International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2019): 7.583

System Memory Impacts Over SoC Dynamic Power Regression

Apoorva Reddy Proddutoori

San Diego
Email: *apoorvaproddutoori[at]gmail.com*

Abstract: ***A customary mixed media SoC is comprised of parts for taking pictures and recordings, packing and de-pressurizing them, and taking care of PC vision, illustrations, and how things are shown. This large number of parts need to have a similar restricted space in the outer DDR memory. The typical approach to utilizing store is somewhat terrible for taking care of media since it's excessively sluggish. In this paper, we will discuss a better approach for sorting out store that is better for mixed media, saving space in the DDR memory, and making things run smoother. Our new store arrangement incorporates an extraordinary sort of parted, a reserve with various openings that can be changed, and a chief. We tried our new arrangement and found that it made things half better for dealing with video disentangling. A sight and sound SoC need to share restricted space in the outer DDR memory. The typical approach to utilizing reserve isn't great for taking care of media. Another reserve arrangement incorporates an exceptional sort of parted, a store with various spaces that can be changed.***

Keywords: SoC, DDR memory, mixed media, sorting

1. Introduction

With the interest for longer battery duration in versatile electronic gadgets, there is an always expanding interest for executing higher levels of forceful power- the board procedures in SoCs. Power the board techniques incorporate a variety of rest modes executed by means of differing levels of clock and power gating. These modes utilize a mix of equipment state machines too as programming directed section and exit. As responsibility changes, portions of the SoC might quickly progress in furthermore, out of rest modes. To accomplish least power, the PMM configuration endeavors to stay in the least utilization state conceivable, limiting inertness to progress all through these modes. Plan intricacy is intensified by various nonconcurrent clock spaces what's more, inexactly coupled IP blocks inside the SoC. This climate presents novel difficulties to the approval of PM modes and troubleshoot of the SoCs.

The progression in chip planning handle has been quick. The recurrence of operation has continuously been on rise. With increment in complexity level of the circuits, it is imperative to note that not each component within the circuit encounters the same clock timings. This leads to diminished productivity and in cases misfortune of information. The three enormous concerns for the coordinates circuits plans are the clock skew, constriction and control misfortune.

Each multi-media component accesses external memory by injecting a unique traffic pattern. This is similar to how an imaging IP predominantly accesses raster data from an image frame, while a video decoding IP does a mix of deterministic and random block-level accesses from a video frame. Additionally, there may be unrelated traffic, such as networking. All of this results in various dynamic traffic scenarios that must still meet the latency and throughput requirements of the specific use case.

To influence inconstancy, the sensational increment of spot error rate, this paper examines the store access conduct to give the store the executives procedures to low-voltage conditions and balances the frequency to acquire the greatest energy saving. However a variable-inactivity store can endure a enormous number of access-time disappointments, the high multi duty cycle access rate will cause huge execution corruption in L 1 and furthermore result into more spillage. As a result, a dynamic store the executives is required, expecting to decrease the inertness and to involve quick lines as much as possible. One of the benefits of the proposed arrangements is to use core run-time application data [or improving the entire framework.

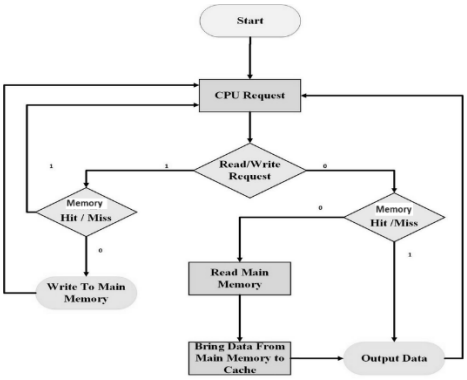


Figure 1: Memory hit/miss

The execution cores of advanced superscalar microprocessors demand high-performance data path circuits characterized by single-cycle latency and throughput. To achieve this, innovative circuit design solutions are necessary to deliver such performance despite the challenges posed by increasing frequencies, reverse scaling of interconnects, and rising device leakage. In this paper, we will explore key circuit solutions employed in the design of high-performance, energy-efficient data path circuits.

