried out simultaneously in a portion-by-portion manner. As a result, the occurrence of hot data swap in DRAM is largely reduced.

We briefly discuss the reference image access energy consumption cost estimation when using the above modified processing flow. Let M denote the number of reference frame and N denote the number of frames predicted by the reference frames. Assume each image frame requires I bits of storage capacity, and the hot data zone has a capacity of C bits. Since the data access energy consumption of the hot data zone is a function of its storage capacity, we use $E(C)$ denote the access energy consumption per bit of the hot data zone. In addition, let E_g denote the access energy consumption per bit of the main memory array. When using the above modified video decoding process flow in heterogenous DRAM, the reference frame access energy consumption can be estimated as

$$E = 2 \cdot \left(\frac{M \cdot I}{C} - 1\right)^{+} \cdot (E_g + E(C)) \cdot C + N \cdot I \cdot E(C) \quad (2)$$

where $(\cdot)^{+}$ denotes the positive part of the operand, i.e.,

$$(x)^{+} = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases}$$

The first term in (2) represents the energy consumption induced by DRAM internal data swap, and the second term is the energy consumption induced by retrieving reference image data for the video decoder.

V. Simulation Results

To demonstrate and compare the above presented design techniques, we use video decoding as a test vehicle and carry out extensive simulations. Regarding the DRAM, we set its parameters as 64 bits of on-chip data bus width, 8 bits of I/O width, one read/write port, and 8 banks. The DRAM modeling is carried out at 45 nm technology node using CACTI [22] which is further enhanced to capture the energy consumption induced by both self-transition and coupling transition. We consider two different DRAM capacities, including 512 Mb and 2 Gb. The DRAM modeling results used in our evaluation are listed in Table. II. The energy consumption is calculated by

TABLE II
DRAM PARAMETERS

Capacity	512Mb	2Gb
Area (mm^2)	12.11	49.88
Read energy from a closed page (nJ)	1.07	1.44
Read energy from an open page (nJ)	0.22	0.59
Energy of data bus (nJ)	0.21	0.57

TABLE III
POWER CONSUMPTION (MW) COMPARISON AND SAVING OF ENTIRE VIDEO DECODER

video format	decoder core	512Mb DRAM		saving
		baseline	proposed	
QCIF@15fps	0.125	0.19	0.12	22.2%
D1@30fps	12.4	5.14	3.17	11.2%

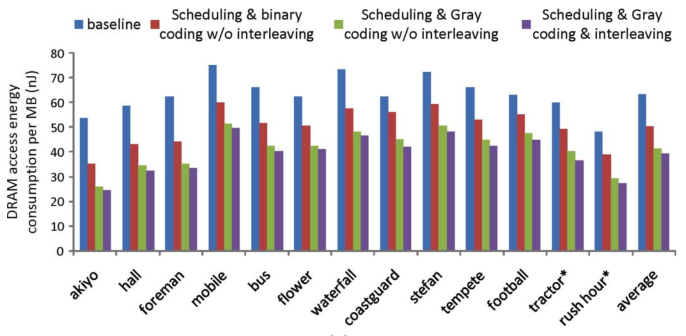
video format	decoder core	2Gb DRAM		saving
		baseline	proposed	
QCIF@15fps	0.125	0.50	0.31	30.4%
D1@30fps	12.4	13.69	8.36	20.4%

assuming 50% switching activity. With a larger capacity, the 2 Gb DRAM has a bigger footprint, which leads to longer routing bus and hence higher energy consumption. Regarding video decoding, each macroblock contains 16×16 pixels and each pixel's luminance intensity is represented by 8 bits. The chroma format is 4:2:0, hence the amount of chrominance data is half of that of luminance data. The Gray code encoder and decoder are implemented with Verilog design entry and synthesized with Synopsis design tools.

