

Utilizing Virtex-4 FPGA-Immersed IP for Hardware Acceleration for RAID 6 Parity Generation/Data Recovery

FPGAs and immersed IP blocks increase data reliability and availability in storage systems.

by Matt DiPaolo
Sr. Product Applications Engineer
Xilinx, Inc.
matthew.dipaolo@xilinx.com

Steve Trynosky
Product Applications Manager
Xilinx, Inc.
steve.trynosky@xilinx.com

A redundant array of independent disks (RAID) is a hard disk drive (HDD) array where part of the physical storage capacity stores redundant information. This information can be used to regenerate data if there is a failure in one or more of the arrays' member disks (including a single lost sector of a disk), or in the access path to the member disk.

There are many different levels of RAID. The implemented RAID level depends on several factors, including overhead of reading and writing data, overhead of storing and maintaining parity, plus the mean time to data loss (MTDL). The most recent level is RAID 6, which can have two implementations (Reed-Solomon P+Q or Double Parity). RAID 6 is the first RAID level that allows the simultaneous loss of two disks, which improves its MTDL over RAID 5.

In this article, we'll summarize the Xilinx® reference design for RAID 6, illustrating advantages of immersed hard-IP blocks of the Virtex™-4 architecture, including block select RAMs, distributed memory, FIFO memory, digital clock managers (DCM), the PowerPC™ PPC405, and DSP48 blocks. Our hardware acceleration block supports Reed-Solomon RAID 6 and can support other RAID levels when coupled with appropriate firmware.

Why RAID 6?

In RAID levels such as RAID 5, when a disk failed system firmware would use the remaining disks to regenerate the data lost from the failed disk. If another disk failed before regeneration was complete, the data was lost forever, which becomes the point of MTDL. Up to now, because of the size and probability of disk failure, the MTDL of RAID 5 was acceptable.

With the rising popularity of inexpensive disks such as Serial ATA and Serial Attached SCSI (SAS), as well as larger capacity disks, the mean time between failures (MTBF) of a disk has increased dramatically. For example, 50 disks, each with 300 GB capacity, would have an MTBF of 5×10^5 hours (a 10-14 read error rate), resulting in an array failure in less than eight years. According to Intel's TechOnline, "High-Performance RAID 6: Why and How," RAID 6 improves this to 80,000 years. Achieving a large MTDL justifies the increased overhead of both disk space for additional parity data, additional reads and writes to disk drives, and the system complexity required for handling multiple disk failures.

Another aspect of RAID 6 is the method in which data and redundancy information is stored on multiple disks. Figure 1 shows an example of a five-disk system. Data and parity information is striped horizontally across the drives in blocks of data. Each block is typically a multiple of 512 bytes. To keep the parity drives from being a system bottleneck, as can occur in RAID 4, the parity information rotates around the drives, in integer increments of blocks of data. Note that the five-disk case has a 40% overhead for parity; larger disk arrays can reduce this to 16% in a 12-disk system.

Mathematics Behind the Technology

To understand Reed-Solomon RAID 6, you must have some familiarity with Galois Field (GF) mathematics. We will summarize the GF mathematics involved in RAID 6 implementation here, but for a more detailed description, please refer to:

- Xilinx XAPP731 (www.xilinx.com/bvdocs/appnotes/xapp731.pdf)
- Intelligent RAID 6: Theory, Overview and Implementation (www.intel.com/design/storage/papers/308122.htm)
- A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems (www.cs.utk.edu/~plank/plank/papers/CS-96-332.html)
- The Mathematics of RAID 6 (<http://kernel.org/pub/linux/kernel/people/hpa/raid6.pdf>)

GF mathematics define that any calculation continues to have the same number of bits as the two operands from which it was generated (that is, two 8-bit numbers using GF multiplication result in an 8-bit number). Here are the definitions of GF multiplication and division. Note that the

addition/subtraction is regular integer addition/subtraction.

$$2 \otimes 8 = gflog[gflog[2] + gflog[8]] = gflog[1 + 3] = gflog[4] = 0 \times 10$$

$$0 \times d \div 0 \times 11 = gflog[gflog[0 \times d] - gflog[0 \times 11]] = gflog[0 \times 68 - 0 \times 64] = gflog[4] = 0 \times 10$$

The reference design implements the gflog and gfilog values using the polynomial $x^8 + x^4 + x^3 + x^2 + 1$. Lookup tables, stored in block RAM inside the FPGA, are placed in the data path to provide fast access, enabling calculations at DDR memory speeds.