2. Memory Traffic

A memory controller determines whether an access is a hit or miss by comparing the tag. It can separate the search for memory tags from the actual access to memory data. Once a memory hit is confirmed, the memory controller activates the

Write-Level Access (WL) to read the bit line corresponding to the specified cycle count. However, the bit line is pre-charged to a high voltage level and discharges in every regular access cycle.

To implement extended access, the SRAM pre-charge signal must be gated, as illustrated in the paper. This represents the concept of latency-aware memory management (LCM). With this approach, the bit line discharge phase is extended, allowing for the read of weaker cells. Consequently, the correct data is read out in the extended cycle. Additionally, it is possible to analyze the delay distribution of cells, which helps in arranging the discharge phases of different cells within a cycle. Cells with delays longer than 2 cycles are considered timing defects and are disabled for the benefit of the entire system.

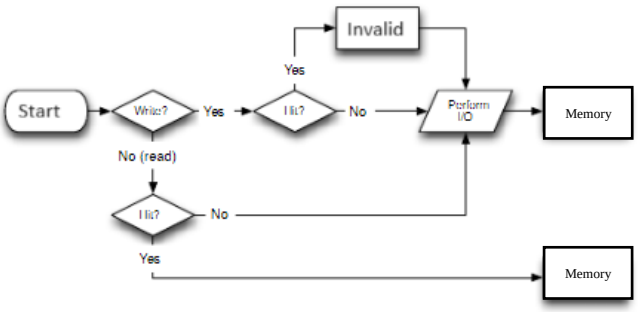
Therefore, the hit/miss arbiter only needs to process one bit for the multiple-access part (with slower lines). The multiple-access part is designed to accommodate different latency fault rates and can be configured a priori to store timing information in a timing table [13]. This means there is an extra layer of storage for setting up the configuration. The CPU fetches data from the fastest lines as much as possible, but if a miss occurs, the data is moved to the faster line, and the data from the slower lines is moved to the slower parts of the memory. This data eviction is part of the memory's replacement policy.

In cases where there is a hit on the slower lines, it presents an opportunity to mitigate the penalty for a miss on the first line. The data is then moved to a 1-cycle line. This strategy optimizes the use of both fast and slow lines in the memory. The operation and access flow for this strategy are illustrated in Figure 3. The CPU sends a request to the L1 memory to check both the fast and slow lines simultaneously. If a hit is found in the fast line, it finishes within a single cycle as a normal memory operation. If a miss occurs in the L1 memory, the data is moved to the faster line, and the data from the slower lines is moved to the slower parts of the memory. This process is part of the memory's replacement policy.

This method is based on the Least Recently Used (LRU) replacement policy in memory control. It is implemented by slightly modifying the memory management and adding two additional policies: "hit-time-aware replacement policy" and "hit-data migration policy." These policies enhance the functionality of the latency-aware memory management system. The next two sections will delve into the details of these policies within the latency-aware memory management system.

3. Memory State Retention

Many System-on-Chip (SoC) designs incorporate IP blocks that demand exceptionally low sleep-state exit latencies. To meet these requirements, these SoCs utilize dual power rails integrated with the architecture. These rails serve both as a main operation rail and a state retention rail. In normal operation, both rails are powered. However, during power-down (save) and power-up (restore) sequences, the power consumption of an offline block is virtually zero, yet the process affects the block's entry and exit latency. The storage and retrieval of the debug logic state are typically handled by a microcontroller, hardware state machine, or a dedicated debug controller. The exact location of the saved state may vary based on the architecture's needs, using either SRAM of the internal microcontroller, a specified SoC array, or a portion of the main memory.



When the IP block is no longer in use, the power-down sequence initiates, disconnecting power from the main operation rail and keeping the power to the state retention rail. This design enables exceptionally low sleep state exit latencies, eliminating the necessity for state restoration from saved locations. In this configuration, the debug logic is designed to rely on the state retention rail for state preservation. This approach ensures that debug logic states remain available throughout, including during and after the sleep event. To minimize power consumption on the state retention rail, the debug logic must be minimized, utilizing mission-mode clock or power gate capabilities.

