Release Notes

SmartHBA 2100 and SmartRAID 3100 Software/Firmware

Released

January 2019

Revision History

Revision	Revision Date	Details of Change
17	January 2019	SR2.4 Production Release
16	June 2018	SR2.3 Production Release
15	June 2018	Updated for RC Release
14	October 2017	Update supported OSs
13	October 13, 2017	First Production Release
1-12	June 2016-July 2017	Pre-Production Releases.

Software License Agreement

PLEASE READ CAREFULLY: THE USE OF THIS SOFTWARE IS SUBJECT TO THE SOFTWARE LICENSE TERMS OF MICROSEMI, INC. AND OTHER LICENSORS WHOSE SOFTWARE MAY BE BUNDLED WITH THIS PRODUCT.

BY YOUR USE OF THE SOFTWARE INCLUDED WITH THIS PRODUCT YOU AGREE TO THE LICENSE TERMS REQUIRED BY THE LICENSOR OF THAT SOFTWARE, AS SET FORTH DURING THE INSTALLATION PROCESS. IF YOU DO NOT AGREE TO THE LICENSE TERMS APPLICABLE TO THE SOFTWARE, YOU MAY RETURN THE ENTIRE UNUSED PRODUCT FOR A FULL REFUND.

In return for acquiring a license to use the Microsemi software, which may include software from third party licensors and patches made available by Microsemi (“Software”), and the related documentation, you agree to the following terms and conditions:

1. License. This Agreement grants you, the Licensee, a license to:
   a. Use the Software on a single computer system, which is not intended for use by more than five (5) users; and:
   b. Make one copy of the Software in machine readable form solely for back-up purposes, provided you reproduce Microsemi's copyright proprietary legends. Notwithstanding the foregoing, the Software may be used on the home, laptop or other secondary computer of the principal user of the Software, and an additional copy of the Software may be made to support such use. As used in this license, the Software is “in use” when it is either loaded into RAM or installed on a hard disk or other permanent memory device. The Software may be “in use” on only one computer at any given time. (Different license terms and fees are applicable for networked or multiple user applications.) As a specific condition of this license, you agree to use the Software in compliance with all applicable laws, including copyright laws, and that you will not copy, transmit, perform or distribute any audio or other content using the Software without obtaining all necessary licenses or permissions from the owner of the content.
2. Restrictions. You may not distribute copies of the Software to others or electronically transfer the Software from one computer to another over a network. You may not post or otherwise make available the Software, or any portion thereof, in any form, on the Internet. You may not use the Software in a computer service business, including in time sharing applications. The Software contains trade secrets and, in order to protect them, you may not decompile, reverse engineer, disassemble, or otherwise reduce the Software to a human-perceivable form. YOU MAY NOT MODIFY, ADAPT, TRANSLATE, RENT, LEASE, LOAN, RESELL FOR PROFIT, DISTRIBUTE, NETWORK OR CREATE DERIVATIVE WORKS BASED UPON THE SOFTWARE OR ANY PART THEREOF.
3. Ownership of Software. As Licensee, you own the media upon which the software is recorded or fixed, but Microsemi and its licensors retain title and ownership of the Software recorded on the original media and all subsequent copies of the Software, regardless of the form or media in which or on which the original and other copies may exist. This license is not a sale of the Software or any copy.
4. Confidentiality. You agree to maintain the Software in confidence and that you will not disclose the Software to any third party without the express written consent of Microsemi. You further agree to take all reasonable precautions to preclude access of unauthorized persons to the Software.
5. Term. This license is effective until January 1, 2045, unless terminated earlier. You may terminate the license at any time by destroying the Software (including the related documentation) together with all copies or modifications in any form. Microsemi will have the right to terminate our license immediately if you fail to comply with any term or condition of this Agreement. Upon any termination, including termination by you, you must destroy the Software (including the related documentation), together with all copies or modifications in any form.
6. Special Terms Applicable to Databases. Where a database is included with the Software, you acknowledge that it is licensed only in connection with the use of the Software to perform disc creation, and that the

database and all data derived therefrom must be maintained in confidence in accordance with the provisions of Section 4. This license does not grant you any rights to distribute or disclose such database or data.

7. Limited Warranty. Microsemi and its Licensor warrant only that the media upon which the Software is furnished will be free from defects in material or workmanship under normal use and service for a period of thirty (30) days from the date of delivery to you. MICROSEMI AND ITS LICENSORS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE OR DOCUMENTATION. THE FOREGOING STATES THE SOLE AND EXCLUSIVE REMEDIES MICROSEMI AND ITS LICENSORS WILL PROVIDE FOR BREACH OF WARRANTY. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, MICROSEMI AND ITS LICENSORS MAKE NO WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED, AS TO NON-INFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow the exclusion of implied warranties or limitations on how long an implied warranty may last, so the above limitations may not apply to you. This warranty gives you specific legal rights and you may also have other rights which vary from state to state.
8. The entire liability of Microsemi and its licensors, and your exclusive remedy for a breach of this warranty, shall be:
    a. The replacement of any media not meeting the above limited warranty which is returned to Microsemi; or:
    b. if Microsemi or its distributor is unable to deliver replacement media which is free from defects in materials or workmanship, you may terminate this Agreement by returning the Software and your money will be refunded.
9. Limitation of Liability. IN NO EVENT WILL MICROSEMI OR ITS LICENSORS BE LIABLE TO YOU FOR ANY INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR LOSS OF DATA, EVEN IF MICROSEMI OR A LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY. Some states do not allow the exclusion or limitation of special, incidental, or consequential damages, so the above limitation or exclusion may not apply to you.
10. Export. You acknowledge that the laws and regulations of the United States and other countries may restrict the export and re-export of the Software. You agree that you will not export or re-export the Software or documentation in any form in violation of applicable United States and foreign law.
11. Government Restricted Rights. The Software is subject to restricted rights as follows. If the Software is acquired under the terms of a GSA contract: use, reproduction or disclosure is subject to the restrictions set forth in the applicable ADP Schedule contract. If the Software is acquired under the terms of a DoD or civilian agency contract, use, duplication or disclosure by the Government is subject to the restrictions of this Agreement in accordance with 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors and 49 C.F.R. 227.7202-1 of the DoD FAR Supplement and its successors.
12. General. You acknowledge that you have read this Agreement, understand it, and that by using the Software you agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the agreement between Microsemi and you, and supersedes any proposal or prior agreement, oral or written, and any other communication between Microsemi and you relating to the subject matter of this Agreement. No additional or any different terms will be enforceable against Microsemi unless Microsemi gives its express consent, including an express waiver of the terms of this Agreement, in writing signed by an officer of Microsemi. You assume full responsibility for the use of the Software and agree to use the Software legally and responsibly. This Agreement shall be governed by California law, except as to copyright matters, which are covered by Federal law. This Agreement is deemed entered into at Sunnyvale, California by both parties. Should any provision of this Agreement be declared unenforceable in any jurisdiction, then such provision shall be deemed severable from this Agreement and shall not affect the remainder hereof. All rights in the Software not specifically granted in this Agreement are reserved by Microsemi.

Should you have any questions concerning this license, contact:

Microsemi Corporation
Corporate Headquarters
One Enterprise
Aliso Viejo, CA 92656 USA

Contents

1 About This Release

The development release described in this document includes firmware, OS drivers, tools, and host management software for the SmartHBA 2100 controller solutions from Microsemi.

1.1 Release Identification

The firmware, software, and driver versions for this release are shown in *Table 1 • Release Summary* on page 1

Table 1 • Release Summary

Solutions Release	2.4
Package Release Date	January 9, 2019
Firmware Version	1.98 B0[1,2] (basecode 06.04.031.000)
UEFI Version	1.3.5.2
Legacy BIOS	1.3.5.2
Driver Versions	Windows SmartPQI: • Windows 2012/2016/2019: 106.84.2.64 • Windows 2008: 6.84.2.64 Linux SmartPQI: • RHEL 6/RHEL 7/RHEL 7.5/SLES 11/SLES 12/SLES 15: 1.2.4-065 • Ubuntu 16/18: 1.2.4-065 • CentOS 7.5: 1.2.4-065 • Debian 8/9: 1.2.4-065 VMware SmartPQI: • vSphere 6: 1.0.3.2035 • VMware 6.7: 1.0.3.2035 FreeBSD/Solaris SmartPQI: • FreeBSD 10.2/10.3/11: 1.0.3.2035 • Solaris 11: 1.0.3.2035
arcconf/Maxview	B23484

Note:

1. Downgrading to 1.04 B0 or older builds from this release or prior 1.29 releases may cause the board to not boot or have supercap errors due to an incompatibility in SEEPROMs between this release and prior releases. Please refer to the section "*Updating the Board Firmware*" to downgrade an existing board.
2. If the firmware running on the board is older than 0.01 B594, existing data in the logical volumes must be backed up if it needs to be used after the upgrade. After the upgrade from firmware prior to 0.01 B594, the logical volumes will need to be recreated.
3. Only run the driver on firmware 0.01 build 500 or later.

1.2 Components and Documents Included in this Release

Download the firmware, drivers, host management software, and supporting documentation for your SmartHBA 2100 controller and SmartRAID 3100 controller solutions from the Microsemi Web site at *https://start.microsemi.com*

1.3 Files Included in this Release

This release consists of the files listed in the following tables:

Firmware Files

Table 2 • Firmware Files

Component	Description	Pre-Assembly Use	Post-Assembly Use
SmartFWx100.bin	Programmable NOR Flash File Use to program NOR Flash for boards that are already running firmware.		X

Table 3 • Firmware Programming Tools

Tool	Description	Executable
Arcconf romupdate	The command allows to upgrade/downgrade the firmware and BIOS image to the controller.	Refer to *Table 7 • Host Management Utilities* on page 4

Driver Files

Table 4 • Windows Storport Miniport SmartPQI Drivers

Package	Drivers	Binary	Version
2012	Server 2019 Server 2016 and Windows 10 Server 2012, R2 and Windows 8.1, 8	SmartPqi.sys	x64
		SmartPqi.inf	x64
		Smartpqi.cat	x64
2008	Server 2008, 2008 R2 and Windows 7	SmartPqi.sys	x64
		SmartPqi.inf	x64
		SmartPqi.cat	x64

Table 5 • Linux SmartPQI Drivers

Drivers	Version
Red Hat Enterprise Linux/CentOS 7.6, 7.5[1], 7.4[1], 7.3, 7.2, 7.1, 7.0	x64
Red Hat Enterprise Linux/CentOS 6.10, 6.9[1], 6.8, 6.7, 6.6	x64
SuSE Linux Enterprise Server 12[1], SP4, SP3, SP2, SP1, and Base	x64
SuSE Linux Enterprise Server 15[1]	x64
Oracle Linux 7.5, 7.4, 7.3, 7.2	x64
Oracle Linux 7.2 with UEK3 (3.8.13-*)	x64
Oracle Linux 7.3 with UEK4 (4.1.12-61)	x64

Drivers	Version
Oracle Linux 7.4 with UEK4 (4.1.12-94)	x64
Oracle Linux 7.5 with UEK4 (4.1.12-112)	x64
Ubuntu 18.04 LTS, 18.04.1	x64
Ubuntu 16.04.5, 16.04.4, 16.04.3, 16.04.2, 16.04.1, 16.04.0	x64
Debian 9.3	x64
Debian 8.10, 8.9	x64
Citrix xenServer 7.1	x64

Note: To mitigate against the Spectre Variant 2 vulnerability, the RHEL 6u9/RHEL7u4/RHEL7u5 and SLES11 SP3 and higher drivers have been compiled to avoid the usage of indirect jumps. This method is known as "Retpoline".

Table 6 • FreeBSD, Solaris, and VMware SmartPQI Drivers

Drivers	Version
FreeBSD 11.2, 10.4, 10.3	x64
Solaris 11.3	x64
VMware 6.0, 6.5, 6.7	x64

Host Management Software

Table 7 • Host Management Utilities

Description	OS	Executable
ARCCONF Command Line Utility	Windows x64 Linux x64 VMware EXSi 5.5/6.0 XenServer FreeBSD x64 Solaris x86	See the Arcconf download package for the OS-applicable installation executable.
ARCCONF for UEFI		Included as part of the firmware downloadable image.
maxView Storage Manager	Windows x64 Linux x64 VMware EXSi 5.5/6.0 XenServer	See the maxView Storage Manager download package for the OS-applicable installation executable.
maxView vSphere Plugin	vCenter 5.5 and 6.0	See the VMware maxView Storage Manager download package for the OS-applicable installation executable.
Boot USB (offline or pre-boot) for ARCCONF and maxView Storage Manager	Linux x64	See the maxView BootUSB download package for the .iso file.

2 What is New?

2.1 Features

Table 8 • Feature Summary on page 5 lists features supported for this release. Features new to this release are designated as "New".