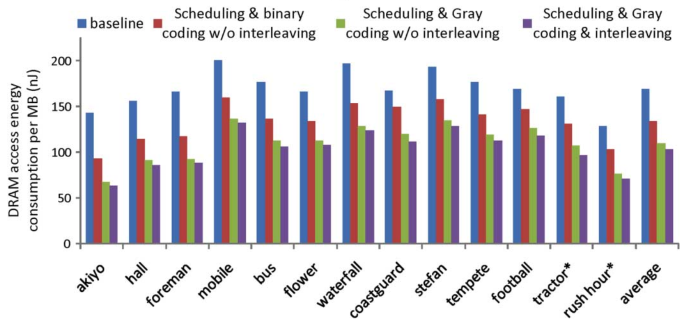
A. Evaluation of Data Manipulation Techniques

First, we evaluate the image data access energy consumption when using the data manipulations techniques presented in Section III, including pixel transfer scheduling, Gray coding, and bit-level interleaving. For the purpose of comparison, we also consider the case when none of these techniques are used and image data are simply accessed in a row-by-row manner, which is used as the baseline. Fig. 14 shows the average DRAM energy consumption to read or write both luminance and chrominance data of a 16×16 macroblock under various video sequences, where different combinations of these data manipulation techniques are considered. The results clearly demonstrate the effectiveness of these data manipulation techniques. For 512 Mb DRAM, the average access energy consumption per macroblock is 63.42 nJ in the case of baseline, which reduces to 39.20 nJ when all the three techniques are used, representing 38.2% reduction; for 2 Gb DRAM, the average access energy consumption per macroblock is 169.07 nJ in the case of baseline, which reduces to 103.27 nJ when all the three techniques are used, representing 38.9% reduction.

To further demonstrate effectiveness of these data manipulation techniques in the context of entire video decoding, including both processing core and DRAM energy consumption, we estimate the corresponding DRAM power consumption under different video formats when using either the baseline or the combination of all the three data manipulation techniques. Moreover, we extract the video decoder core power consumption for two video formats QCIF (176×144) and D1



(a)



(b)

Fig. 14. DRAM access energy consumption per macroblock in the case of (a) 512 Mb and (b) 2 Gb DRAM. Both luminance and chrominance data are accessed.

(720×480) based upon the results presented in [27]. The decoder is implemented at 0.18 μm technology node. The results are listed in Table III, in which all the power are in the unit of mW. The Gray encoding and decoding power consumption for two video formats is 1.3 μW and 0.21 mW, respectively, which are negligible compared to the decoder and memory power consumption. The latency incurred by Gray coding is very small and can be hidden by pipelining. The power reduction on DRAM access contributes substantially to the total power reduction of the entire decoding system.

Next, let us consider the scenarios when image frame recompression is used. Fig. 15 shows the estimated 2 Gb DRAM access energy consumption per macroblock when using different schemes. We note that all the schemes use the pixel transfer scheduling technique, hence for the purpose of being concise the legends in Fig. 15 do not explicitly contain the term "scheduling". For the purpose of comparison, the figure also shows the energy consumption when using all the three data manipulation techniques without recompression. Four video sequences "akiyo", "hall", "foreman", and "mobile" sequences are used in the simulation. As demonstrated in Fig. 15, in general, Gray coding can realize better energy efficiency than binary coding under the same QP value. In addition, as we improve the compression ratio by increasing the QP value, we can reduce the memory access energy consumption because less amount of data need to be accessed. With highest compression ratio (i.e., $\mathrm{QP} = 3$), our techniques can achieve up to 80% of power reduction, compared to baseline as shown in Fig. 14(b). By varying the recompression quantization parameter QP from 1 to 3, Fig. 16 shows the trade-off between image fidelity degradation and reference image access power

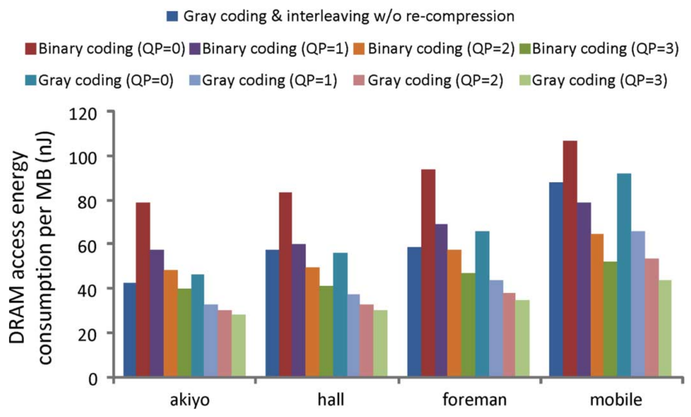


Fig. 15. 2 Gb DRAM access energy consumption per macroblock when frame recompression is being used. Both luminance and chrominance data are considered.

