White Paper

Title: **Quazar QPR (Quad Partition Rate) Architecture**
Author: **M Baumann**
Date: **2020, 07, 20**

Document Number:
WP_QUAZAR_QPR ARCHITECTURE_200720

SUMMARY

This document describes the flexibility and power of the unique QPR (Quad Partition Rate) memory architecture.

This architecture begins with a focus on solving the issues with the traditional QDR architecture.

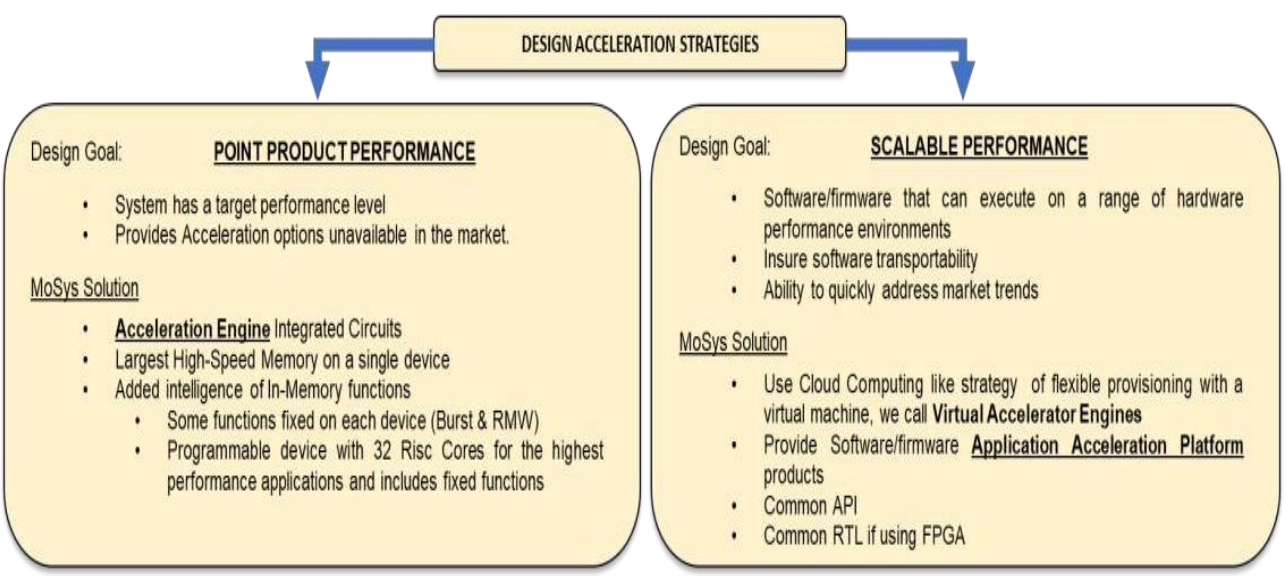
By using the MoSys' 1T memory cell technology, large capacity memories can be created that are able to selectively choose between several modes of operation.

Using a memory element call a Partition and dividing it into a number of Banks, it allows multiple independent random-access SRAM memories to exist in one device

The flexibility of the architecture allows a single device to achieve a Bandwidth up to 640Gb/s (320 Gb/s full duplex).

KEY POINTS:

- High performance memory architecture
- Configurations allow up to 8 QDR type memories in one device
- Memories are independent random-access memories
- tRC of 2.6ns-3.2ns
- Bandwidth up to 640Gb/s
- Word width up to 576b
- Perfect for QDR replacements or system upgrade to higher capacity
- Achieve 576Mb or 1Gb per device
- MoSys RTL Memory Controller simplifies User RTL with Memory accesses



In this paper we will be delving into the architectural features of the Quazar family of products from MoSys called “Quad Partition Rate” architecture. The Quazar family is group of parts that is designed to address the needs for a next generation of Synchronous SRAM devices on the market, as of the writing of this paper the device with the largest share of this market is QDR (Quad Data Rate).

As MoSys approached this issue, it was realized that there are some limitations that needed to be addressed with the present QDR devices and some goals that were desired as a result. In reviewing the present QDR devices that are on the market, the issues were:

- Density of 144Mb with one offering at 288Mb
- Use of wide parallel busses that run at very high frequency
- Strict rules to layout boards to accept these fast-wide busses
- Sourced by multiple vendors (Cypress (Infineon) and GSI) but each vendor uses a slightly different pinout
- No commitment to a future roadmap

In seeking to ensure that the new MoSys device addresses each of these issues the goals set out were:

- Design a memory system that is optimized for bandwidth and low latency
- Develop a device that is at least 576Mb in density, with a goal of 1Gb
- Use an Interface bus structure that is available today and has a growth to future speed and bandwidth (Since SerDes are becoming more ubiquitous on FPGAs, ASICs and ASSPs this is a strong option)
- Simplify hardware design efforts
- Optimize Highest bandwidth vs latency vs I/F pins

The design approach that MoSys is presenting is a significant improvement over a standard QDR device. MoSys has architected a device that provides a device that is 4 to 8 times the density of a QDR device and achieves this with comparable system latency and 2 to 5 x the system access bandwidth.