Table 8 • Feature Summary

Feature		Supported in this Release	Future Release
UEFI Driver, Boot Support		X	
Legacy Boot Support		X	
Dynamic Power Management		X	
SMR Drive Support	Enumeration, Unrestricted Command Flow-Through	X	
	SATL Translation for HA/HM SMR Management	X	
	Identify All Drive Types	X	
Driver Support	Windows	X	
	Linux	X	
	VMware	X	
	FreeBSD	X	
	Solaris	X	
	OS certification	X	
Flash Support		X	
Configurable Big Block Cache Bypass		X	
Green Backup Support for SmartRAID		X	
4Kn Support in RAID		X	

2.2 Fixes

2.2.1 Firmware Fixes

2.2.1.1 Fixes and Enhancements for Firmware Release 1.98 B0

This release includes the following fixes and enhancements:

- Added an enhancement of the '100% read cache upon backup power source failure' feature to include offline backup power source failure cases.
- Fixed an issue where controller hangs when running I/O with pending DDR cache retries.
  - Root Cause: When there are many DDR cache retry internal requests queued, there is a corner case where the thread which is responsible for handling the host IOs thinks that there are coalescing opportunities and will keep on accumulating all host requests without submitting it to be executed to the drives. At one point, firmware runs out of resources and incoming IO's are stuck resulting in a controller hang.
  - Fix: The check for deciding whether firmware needs to continue parsing incoming requests without submitting should be done only if we have coalescing enabled.
  - Risk: High
- Fixed controller lockup 0x1A91 during active path failure.
  - Root Cause: During path failure, when scanning the expander indexes to determine if the phys are up or not, the phys that are up are not validated properly causing the lockup.
  - Fix: When validating if the phys are up or not, qualify that the PHY is truly a child of the parent PHY.
  - Risk: Medium
- Fixed an issue when a cable is removed from a wide port configuration that causes the host tools to display two connectors.
  - Root Cause: Internal firmware counter increased for the PHY more than once during PHY link up, causing the issue.
  - Fix: For each PHY link up, firmware only increases the internal counter once.
  - Risk: Low
- Fixed a controller hang issue due to deadlock between I/O processing and MaxCache flush process that was triggered by a transient error returned by the physical drives.
  - Root Cause: When MaxCache is present in the system, a transient device error such as time out can trigger a dead lock condition between the MaxCache flush thread and the host I/O thread, resulting in the controller hang condition with several commands outstanding. This is because the host I/O processing thread triggers the MaxCache flush, which is waiting for each other to complete.
  - Fix: Route MaxCache flush I/O retries to avoid a deadlock condition in such cases.
  - Risk: Medium
- Fixed an issue where all logical drive status in controller changes to "Not available", when one of the logical drives is experiencing a loose cable condition because drives are missing during the reboot.
  - Root Cause: The firmware was preventing the logical drive from transitioning to the failed state, when it detects a loose cable condition between the controller and the drives.
  - Fix: Allow the logical volume in a loose cable state to go to a failed state, if there is no dirty data in DDR cache or no transformation in progress. All other logical volumes will be available to the OS.
  - Risk: Low
- Fixed an issue where the controller cache contents were backed up unnecessarily, when there is no dirty cache data to backup.
  - Root Cause: During controller run time, if a backup power source error occurs, the backup power source should be turned off after completing the cache flush. However, if the backup power source is not turned off and an unsafe shutdown occurs, an unwanted backup occurs causing unnecessary back up events and the cache gets disabled.
  - Fix: Check for power source errors and if no dirty cache then disable the backup power source. This will avoid the false backup and its events.
  - Risk: Low

- Fixed an issue where 0x1E30 lockup occurs after enclosure cable restore.
  - Root Cause: Fan-out enclosure detection logic during hot plug was trying to allocate and free up a buffer from a critical section which was due to incorrect handling of correcting the box ID for child and parent expanders.
  - Fix: Fan-out enclosure detection logic during hot-plug is changed to handle this situation better. If an expander has the same box ID as its parent, then they are both in a fan-out enclosure. On hot plug if it's discovering a hub-expander, it should look for a mirror expander that's in fan-out enclosure. If it doesn't find one then it should wait. When the children are discovered, it will get marked as being in a fan-out enclosure.
  - Risk: Low
- Fixed a 0x65016 lockup after host transport change to SIS mode when there are outstanding IOs in PQI mode.
  - Root Cause: When there are outstanding IOs in PQI host transport and OS driver is initiating a request to change to synchronous host transport, firmware hits this lockup while trying to complete the pending IOs.
  - Fix: When the OS driver sends a PQI RESET, firmware will check for outstanding IOs on RAID path and if any, then reset the corresponding drive to force the completions back. In case of IOBypass, reset all the drives.
  - Risk: Low
- Fixed an issue where enabling MaxCache causes cache and data logical drive to be marked as failed/degraded.
  - Root Cause: Certain SSDs do not handle large single transfer IO. This causes failure when snapshot meta-data read / write occurs to MaxCache Unit.
  - Fix: For snapshot meta-data of MaxCache Unit, limit the block size of IO to be 512 KiB. 512 KiB is the typical strip size in logical devices and known to be a reliable transfer size.
  - Risk: Low
- Fixed an issue where firmware repeatedly fixes URE at the same LBA.
  - Root Cause: During URE recovery, while performing a read retry the LBA is truncated to 32 bits incorrectly. So, the read succeeds as it was reading a wrong LBA and firmware ends up in fixing the wrong LBA over and over.
  - Fix: Use the right data type to avoid the truncation.
  - Risk: Low
- Fixed a 0x1E30 lockup on OS shutdown, when an expansion is in progress.
  - Root Cause: On OS shutdown event, expansion progress is being flushed to drives and DDR cache gets re-enabled for the host IO. Now, incoming host requests use cache and firmware sets an internal flag on those requests to determine whether it should serve the IOs using “yet-to-expand” region or already expanded region of the logical drive. If the host request is queued for cache retry without clearing this internal flag as it assumes expansion is not running because requests are being processed through the cache. While executing the request from cache retry logic, lockup is observed because firmware is trying to set this internal flag which is already set.
  - Fix: Do not enable the DDR cache for host IOs until the expansion is completed or failed.
  - Risk: Medium
- Fixed a problem where controller gets stuck after running IO.
  - Root Cause: Firmware gets into an infinite while loop between a gate-keeping check for coalescing opportunities and a condition that decides to submit the accumulated commands since the memory resources are going to be exhausted soon.
  - Fix: Coalescing gate-keeping check should account for the memory resources appropriately and submit the requests accumulated instead of waiting for more opportunities.
  - Risk: Medium

- Fixed a 0xFFFFF001 Lockup while RAID 1 IO is running.
  - Root Cause: For buffered write in RAID 1, firmware is populating more scatter-gather entries than supported, which is 64. This results in the lockup when we try to deference the invalid memory.
  - Fix: Enable buffered writes for RAID 1 only when scatter-gather count is less than or equal to supported.
  - Risk: High
- Fixed an issue that causes Windows to lose drive slot number information.
  - Root Cause: Controller firmware response to Inquiry for VPD page 83h was modified to return the WWN of device. However, some applications use SAS address rather than WWN to communicate with the device and non-existance of the drive's SAS address in the Inquiry response can cause failure of application to get correct slot number.
  - Fix: Update SAS address in the second descriptor of VPD page 83 instead of WWN so that both WWN and SAS address is presented to the host.
- Fixed a corner case issue found in wide port Loss Of Sync (LOS) testing where Identify Device (IDD) was getting sent to a drive erroneously.
  - Root Cause: When the controller experiences a LOS event (surprise SAS PHY down) on a wide port, the link between controller and expander goes away but the link between expander and the drive is not impacted. This means the IOs out to the drive stay at the drive and are not flushed. In this scenario, the controller must not send IDD when the wide port comes back up because some NCQ IOs are already outstanding to the drive and IDD is a non-NCQ command. These commands can not be mixed.
  - Fix: Only send internal IDD when the drive/link recovers if there are no IOs out to the drive and the drive is in a state that it can process the IDD.
- Fixed an issue where direct attached tape device's bay count was incorrectly appearing as 255.
  - Root Cause: During boot, firmware defaults all direct attached drive bay numbers to PHY + 1 which will also cover the tape drives. When a change was added to perform this for drives and for the devices which have not been assigned a box yet, it impacted the tape drive's bay assignment.
  - Fix: Set the bay number for tape devices accordingly and on hot-plug move the check for unknown box and not in enclosure for all devices.
  - Risk: Low
- Fixed a controller lockup issue while updating the firmware when another update is already in progress.
  - Root Cause: Incorrect address was being freed up.
  - Fix: Firmware is changed to pass the correct buffer address while freeing up the allocated memory.
  - Risk: Low
- Fixed a controller lockup problem during boot due to invalid memory access while processing errors.
  - Root Cause: Memory resources for the thread that process error responses are allocated in another thread, which run in parallel during boot sequence. Hence, the necessary resources are not allocated yet, but a drive returns failure during boot time when trying to read if it was part of a logical drive or not. While handling this failure, a NULL pointer access exception is seen.
  - Fix: Firmware is changed to allocate the memory resources required for the error processing thread just before creating the thread, which makes sure that it won't process any error until all necessary resources are initialized.
  - Risk: Low
- Fixed a controller assertion lockup 0x1E30 problem when the cache ratio is changed.
  - Root Cause: Firmware locksup when an unexpected thread calls the cache shutdown handling code when cache ratio changes are made.
  - Fix: Firmware is changed so that only the expected thread calls the cache shut down handling code while processing host IOs.
  - Risk: Low

- Fixed an issue where rebuilding is not getting started when predictive spare rebuild is triggered on MaxCache volume.
  - Root Cause: When activating the spare, firmware stalls the thread that dispatches the queued MaxCache I/O requests. Later when swapping the data and spare drive, firmware also expects the mentioned thread to process these requests while quiescing the host IOs which causes controller to be in hung state.
  - Fix: Firmware is now changed to avoid the stalling of the thread which dispatches MaxCache IO requests while activating the spare and disables MaxCache flush requests after quiescing host IOs will make sure that no other MaxCache flush requests will come in between swapping of spare and data drive.
  - Risk: Medium
- Fixed an issue where status LED is not blinking on volume transformation after reboot.
  - Root Cause: Volume transformation such as expansion/rekeying is suspended if the power source state is charging and in this state, firmware does not blink the drive status LED
  - Fix: when the suspended volume transformation is clear from the power source completely charged, update the drive status LED as required.
  - Risk: Low
- Fixed an issue where server is not booting after backup power source is removed while volume rekey is in progress
  - Root Cause: When controller's backup power source is hot removed while rekeying is in progress, the meta data on the disks is not updated and the rekey progress is lost.
  - Fix: Firmware is changed to unconditionally save the expansion/rekey progress on the drives metadata irrespective of the power source state changes.
  - Risk: Low
- Fixed an issue where firmware is accepting SSD to be part of HDD volumes.
  - Root Cause: Firmware had a logic to update the ‘drive type mix allowance’ if SSD was present which is allowing an SSD to be part of HDD volumes during rebuild.
  - Fix: As the logical drive should be made of the same type of physical drives, there is no need for the logic for fixing the drive type mix allowance.
  - Risk: Low
- Fixed an issue where firmware is not changing the PHY rate settings when controller goes into survival mode
  - Root Cause: While separating the power modes and survival modes, code was rearranged so that PHY settings were not set on over temperature condition. Hence, the PHY rate settings never changed.
  - Fix: Populate the PHY rate settings accordingly during over temperature condition.
  - Risk: Low
- Fixed a data corruption when a SATA drive overruns a data transfer.
  - Root Cause: A data miscompare is detected when certain SATA drives (Seagate EXOS NL 7200.3) are used for URE testing with RAID volumes. It was discovered that certain SATA drives would return a large amount of garbage data when it returns the data for a read operation to a block with a URE.
  - Fix: Any SATA command that triggers an overrun will be failed back to the raid stack firmware. This change will fail any physical request without retrying or sending to error task to prevent further overruns. In addition, any SATA overrun will now trigger a device reset to prevent data corruption. All outstanding request to the drive will now get returned to the firmware with appropriate status. This status just requeues the requests to the drive without printing any errors.
  - Risk: Medium
- Incorrect POST messages are observed when a volume under expansion is failed.

- Root Cause: In cases where logical drives were going through expansion task and physical drives were removed offline, though firmware had information about both old and new logical drive configuration, it was considering only the new logical drive configuration to clear the abnormal volume state. This lead to firmware changing the volume state to expanding for the new logical drive even though the old logical drive is in failed state and hence caused incorrect POST messages.
- Fix: Firmware is changed to check both the old and new logical drive configuration state of the expanding volume and stay in abnormal volume state if any of the configuration is in FAILED state.
- Risk: Medium

- Drives are not being listed under enclosure in host management tools when using fan-out enclosures.
  - Root Cause: Bays are assigned based on when expanders are discovered and few internal data structures are not valid for fan-out enclosures. If many expanders are discovered at once, it is possible that incorrect information is reported to the host management tools.
  - Fix: Firmware is changed not to fill in the unnecessary data structures for fan-out enclosures and report the correct information about the drives in the enclosure.
  - Risk: Medium
- Fixed an issue where cache Status changed from "OK" to "Permanently Disabled" after host server power cycle.
  - Root Cause: Upon cache flush completion the batteries are turned off by firmware and the erase might be in progress, during which the firmware can proceed to destroy the cache signature. This results in cache signature failure during host power cycle.
  - Fix: The cache signature destruction is removed in the path where batteries are turned off. The main intention was to destroy part of the cache signature, so the firmware can tell if the cache contents are invalid.
  - Risk: Low
- Fixed an exception within the cache exception handler caused by register corruption should a cache error (for example, MIPS L1 cache parity error) result in both VPEs on a core executing their exception handlers at the same time.
  - Root Cause: The usertracedata CP0 registers used as a scratchpad for r26 and r27 in the cache exception handlers are per core rather than per VPE. Therefore if two VPEs are in the cache exception handler at the same time r27 and r26 will be corrupted when we return to the ISR resulting in a TLB exception when the ISR tries to jump to r27.
  - Fix: The fix is to use WatchLo0 and WatchLo1 registers to save r26 and r27 during a cache exception then clear WatchLo0 and WatchLo1 before we return from the cache exception. WatchLo registers are per VPE but we must clear them so they don't trigger Watch exceptions immediately when the ISR returns.
- Fixed an issue where controller memory corruption could occur when the response to an Identify Device command (IDD) is received from a drive.
  - Root Cause: The internal IDD command did not setup data buffers to store the incoming data from drive which resulted in the data being stored at random locations causing a data corruption.
  - Fix: Allocate a data buffer before trying to dispatch internal IDD and release the buffer after IDD response received.
- Fixed a rare controller lockup issue caused by a misbehaving SATA drive during an NCQ error handling sequence.
  - Root Cause: The issue is caused by a buffer leak which is exposed in the presence of a bad SATA drive. The drive was seen to have returned an invalid NCQ tag in response to the controller having sent an RLE to determine which NCQ command had failed. NCQ error handling normally involves the controller retrying all queued IOs but aborting the failed IO. In this case, due to the invalid NCQ tag, the controller aborts an incorrect IO which does not have an error and controller ends up retrying the bad IO. The bad IO was supposed to be aborted. The IOs to be retried are stored in a retry queue rather than the main queue which is used for the first transmission of each IO. There was a firmware defect in the processing of the retry queue which resulted in what's known

as an event buffer being allocated but not free'd, resulting in the buffer leak. Eventually after numerous occurrences, the buffer becomes depleted resulting in lockup.

- Fix: The buffer was allocated before it was known with certainty that it would be used. Modified firmware to allocate the buffer only when it is needed and ensure the accompanying buffer free operation is in the right place in the logic.