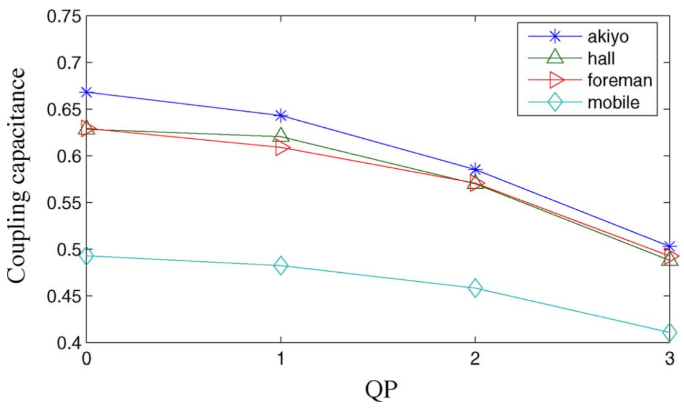


Fig. 17. Threshold of normalized coupling capacitance, below which Gray coding can outperform binary coding in terms of energy consumption.

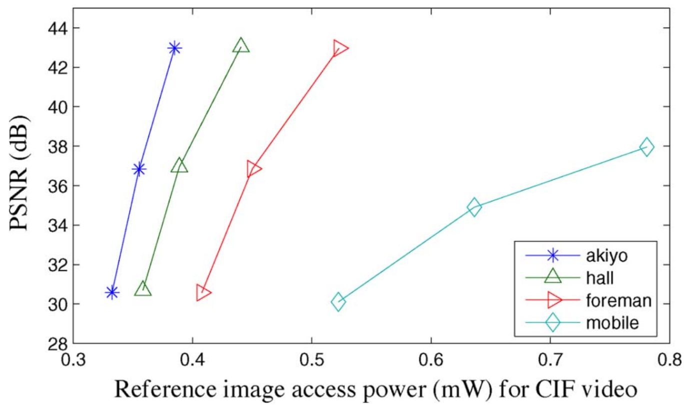


Fig. 16. Reference image access power consumption versus PSNR under different recompression quantization parameter QP. Points on each curve from top to bottom represent QP of 1, 2, and 3, respectively.

consumption in real-time CIF video decoding. As mentioned in Section III-B, Gray coding may not necessarily result in less energy consumption than binary coding in the context of image recompression. Because Gray coding can reduce the self-transition activity but tends to induce more coupling activity as shown in Fig. 11, Gray coding can only outperform binary coding when the coupling capacitance is sufficiently small compared with the total wire capacitance. Based on the simulation results, Fig. 17 shows the threshold of normalized coupling capacitance versus total wire capacitance, below which Gray coding can outperform binary coding in terms of energy consumption. The threshold drops as we increase the compression strength. In this study, we set the normalized wire coupling capacitance to 26% in the DRAM modeling, well below the thresholds shown in Fig. 17. Therefore, as shown in Fig. 15, Gray coding outperforms binary coding in our study.

B. Evaluation of Heterogeneous DRAM Architecture

Fig. 18 shows the energy to access one macroblock in the hot data zone when varying the capacity of the hot data zone and the size of each memory sub-array. All the three data manipulation techniques are used in the simulation. We also consider the use of two different supply voltages 1.2 V and 1.0 V. The results suggest that the memory sub-array size of 256 $\times$ 512 appears to be the best option that can minimize both the footprint and energy consumption. Its small footprint is simply due to the relatively large sub-array size compared with the other options. Since the energy consumed by routing dominates the total image data access energy consumption, the sub-array size of 256 $\times$ 512 results in minimal routing energy consumption and hence the minimal overall access energy consumption. When using high supply voltage and 256 $\times$ 512 sub-array size, the average energy to read one macroblock from a 32 Mb hot data zone is 9.21 nJ, which is only 21% of the energy consumed to read one macroblock from a homogeneous 2 Gb DRAM. By decreasing the supply voltage from 1.2 V to 1.0 V, the energy is further reduced by approximately 30%. In addition, such a 32 Mb hot data zone only occupies 1.45 mm^2 compared to 50 mm^2 of a 2 Gb DRAM.