RAID 6 Parity (P & Q) Equations

To recover from two disk failures, RAID 6 stores two unique parity values, P and Q. This second parity creates a second equation to allow you to solve for two unknowns (or disk failure points).

The equations discussed next assume a RAID 6 disk array comprising three data disks and two parity disks for each block of data to simplify the discussion. These equations extend to 255 disks, including the two parity disks, which is the mathematical limit of the equations.

The first is the P parity, which is identical to RAID 5. A simple exclusive or (XOR)

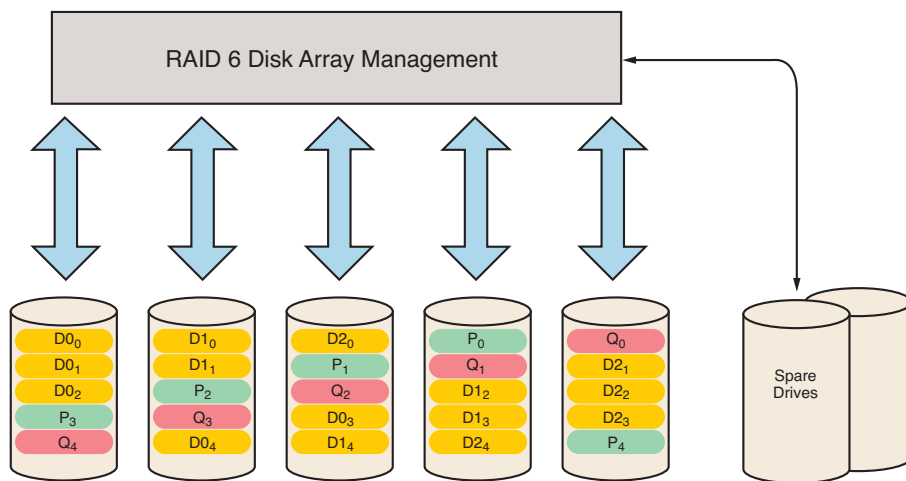


Figure 1 – RAID6 data block organization

The FPGA contains a SATA protocol controller that interfaces with the memory controller and PowerPC405.

function generates the parity block from the data values in the same sector horizontally across the data drives in an array group. In Figure 1, P_0 XORs the $D0_0$, $D1_0$, and $D2_0$.

$$P_N = D0_N \oplus D1_N \oplus D2_N \oplus \dots \oplus D(M-1)_N$$

- $N = 0$ to the maximum number of blocks (sectors) on the disk drive
- $M =$ number of data disks in the array group

The equation for the first sector of a three-data-drive system is:

$$P_0 = D0_0 \oplus D1_0 \oplus D2_0$$

In RAID 6, Q parity assigns a GF multiplier constant associated with each data disk drive. The constant applies only to the data blocks stripped horizontally across the array, not to each drive in the array. Each data block is GF multiplied by a constant, before adding to the data elements of the next data drive. The g constants are determined from the GFILOG table. If another drive was added to the array, then $g^3 = \text{gfilog}(3) = 0x8$.

$$Q_N = (g^0 \otimes D0_N) \oplus (g^1 \otimes D1_N) \oplus \dots \oplus (g^{(M-1)} \otimes D(M-1)_N)$$

Here is the equation for the third sector of a three data drive system:

$$Q_2 = (0x1 \otimes D0_2) \oplus (0x2 \otimes D1_2) \oplus (0x4 \otimes D2_2)$$

FPGA-Based Hardware Acceleration Solution

The Xilinx reference design for RAID 6 is based on a sample configuration. In this case, the system contains a RAID host controller on an ML405 demonstration board. This board contains a Virtex-4 FPGA along with DDR memory and Serial ATA (SATA) connectors attached to a port multiplier, which in turn connects to five SATA HDDs.

The FPGA contains a SATA protocol controller that interfaces with the memory controller and PowerPC405. It is possible to replace the SATA protocol controller

with Serial Attached SCSI (SAS), Fibre Channel (FC), or any other disk interface protocol depending on the overall system architecture.

The design concentrates on the hardware acceleration portion of a RAID 6 system (depicted in Figure 2). An embedded PowerPC405 block controls the RAID 6 hardware, setting up pointers to the data and parity blocks of memory as well as setting up the hardware. The reference design

doesn't include the protocol link between the memory and the HDDs, as the possibilities here are endless.