If you examine the MoSys Quazar family of devices, you will see that at many levels the architecture has been developed to support the list of feature goals. This resulted in a memory architecture that took the following into consideration:

- Cell design
- Bank architecture
- Partition architecture

- Quad Partition structure
- Internal Bus structure
- Internal clocking and tRC

These features, combined with a high speed, low pin count FPGA connection, has resulted in an SRAM with unmatched size, speed and bandwidth capabilities.

ARCHITECTURE OF THE QPR (QUAD PARTITION RATE) MEMORY

In the next part of this paper, we will discuss the MoSys approach to achieve the desired goals.

Cell Design

The cell design that MoSys has chosen to use in the Quazar product line uses an embedded DRAM design. What MoSys feels is the benefits of this design vs either a straight DRAM or and SRAM cell is two-fold:

1. By using an E-DRAM cell it is designed using a “logic” process at fabrication facilities like TSMC. By using a logic vs. a DRAM process the cell is slightly larger than a pure DRAM cell would be but the process, as it is named, also allows for integration of large amounts of Logic to surround the memory array, in the case where additional embedded functionality is desired.

2. By using this version of the process, it is also possible to design the array with the desired characteristics to enable very fast access.

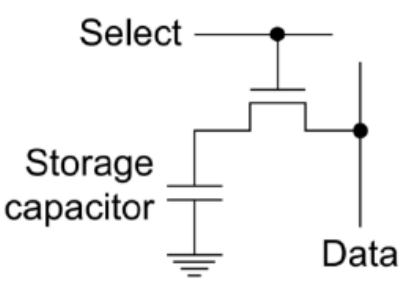
What is meant by this is that to achieve the desired speed (which in our case is close to SRAM speed) it is necessary to keep bit lines and word lines at a reasonable length. By doing this, the capacitance of these metal lines is kept to a minimum which in turn, defines the size of the line drivers and sense amps that will be needed to drive these lines. In the case of the MoSys devices, we designed the array with lines that had only 144 bits per line vs. the approx. 2000 bits per line that is standard in normal DRAM devices. This has the impact of reducing the load that both needs to be driven by line drivers and by the cells themselves when they are activated.

Reduced driver size. When the line drivers do not have to drive a large capacitive load the sizing of the drivers can be reduced. This has a positive effect in that the drivers can be smaller, but it has a slightly negative effect in that you need more of them to because they are driving smaller portions of the array.

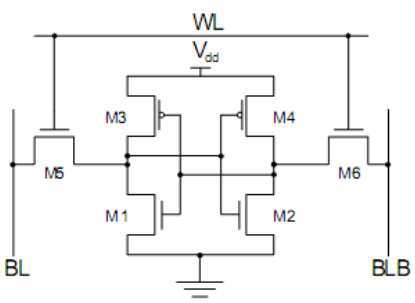
Power is also impacted. When reviewing the power impact of utilizing E-DRAM any one access will require less power than an access of a larger number of bits. The equation of $P=CV^2F$ is directly applicable. *Since the C factor is reduced on any one access, by the need to drive shorter bit and word lines, the resultant power dissipation is reduced.*

Power vs speed. As mentioned in the previous paragraph, the individual power of a single access is reduced, by reduction of the capacitive loads that need to be driven. By reducing the capacitance, the resulting speed is increased by being able to drive and recover the lines faster. This allows the array to run faster, which results in a higher bandwidth device. The impact of this is a slightly higher power dissipation in that the same $P=CV^2F$ equation now increases because the F (frequency) factor is increased. *The result is still a reduction in power over an equivalent density of QDR devices.*

Cell Size. An E-DRAM cell in 40nm TSMC process each takes approx. $0.242um^2$. Whereas a single SRAM cell in a comparable technology takes approximately $0.370um^2$, which is approximately 53% larger than the E-DRAM cell. When one places 576 M of these cells on a die it has a very large impact on overall die size. (This is the major reason why even with newer technologies, the size of an SRAM cell will be the limiting factor in the density of the available arrays. The densest SRAM is 288Mb and DRAM are in the Gb and higher density).