- Fixed a READ IO data corruption which can occur in the presence of a SATA drive with a defect causing NCQ overrun.
  - Root Cause: A SATA NCQ overrun, while there is atleast one other NCQ command outstanding to the drive, can lead to data corruption of a READ buffer. It was found that a defect in SATA drive firmware caused NCQ overrun, thereby exposing the issue.
  - Fix: Modified controller firmware such that on an overrun condition, reset the drive and retry all IOs. In addition the drive vendor is working on a future fix for the drive firmware.
- Fixed a potential data corruption issue with SATA NCQ error handling.
  - Root Cause: When NCQ error occurs, controller firmware was retrying IOs which are only a portion of an inline XOR sequence resulting in incorrect parity.
  - Fix: Modify controller firmware such that when NCQ error occurs, re-initialize the parity buffer and retry all physical IOs associated with the logical IO.
- Fixed an issue where super-cap charge cycles were timing out.
  - Root Cause: An issue was introduced to the controller super-cap charge code that inadvertently turned off the super-cap node voltage balancing circuit when the charge circuit was turned off.
  - Fix: Ensure balancing circuit remains enabled unless controller is operating from super-cap power.
- Fixed an issue where the host system could not communicate with a RAID5/6 volume after an I/O timeout on one of the drives in the array. This may last as long as an hour. Eventually the drive is failed and communication with the degraded volume will resume.
  - Root Cause: This issue is triggered by a command timeout on a drive. Once this happens, the controller will issue a command that checks for PHY DWORD errors. This command will also fail due to a drive bug and result in 2 retries. The entire sequence will last for at least 60 seconds. Contributing to this issue is the fact that these commands are non-NCQ, which means there can be no host interaction while these commands are processed. When the final attempt to retry this command expires after 60 seconds, we are again facing a situation in which the PHY DWORD error is issued. The end result is that the controller will continue in a loop consisting of command timeouts resulting in DWORD error checking commands which in turn timeout. This causes the host I/O starvation and inability to communicate with the volume.
  - Fix: There are three contributing changes to mitigate this issue. It is likely that all three will be implemented in the code base:

    1. Checking for PHY errors should not be an automatic response. Currently the default behavior is to issue the DWORD error check after I/O timeout. This logic should be reversed so that PHY errors are only checked when needed (e.g., after explicit response indicating a DATA_PHASE_ERROR).
    2. If a calling function does not care about the success of a called function, set a flag that indicates the command should not be retried. This mechanism already exists in the FW stack, but was not being used by these processes.
    3. Increase the time between calls to check for PHY errors. Issuing a retry in 2–3 minutes instead of 20 seconds would reduce both the number of instances and recurrences of I/O starvation.

  - Risk: Creating a test harness that first reproduces the problem and then tests the possible fixes is complex and will probably take as much or more work than generating the fixes themselves.
- Fixed a firmware lockup 0x1E30 when discovering ATA locked drive.
  - Root Cause: The controller discovers that the ATA drive is locked and marks it as non-disk device. Firmware asserts if either the drive has not gone through initialization or it was not already marked as non-disk device.

- Fix: Firmware is changed to mark the device as non-disk device if not already set, instead of asserting.
- Risk: Low

- Fixed an I/O performance drop issue during sequential read operations on caching enabled volumes.
  - Root Cause: The builds prior to 1.32 had coalescing enabled on cached volumes. It cannot be re-enabled on volumes that have caching enabled due to potential data corruption. By invoking the read ahead calls on every cache hit and read extend on every incomplete cache hit takes more time.
  - Fix: The firmware now avoids the read ahead on every cache hit as well as optimize the current read extend logic that reduces the chances of suspending a request. Without suspended read ahead cache hit rate will be less as most of the request will be an incomplete cache hit. For smaller sequential read workloads, suspended read ahead helps in improving performance.
  - Risk: Low
- Fixed a firmware lockup problem during an enclosure hot-plug.
  - Root Cause: The firmware allocates a buffer to track the hot plug count during SES initialization layer. In this situation, the firmware tries to access the buffer prior to allocation.
  - Fix: Firmware is changed to check the allocation before accessing the buffer during enclosure hot plug.
  - Risk: Low
- Fixed a problem where logical drive goes into failed state when Rapid Parity Initialization (RPI) is in progress and when physical drive write caches were enabled.
  - Root Cause: When drive write cache is enabled on specific drives that do not support UNMAP/TRIM commands, the drives are not responding to the WRITE IOs to zero out the blocks during RPI, within the timeout when connected behind a specific expander. The RPI logic fails the drive without retrying the request.
  - Fix: Firmware is changed to retry the request until the retry count gets exhausted.
  - Risk: Low
- Fixed a controller lockup/hang problem on creating logical drive when abnormal volume state is set.
  - Root cause: When abnormal volume state is set and user tries to create logical drives using SSDs supporting UNMAP/TRIM commands, the set configuration command waits for Over Provisioning Optimization process to complete. The RPI background code never proceeds through its process because the abnormal volume state is set. This results in the command not being completed and the system becoming hung.
  - Fix: Firmware will return error and does not allow the set configuration command when the controller is in abnormal volume state.
  - Risk: Medium
- Fixed a command timeout issue which was occurring when a command was sent from controller to a drive performing sanitize operation.
  - Root Cause: When Sanitize operation is in progress on a drive, all the incoming I/Os are rejected by the drive (SAS) or by the controller firmware (SATA) with status of Sanitize in progress and this status was being dumped to the (slow) UART interface. The request to the drive and completion from the drive were being queued for a time duration longer than the timeout dictated by the higher level controller firmware. A timeout would occur, followed by an attempt by firmware to reset the drive. But since the reset command and completion also were also being queued for a duration longer than the reset, it was resulting in a lockup.
  - Fix: Modified the controller firmware to eliminate the status prints when the I/O is rejected with a Status of Sanitize in Progress (this info is still logged in internal firmware logs). This relieves congestion in the lower firmware queues and eliminates the timeouts.
- Fixed an issue in the assignment of WWN to SATA drives in cases where there are more than eight directly attached SATA drives.

- Root Cause: Higher port count devices (24) broke the WWN naming scheme controller firmware used to assign unique WWN to each PHY in a system containing more than eight directly attached SATA drives. This resulted in WWN overlap where multiple SATA drives had the same WWN assigned to them.
- Fix: Change the WWN naming scheme to correct the problem. Now all the WWN names assigned to the directly connected SATA drives are based off a WWN associated with the controller card.

- Fixed an issue found in Microsoft Storage Spaces testing where the host application expects drive dependent WWN name (from IDD response) to be returned by the controller but the controller is returning locally generated slot dependent WWN or expander generated WWN.
  - Root Cause: Controller firmware returns WWN reported during device add. This WWN is either internally generate by controller for direct attached drive or generated by expander for expander attached SATA drives. The issue is that this makes WWN phy/slot dependent and not drive dependent which is causing issue with Microsoft storage spaces tests.
  - Fix: Modified controller firmware for INQUIRY to return the correct type of WWN based on what the host application requires.
- Created a firmware workaround for a CPLD implementation issue with reset (driving controller chip RSTB pin).
  - Root Cause: If power returns in the window between [a] and [b], a corner case in the CPLD implementation is exposed.

    [a] after the final DMA to complete the backup to NAND flash

    and

    [b] controller firmware asserting the backup_done GPIO which removes backup power

    In this window if power returns, the CPLD won't drive the device RSTB pin after firmware asserts the backup_done GPIO. The system requirement that when the GB state machine transitions to `s=power_down` ([b]) the GB subsystem goes dark, is broken in this instance.
  - Fix: The firmware will force the top level RSTB. Without the workaround, firmware may transition from power_down to power_up and expose the GB subsystem to unnecessary corner cases including re-initializing NAND device, NFC (NAND Flash Controller), and so on.
- Fixed an issue where controller write cache is not turned on when backup power source cable error is restored.
  - Root Cause: Controller cache will switch to 100% read cache on cable error and when the cable error is restored, the required functions are not called to turn the cache back on with preferred write/read ratio.
  - Fix: Firmware is now changed to call the necessary function to turn cache on with preferred settings once the backup power cable error is restored.
  - Risk: Low
- Fixed an issue where controller does not report when it is in survival mode after POST during boot sequence.
  - Root Cause: In certain scenarios where pre-boot driver is not executed, the controller does not report the correct survival mode thresholds.
  - Fix: Firmware is changed to handle the quick boot scenario wherein it will save and restore the host address provided by pre-boot drivers. This will ensure necessary steps are taken once the host boot process is complete which includes survival mode settings.
  - Risk: Low
- Fixed an issue where connection information was not getting updated to the devices attached during hot plug/remove of wide port expander configuration.
  - Root Cause: When a wide port expander had a change to its connector number, the devices attached to it were not getting their connector number updated to the same value.

- Fix: Firmware is changed to iterate through all attached devices to an expander after it has its connector information updated due to a single cable hot plug or remove event.
- Risk: Low

- Fixed an issue where an incorrect connector status was reported when both SAS cables were removed from a wide port expander connection at the same time.
  - Root cause: No PHY down event was being sent in this situation.
  - Fix: Firmware sends a PHY down event when an expander directly connected to the controller was hot removed which was in a wide port configuration. When processing this event, check the controller active PHY map to see if the expander has been hot removed entirely then take appropriate PHY down on the expander so when the hot remove event is received it must process only one connector on one path.
  - Risk: Low
- Fixed an issue where IO may never get completed when disabling IOBypass.
  - Root Cause: When disabling IOBypass, IOs could be stuck at drives.
  - Fix: Firmware is changed to wait for outstanding SmartPath commands count to become zero within a time bound limit to make sure no IOs are pending before disabling SmartPath.
  - Risk: Low
- Added firmware support for registered message types in `Get MCTP Version Support` command.
  - Root Cause: The `Get MCTP Version Support` command only supported a hard-coded set of message types. It should also be able to supply the version of dynamically-added message types.
  - Fix: Modified firmware to include "version" parameter to specify the version of the registered message type.
- Firmware added support for Control Message type in `MCTP Get Message Type Support` response.
  - Root Cause: The `MCTP Get Message Type Support` command did not return the built-in Control Message type in its list of registered types.
  - Fix: Modified firmware to include Control Message type in `Get Message Type Support` response.
- Fixed an issue in SATA firmware where an exception was occurring due to misinterpretation of a context field as an address.
  - Root Cause: The context field called "active request" for internally generated SATA request (for example, IDD) are custom values and not memory addresses. Active request for a device is updated with the context to track and block other IOs when the internal IO is outstanding. A firmware exception occurs when the active request is dereferenced like a pointer when an internal request is outstanding. This is an incorrect operation as this active request value is not a pointer.
  - Fix: Modified SATA firmware to add a check on the type of context before dereferencing the active request.
- Added firmware workaround for M83C NAND flash timing mode 6 errata.
  - Root Cause: M83C NAND timing mode 6 errata is for industrial temperature M83C devices. When operating in the temperature range of 0 °C to 85 °C in the NV-DDR2 interface at timing mode 6 (166 MHz), the tDH and tDS min timing parameters are required to follow the NV-DDR2 timing mode 5 (133 MHz) values.
  - Fix: If the NAND is M83C and the timing mode is set to 6 (166 MHz), firmware will set timing mode to 5 (133 MHz).
- Added an enhancement to improve the controller's ability to self-test discharged degraded super-caps.
  - Root Cause: With degraded super-caps, multiple self-test cycles are required to get reliable estimates of super-cap capacitance and remaining lifetime.
  - Fix: Manufacturer's recommended self-test scheme is charge to rated voltage, then discharge to 1/2 rated voltage, repeat the charge/discharge cycle and to use results from second cycle to estimate capacitance. This was previously tested using good capacitors and it was found the

second cycle wasn't necessary. With degraded capacitors, the second test cycle is necessary to get reliable capacitance estimate.

- Fixed an issue in an unstable SAS link environments where controller firmware would prematurely allow commands to be sent to a drive after the drive reset attempt failed.
  - Root Cause: When the higher layer firmware requests a reset command to be sent to a drive and the command fails to be sent due to a link down, drive removal, or an unstable SAS link, the higher layer firmware gets a failure indication, but the lower level firmware incorrectly re-enabled the transmission of commands to that drive. This results in the drive not being reset correctly.
  - Fix: Modify the firmware to not re-enable commands to be sent to a drive associated with this type of reset failure where the reset was not actually sent to the SAS link.
- Added a firmware feature to send Identify Device (IDD) after device reset to work around problems seen in expanders.
  - Root Cause: After a SATA drive has been reset, some expanders require an IDD command to be sent to the drive, otherwise the expander will not forward any commands to the drive causing I/O timeouts that can lead to drives being reported as failed.
  - Fix: Modified SATA firmware to send an IDD after device reset.
- Fixed an issue where the processing of asynchronous events from drives and expanders gets delayed when there is a flood of such events.
  - Root Cause: A single buffer pool was used for messaging between different firmware layers and lower layers of firmware for asynchronous events and for synchronous messaging. A flood of asynchronous events can cause buffer depletion which delays response for error handling between firmware layers.
  - Fix: Split the single buffer pool into two buffer pools to improve the firmware layers intercommunication.
- Fixed an issue where TLR control bit was not getting set for tape drives.
  - Root Cause: Tape drives supporting TLR did not have an appropriate bit set in the corresponding mode page on discovery.
  - Fix: Firmware now sends mode and mode select for all tape drives to enable TLR as long as the TLR enable bit is not already set.
  - Risk: Low
- Fixed a problem where incorrect drive write cache was reported for HBA drives.
  - Root Cause: If the policy was set to default for HBA drives, the host operating system changes the drive write cache, but firmware was reporting the older status. Mode sense and mode select to the caching mode page were also being sent in cases that were not necessary.
  - Fix: Firmware does not send mode sense/mode select command to caching mode page unless it is required to keep track of the current drive write cache status. In case of HBA drives, the firmware checks the current setting always since the host OS might have changed it.
  - Risk: Low
- Fixed a deadlock situation in cases where DDR cache is completely dirty that results in a 0x27006 lockup.
  - Root Cause: When cache is completely dirty, new IOs are put into a cache retry queue. Once all the internal resources are used, cache flush logic and cache retry logic end up in a dead lock situation waiting for internal resources to be available.
  - Fix: Added checks to avoid exhausting all the internal resources.
  - Risk: Low
- Fixed a problem where the controller clear configuration fails when sanitize erase is in progress on physical drives.
  - Root Cause: A race condition between clear configuration code and sanitize erase code while modifying the meta data stored in disks results in this problem.