There is a microprocessor to run the RAID firmware and a RAID hardware accelerator block. This firmware also generates typical data that would come from some type of HDD connection. The PPC405 firmware creates example data blocks in the DDR memory to emulate disk sectors retrieved from a disk array

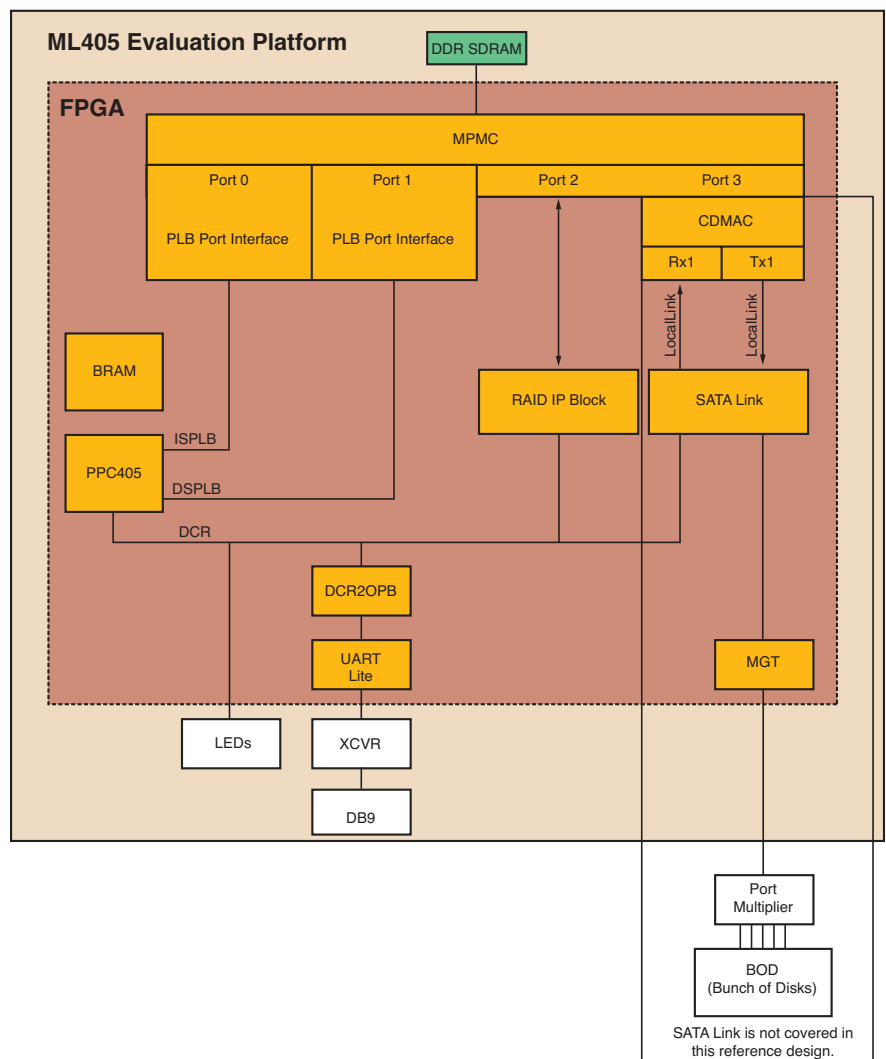


Figure 2 – Theoretical typical system

system. These data blocks are then used by the RAID accelerator to generate P and Q parity blocks.

A Xilinx multiple port memory controller (MPMC) controls the DDR memory. The MPMC provides multiple memory masters to access a shared memory. The MPMC connects to each of the PowerPC405 instruction and data-side processor local bus interfaces. Two other ports provide system-specific implementations for this reference design; one is used for the RAID hardware accelerator and the other is available for access to the hard disk. For additional information on the MPMC, refer to XAPP535, “High-Performance Multi-Port Memory Controller” (www.xilinx.com/bvdocs/appnotes/xapp535.pdf).

A RAID 6 hardware accelerator is mathematically intensive. The RAID 6 calculations require each block of data to be stored in a memory buffer and then read into a temporary data buffer while being XOR'ed or added to other data elements. There is a large amount of data manipulation that needs to be done. This manipulation is also time-intensive, but hardware implementations are faster than processor-only register calculations because hardware can provide parallel manipulation of data blocks, clock rate lookup table access, and multipliers that operate much faster than a processor alone. This tends to be a small portion of the overall period when including the seek time of the disks themselves.