To demonstrate the impact of the hot data zone capacity on video decoding system energy consumption, we use HDTV sequences "tractor" with multiple reference frames decoding as an example. In the case where the hot data zone is not large enough to hold all the current reference frames, we use the run-time hot data swap and accordingly modify the decoding process flow as discussed in Section IV-B to minimize the data swap energy consumption. We set the entire DRAM is 2 Gb. The number of reference image frames varies from 1 to 5. In this HDTV video sequence, each frame requires 24 Mb storage capacity, and each reference image frame is involved in the motion compensation for 10 frames. We consider three different capacity of the hot data zone, including 8 Mb, 16 Mb, and 24 Mb. Fig. 19 shows the average power consumption for retrieving each reference image under different hot data zone capacity and different number of reference frames being used in motion compensation. As more reference image frames are involved in the motion compensation, hot data swap will be carried out more frequently, leading to higher energy consumption overhead. Therefore, the average reference image frame retrieval energy consumption increases as the number of reference image frames increases, as shown in Fig. 19. For the purpose of comparison, this figure also shows the case without using the hot data zone, where the reference image retrieval energy consumption is independent on the number of reference image frames. Our results

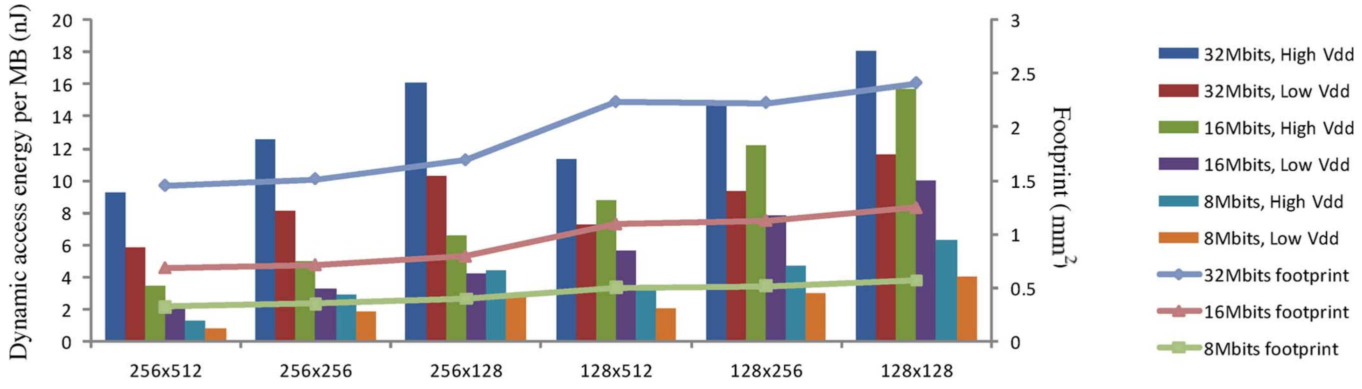


Fig. 18. Average energy consumption to access one macroblock from the hot data zone with various configurations and two different supply voltages, and footprint of the hot data zone with different configurations.

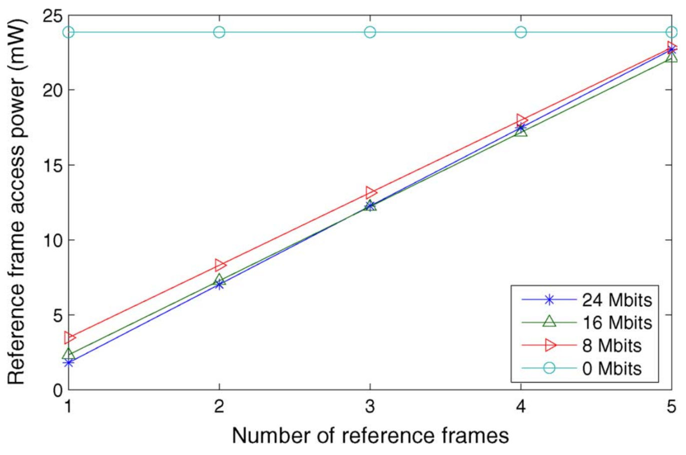


Fig. 19. Average power consumption for retrieving each reference frame for real-time 1080p video decoding with various capacities of hot data zone in a 2 Gb DRAM.

show that heterogeneous DRAM architecture outperforms conventional architecture if less than 5 reference frames are used. The power can be saved as much as 92% if all the techniques are used together. In addition, energy increases more rapidly with number of reference frames increasing if larger hot data zone is used, because access energy from that region is higher.