- Fix: Firmware is changed to suspend background at the very beginning to avoid this condition.
- Risk: Low

- Fixed a problem where drive appears twice after hot remove/add of expander cables.
  - Root Cause: When a drive failed, the expander IO module is restored in a dual domain configuration with asymmetrical cabling and drives were added back as new single path drives due to the PHY routing attribute checks which were required for legacy JBODs that are not compliant to specifications.
  - Fix: Skip the PHY routing attribute check if the expanders are compliant to specifications.
  - Risk: Low
- Fixed an issue where typical occurrences of SAS PHY were not recovering during continuous drive hot unplug/re-insertion test cycle.
  - Root Cause: When running a strenuous drive hotplug test (repeating 10,000 hotplugs), it has been observed that after ~7500 hotplugs, when the drive is re-inserted the SAS PHY gets stuck and never transitions to the state where it can pass traffic between the controller and the drive. The root cause is due to not disabling a clock referred to as LOS_REFCLK on a PHY disable action.
  - Fix: Per hardware specification firmware was modified to stop LOS_REFCLK during power down (unplug) sequence and the clock was restarted during power up (re-insertion).
- Added a firmware feature to provide backup related information to host applications such as expected backup power usage, backup time and RAID cache size for underlying hardware which helps in improving management of the backup power supply from user's perspective.
  - Root Cause: Information available to the application was not explicit enough to provide to the host tools and user.
  - Fix: Added an API in firmware, which applications can call to gather information related to backup power supply mentioned above.
- Fix an issue where supercap charge cycle does not resume after an over temperature condition is cleared.
  - Root Cause: The supercap charge cycle is stopped when the supercap temperature exceeds the maximum or minimum thresholds as configured in the charge parameters. Due to a flaw in the logic, the firmware does not automatically resume the charge cycle when the supercap temperature transitions back into the operating range.
  - Fix: Fixed the firmware flaw, such that when the supercap temperature transitions back to the valid operating range, the charge cycle is resumed.
- Fixed an issue found where LED control was not functioning properly - that is, LED blinking was not working as expected.
  - Root Cause: There was an error in the values stored in an LED array holding LED blink patterns.
  - Fix: Made changes in LED array so that it matches the design pattern used for SGPIO protocol.
- Fixed a timeout followed by LUN Reset caused by the SATA firmware for the SCSI Send Diagnostic command during a drive's default self test.
  - Root Cause: SATA layer firmware queued incoming Send Diagnostic commands while the drive default self test was in progress. Eventually before the self test completed the requests timed out resulting in upper layers sending LUN reset.
  - Fix: Modified firmware to use the same logic for default self test as it does for background self test whereby the incoming command is rejected with the reason of self test in progress. The avoids the subsequent LUN Reset in the recovery from upper layers and allows for a more graceful wait until self test is complete.
- Fixed an issue where the SMART status fields in ATA IDENTIFY data and SCSI ModeSense Informational Exceptions (IE) page data are inconsistent.
  - Root Cause: SATA layer firmware caches the SMART feature status settings when they are enabled/disabled through Mode Select. Firmware reports the cached value when requested through Mode Sense. If the SMART feature status was modified using an ATA Passthrough, SATA

layer firmware would not be aware of this and does not update the internal cached value. Therefore after an update using ATA Passthrough, subsequent requests for this status through mode sense can result in the wrong status values being returned.

  - Fix: Eliminated the usage of cached values for SMART status fields. Changed firmware to manage only one copy of the status fields.

- Fixed a rare issue where drive contexts do not get dropped from firmware device list after they are hotswapped within two second time window.
  - Root Cause: Lower level firmware has a grace period (device immunity) between when discovery determines that the device is absent and the removal of the device context from the firmware device list. This is to prevent RAID volumes from instantly being degraded when a device disappears for a short period of time due to temporary loss-of-sync (LOS) on the SAS links. Due to a flaw in the firmware, if a second device is found associated with the same expander PHY resulting in the device list containing two device contexts associated with the same expander PHY, the device list could remain in a corrupted state with the original device context not being removed. This can happen when one device is unplugged and very quickly a new device is inserted (within a two second time window).
  - Fix: Modify firmware to immediately remove a previous device context from the device list when a new device is detected on the same PHY.
- Fixed an issue where the super-cap self-test cycle does not properly handle cap node voltage balancing.
  - Root Cause: Firmware was monitoring to maintain voltage balance between super-cap nodes during a non-charging stage of the self-test instead of only monitoring during charging stages. The problem was observed while testing with a degraded super-cap and running self-test cycles. This can lead to self-test cycle being continuously interrupted to wait for the super-caps node voltages to balance resulting in self-test not completing.
  - Fix: Removed super-cap node voltage monitoring from self-test stages when charger is off.
- Fixed an issue where device discovery was taking a long time to complete on unstable topologies.
  - Root Cause: When discovery firmware is notified of a port event (like Broadcast change) or when a device got reset on the port, any ongoing discovery process on the port was halted and re-started. The reasoning was to avoid mistakenly removing a device when it's only expected to be temporarily absent due to a glitch on a SAS link. However, with device immunity feature added to firmware for all device removals to address temporary LOS (loss-of-sync), a temporary absence will not result in device removal. On the other hand, stopping discovery on device resets and port events caused discovery to take a long time to complete on unstable topologies. This results in delays in reporting topology changes.
  - Fix: Modified firmware to no longer stop and fail current discovery instance due to resets outside of discovery or due to port events.
- Fixed an issue where a firmware feature to auto-detect a new super-cap attached had been mistakenly removed.
  - Root Cause: SR2.1 firmware monitors for a new super-cap being attached. This was based on a capacitance increase of more than 20%. Detecting this would reset the super-cap meta data and it would begin tracking the new super-cap health. This code was inadvertently removed from SR2.2.
  - Fix: Restore code to detect a 20%+ increase in capacitance to determine that a new cap has been attached. Also, added support for detection of hot-swap when super-cap has failed.
- Added a firmware API for additional LED control required by customers.
  - Root Cause: Customer's required a third LED blink rate and two new activity LED commands.
  - Fix: Modified firmware to add required features.
- Fixed an issue with support for `subtractive-to-subtractive` inter-expander connection.
  - Root Cause: In relaxing the restrictions on mixed topology support, the fix mistakenly treats a `subtractive-to-subtractive` inter-expander connection as a loop.
  - Exposure: All releases

- Fix: Correct code to not treat expander `subtractive-to-subtractive` connection as a loop.

- Fixed an issue where cache ratio is not getting reconfigured with previous settings when cable error is restored.
  - Root Cause: In case of temporary cable error, cache settings will be changed to 100% Read, 0% write and these settings were written into drive metadata which will override the user selected settings.
  - Fix: Instead of updating metadata with 100% read, preserve and restore the drive metadata which has the previous user selected settings, in case of temporary cable error.
  - Risk: Low
- Fixed an issue where firmware returns error while creating logical drives after backup power source cable is removed.
  - Root cause: Cache configuration buffer was found to be invalid after backup power source cable is removed and cache is set to 100% read ratio.
  - Fix: Added a condition to check if firmware changes over to 100% read cache ratio and allow the logical drive creation.
  - Risk: Low
- Fixed an issue where firmware was not failing the SSDs which replaced HDDs on reboot.
  - Root Cause: Even though the replacement drives were of not valid type, they were getting marked OK state
  - Fix: Firmware is changed to validate the drive type and size for the replacement drives.
  - Risk: Low
- Fixed an issue where serial output buffer may not accept new data if it is full at times.
  - Root Cause: When the serial output buffer is full and a consumer is extracting data slower than the rate at which new data is produced, it was possible that new data is not added into the buffer. This approach is prioritizing the older data over newer.
  - Fix: Firmware is changed to prioritize the newer data over older older data when the serial output buffer is full and there are active consumers of the buffer.
  - Risk: Low
- Fixed a controller 0x1EF0 lockup during cache flush at boot time.
  - Root Cause: Firmware paused the background thread intentionally during cache flush during controller boot time, but the code which detects possible hang condition did not realize the situation.
  - Fix: Firmware is changed track the intentional background pausing and not to consider this as a possible hang condition.
  - Risk: Low
- Fixed an issue where connector information of sub-expander’s children is not getting updated in an enclosure.
  - Root Cause: When wide port configuration changes, the connector information is updated only for the directly attached expander and its children including sub expander and drives, but not for the downstream expander's children. Due to this the box number is shown as 0 for the devices attached to the enclosure.
  - Fix: Update connector information appropriately for sub expander's children.
  - Risk: Low
- Fixed an issue where IGNORE ZONE GROUP bit is not set when needed while sending SMP DISCOVER command.
  - Root Cause: Firmware was not setting the IGNORE ZONE GROUP bit when sending the SMP DISCOVER command to an expander due to which firmware received PHY VACANT response back. This resulted in devices are listed with 255 bay number on expanders with zoning enabled.

- Fix: Add the code to set "IGNORE ZONE GROUP" bit in byte 8 of SMP DISCOVER request.
- Risk: Low

- Fixed a 0x1E30 lockup on boot controllers with more than 24 PHYS.
  - Root Cause: SmartRAID 3154-24i2b boards use 26 PHYs, not 24 PHYS. This caused an incorrect calculation while determining the maximum ports which resulted in ASSERTION lockup.
  - Fix: Calculate the maximum value correctly based on maximum number of PHYS supported.
  - Risk: Low
- Fixed an issue where a failed request retried forever when active path of drive is changed.
  - Root Cause: When active path for a drive changes, physical requests sent to drive with old path fails with a specific error. When this request is retried, firmware is no not updating active path of request with this results in infinite retries of the same request in error handling.
  - Fix: New active path for a drive needs to be updated when path changes in order to avoid retrying failed request forever and retry only within the limit set.
  - Risk: Low
- Fixed an issue where a drive is made offline during hot plug which was undergoing sanitize operation earlier.
  - Root Cause: In certain cases, a drive that was previously going under sanitization was reporting sense data of LOGICAL UNIT IS IN PROCESS OF BECOMING READY when being hot added. This mean that explicit sanitization checks did not detect that the drive was sanitizing at that time. Code which handles hot plug, had a check to wait for a drive to become ready before indicating that the device was available. If it did not detect that the device was ready, it would fail the device. In the case of the sanitized drive, it eventually returned LOGICAL UNIT NOT READY, SANITIZE IN PROGRESS.
  - Fix: When waiting for the drive to become ready, check if the drive indicates that sanitization is undergoing and mark it as being sanitized.
  - Risk: Low
- Fixed a data integrity problem while running IO on RAID 1 volume.
  - Root Cause: When host IOs and surface scan are running on the same logical drive, surface scan does not lock the correct stripe while performing its operation on RAID 1. Due to this host written data might be overwritten with the incorrect data from surface scan operation in case of race condition between READ from surface scan and WRITE from host IO.
  - Fix: Corrected the logic which determines the stripe and stripe length for RAID1 during surface scan. This will make sure the complete stripe is locked during surface scan.
  - Risk: Medium
- Fixed an issue where transformation/expansion task does not resume on backup power source flickering.
  - Root Cause: Transformation task is shut down when backup power source goes offline. When the power source comes back online, expansion task is not triggered properly, instead it will be resumed only on reboot.
  - Fix: When firmware receives asynchronous notification from backup module that power source is ready, expansion task will be triggered to resume.
  - Risk: Low
- Fixed a controller hang problem after sending the rekey command.
  - Root Cause: There was deadlock situation between the thread responsible for processing host request and the thread which is responsible for flushing the cache while handling the cache retries due to errors. When rekey command is initiated, request processing thread was waiting for cache data to be flushed, this thread is also responsible for retrying any requests which are queued. But flush thread encountered errors while flushing cache data, so it posts these requests for retry.
  - Fix: Host requests processing thread will first dispatch the retry requests before waiting for cache data to be flushed.

- Risk: Low

- Fixed a balancing circuit timeout value for the super-cap charge logic for smaller super-caps.
  - Root Cause: The balancing circuit on the 3162 boards using the 17F caps is not the same as the the boards using the 35F caps, and the 15 minute balancing timeout is not sufficient. As an interim work-around, the timeout is being doubled in firmware.
  - Fix: Doubled the balancing timeout to accommodate balancing the circuit used for 17F super-cap packs.
  - Risk: Low
- Fixed a firmware deadlock issue with multiple controllers in a large SATA configuration.
  - Root Cause: During controller boot up, a firmware deadlock can occur when connected to a large SATA topology that contains multiple controllers. If the SATA drives were previously affiliated with a different controller, a large number of IOs can fail with a STP Resource Busy error. The high number of STP Resource Busy errors deplete a firmware buffer pool and the buffer empty condition is not handled correctly by firmware resulting in the deadlock.
  - Fix: Correct the firmware logic to handle the buffer empty condition.
  - Exposure: Firmware 1.60 and later releases
- Fixed B16A NAND flash backup times reported by the controller.
  - Root Cause: Backup up times were incorrect.
  - Fix: Corrected the B16A backup times stored in the firmware.

2.2.2 UEFI Fixes

Note: Microsoft signed and secure boot is supported**.**

2.2.2.1 Fixes and Enhancements for UEFI Build 1.3.5.2/Legacy BIOS Build 1.3.5.2

This release includes the following UEFI fixes and enhancements:

- Fixed an issue where an incorrect enclosure serial number was getting displayed in driver health message when I\O modules have different firmware versions.
  - Root Cause: Serial number was not included in driver health message.
  - Fix: Included serial number in driver health message.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the disk utilities menu shows multiple enclosure entry when connection is in dual mode.
  - Root Cause: No filter considered for multiple SEP entries from the enclosure while listing devices.
  - Fix: Enclosure number to be considered for sorting devices. Added filter to skip multiple SEP entries from the enclosure while listing devices.
  - Exposure: All previous releases
  - Risk: Low
- Fixed an issue where No Battery Write Cache was not getting enabled on configured controller.
  - Root Cause: API had a circular dependency scenario when the backup power source is missing.
  - Fix: Modified API to handle dependencies individually.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where improper error message was displayed when trying to create multiple MaxCache logical drive with no minimum space available.