EDRAM Cell 0.242um



SRAM Cell $0.370um^2$

The result of using this cell structure to be the basis of the MoSys Quazar MSQ220 (0.5Gb) and MSQ230 (1.1Gb) devices, is that one can achieve the density of 2X to 8X that of available SRAM devices at a comparable technology node. *The device can run at comparable system speeds and achieve a lower power consumption than an equivalent SRAM and still be competitively priced.*

Bank Architecture

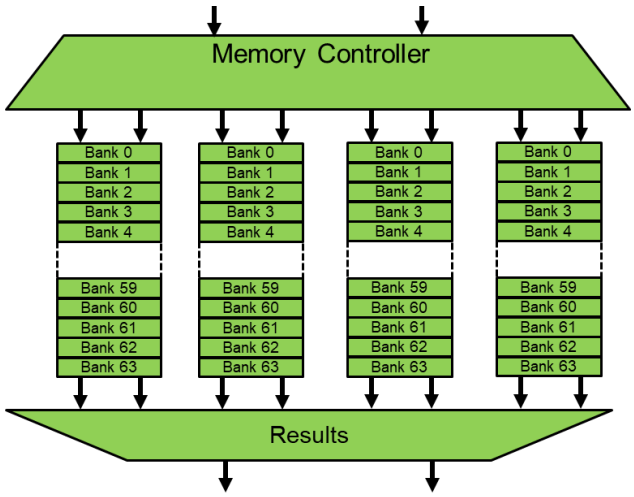
After settling on the cell design, and realizing the advantages of the E-DRAM cell, the next level of hierarchy, is to decide if the array would be best served if it were a flat unified structure like an SRAM or if a benefit can be gained by subdividing the array into smaller elements. When looking at a flat array, it is important to keep in mind that the cell is still a capacitive storage element and will require refresh on occasion. The issue with refresh is, what if any, impact it has on system throughput and performance.

In virtually all systems with DRAM, the need to refresh a location or array is given a higher priority than performing a memory access. This is because if a cell is not refreshed or accessed in a specified amount of time it is possible that it will lose the content and it is not guaranteed that the user will have any indication that the state of the cell is potentially corrupted, and data is lost.

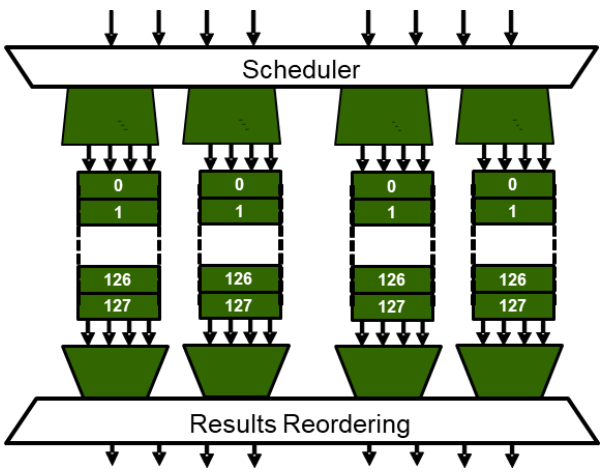
In the Quazar family of devices, the act of refreshing the array is a function that is handled internally. Again, the benefit of building the array in a logic process is that the device can support the counters and support logic needed to implement this required refresh function internally and the user does not need to be concerned with this logic or its internal timing requirements.

This is another reason why separating the memory into banks is beneficial, since the internal refresh logic is designed to take advantage of the fact that if an access is being made to one section or bank of the memory the logic will perform a refresh on the bank(s) that are not being accessed. The conclusion that was reached is that if the memory is truly being accessed in any sort of random fashion, and it is subdivided down into “banks” of 32K x 72 arrays which results in 256 banks in the MSQ220 device and 576 banks in the MSQ230 device there will be an incredibly small possibility that a collision between a refresh and an access will ever occur. It is also understood that although the odds are small it is not zero. So, if it does happen the GCI protocol is designed to allow the refresh to complete and then resume normal transactions. This “back-pressure” is handled by a “ready” signal that informs the host if the QPR is capable of accepting new requests.

The following is a figure depicting the memory bank structure in the MSQ220 device.



The following is a figure depicting the memory bank structure in the MSQ230 device.