VI. Conclusion

This work concerns how to reduce DRAM image data access energy consumption in video processing applications. Its motivation is two-fold: 1) video processing is one of the most energy-hungry functions in pervasive mobile devices, and 2) DRAM image data access energy consumption increasingly outweighs video processing logic energy consumption, but prior efforts mainly focused on how to reduce the latter. The objective of this work is to develop domain-specific techniques to reduce DRAM energy consumption by appropriately exploiting image data access characteristics, in particular the image data spatial and temporal correlation and unbalanced image access in most video processing. We propose to use three simple yet effective data manipulation techniques to exploit the spatial/temporal correlation to reduce DRAM data routing energy consumption, and propose a heterogeneous DRAM architecture to embrace the unbalanced image access to further reduce DRAM energy consumption. Using CACTI-based DRAM modeling and video decoding as a test vehicle with representative video sequences, we quantitatively demonstrated the effectiveness of these design techniques.

References

[1] D. Zhou, J. Zhou, X. He, J. Zhu, J. Kong, P. Liu, and S. Goto, "A 530 Mpixels/s 4096 × 2160@60 fps H.264/AVC high profile video decoder chip," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, pp. 777–788, 2011.

[2] V. Sze, D. Finchelstein, M. Sinangil, and A. Chandrakasan, "A 0.7-V 1.8-mW H.264/AVC 720p video decoder," *IEEE J. Solid-State Circuits*, vol. 44, no. 11, pp. 2943–2956, 2009.

[3] M. Ghosh and H. Lee, "Smart refresh: An enhanced memory controller design for reducing energy in conventional and 3D die-stacked DRAMs," in *Proc. 40th Annu. IEEE/ACM Int. Symp. Microarchitecture*, 2007, pp. 134–145.

[4] H. Zheng, J. Lin, Z. Zhang, E. Gorbatov, H. David, and Z. Zhu, "Mini-rank: Adaptive DRAM architecture for improving memory power efficiency," in *Proc. 41st Annu. IEEE/ACM Int. Symp. Microarchitecture*, 2008, pp. 210–221.

[5] A. Udipi, N. Muralimanohar, N. Chatterjee, R. Balasubramonian, A. Davis, and N. Jouppi, "Rethinking DRAM design and organization for energy-constrained multi-cores," in *Proc. 37th Annu. Int. Symp. Computer Architecture*, 2010, pp. 175–186.

[6] E. Cooper-Balis and B. Jacob, "Fine-grained activation for power reduction in DRAM," in *Proc. 43th Annu. IEEE/ACM Int. Symp. Microarchitecture*, May 2010.

[7] A. Amin and Z. Chishti, "Rank-aware cache replacement and write buffering to improve DRAM energy efficiency," in *Proc. 16th ACM/IEEE Int. Symp. Low Power Electronics and Design*, 2010, pp. 383–388.

[8] H. Kim and I. Park, "High-performance and low-power memory interface architecture for video processing application," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 11, pp. 1160–1170, Nov. 2001.

[9] T. Chuang, L. Chang, T. Chiu, Y. Chen, and L. Chen, "Band width efficient cache-based motion compensation architecture with DRAM friendly data access control," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2009, pp. 2009–2012.

[10] Y. Li, Y. Liu, and T. Zhang, "Exploiting three-dimensional (3D) memory stacking to improve image data access efficiency for motion estimation accelerators," *Elsevier J. Signal Process.: Image Commun. (Special Issue on Breakthrough Architecture for Image and Video Systems)*, vol. 25, no. 5, pp. 335–344, Jun. 2010.

[11] S. Dikbas and F. Zhai, "Lossless image compression using adjustable fractional line-buffer," *Elsevier J. Signal Process.: Image Commun. (Special Issue on Breakthrough Architecture for Image and Video Systems)*, vol. 25, no. 5, pp. 345–351, Jun. 2010.

[12] T. Lee, "A new frame-recompression algorithm and its hardware design for MPEG-2 video decoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 529–534, Jun. 2003.