The hardware accelerator comprises four main blocks:

- Data manipulation block (DMB) – this block performs the mathematical operations on one byte of data (depending on the system data width, more DMB logic can be added to support larger data widths). It can create parity and regenerate data of any case discussed in the RAID 6 equations.
- RAID FSM – the main logic of the reference design. It controls the MPMC_IF block along with all of the DMB signals, including block RAM address, read, write control, and MUX select lines.

- DCR FSM – communicates with the PowerPC405 device control register (DCR) bus.
- MPMC_IF – Controls the native interface of the MPMC block to transfer data. The MPMC is a four-port DDR memory controller time-shared with the processor instruction and data cache processor local bus (PLB) interfaces.

Firmware plays a major role in all RAID systems. However, the hardware changes little for the different RAID levels. The hardware accelerator can remain the same while the firmware changes to incorporate the different ways the data is organized on the HDD, and how the parity is generated.

The hardware accelerator can support other RAID levels beyond RAID 6 with minimal (if any) modifications. RAID Double Parity (DP), RAID 5, RAID 4, and RAID 3 are among the supportable RAID levels.

Conclusion

The Xilinx reference design for RAID 6 incorporates advantages of immersed hard-IP blocks of the Virtex-4 architecture, including block select RAMs, PowerPC PPC405, and DSP48 blocks in a hardware acceleration block that supports Reed-Solomon RAID 6 and can support other RAID levels when coupled with appropriate firmware.

The block RAMs are used for the GF mathematic look-up tables, the DSP48 blocks perform fast integer addition, and the PowerPC405 handles memory/address management, plus the potential to handle RAID-level data structures in HDD arrays. As in all systems, you must evaluate hardware and firmware trade-offs when architecting redundant systems.

For more information about the RAID 6 reference design, visit www.xilinx.com/bvdocs/appnotes/xapp731.pdf or e-mail storage@xilinx.com.

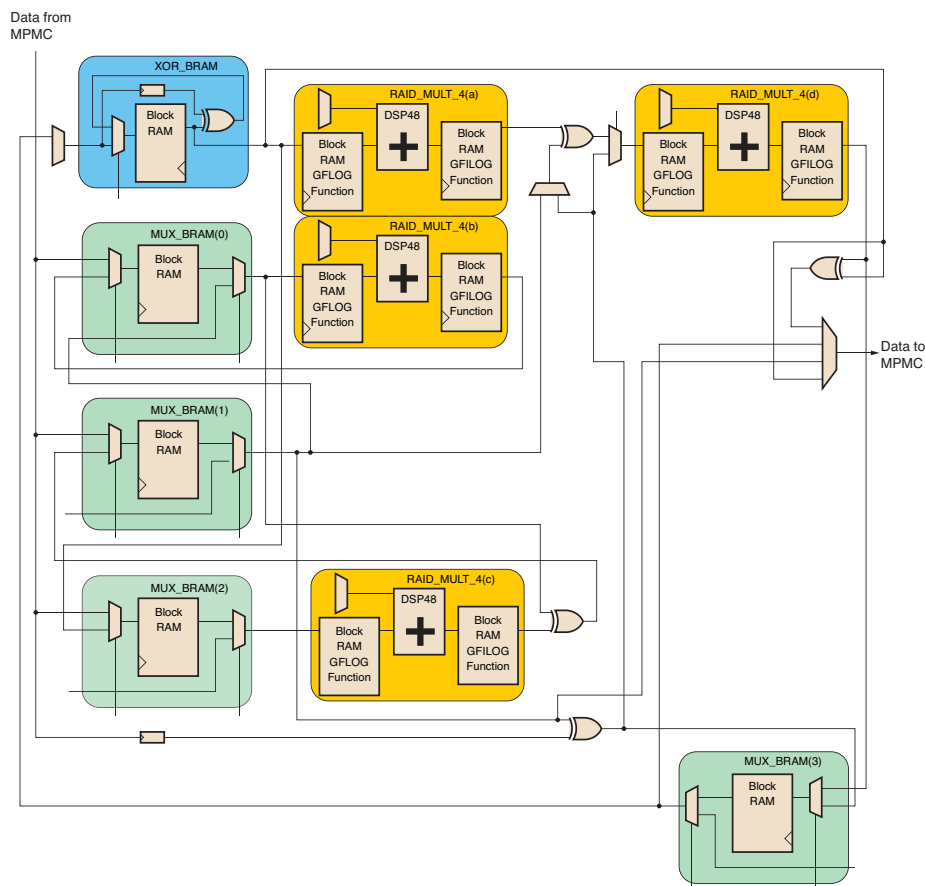


Figure 3 – Data manipulation block logic