- Root Cause: No minimum size check present while listing data logical drive for MaxCache logical drive creation.
- Fix: Added minimum size check and error handling while listing data logical drive for MaxCache logical drive creation.
- Exposure: All previous releases.
- Risk: Low

- Fixed an issue where user was allowed to delete any logical drive while abnormal state exists.
  - Root Cause: Checks were not done on all volumes to determine if any volume on the controller was in an abnormal state. The case of a volume with abnormal state therefore could not prevent volume deletion.
  - Fix: All volumes on the controller are checked to determine if they are in an abnormal state. If any such volume exists, a controller flag is set to indicate. This flag is now checked before volume deletion operations, which if set prevents the operation. Virtual configurations also cannot be created if this flag is set.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where clear configuration operation failed when all the connected drives removed and logical device is in failed state.
  - Root Cause: This rule is a legacy rule that blocked clear configuration when no drives were present to update RAID metadata. Booting the controller and re-inserting the missing physical drives would cause the original configuration to be restored. Since physical drives are not required to be part of a RAID volume or masked from the host, there are more use cases when a user may want to clear a failed logical drive configuration even if no physical drives are present.
  - Fix: Remove the restriction that any physical drive must be present to perform a clear configuration. UEFI will allow deleting all logical drives even if all physical drives are missing.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue to enable the "No Battery Write Cache" on a configured controller.
  - Root Cause: Software requires at least one logical drive with 'ControllerCache' as the acceleration method to allow setting No Battery Write Cache. However, users cannot configure a logical drive with 'ControllerCache', if the backup power source is missing and No Battery Write Cache is enabled. So, there is a circular dependency issue when the backup power source is missing.
  - Fix: Software was changed to ignore checking if any logical drives are using 'ControllerCache'. This will allow users to configure No Battery Write Cache as long as any logical drive exists. Afterwards, the user will be able to create new logical drives or edit existing ones with 'ControllerCache' even when the backup power source is missing.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the AdapterInfo protocol DriveInventory is causing memory corruption.
  - Root Cause: DriveInventory method of AdapterInfo protocol is passing invalid target index which is outside the buffer limit causing invalid memory access.
  - Fix: DriveInventory method of AdapterInfo protocol to pass valid target index.
  - Exposure: 1.3.0.89 and later
  - Risk: Low
- Fixed an issue where the Acceleration method could be changed when the logical drive state is failed.
  - Root Cause: `SA_CanSetEditableLogicalDriveAccelerator()` made no check for logical drive status.
  - Fix: For existing logical drives, check the logical drive status.
  - Exposure: All previous releases

- Risk: Low

- Fixed an issue where array operations are not blocked while a rekey task in progress.
  - Root Cause: SA_CanChangeArraySpareType( ) is not restricting the operation when any of the Array logical drives are rekeying.
  - Fix: Add check to SA_CanChangeArraySpareType( ) that will return if the Array status is not reported as "OK". This will catch the case when any of the logical drives in the Array are rekeying.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where proper Port/box/bay was not appearing for EFI_ADAPTER_INFORMATION_PROTOCOL.
  - Root Cause: Incorrect logic used to collect number of physical drives that are part of logical drives.
  - Fix: Corrected the logic which is used to collect number of physical drives that are part of logical drives.
  - Exposure: All previous releases
  - Risk: Low
- Added an option under Administration menu to reset controller to factory defaults.
  - Fix: Option added under Administration menu to reset controller to factory defaults.
  - Risk: Low
- Fixed an issue where the OS does not boot when the boot device is connected to Phy0.
  - Root Cause: Driver is not handling I/O errors for EFI_BLOCK_IO_PROTOCOL Read/Write.
  - Fix: Added error handling for I/O errors in EFI_BLOCK_IO_PROTOCOL Read/Write.
  - Exposure: All previous releases
  - Risk: Low
- Fixed an issue where the utility allows to set the logical drive label when Rapid Parity Initialization (RPI) is in progress.
  - Root Cause: API to set logical drive label was not checking for RPI queued or in progress.
  - Fix: Add a check for the logical drive RPI queued or in progress.
  - Exposure: All previous releases
  - Risk: Low
- Added a fix where the logical drive transformation progress is not updated and displayed as 0.
  - Root Cause: The progress percentage calculation was using the incorrect number for total number of blocks within the logical drive. The resulting total number of blocks processed would always calculate to zero.
  - Fix: The correct field for the total number of blocks in the logical drive is now used in the progress percentage calculation.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where an unassigned drive count under controller information menu includes un-configurable drives.
  - Root Cause: API to get unassigned drive count includes unsupported/unconfigurable drives in the result.
  - Fix: Add a drive usage type to apply to physical drives that are unsupported-for-RAID.
  - Exposure: All previous releases.
  - Risk: Low
- Provided additional fields in controller information menu such as password attempts, cache, logical, and physical drive count related information.

- Implementation : Included number of login attempts remaining, cache, logical and physical drive count related information in controller information menu.
- Risk : Low

- Fixed an issue where the driver health message 1778 is not observed when a volume is failed and a reboot is performed.
  - Root Cause: Improper check when the logical drive state is LOOSE_CABLE.
  - Fix: Improved check for driver health message 1778 when logical drive state is LOOSE_CABLE.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the BlockIo protocol handles not updated after any configuration changes.
  - Root Cause: BlockIo protocol was not re-installed for any configuration changes.
  - Fix: Re-install BlockIo protocol after the configuration changes like logical drive creation/deletion, and so on.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the controller Information shows the cache status as "*Cache disabled low backup power source*" even if the backup power source is charged.
  - Root Cause: Improper check for cache status.
  - Fix: Rectified cache status check.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the sanitize erase operation does not display relevant driver health messages.
  - Root Cause: Deprecated feature bit was being used to check sanitize support.
  - Fix: Added appropriate check for sanitize support.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where modifying the cache ratio fails when array transformation in progress.
  - Root Cause: No check for cache operation support before cache ratio modification.
  - Fix: Added check for cache operation support before cache ratio modification.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where encoding and transformation progress are not shown in logical drive information menu.
  - Root Cause: Progress information not considered for encoding and transformation states.
  - Fix: Progress information added for encoding and transformation states.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the unsupported default MaxCache logical drive size is displayed for selection.
  - Root Cause: Default MaxCache logical drive size for selection is not calculated based on cache line size selection.
  - Fix: MaxCache size calculated based on data logical drive size selected and cache line size.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the save support archive operation is not capturing controller crash dump.
  - Root Cause: Controller crash dump not considered for archive operation.

- Fix: Added support to capture crash dump information in save support archive operation.
- Exposure: All previous releases.
- Risk: Low

- Fixed an issue where the setup was allowed to proceed when password or master key is left blank or encryption left at disabled.
  - Root Cause: No validation done for password or master key in full setup encryption form.
  - Fix: Implemented validation for full setup encryption form before proceeding to the next form.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed the controller settings and advanced controller settings submit form that does not provide specific error messages.
  - Root Cause: Controller settings and advanced controller settings submit does not collect specific error messages.
  - Fix: Option specific error messages collected and provided to the form.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed the ability to identify a failed path connector in logical drive info and in disk utilities menu.
  - Root Cause: Failed path is not represented for a multi path connected drive.
  - Fix: Physical drive representation in logical drive info and disk utilities modified to show (F) for failed path.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where during the MaxCache array creation, the selection of data logical drive is after the selection of RAID level.
  - Root Cause: Selection of data logical drive has to be before RAID level selection.
  - Fix: Moved selection of data logical drive to the beginning right after drive selection menu.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed where the controller cache option is shown as **No Battery Write Cache** in other tools.
  - Root Cause: Controller Cache option name is not consistent with other tools.
  - Fix: Controller Cache option renamed as No Battery Write Cache.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the logical drive status string shown on logical drive information menu is not same as shown on other tools.
  - Root Cause: Logical drive status strings are not consistent with other host tools.
  - Fix: Logical drive status strings modified to match other host tools.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where MaxCache logical drive information menu does not have cache status fields.
  - Root Cause: Cache status fields not added in MaxCache logical drive information menu.
  - Fix: Added cache status fields in MaxCache logical drive information menu.
  - Exposure: All previous releases.
  - Risk: Low
- Fixed an issue where the array creation fails when maximum fraction numbers are specified.

- Root Cause: Logical error in conversion of floating point to decimal calculation.
- Fix: Added boundary checks for floating point to decimal conversion.
- Exposure: All previous releases.
- Risk: Low

This release includes the following legacy BIOS fixes and enhancements:

- Fixed a BIOS red screen failure when a DVD/CD-ROM is attached to the system and the SATA controller is enabled.
  - Root Cause: The Interrupt Flag set by the INT 13h handler caused the AHCI option ROM to cause a red screen BIOS failure.
  - Fix: Clear the Interrupt flag at the exit of the INT 13h handler.
  - Exposure: All previous releases
  - Risk: Low

2.2.3 Driver Fixes

2.2.3.1 Fixes and Enhancements for Linux Driver Build 1.2.4-065

This release provides the following fixes and enhancements:

- Fixed an issue to correct the volume status when a unit is deleted between “Report Logical LUNs” and “Logical volume status vendor-specific Inquiry page".
  - Root Cause: There is a race condition when a unit is deleted after a “Report Logical LUNs”, and before getting the “Logical Volume Status” page of the unit. In this case we will get a standard inquiry, rather than the vendor specific page “Logical Volume Status”. This will result in a unit presented which no longer exists.
  - Fix: Insure we receive the “Logical Volume Status” page to report the correct logical volume status.
  - Risk: Low
- Fixed an issue to not offline a device when receiving transient did NO_CONNECT.
  - Root Cause: This error may be transient, and should schedule a re-scan. During the rescan if everything is normal, assume this was a transitory issue, which has been resolved. Do not force the device offline, as the error handling strategy will not be used to validate the transitory nature of the error.
  - Fix: Do not force the device offline, schedule a device scan to determine if the device is still present
  - Risk: Low
- Fixed an issue where Linux driver RAID map is too large.
  - Root Cause: Currently, the code statically allocates array memory for RAID map structure.
  - Fix: Now, the code dynamically allocates the memory for RAID map entries based on the response from the firmware.
- Fixed an issue to change the NMI reset to controller soft reset.
  - Root Cause: When there is no response from firmware during online firmware update after acknowledgment for quiesce IO event, the firmware might lockup.
  - Fix: If there is no response from firmware, the driver performs a controller soft reset during that juncture.
- Fixed an issue to serialize LUN reset.
  - Root Cause: In some cases, the LUN reset process did not clean up the IO inbound queues. This would result in the LUN not being reset.
  - Fix: Fixed the LUN reset process to clean up the IO inbound queues so the LUN was reset correctly.
  - Risk: Low

- Fixed an issue where an Inquiry command never completed.
  - Root cause: In some cases, an Inquiry response would not be received, which would cause the device scan to hang.
  - Fix: Added timeout for Inquiry command and reset the LUN when the timeout occurred.
  - Risk: Low
- Fixed an issue to check for null device pointers
  - Root Cause: While drive removal is in progress and if OS triggers LUN reset, firmware sends incorrect LUN task management response. This results in the driver sending a failure to OS and OS marks device as offline. A device can be deleted causing NULL pointer exceptions.
  - Fix: Wait on all outstanding IO to complete before the device is removed. Check for null device pointers in IO entry/completion functions.
- Fixed an issue where the system with NUMA node other than NO_NUMA_NODE(non-zero) fails to load the smartpqi driver.
  - Root Cause: When allocating ctrl_info structure, if the gendevice is defined to have a NUMA node of NO_NUMA_NODE, we set the numa node to 0.
  - Fix: Set `pci_dev->dev` to 0 only, if the node is NO_NUMA_NODE. If not, then do not reset the value but retain it.
  - Exposure: Applicable for all servers, which allocate NUMA differently.
  - Risk: Low
- Fixed an issue where the Expander SMP target is hidden when controller or connector is in mixed/RAID mode.
  - Root Cause: Driver explicitly unmasks the enclosures SMP target during initialization or hotplug processing. This prevents the users from resetting an expander-attached drive that is part of a RAID volume.
  - Fix: Make the driver honor the firmware mask setting for the Expander SMP target.
  - Exposure: Whenever connector is in mixed/RAID mode and an expander is attached.
  - Risk: Low
- Fixed an issue to wake up drives after OS resumes from SUSPEND.
  - Root Cause: When a server is suspended, the driver call-backs send FLUSH_CACHE to the firmware, which spins down all the drives to sleep. Once the server wakes up, the drives are still spun down and any request to that drive is errored with a ASC/ASCQ : 4/2 condition.
  - Fix: Set `allow_restart` option during scsi_device init. This allows the kernel to send a START/STOP Unit command to the drive if it encounters a 4/2 check condition in sense data.
  - Exposure: It happens for hard disk drives (SATA) and every time server resumes from suspend.
  - Risk: Low
- Fixed an issue where kernel panics during device discovery state.
  - Root Cause: During device discovery, there is a INQUIRY timeout on one of the drive. So there is a crash while handling the inquiry command failure in above layer. The reason for timeout was, on SLES 15 the CONFIG_HZ value is set to 250 by kernel, hence the 30 secs(30*HZ) timeout value was resulting in 7.5 seconds.
  - Fix: Set the value of PQI_HZ to 1000 on the OS with HZ value less than 1000 to get an appropriate 30 second timeout value.
  - Exposure: This can happen on setups that have low speed drives.
  - Risk: Low
- Fixed an issue where more then two disks are not exposed in Windows guest OS in Xen 7.4.
- Fixed an issue where the mount point/Disk name was appearing as "Not Applicable" for OS boot volume in management tool.
  - Root Cause: This issue was caused due to one of the recent device attribute additions.

- Fix: During Unique id string building, there was a formatting issue. It's been fixed by using correct format specifier.

- Added support for the device `sysfs` attributes like `unique_id`, `lunid`, and `path_info`.
- Added an enhancement to provide support for 4.15+ kernels.
- Added `smp_utils` support.
  - Root Cause: The driver sas transport entry point for `.smp_handler` was not implemented.
- Fixed an issue where there was a firmware lockup since INQUIRY command for VPD page 0 was stuck in LUN for five minutes.
  - Root Cause: If the INQUIRY command for VPD page 0 is stuck and never gets completed by the firmware, then the driver will not attempt to issue a reset to the LUN that the INQUIRY was submitted to.
  - Fix: Add support to the driver for handling this situation. If INQUIRY for VPD page 0 command does not get response in 30 seconds the driver recovers the device by issuing device reset.
- Fixed WRITE_SAME issue for logical volumes.
  - Symptom: WRITE_SAME template setting was not enabled for logical volumes.
  - Root Cause: Add no_write_same for logical volumes.
  - Fix: For logical volumes include no_write_same into the scsi_device structure. This will insure that WRITE_SAME will not be used for logical volumes.
- Fixed LUN reset issue for rejected Task Management function.
  - Symptom: Retry LUN reset, when the Task Management function is rejected by the controller.
  - Root Cause: If the driver sends a LUN reset for a device already being reset by the controller, then the controller will reject the LUN Reset from the OS.
  - Fix: Driver will wait for ten seconds and try again.