[13] R. Osorio and J. Bruguera, "A combined memory compression and hierarchical motion estimation architecture for video encoding in embedded systems," in *Proc. 9th EUROMICRO Conf. Digital System Design*, 2006.

[14] Y. Jin, K. Yum, and E. Kim, "Adaptive data compression for high performance low-power on-chip networks," in *Proc. IEEE/ACM Int. Symp. Microarchitecture*, 2008, pp. 354–363.

[15] M. Thuresson, L. Spracklen, and P. Stenstrom, "Memory-link compression schemes: A value locality perspective," *IEEE Trans. Comput.*, vol. 57, no. 7, pp. 916–927, Jul. 2008.

[16] M. Mehendale and S. D. Sherlekar, *VLSI Synthesis of DSP Kernels—Algorithmic and Architectural Transformations*. New York: Springer, 2001.

[17] L. Benini, G. DeMicheli, E. Macii, D. Sciuto, and C. Silvano, "Asymptotic zero-transition activity encoding for address busses in low-power microprocessor-based systems," in *Proc. 7th Great Lakes Symp. VLSI*, Mar. 1997, pp. 77–82.

[18] E. Musoll, T. Lang, and J. Cortadella, "Working-zone encoding for reducing the energy in microprocessor address buses," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 4, pp. 568–572, Dec. 1998.

[19] H. Mehta, R. Owens, and M. Irwin, "Some issues in Gray code addressing," in *Proc. 6th Great Lakes Symp. VLSI*, 1996, pp. 178–181.

[20] K. Hirose and H. Yasuura, "A bus delay reduction technique considering crosstalk," in *Proc. Design, Automation and Test in Europe Conf. Exhib.*, 2000.

[21] D. Suresh, B. Agrawal, J. Yang, and W. Najjar, "Tunable and energy efficient bus encoding techniques," *IEEE Trans. Comput.*, vol. 58, no. 8, pp. 1049–1062, Aug. 2009.

[22] CACTI: An Integrated Cache and Memory Access Time, Cycle Time, Area, Leakage, and Dynamic Power Model. [Online]. Available: http://www.hpl.hp.com/research/cacti/.

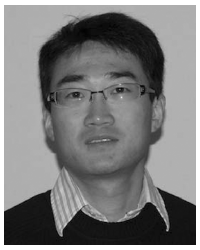
[23] Micron Technology, Inc. [Online]. Available: http://www.micron.com.

[24] Y. Lee, C. Rhee, and H. Lee, "A new frame recompression algorithms integrated with H.264 video compression," in *Proc. IEEE Int. Symp. Circuits and Systems*, 2007, pp. 1621–1624.

[25] T. Yng, B. Lee, and H. Yoo, "A low complexity and lossless frame memory compression for display devices," *IEEE Trans. Consum. Electron.*, vol. 54, no. 3, pp. 1453–1458, Mar. 2008.

[26] H. You, J. Jo, and J. Jeong, "A hierarchical lossless image compression based on modified Hadamard transform," in *Proc. 10th Workshop Image Processing and Understanding*, 1998, pp. 305–310.

[27] T. Liu, T. Lin, S. Wang, W. Lee, J. Yang, K. Hou, and C. Lee, "A 125 μW, fully scalable MPEG-2 and H.264/AVC video decoder for mobile applications," *IEEE J. Solid-State Circuits*, vol. 42, no. 1, pp. 161–168, Jan. 2007.



Yiran Li received the B.S. degree in electronics engineering from Shanghai Jiaotong University, Shanghai, China, in 2003 and the M.S. degree in electrical engineering from University of Florida, Gainesville, in 2007. He is currently pursuing the Ph.D. degree at the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY.

From 2003 to 2006, he was with SONY Corporation, Mobile Products R&D Group, Tokyo, Japan. In 2008, he worked at Marvell Semiconductor, Inc., Santa Clara, CA. His research interests include image and video processing, memory architecture design, VLSI architecture, and circuit design for multimedia processing.

Tong Zhang (M'02–SM'08) received the B.S. and M.S. degrees in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 2002.

He is currently an Associate Professor with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. His research activities span over circuits and systems for various data-storage and computing applications.

Dr. Zhang currently serves as an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-II and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.