Partition Architecture

The next level of hierarchy is also visible in the previous figure. The memory organization has been designed to provide yet another level of functional division. In looking at the above figure, it is seen that the memory has been divided into “Banks” and the banks are grouped into blocks of 64 banks on the 576b device and 126 on the 1Gb device. A group of banks is defined as a “Partition”. In the above picture the memory has 4 partitions that each has 64 or 128 banks of memory. In the smaller device, the MSQ220 device each bank is 32K x 72 and there are 64 per partition so each partition is 2M x 72. In the MSQ230 device each bank is also 32k x 72 and there are 128 per partition so each partition is 4M x 72.

When accessing these partitions, the internal memory controller will rotate through the four partitions rotation. The rotation time is designed to align with the FPGA Cycle time in a TDM fashion. This resulted in a Trc of approx. 3ns which aligns with the FPGA cycle time of approx. 3 ns. The access will start at partition 0 then 1, 2, 3, and back to partition 0. This is a fixed access rotation which aligns

with a 3.2ns clock. This is exactly what the Trc is for the 12.5Gbps MSQ220.

Device	Density	SerDes Rate	Trc	FPGA Clk Rate
MSQ220	576 Mb	10.3125Gbps	3.8 ns	257.8MHz
MSQ220	576 Mb	12.5Gbps	3.2ns	312.5MHz
MSQ230	1.1Gb	12.5Gbps	3.2ns	312.5MHz
MSQ230	1.1Gb	15.6Gbps	2.6ns	390MHz

The above table relates the rate at which the FPGA clock rate runs compared to the tRC required for the MSQ devices need to run. `

Due to the unique Quad Partition structure of the MSQ devices, and the fact that during each FPGA clock cycle the MSQ device will access all 4 partitions in a TDM fashion, results in a requirement that each partition must complete the memory access in the specified Trc associated with the FPGA clock rate and within that clock time an access will happen in each of the 4 partitions. This results in 4 independent access in one FPGA clock cycle.

This unique partitioning and round robin access pattern allows for the flexibility and increased Bandwidth over SRAM like QDR. What the MSQ devices can be considered as a device that combines 4 independent QDR devices into a single package, while maintaining independent access to each of the 4 devices. This then allows a user to initiate 4 parallel reads (per GCI port) to 4 independent locations all within one FPGA clock cycle. The return will be 4 separate 72 words, again in one FPGA clock cycle. The MSQ220 is a fully synchronous device and has a fixed pipeline and latency. This is ideal for applications such as table lookups that require a known request to data latency.

In effect, since every partition can be accessing a different location, and all accesses happen in the same Trc cycle, the device operates like 4 independent random-access SRAM.

In reviewing the above diagram, it is also seen that each partition has two read ports and two write ports. What this allows is an additional source of bandwidth. When a controller is designed to use one GCI port it is possible that through the port one read, and one write can be issued per partition per clock cycle. This allows the issuance of as many as 4 reads and 4 writes per FPGA cycle. Since

the MSQ220 and MSQ230 devices have two ports each this enable twice the throughput as a single port and allows two reads and two writes to be issues from the two ports (1 from each) and the memory core is designed to support this throughput. Again, as long as all 4 access requests are to separate banks within each partition.

If we look at what this partitioning and banking allows, the result is 2 Reads and 2 Writes per partition per access or a total of 8 read and 8 writes per access. A significant bandwidth increase over SRAM like QDR. If we take this even a step further in the MSQ230 device the bandwidth is again doubled to allow up to 4 reads and 4 writes per partition per FPGA cycle. This allows for a total of 16 Reads and 16 Writes per FPGA cycle.

Quad Partition

The next innovation in the MoSys memory architecture is the division of the memory array into Quad Partitions. A partition in the case is 2M x 72 (144Mb) in the case of the MSQ220 (the equivalent of 1 QDR-144Mb), or 4M x72 (288Mb) in the case of the MSQ230 (the equivalent of 2 QDR-144Mb). In total memory capacity, since the MoSys device has 4 partitions, the MSQ220 is equivalent to 4 QDR devices in one package and the MSQ230 is equivalent to 8 QDR devices in one package.

In addition to the density, the bus structure allows each partition to be accessed as a fully independent memory structure or as part of a unified memory. This enables the user to access 4 independent 72-bit words, one in each of the 4 partitions, from each of the GCI ports. That equates to 288 bits of data from each of the ports or potentially a total of 576 bits of data during each of the associated FPGA clock cycles. As with all previous access patterns it remains important that each of the accesses into any one partition be to a separate bank within the partition. This is because even though a partition can support multiple reads and multiple writes (simultaneously) any individual bank is limited to a single access at a time. (Banks are single ported, limiting it to a read or write per cycle.)