2.2.3.2 Fixes and Enhancements for FreeBSD Driver Build 1.0.3.2035

The fixes and enhancements in this release.

- Added support for FreeBSD 11.1 i386.
- Removed OS_SLEEP form Admin queue request.
  - Root Cause: Firmware was taking time to write response IU, that is why driver was waiting for 20 millisecond to complete response IU.
  - Fix: Firmware added the fix for it so that response will be written properly without waiting in driver side.
- Introduced a new log level for discovery.
  - Root cause: There was only one log level for initialization and driver discovery.
  - Fix: Added two separate log level `DBG_INIT` and `DBG_DISC` for debugging initialization and discovery part.
- Added OS specific macro to check request type.
  - Root Cause: Driver needs to check whether a pending request is an internal request or a command sent by the upper layer.
  - Fix: Added OS specific macro to check the request type.
- Fixed an issue to change the bus transfer speed.
  - Root cause: The bus transfer speed was mentioned incorrectly.
  - Fix: To maintain the device list driver is using array implementation.
- Added Legacy interrupt support.

- Root Cause: If system fails to allocate interrupt type MSIX. Smart controller will not be initialized and driver return error status while loading.
- Fix: If Interrupt type MSIX is not available, driver will try to allocate and use interrupt type legacy interrupts.

2.2.3.3 Fixes and Enhancements for Solaris Driver Build 1.0.3.2035

There are no fixes and enhancements for this release.

2.2.3.4 Fixes and Enhancements for Windows Builds 106.84.2.64

This release includes the following fixes and enhancements:

- Fixed a potential negative value passed to MemCpy in CopyRaidMapData. If RaidMap StructureSize is 0, calculate the disk data size as (0 - 64) resulting in negative/invalid value passed to storport memcpy routine.
  - Root Cause: Code Changes during dynamic RAID MAP Allocation.
  - Fix: Check for the structure size returned in RAID MAP Completion being greater than RAID_MAP_HEADER_SIZE before calculating the disk data size.
  - Risk: Medium
- Fixed an issue on RAID MAP DATA Memory Allocation. In Mixed RAID Configuration the RaidDiskData size stored in PER_LUN_MEMORY structure is not reset to the default supported size. This will cause the driver to use an incorrect Raid Map Disk Data Size to find out if the next Raid Map requires a reallocation, which causes the driver to not to call the Raid Map memory reallocation code and corrupt the RAID_MAP_DATA.
  - Root Cause: Not Resetting the RAID disk data size to the default size after completing the CISS Get Logical disk RAID Map.
  - Fix: Set the RAID disk data size to the Default size after completing the Get Device Raid Map. During rescan RAID MAP allocation make sure IOBypass is not accessed. Add ASSERT to check Global IOBypass flag is enabled during RAID MAP memory reallocation.
  - Risk: Medium
- Implemented the logic to process all the configuration table entries.
  - Root Cause: The current algorithm used for reading the configuration table has an issue that prevents reading the last entry from the configuration table.
  - Fix: Modified the logic to process all the configuration table entries.
  - Risk: Low
- Fixed an issue where SATA Device doesn't report Unique WWN ID.
  - Root Cause: Feature is not supported/enabled.
  - Fix: Add Registry settings for users to enable the SATA drive unique WWN ID feature. During driver initialization read the registry and request the firmware feature to enable the Unique SATA WWN ID through Vendor general command.
  - Exposure: Windows
  - Risk: Low
- Fixed an issue where the RAID_MAP_DATA structure is dynamically allocated based on the required size from the logical Raid map data. This structure is in the middle of the PER_LUN_MEMORY structure and it can overwrite the next variable in the structure, if it is dynamically allocated.
  - Root Cause: Code change fix done for dynamically reallocating the RAID MAP DATA.
  - Fix: Move the RAID_MAP Data structure to end of the PER_LUN_MEMORY structure.
  - Exposure: Windows
  - Risk: Low

- Fixed an issue where after freeing the Storport pool memory the buffer pointer was not set to NULL. This caused the ReleaseLun Memory to call the `StorPortfreePoolMemory` again, which causes the BSOD.
  - Root Cause: During a code fix to support 4kn drives, freeing of Pool Memory was moved into a generic function and resetting the Buffer Pointer to NULL was missed.
  - Fix: Set the Buffer Pointer to NULL after successfully freeing the Pool memory.
- Fixed the BSOD DRIVER_IRQL_LESS_OR_EQUAL(SmartPqi.sys) after controller lockup and server reboot.
  - Root Cause: While executing controller lockup test, the SmartPqi driver will change the controller state to failed. This prevents all the subsequent PNP and POWER SRB to complete without submitting to the controller. During restart/uninstall, the storage stack calls the StorRemoveDevice/StorStopDevice PNP action. As the HBA state is failed, the driver will immediately complete with SUCCESS status. Eventually for all the PNP actions no cleanup will be done. As no cleanup is done, the timer will be active and it will execute periodically. BSOD will happen when the code section is swapped out as part of unloading the driver and subsequently the timer callback executed.
  - Fix: Cancel all the timers in "`ScsiStopAdapter`" when the HBA state is failed.
- Fixed the issue where the RAID_MAP Disk Data Buffer in PER_LUN_MEMORY structure is statically allocated and it is limited to the size of 256. It can accommodate only 256 drive handles, so RAID volumes with more than 16 drives require more than 256 handles.
  - Root Cause: Provide static allocation of RAID_MAP_DISK Data buffer.
  - Fix: Recalculate the RAID_MAP_DISK date memory requirement and dynamically reallocate the memory buffer for RAID_MAP_DATA and PER LUN Memory.
- Fixed an issue where the driver version was not appearing in the controller UART.
  - Root Cause: The BMIC Host Driver version structure has field "driver_version" and it can accommodate a maximum of 32 characters. If we create a string "DRIVER_VERSION_HEALTH_WELLNESS_STR" with more than 32 characters then the string copy will be failed and it makes the destination string to be NULL terminated.
  - Fix: Modify "DRIVER_LOCAL_STR" on both DBG and release version to fit into maximum of 32 characters.
  - Risk: Low

2.2.3.5 Fixes and Enhancements for VMware Driver Build 1.0.3.2035

This release provides the following fixes and enhancements:

- Fixed an issue where the crash dump was not successfully generated in NUMA machine.
  - Root Cause: It is due to a recursive panic, that is, the driver crashes while saving the crash dump. A page fault is happening in the dump handler function in the driver. In the crash dump handler function, driver is supposed to poll each response queue for the IO completion. Number of response queue to iterate is decided based on the number of interrupt vectors allocated which is not correct.
  - Fix: The driver should decide based on the number of operational queue created instead of number of interrupt vectors allocated.
- Fixed an issue for data store unmount failure after trying driver unload. When the OS is on the controller and trying to unload the driver, unload will fail. We will not be able to mount datastore back after this and system will hang.
  - Root Cause: ESXi will try to unload the driver and it fails to do so as the OS is on controller. As a part of driver unload, ESXi sets "ioAllowed" for HBA to false and notifies the driver. After that, the driver will reject all the IOs to the HBA and that can lead to OS hang.
  - Fix: No need to block IOs based on ioAllowed flag. If OS send IOs, driver shouldn't block that as long as controller and drives are up.

- Fixed an issue for the discovery failure. During rescan devices a controller command is failing and that result in rescan failure.
  - Root Cause: The controller command was completed with underflow status and the driver was treating that as failure.
  - Fix: Corrected the error handling for internal controller commands.
- Fixed an issue where "Raid map too large" when RAID5 created with 33 SSDs. The RAID map data exceeds the statically allocated RAID map structure size.
  - Root Cause: The structure has raid map data and the size of raid map data is restricted. This may not be enough always. If there are more than 1024 entries, the driver will throw a warning and IOBypass will get disabled.
  - Fix: Dynamically allocate memory for the RAID map structure.
- Fixed an issue for PSOD after initiating two controller lockup. The PSOD was observed while performing array creation and deletion in a loop followed by two controller test lockup.
  - Root Cause: Since test lockup was initiated on two connected controllers one after the other during array create and delete operation, rescan devices and controller lockup handling paths are executing concurrently and trying to access the device list.Thus, it's causing rescan devices to access device memory which is de-allocated in controller lockup handling path.
  - Fix: Wait for rescan to complete in controller lockup handling path before un-claiming and removing the devices.
- Fixed an issue for PSOD during driver unload after controller lockup. The PSOD occurred after controller test lockup and followed by a reboot from ESXi shell
  - Root Cause: After controller firmware has locked up or went to offline state, the driver will unclaim all scsi devices paths and de-allocate the device memories. Since reboot was initiated after controller lockup, the drivers will be unloaded before system is booting up. During unload, the driver is trying to free the SES (Expander) device memory which was already deallocated at controller lockup handling path. So deallocating the non-existing memory is causing PSOD.
  - Fix: Once SES device memory has freed, make NULL assignment to the SES device entries in scsi device list array, ensure it does not get freed a second time.
- Fixed an issue for driver unload failure after expander device was removed. The driver unload operation failed after performing expander device cable hot remove and re-insert.
  - Root Cause: Unload operation was failing due to memory leak from driver. VMware driver was not exposing Enclosure device (SES Device) entry to the PSA layer of VMkernel. All physical and logical target device memory was freeing from Scsi Adapter discover probe function, which will be invoked by PSA when device path has been destroyed. Since Expander device is not exposed to the PSA, SES device Memory and Target ID are never freed even if path has been destroyed.
  - Fix: Free the SES device memory and Target ID when device has been removed.
- Added a fix for PSOD while shutting down device driver. After initiating reboot from ESXi shell, PSOD was observed while doing hard drive hot remove/re-insert. Hard drive hot plug was done while ESXI DCUI displaying "shutting down device drivers" message. At this point system will try to unload the all device drivers.
  - Root Cause: Since hot plug event reported by firmware, the driver hot plug event handling and driver unload operations are executing concurrently. After processing the event, driver will schedule the Helper queue request (thread) to acknowledge the event and to update the new devices list. The moment driver sends acknowledge event to the firmware, other thread has freed the driver resources such as inbound and outbound queue memory. Thus it is leading to the PageFault(PF) exception from the SmartPQI driver.
  - Fix: During a driver unload operation, before freeing the PQI Queue resources, check and cancel any pending requests in the helper world and then destroy helper queue. Also modify the sequence of destroy interrupt resources in driver unload path.
- Added an enhancement to honour the CPU count while creating operational queues.

- Root Cause: Number of online CPUs are not considered for creating operational queues. The queues were created based on the number of MSIX available.
- Fix: Queues are created based on the number of active CPU cores and interrupt count allocated to the driver.
- Risk: Low

- Fixed an issue to resolve unresponsive disks after a vSphere Quick Boot is executed.
  - Root Cause: During the driver unload, driver requests the firmware to flush the cache. While sending the cache flush request, the driver was giving the hint of system shutdown. Due to this, the firmware was sending START/STOP UNIT command to all direct attached drives to spin down them. During quick boot, driver will get loaded again, drives will remain in spin down state and all the medium access commands to those drives will fail with sense data as 02/04/02 (NOT READY/LOGICAL UNIT NOT READY/INITIALIZING COMMAND REQUIRED). Upon receiving this response, ESXi storage stack is supposed to send a START/STOP unit command to spin up those drives. Currently it is not supported in ESXi. Due to this limitation in ESXi, drives will not be accessible and remain offline and datastores will not be available.
  - Fix: During driver unload, driver should request cache flush without giving hint of shutdown, so that the firmware will not spin down the drives. Since the live-install is not supported, a system reboot is required to do an upgrade (for the new driver with this fix).
  - Exposure: Direct attached drives connected through internal ports.
  - Risk: Medium
- Fixed an issue where the unload driver operation was failing.
  - Root Cause: During a driver unload operation, a cache flush is performed. If the controller is in a lockup condition, firmware becomes inactive and the cache flush request fails. Due to the cache flush request failure, the driver detach function returns an exit status of failure and that caused the unload operation to fail.
  - Fix: When the cache flush returns failure due to controller offline condition, the detach function takes it as a success condition and unloads the driver.
  - Risk: Medium
- Fixed an issue where the memory allocation failure was happening under heavy load.
  - Root Cause: DMA memory allocation is non-blocking. This may result in allocation failure if there is no enough memory available.
  - Fix: Memory allocation logic is changed to blocking. It should be OK as all the dynamic memory allocation is in blocking context.
  - Risk: Low
- Fixed an issue when in vSAN multipath environment, the drives became offline when the active path is removed.
  - Root Cause: When a path is removed, the controller firmware returns PQI_AIO_STATUS_AIO_PATH_DISABLED error status for the pending IOs. Upon receiving this, the driver reports "VMK_SCSI_HOST_ERROR" to PSA, and that results in IO retry through the same path. This might delay the path switching in dual paths configuration and vSAN reports disk failure.
  - Fix: Return "VMK_SCSI_HOST_NO_CONNECT" instead of "VMK_SCSI_HOST_ERROR" in case of PQI_AIO_STATUS_AIO_PATH_DISABLED error, which will trigger the path switching and IOs will be retried immediately through the alternate path.
  - Risk: Medium
- Fixed an issue when there is a failure in mounting data store after the driver unload-load sequence.
  - Root Cause: In scenario of both driver load/unload test and system upgrading, the device will go through detach->attach without power cycle happened between these two stages. During "detach" driver was sending cache flush with "SHUTDOWN" indication and the firmware spin down the devices, but no one spin them up during attach. This will cause the device to be in accessible after driver load.