4. Memory Hit/Miss Management

The video codec engine consists of several hardware execution units, each tasked with specific codec-related operations, such as entropy decoding. These execution units operate in parallel, enabling them to access pixel and control data inter-unit. This functionality is analogous to that of a conventional multithreaded, multi-core CPU. The external memory traffic accessed by the video decoder engine typically involves two types of access patterns:

a) Deterministic Block Accesses: This involves storing the output of de-blocking filters into external memory. This data is used for both video display and future decoder frame processing. Typically, this data is organized in 16x6 pixel blocks, also known as Macro Blocks (MB).
b) Random Block Accesses: During motion compensation in a video decoding engine, macro blocks are retrieved from various reference frames to calculate interpolated data necessary for video reconstruction.

Traditionally, caches have been constructed with a write-through scheme. However, this approach can lead to inefficiencies in video decoder traffic, especially with smaller transactions at the block level. For instance, a 16-byte transaction for a Macro Block (MB) can result in up to 80 percent loss in efficiency due to alignment issues with 64-bit memory. To mitigate this, Write Merging has been introduced, which optimizes data transfer by performing write back only on evictions. Additionally, when writing to a section of a write frame buffer that has already been read from memory, there's no need to cache the line again, further enhancing data transfer efficiency.

Prefetching is another cache feature that streamlines the process of retrieving the next set of memory addresses, often in a speculative manner. This, particularly in traffic patterns characterized by predictability and incremental nature, does not lead to loss of cache utilization. Instead, it facilitates the execution of multiple transactions on the same memory page. This optimization can allow the memory controller to issue more transactions to open DDR pages, thereby avoiding the penalties associated with page close and page open penalties. For image data, prefetching may need to be done in a 2D fashion, requiring the fetching of multiple lines at once.

The Least Recently Used (LRU) replacement policy offers a unique advantage in scenarios where incremental address patterns are expected. Implementing an LRU policy in traffic patterns of stochastic nature can be challenging, as it requires tracking, sorting, and selecting each cache line's access time for eviction. In contrast, for incremental patterns, an alternative approach using a circular buffer can be more efficient. This setup allows the cache to evict the cache line corresponding to the write operation that is no longer necessary.

5. Conclusion

Memory blocks are employed to diminish the frequency of external memory accesses and to decrease the latency associated with these operations. This document addresses the challenges encountered with conventional cache systems in multimedia SoC design. It introduces a novel cache architecture that integrates a qualifier-based splitter, multiple fully associative cache modules, and an arbiter. Furthermore, the document explores innovative use cases for this architecture, including the application as an infinite circular buffer for data buffer sharing among hardware components. Simulation results demonstrate a 50% enhancement in DDR bandwidth for video decoder traffic.

References

[1] Sanu Mathew, “ Advanced Circuit Techniques for High-Performance Microprocessor Design Challenges”, IEEE 2003
[2] Prashant Karandikar, Mihir Mody, Hetul Sanghvi, Vasant Easwaran, Prithvi Shankar Y A, Rahul Gulati, Niraj Nandan and Subrangshu Das, “ Efficient System Level Cache Architecture for Multimedia SoC”, IEEE, July 2014
[3] Yung-Hui Уи1, Po-Hao Wang1, Tien-Fu Chen1, Tay-Jyi Lin2, Jinn-Shyan Wang3, “Adaptive Variable-Latency CacЬe Management for Low-V oltage Caches”, IEEE 2014
[4] Sankaran Menon1, Chinna Prudvi1, Rolf Kuehnis1, Sukhbinder Singh Takhar1, Spencer Millican2, Eric Rentschler3, Pandy Kalimuthu4, Preeti Ranjan Panda5, Priyadarsan Patra6, Ashish Gupta7, “Techniques for Debug of Low Power SoCs”, 2019 20th International Workshop on Microprocessor/SoC Test, Security and Verification (MTV)
[5] PAI CHEN, JIANHUI YUE , XIAOFEI LIAO , AND HAI JIN, “Trade-off Between Hit Rate and Hit Latency for Optimizing DRAM Cache”, IEEE 2018
[6] Prashant Karandikar , Mihir Mody, Hetul Sanghvi , Vasant Easwaran, Prithvi Shankar Y.A., Rahul Gulati, Neeraj Nandan, Dipan MandaI and Subrangshu Das, “ Understanding System Level Caching Behavior in Multimedia SoC”, International Conference on Communication and Signal Processing, April 3-5, 2014, India