Internal Bus Structure

To understand how this much bandwidth is supported with a MoSys devices it is helpful to understand the bussing structure(s) within the device. As has been mentioned throughout the paper, starting with the I/Os. The I/Os that are used with the MSQ devices are SerDes based. This was done with the intent of being

high throughput, Low latency (using a low latency protocol), supported by an industry approved (SerDes) roadmap (10Gbps through 112Gbps specifications already exist), proven reliable, low pin count (as few as four SerDes lanes and a maximum of 16 SerDes lanes per device), ease of routing, and availability of many lanes on both FPGAs and ASICs. Allowing for a wide variety of options on how to interface with a MoSys device.

Once data is received at the I/O the internal bus structures are independent Address and Data buses to each of the 4 partitions and for a MSQ220 there are two read busses and 2 write busses per partition so that each of the busses support full bandwidth access to separate banks within a partition. Within the device there are 8 independent read busses and 8 independent write busses, running at full bandwidth to supply simultaneous access to 16 separate banks of memory within one clock cycle. This is doubled in the MSQ230 device where 32 separate and simultaneous accesses can take place within the device. This is approx. 0.75Tbs of internal bandwidth.

Conclusion

The unique architecture of the MoSys Multi-Partition-Rate devices offers users SRAM like access to a next generation memory. This device offers bandwidth, speed, and availability, with a solid roadmap to increased functionality.

These devices are ideal for replacement and enhancement of present day QDR devices. MoSys does not only support the device but in addition it offers sample RTL for the controllers that have been proven functionality in most families of FPGAs.

The QPR architecture is extremely flexible and have many performance modes of operating. Due to MoSys' 1T memory cell technology, they are the lowest cost SRAM at this high capacity and performance. The QUAZAR QPR4 and QPR8 are the base products. Higher functionality can be found in the BLAZAR family of Accelerator engines.

In-Memory	Part Number	Description	Package	Interface					Memory		Access Rate		In-Memory Functions		
			Pkg Size	Lanes	Rate per Lane Gb/s				BW MAX.	tRC	Size	Billion Transactions per second	R/W	BURST for Data Movement	RMW / ALU for Compute and Decision	Custom & User Functions with 32 RISC
			mm	Tx/Rx	10.3	12.5	15.6	25	Gb	ns	Gb					
QPR4	MSQ220	QPR4 (Quad Partition Rate) 0.5 Gb	FCBGA 19X19	16	✓	✓			320	3.2	0.5	2.5	✓			
QPR8	MSQ230	QPR8 (Quad Partition Rate) 1Gb	FCBGA 27X27	16			✓	✓	640	2.7	1	5	✓			
BURST	MSR622	Bandwidth Engine 2 Burst Serial 0.5Gb High Access Memory	FCBGA 19x19	16	✓	✓			320	3.2	0.5	3.3	✓	✓		
BURST	MSR630	Bandwidth Engine 3 Burst Serial 1Gb High Access Memory	FCBGA 27x27	16		✓	✓	✓	640	2.7	1	6.5	✓	✓		
RMW	MSR820	Bandwidth Engine 2 RMW Serial 0.5Gb High Access Memory with ALU for RMW functions	FCBGA 19x19	16	✓	✓			320	3.2	0.5	3.3	✓	✓	✓	
RMW	MSR830	Bandwidth Engine 3 RMW Serial 1Gb High Access Memory with ALU for RMW functions	FCBGA 27x27	16		✓	✓	✓	640	2.7	1	6.5	✓	✓	✓	
Program	MSPS30	Programmable HyperSpeed Engine Serial Interface, 1Gb Memory, 32 RISC Processor cores for custom algorithms, compute, functions	FCBGA 27x27	16		✓	✓	✓	717	2.7	1	24 Internal	✓	✓	✓	✓
RTL	RTL-AE	RTL Memory Controller for Bandwidth Engine and Programmable HyperSpeed Engine. Manages memory and the serial interface signals. Presents a QDR like parallel RTL interface to the user.	FPGA RTL Code		✓	✓	✓	✓			576Mb & 1Gb	6.5	✓	✓	✓	

BW MAX. = Aggregate of all SerDes laneS at the highest serial interface speed

MoSys Applications Engineers are available to discuss in more detail how the Quazar QPR Memories or the Blazar Bandwidth Engine products can accelerate an FPGA application.

Applications also has experience in High Speed board design, layout and signal integrity. If you wish to discuss the issues of memory tradeoff or board layout, please contact:

MoSys Applications (Link)

MoSys Sales (Link)

Samples are available for Qualified Applications.