- Fix: In detach, do cache flush without SHUTDOWN.
- Risk: Low

- Fixed an issue when the system crashes during driver reboot.
  - Root Cause: When the driver is unloaded, we are not waiting for all submitted internal commands to get completed. If any internal commands are pending at the time of driver unload, driver will not wait for it and proceed with clean up of resources. This will cause a page fault when the command is completed later.
  - Fix: Wait for all internal commands to get completed before clean up of the resources.
  - Risk: Low
- Fixed an issue where the incorrect target SAS address was displayed under multipath information for external RAID arrays.
  - Root Cause: SAS address assignment was not correct for external raid device. Driver was using "lun id" of the `report_logical_lun` response data as the SAS address that is incorrect. We have to use the `report_phys_lun` response data for the corresponding device to get the SAS address.
  - Fix: Corrected SAS address for logical,physical and external raid targets.
  - Risk: Medium
- Added an enhancement to remove "Below 4 GB" DMA memory usage. The driver always requests the physical memory address below 4 GB range and there might not be any DMA memory available in that range.
  - Root Cause: DMA memory allocation was failing when there is no DMA memory available below the 4 GB range.
  - Fix: Removed "Below 4GB" DMA memory usage.
  - Risk: Low

2.2.4 Management Software Fixes

2.2.4.1 Fixes and Enhancements for Arcconf/maxView Build B23484

This release includes the following fixes and enhancements for Arcconf/maxView Build B:

- Fixed an issue where modifying the logical device size results in a "Segmentation Fault", if an incorrect size is provided as an input.
  - *Root Cause*: Accessing invalid command line argument pointer resulted in a crash.
  - *Fix*: Added a command line argument check to fix the crash and display a valid error message.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the "LEGS" option is ignored while creating logical devices for RAID levels 50 and 60 using arcconf.
  - *Root Cause*: Spanned segments for RAID50 and RAID60 is set to default values always irrespective of the input provided for the ‘LEGS’ option.
  - *Fix*: Added a condition to check, if you have specified the ‘LEGS’ option value while creating logical devices with RAID50 or RAID60, to set the input value instead of the default value.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf fails to modify the logical device size with the different stripe size.
  - *Root Cause*: Missed the validation to check for the supported stripe sizes while assigning the stripe size value.

- *Fix*: Assigned the provided stripe size value as an input to modify and added a condition to check about the supported stripe sizes.
- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where arcconf does not remove the "Mount Point(s)" property for physical devices used while creating the Windows storage pool.
  - *Root Cause*: arcconf displays "Mount Point(s)" property for physical devices, when a device is associated with a logical device, array or Windows storage pool.
  - *Fix*: Added changes to display the "Mount Point(s)" property only when device is not associated with any logical device, array or Windows storage pool
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the arcconf modify command does not consider the stripe size parameter while modifying the RAID level.
  - *Root Cause*: Missed the controller specific check for Smart controller when modifying both RAID level and stripe size of the logical device.
  - *Fix*: Added the controller specific check when modifying both RAID level and stripe size of the logical device.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf does not allow to modify the logical device name when the expansion task is in progress.
  - *Root Cause*: Missed the controller specific check, when verifying the state of logical device and tasks are in progress, resulted in failure.
  - *Fix*: Added the controller specific check, when verifying the valid state of logical device and tasks are in progress.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf command output shows the "operation completed successfully" message. Though, it fails.
  - *Root Cause*: Return code was successful by default due to which the command displays the successful completion message even when the operation reason is not available.
  - *Fix*: The default return code is set to failure to display a proper failure message.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf does not allow to set the "Boot Type" when the expansion task is in progress.
  - *Root Cause*: The operation to set the "Boot Type" is blocked when the expansion task is in progress.
  - *Fix*: Added changes to allow users to set the "Boot Type" while the expansion task is running on a logical device.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the SMP passthrough command does not accept the legacy commands.
  - *Root Cause*: Missed the changes to make the enclosure and connector as optional in the SMP command for smart controllers to accept the legacy SMP command.
  - *Fix*: Added code to implement enclosure and connector as an option in the SMP command for smart controllers.
  - *Exposure*: All releases.

- *Risk*: Low.

- Fixed an issue where arcconf does not allow to change the device cache policy when all connector modes are in HBA.
  - *Root Cause*: The SETCACHE command is blocked if a controller is in HBA mode.
  - *Fix*: Added changes to allow setting for device write cache policy when controller is in HBA mode.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the controller lockup is observed when a device under erase is hot removed while firmware flash is in progress.
  - *Root Cause*: arcconf sends the pause and resume background activity with "disable hotplug events" set to true while waiting for the erase operation to be completed, which results in the issue.
  - *Fix*: Code changed to remove the pause and resume background activity operation as lower layers internally do the same on firmware flash.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf does not show the minimum allowed size message during the maxCache creation, resulting in failure to create maxCache.
  - *Root Cause*: Size for the minimum maxCache creation is mapped to a wrong value.
  - *Fix*: Changed the minimum size value for creating maxCache to an appropriate one for a logical device.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the failure response was observed while deleting all logical devices.
  - *Root Cause*: Whenever arcconf deletes all logical devices, it deletes cache arrays first followed by data arrays. This causes a failure to delete the stale cached array.
  - *Fix*: Added changes to delete the data arrays which in turn deletes the cached array in the correct order.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the clear configuration operation is allowed for HBA controller without any devices connected.
  - *Root Cause*: arcconf did not check for the boot volume and clear it while setting the controller configuration to default in HBA controller.
  - *Fix*: Added code to check for the boot volume and clear it while setting the controller configuration to default in HBA controller.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the LEGS option is ignored when modifying a logical device in arcconf.
  - *Root Cause*: Validation for the LEGS option was missed in arcconf while blocking the modify command.
  - *Fix*: Added validation based on the LEGS option so if there is no change in all parameters. The user requested changes will be applied to the existing configuration on any input.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where arcconf is unable to collect the support archive in ESXi6.5_U1 OS.
  - *Root Cause*: The buffer to hold the firmware log was getting corrupted due to stack overflow.
  - *Fix*: Added changes to hold the the firmware log buffer by resolving stack overflow.

- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where modifying the logical device size to the maximum value failed.
  - *Root Cause*: Maximum size for the expand logical device operation was not updated to the valid size from the lower layer.
  - *Fix*: Added code to set the valid maximum size from the lower layer for expanding the logical device size.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where only one spare gets assigned to array, which has predictive failure device when provided with two devices to set as spares.
  - *Root Cause*: While the assignment for the second spare is in progress, the rebuild starts with the earlier spare assigned, resulting in failure to assign the second spare.
  - *Fix*: Added changes to assign multiple spares at the same time to array.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where SSD Over Provisioning Optimizatin (OPO) is not enabled on the SSD logical device creation.
  - *Root Cause*: On the SSD logical device creation, the value for SSD OPO was always set to false instead of accepting the user input.
  - *Fix*: Added changes to accept the SSD OPO input from the create logical device wizard.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the device information logs are dumped in Linux kernel every 5 seconds, when maxView service is installed.
  - *Root Cause*: As the event poll rate was fixed at 5 seconds, OpenStatus was printed periodically while trying to detect both ARC and Smart family of controllers.
  - *Fix*: Made the event polling rate configurable in the installation directory.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the "Unsupported" devices are listed for array creation, which results in operation failed when selected.
  - *Root Cause*: In maxView, the check to block devices of type "Unsupported" to the list in the create logical wizard was missed.
  - *Fix:* Added changes to block devices of the "Unsupported" type from the list in the create logical device wizard.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an uninstallation issue with StorMan in Ubuntu OS.
  - *Root Cause*: Firewall entries that are not required for the `StorMan.deb` package is disabled, which results in failure of the script during uninstallation.
  - *Fix*: Made changes to the maxView .deb package to process the invalid firewall entry correctly.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxView does not allow to shrink the parity groups for RAID level 50.
  - *Root Cause*: When the user selects the Shrink by parity group option for the shrink array operation, it does compare with the wrong option.

- *Fix*: Added code changes to compare the user input with the valid shrink option for execution.
- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where maxView webserver does not get started with SLES 15.
  - *Root Cause*: The maxView webserver uses the default JRE installed with SLES 15, which fails to start the webserver.
  - *Fix*: Made changes in the script to use the JRE packed with maxView instead of the installed JRE in SLES 15.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the system is in the "Critical" state under the "Enterprise View" even after re-enabling the failed logical device.
  - *Root Cause*: The logic to parse through the child nodes to fetch the highest severity was not used for updating the tree to display the system status.
  - *Fix*: Made code changes to parse the child nodes to fetch the highest severity while updating the tree for displaying the system status.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the maxCache statistics for the "Read-Write Cache Hits" property shows a negative number.
  - *Root Cause*: By clicking the "Advanced statistics" tab for maxCache device, the calls are issued for maxCache statistics and advanced statistics, which display wrong values for the properties.
  - *Fix*: Removed the call for advanced statistics and used maxCache statistics for maxCache device to fetch correct values.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the "Task" Tab displays RPI instead of BPI after the expansion task is completed on RAID 50 logical devices.
  - *Root Cause*: After completing the expansion task, the state of the logical device was not updated and displayed as "Rapid parity initialization" instead of "Background parity initialization" for few seconds.
  - *Fix*: Added the condition to check the state of the logical device and display the proper string in the "Task" tab.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the "Create logical device" icon is enabled when all devices are in optimal state and no space is available for Array.
  - *Root Cause*: In maxView, the check for free space was also considered for cache array which resulted in enabling the "Create logical device" icon.
  - *Fix*: Made the code changes to ensure only data arrays are checked for unused space to enable or disable the "Create logical device" icon.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the maxView GUI does not automatically update the list of added or removed devices from the server.
  - *Root Cause*: One of the attributes for events was not assigned with a proper value which failed to update the new configuration in GUI.

- *Fix*: Added the proper value to the event attribute and the logic to update the configuration that comes along with the event object.
- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where the maxView installation summary displays "Not Installed" instead of "Repair" after performing the repair task in maxView.
  - *Root Cause*: Installer did not refer to the "Repair" operation return value, displaying wrong messages.
  - *Fix*: Made changes to check the return values for the features during the "Repair" operation.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxView freezes after applying changes to the web server session timeout settings.
  - *Root Cause*: While writing a new value to the `web.xml` configuration file for web server session timeout, the atmosphere connection was broken, and the web page is frozen.
  - *Fix*: Added changes to store the web server time-out value selected by the user in the memory and write it to the `web.xml` file during shutdown of tomcat.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the spare device type from "Auto-Replace" to "Dedicated" is not updated in the tree view and in the "Array Resources" tab.
  - *Root Cause*: In maxView, while changing the spare device type, the physical device resource is called, which resulted in a failure.
  - *Fix*: Added changes to call array resource instead of physical device to change the spare type.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxView does not show the maxCache Statistics when Advance Statistics is disabled.
  - *Root Cause*: The "maxCache Advanced Stats" tab is disabled when the controller advanced stats are set to disabled.
  - *Fix*: Added a fix to enable the "maxCache Advanced Stats" tab irrespective of the controller advanced stats settings.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue to receive event email notification from maxView when notification type is configured as "Coalesced".
  - *Root Cause*: If the notification type was configured as "Coalesced", maxView did not send the collected events by e-mail as the condition was not set correctly.
  - *Fix*: Added a check to collect the events and send the collection of events in one mail, when the notification type is set as "Coalesced".
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the connector change events are not generated in maxView.
  - *Root Cause*: Events for device status are not generated when the connector changes from one mode to another.
  - *Fix*: Added changes to backup the old device status and compare it with the updated device status to generate proper events.
  - *Exposure*: All releases.

- *Risk*: Low.

- Fixed an issue where the "Force Offline" icon is enabled in maxView for failed devices.
  - *Root Cause*: Even after changing the dedicated spare device to be offline, the state of the device still shows as the hot-spare.
  - *Fix*: Added a check for a physical device to verify whether the device is in failed state and disable the “Force Offline” icon.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxView shows failure for maxCache write-back or write-through changes, but the changes are reflected in ESXi 6.7
  - *Root Cause*: On creating a logical device or maxCache, the OID was not updated properly.
  - *Fix*: Added a fix to update the OIDs properly after creating or deleting a logical device or maxCache.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where no message is displayed while trying to change the boot type property of the logical device of type "Split Mirror Set Primary Device."
  - *Root Cause*: While setting the warning message, boot type related message was overwritten with the default message for the logical device in the AJAX call.
  - *Fix*: Added changes to set the warning message to default help text and a warning message is displayed only when necessary.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxCrypto set configuration operation is failed, when the "maxCrypto mode" is set to enabled and the "Firmware Lock" is set to locked.
  - *Root Cause*: When the "maxCrypto mode" is enabled and the "Firmware Locked for Update" is changed simultaneously, the call is failed to set.
  - *Fix*: Made the code changes to set the "maxCrypto mode" and initiate the "Firmware Locked for Update" call in order.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where a failure occurred while modifying the webserver timeout value for Xen 7.3 OS.
  - *Root Cause*: While stopping the tomcat service, maxView is unable to handle the SIGKILL signal, and it fails to modify the webserver timeout value.
  - *Fix*: Made changes in tomcat installer service to send SIGTERM signal during tomcat service stop to update the webserver timeout value correctly.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where no events are generated after changing the webserver timeout value in maxView.
  - *Root Cause*: The old and new values are not properly retrieved from the properties file for event generation.
  - *Fix*: Made the code changes to retrieve the proper value from the properties file and generate the events only for those properties which are changed.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the maxView desktop icon name is showed as `Storman.desktop` in SLES 15.
  - *Root Cause*: SLES 15 does not provide the "/etc/SuSE-release" file to find the SLES version for installer.

- *Fix*: Added changes in the installer to use "/etc/os-release" for SLES version identification.
- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where maxView does not provide an option to expand or migrate logical devices.
  - *Root Cause*: The Expand or Migrate ribbon icon was disabled as the "Write cache mode per logical device" support check was missed for the controller, which did not support the DDR cache backup power source.
  - *Fix*: Added the check for the "Write cache mode per logical device" support for controllers, which did not support the DDR cache backup power source and enabled the Expand or Migrate ribbon icon.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where a spare is assigned as a dedicated hot-spare for more than one arrays and allows to change as auto replace to any other array.
  - *Root Cause*: In maxView, when the same device is shared as a dedicated hot spare for more than one arrays where no other ready device is available for a new spare assignment, the check to block the user action and a proper warning message are missed.
  - *Fix*: Added changes to block the user action with a proper warning message when the same device is shared as a dedicated hot spare among more than one arrays when there is no other ready device available for spare assignment.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the automatic system refreshed event is observed in maxView for every 5 minutes on Xen 7.3.
  - *Root Cause*: When the web server timeout is set to zero (0), the refresh event is generated continuously.
  - *Fix*: Added changes to reload the web page without generating the event when the web server timeout is set to zero (0) or the user is in idle state for approximately 5 minutes.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the sanitize erase option is disabled in HBA mode.
  - *Root Cause*: The check was missed for the sanitize or legacy erase support on HBA device.
  - *Fix*: Added support for sanitize or legacy erase in the HBA mode along with RAID and Mixed mode.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the move logical device icon is disabled when moving from 512 block sized array to 4 K array.
  - *Root Cause*: While iterating through the array, it considers the last logical device as a selection which does not support the move operation.
  - *Fix*: Added changes to point to the selected logical device for move operation.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the set properties operation fails while changing the connector mode to HBA when there are no devices connected.
  - *Root Cause*: When the controller has a failed device with unknown phy information, the connector mode change fails.
  - *Fix*: Added the check for connector phy information and to block the mode change from RAID or Mixed to HBA when any device phy is set to unknown.

- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where maxView allows the user to change connecter modes from RAID to HBA in a dual-domain configuration.
  - *Root Cause*: The dual-domain check was missed in the parsing logic and the connector mode change option was not blocked for the active path connector(s).
  - *Fix*: Added the logic to block the connector mode change when it is configured with the dual domain active path.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the "Spare Management" icon is disabled while creating a spare for non-redundant logical devices when the spare activation mode is in "Predictive" state.
  - *Root Cause*: For enabling or disabling the "Spare Management" icon, the spare activation check was missed for non-redundant logical devices.
  - *Fix*: Added changes to check the spare activation mode for enabling or disabling the "Spare Management" icon for non-redundant logical devices.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the second logical device in the array is not showed in maxView.
  - *Root Cause*: While parsing the chunks of a logical device, value for start sector attribute was out of range resulting in an exception, and failure to display the second logical device.
  - *Fix*: Made code changes to use proper data type conversion for the start sector attribute to display the second logical device.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where support archive collection from remote maxView does not generate ADU, Smart stats, and SSD Report in ESXi 6.7.
  - *Root Cause*: Support archive freezes inconsistently while executing an operation from firmware.
  - *Fix*: Added a condition to check the response of the operation from firmware and collect logs based on the operations.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxCache write-back and write-through changes show successful from maxView, the changes are not updated when the logical device rekey is in progress.
  - *Root Cause*: For the maxCache write-back and write-through settings, the operation was not called from the logical device node.
  - *Fix*: Made changes to set the maxCache write-back and write-through settings from the logical device node.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue to set write cache policy for HBA devices in maxView for HBA controller.
  - *Root Cause*: The AJAX call for disabling or enabling the OK button for the HBA controller was not defined for the device write cache policy.
  - *Fix*: Made code changes to add the AJAX call for disabling or enabling the OK button for the HBA controller for the device write cache policy
  - *Exposure*: All releases.
  - *Risk*: Low.

- Fixed an issue where there is no event and update in maxView for logical device mount or unmount.
  - *Root Cause*: "Disk Extent Mount Point" was not updating the mount point information while querying for a configuration refresh.
  - *Fix*: Adding changes to update the mount point information along with refreshing the configuration.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where maxView does not show the cache line size information for the RAID 5 or RAID 10 cache logical device.
  - *Root Cause*: For maxCache RAID 5 and RAID 10, the cache line size information was not displayed as one of the properties while calculating the size.
  - *Fix*: Made code changes to display the cache line size information for the RAID 5 or RAID 10 maxCache if it is supported.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the controller mode change operation was enabled even when the failed devices are available.
  - *Root Cause*: In maxView, whenever the controller had any of the degraded logical device or array, the controller mode change operation was enabled.
  - *Fix*: Made code changes to block the controller mode and connector mode change when a degraded logical device or array is available.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the auto replace hot spare operation is successful when the array has a non-redundant logical, and the spare activation mode is set to failure.
  - *Root Cause*: Auto replace spare rebuild mode is not blocked based on the supported array rebuild modes.
  - *Fix*: Added condition to display an error message when the spare rebuild mode is not supported.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the restore configuration fails, when an array with more than one logical device is configured with maxCache.
  - *Root Cause*: Order for creating the logical device was mismatched in restore configuration, resulting in failure.
  - *Fix*: Added code changes for creating logical devices in same order as input configuration.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the heal array operation lists SSDs for an array that is created with HDDs.
  - *Root Cause*: In maxView, the wrong API was called for the heal array operation, which listed the wrong device type.
  - *Fix*: Added code to call the correct API to display device type same as the array for the heal array operation.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the maxView set properties operations fail, when the consistency check is set to high on ESXi 6.7.
  - *Root Cause*: In maxView, when the consistency check is set to high, there is an exception while setting the controller properties for ESXi 6.7.

- *Fix*: Made proper changes to parse the values of consistency check delay and consistency check mode before setting the controller properties in ESXi 6.7.
- *Exposure*: All releases.
- *Risk*: Low.

- Fixed an issue where physical devices disappeared from maxView while creating the Windows Software RAID.
  - *Root Cause*: When partition is created as dynamic in disk management, there is a null pointer exception that stops displaying the physical devices in the tree.
  - *Fix*: Added the check for the dynamic partition information for display by handling the null pointer exception.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Fixed an issue where the crypto erase task cannot be started on a logical device while the background parity initialization runs on a logical device.
  - *Root Cause*: When the "Background Parity Initialization" task runs on a logical device, the crypto erase task is not allowed.
  - *Fix*: Removed the "Background Parity Initialization" task validation to allow the crypto erase operation.
  - *Exposure*: All releases.
  - *Risk*: Low.
- Added support for boot port controller in maxView/arcconf
- Added support maxView vSphere plugin for ESXi 6.7

2.3 Limitations

2.3.1 Firmware Limitations

2.3.1.1 Limitations for Firmware Release 1.98 Build B0

This release includes the following limitations:

- Controller hangs when rebuilding is triggered on a RAID logical volume with the controller cache disabled, while heavy I/O is initiated from the host during this time. The issue is observed only with the HDD based RAID volumes, with the controller cache disabled.
  - Workaround: Enable the RAID volume's cache so that the controller hang condition can be avoided.
- A read performance drop is observed on RAID 1 logical volumes for block sizes 64 KB or higher for the sequential read workloads.
  - Workaround: None
- Controller cache will not be converted into 100% read cache, if any backup power source cable error, charge or charge timeout error occurs when expansion or transformation task is active.
  - Workaround: None
- Host I/O’s run in parallel with a logical volume expansion operation may lead to a controller hang.
  - Workaround: If the system is encountering hangs during an expansion, then the host IOs will need to be minimized until the expansion process completes.
- During a logical volume rekey operation, if a system is shutdown and the physical drives in that logical volume are moved to different slots, then the controller will be in an abnormal volume state on the next boot causing the logical volume to be inaccessible.
  - Workaround: While a logical volume rekey operation is running on a logical volume, ensure it completes before moving any of the physical drives.

- When I/Os are performed on drives that respond slowly or which does not respond to the READ or WRITE commands, and when Secure Erase is performed on other SATA drives, I/Os are stalled for longer time. The time the I/Os are paused depends directly on the amount of unflushed data on the cache and speed with which the device responds to error recovery.
  - Workaround: None
- Performance drop is observed on certain queue depth for the 4 KB sequential write workload on RAID logical volumes with IOBypass and DDR caching disabled.
  - Workaround: Enable the DDR caching for RAID 0 and RAID 1 volumes, to avoid this problem. There are no known workarounds for parity RAID volumes such as RAID 5 or 6.
- When running I/Os on multiple RAID 0 and RAID 1 logical drives, there is a corner timing issue where DMA and COMPLETION threads are dead locked and controller triggers lockup since it does not see requests being completed even after recovery mechanisms.
  - Workaround: None
- When a drive is hot removed from the system, occasionally, the event is recorded as a failure to update the RAID configuration data (event code 0x20), instead of indicating that as a hot removal (event code 0x14).
  - Workaround: None
- I/O performance drop is noticed when running I/Os on four RAID 0 logical drives, with eight physical devices each, compared to three such logical drives.
  - Workaround: None
- In some cases, the logical drive state is reported as “abnormal state” during reboot on encryption-enabled adapters. This is observed while a re-keying operation is in progress and the drive location changed after the reboot.
  - Workaround: Restore the location of the physical drives back to the same location before the system boot

2.3.1.2 Limitations for Firmware Release 1.32 Build 0

- Firmware release 1.32b0 may become unresponsive while attempting to flash firmware or execute other RAID logical volume operations.
  - Description: Refer to entry "Fixed an issue where firmware may become unresponsive while attempting to flash firmware or execute other RAID logical volume operations" in the Firmware fixes section.
  - A fix for this issue is available in the 1.60 B0 firmware release. If a firmware flash failure is occurring, try the following workarounds:
    - *Workaround:* If there are no target devices (expanders or drives) attached to the controller, attach a target device to the controller and try the host management operation again.
    - *Workaround*: If the system is operating using UEFI, the HII tool can be used to flash the firmware to this release as outlined in the *Microsemi SmartIOC 2100/SmartROC 3100 Installation and User's Guide,* appendix entry “Updating the SmartIOC 2100/SmartROC 3100 Controller Firmware”.
    - *Workaround*: If there are target devices attached to the controller and this issue occurs or none of the workarounds can be used, contact Microsemi Support.

2.3.2 UEFI Limitations

2.3.2.1 Limitations for UEFI Build 1.3.5.2 /Legacy BIOS Build 1.3.5.2

There are no known limitations for this release.

2.3.3 Driver Limitations

2.3.3.1 Limitations for Linux Driver 1.2.4-065 Build

This release includes the following limitations:

- Force reboot (reboot -f) might cause controller lockup if outstanding IOs are pending in the controller.
  - Workaround:
    - Avoid using “Reboot -f”
    - Ensure all the IOs are completed before issuing “reboot -f”
- Occasionally, SmartPQI driver might not unload and the OS reports “module is in use”
  - Workaround: Disable SMARTD Daemon.

2.3.3.2 Limitations for Windows Driver Builds 106.84.2.64

There are no known limitations for this release.

2.3.3.3 Limitations for FreeBSD Driver Build 1.0.3.2035

There are no known limitations for this release.

2.3.3.4 Limitations for Solaris Driver Build 1.0.3.2035

There are no limitations for this release.

2.3.3.5 Limitations for VMware Driver 1.0.3.2035

This release includes the following limitation:

- On creating a logical drive in ESXi 6.7, the following events are generated in vmkernel logs. This is just a warning and there is no functional impact.

```
2018-10-11T05:10:09.970Z cpu24:2097701)WARNING: ScsiScan: 2329:
Skipping out-of-range LUN vmhba3:C1:T4:L512 Max LUN ID supported by
the driver is =256
```

2.3.4 Hardware Limitations

This release includes the following hardware limitations:

- Two Wire Interface (TWI) address conflicts can cause system DDR memory to not be discovered.
  - *Description*: The SmartRAID 3100 and SmartHBA 2100 boards include two TWI targets on the host-facing SMBUS interface with the following slave addresses:
    - 0xA0 – Field Replaceable Unit (FRU) SEEPROM
    - 0xDE – PBSI (default)

    According to the JEDEC specification, the default TWI addresses for the DDR SPD is 0xA0-0xAE (the spec uses 7 bit addressing which is 0x50-0x57). On platform system board designs with SMBUS wiring that has both PCIe slots and DDR slots shared on the same TWI bus, the TWI devices for the DDR and Smart controller are exposed to address conflicts which can result in the system memory not being discovered. The Smart controller PBSI interface defaults to a value of 0xDE (0x6F in 7-bit addressing) and is not a problem unless it is changed to an address that conflicts with the JEDEC defined values. The Smart controller FRU SEEPROM is hardwired to 0xA0.

- *Workaround*: None available. If this issue is encountered, contact your Microsemi support engineer to determine the next steps for your system.
- *Performance with workaround*: Not applicable
- *Performance without workaround*: Not applicable

2.3.5 Management Software Limitations

2.3.5.1 Limitations for Arcconf and maxView Build B23484

The following are the limitations of arcconf and maxView in this release:

- Remote-Arcconf tool does not support ESXi 6.7U1
  - *Workaround:* Use maxView GUI or the local Arcconf for configuring ESXi 6.7U1
- maxView GUI may not respond or respond with slowness, in a high config server with more than 100 drives running heavy I/O with enclosure power going off/on frequently.
  - *Workaround:* Restart the maxView redfish server service to bring the maxView GUI to a normal operational state.

3 Updating the Board Firmware for PQI Operation

This section describes how to update the board's firmware components to the latest release.

3.1 Updating Controllers to latest (PQI) Firmware

This procedure describes how to prepare your board to be programmed with the latest board PQI firmware.

Note: Complete these procedures exactly as described for proper functionality. If you do not follow all of the steps correctly, you could encounter unusual runtime behavior.

Flashing the board to the latest PQI firmware:

This section describes how to update all the firmware components on SmartHBA 2100 controller boards to the latest release.

If the controller is currently running 1.60 b0 firmware, follow these steps:

1. **Mandatory:** Flash the target with the provided " SmartFWx100.bin" image with arcconf/maxView software.
2. **Mandatory:** Cold boot the system to refresh all components**.**

If the controller is currently running 1.32 b0 firmware, follow these steps:

1. **Mandatory:** Flash the target with the provided “SmartFWx100.bin" image with arcconf/maxView software.
   - If the arcconf/maxView software becomes unresponsive or hangs then power cycle the system to recover and refer to firmware limitation section *Limitations for Firmware Release 1.32 Build 0* on page 43.
2. **Mandatory:** If flashing completes, cold boot the system to refresh all components.

If the controller is currently running 1.04 b0 firmware, follow these steps:

1. **Mandatory:** Flash the controller with the provided “SmartFWx100_ v1.29_b314.bin" image with arcconf/maxView software.
2. **Mandatory:** Reboot the system to refresh all components**.**
3. **Mandatory**: Flash the target with the provided " SmartFWx100.bin" image with arcconf/maxView software.
4. **Mandatory**: Cold boot the system to refresh all components.

At this point, the controller would be updated and would be ready to use. Install the SmartPQI driver and the latest version of the Arcconf/maxView management utility to monitor and configure the controller.

Note: Downgrading firmware could lead to unexpected behavior due to an incompatibility in SEEPROMs between this release and the prior release.

4 Installing the Drivers

See the "Microsemi Adaptec® SmartRAID 3100 Series and SmartHBA 2100 Series Host Bus Adapters Installation and User's Guide (ESC-2171547)" for complete driver installation instructions.

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions; security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

ESC-2161026