Release Notes

SmartHBA 2100 and SmartRAID 3100 Software/Firmware

Released

August 2019

Revision History

Revision	Revision Date	Details of Change
18	August 2019	Updated for SR 2.4.8
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.

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/SmartRAID 3100 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	2.4.8
Package Release Date	August 13, 2019
Firmware Version	2.30 B0[1,2] (basecode 06.048.003.000)
UEFI Version	1.3.8.2
Legacy BIOS	1.3.8.1
Driver Versions	Windows SmartPQI: • Windows 2012/2016/2019: 106.100.0.1014 • Windows 7/2008: 6.100.0.1014 Linux SmartPQI: • RHEL 6/RHEL 7/RHEL 8/SLES 12/SLES 15: 1.2.8-026 • Ubuntu 16/18: 1.2.8-026 • CentOS 6/7/8: 1.2.8-026 • Debian 8/9: 1.2.8-026 VMware SmartPQI: • VMWare ESXi 6.0/6.5/6.7: 1.0.3-2323 FreeBSD/Solaris SmartPQI: • FreeBSD 11/12: 1.0.3-2323 • Solaris 11: 1.0.3-2323
arcconf/Maxview	B23600

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. Refer to the section " *Updating the Controller 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/SmartRAID 3100 controller SmartHBA 2100/SmartRAID 3100 controller and SmartRAID 3100 and SmartRAID 3100 controller solutions from the Microsemi Web site at *https://storage.microsemi.com/en-us/support/start/*

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
maxView firmware upgrade wizard	The firmware upgrade wizard allows to upgrade/downgrade the firmware and BIOS image to one or more controller(s) of same model in the system.	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 R2 SP1 and Windows 7	SmartPqi.sys	x64
		SmartPqi.inf	x64
		SmartPqi.cat	x64

Table 5 • Linux SmartPQI Drivers

Drivers	Version
Red Hat Enterprise Linux 8.0	x64
Red Hat Enterprise Linux/CentOS 7.6, 7.5[1], 7.4[1], 7.3, 7.2, 7.1	x64
Red Hat Enterprise Linux/CentOS 6.10, 6.9[1], 6.7, 6.6	x64
SuSE Linux Enterprise Server 12[1], SP4, SP3, SP2, SP1, and Base	x64

Drivers	Version
SuSE Linux Enterprise Server 15 SP1[1]	x64
Oracle Linux 7.6 UEK5u2, 7.5 UEK4	x64
Oracle Linux 7.2 with UEK 3.10.0-327.el7	x64
Oracle Linux 7.3 with UEK 4.1.12-61.1.18	x64
Oracle Linux 7.4 with UEK4 (4.1.12-94)	x64
Oracle Linux 7.5 with UEK4 (4.1.12-112)	x64
Ubuntu 18.04.2, 18.04.1	x64
Ubuntu 16.04.5, 16.04.4	x64
Debian 9.8	x64
Debian 8.11	x64
Citrix xenServer 8.0, 7.6	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 12.0, 11.2	x64
Solaris 11.3, 11.4	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	See the maxView Storage Manager download package for the OS-applicable installation executable.

Description	OS	Executable
	XenServer	
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 6 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	
MCTP BMC Management		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 2.30 B0

This release includes the following fixes and enhancements:

- Fixed an issue where logical drive transformation/expansion task fails when one of the physical drive is reset due to error handling.
  - *Root Cause*: The disk requests sent as part of a transformation task are not retried when returned with error completions due to a device reset. These requests should be retried until the retry limit is exceeded before failing the logical request on the logical drive.
  - *Fix*: Retry the failed physical requests instead of failing them immediately.
  - *Risk*: Low
- Fixed a 0x1ABD lockup due to a dead lock situation while processing request completions.
  - *Root Cause*: The completion task was not notified about request completions since the corresponding interrupts were not getting enabled to read the completion queue. This resulted in firmware not noticing the completions even after recovery mechanisms, which resulted in a lockup.
  - *Fix*: Re-enable the non-empty callback interrupts after processing completions to read the completion queue.
  - *Risk*: Low
- Fixed an issue issue where a drive under-going sanitize fails when inserted into a bay that was previously used for a configured drive.
  - *Root Cause*: The drive under-going sanitize is in HBA mode and when inserted into the bay of a configured drive which is in RAID mode, the mode of the HBA drive becomes RAID mode and it tries to become a part of the logical unit, which should not be allowed and it is failed for this reason.
  - *Fix*: Do not allow a sanitizing drive to be a part of a logical volume or become a spare drive.
  - *Risk*: Low
- Fixed an issue where an inquiry command completion is significantly delayed in a controller queue.
  - *Root Cause*: When the controller is run in low power mode, lower layer firmware threads are reduced to achieve power savings. This causes a single thread to handle all IOs and internal operations like rebuilding volumes etc. Queues are serviced based on priority scheduling and this caused the lower priority queues to starve for longer durations. One such low priority request was to DMA error data of an Inquiry command to the host.
  - *Fix*: Firmware modified to round robin service all queues in low power mode.
- Fixed an issue where the firmware attempts communication with a non-SES virtual device using SES protocol.
  - *Root Cause*: Lower layer firmware reported all virtual phys to upper layer firmware. Upper layer firmware treats all virtual PHY connections as SES connections. An error occurs when upper layer firmware tries to talk to a virtual device that cannot talk SES.
  - *Fix*: Lower layer firmware modified to hide virtual devices that are not SES targets from the upper layer firmware.
  - *Risk*: Low
- Fixed an issue where the Western Digital drives would not accept any IOs after running the Sanitize Crypto Scramble command.
  - *Root Cause*: This problem occurred only when using Western Digital drives. After the Sanitize Crypto Scramble had completed, the drive would not accept any IOs until about 9 minutes later.

The hdparm–sanitize-status commands report the Sanitize Crypto Scramble operation is complete; however, sending TEST UNIT READY commands to the drive shows that the firmware was returning LOGICAL UNIT NOT READY, SANITIZE IN PROGRESS. The firmware was out of sync with the state of the drive.

  - *Fix*: If firmware has set the "Sanitize in progress" flag for the target device, send Sanitize Status Ext in the background for any commands sent by host.
  - *Risk*: Low

- Fixed an issue where rebuild starts from zero after server reboot.
  - *Root Cause*: Rebuild progress data which is part of the RAID meta data on the disks, gets saved every 10 minutes during rebuilding. If the server is reboot or shutdown prior to this duration, rebuild will start over again.
  - *Fix*: Update and save the meta data with rebuild progress data in case of server reboot/shutdown.
  - *Risk*: Low
- Fixed an issue where the captured controller log is empty.
  - *Root Cause*: Any unimportant event (with class, subclass and detail as zero) logged to event buffer, is also written to persistent memory (MRAM/NVSRAM) during a controller reset. After the controller is reset, the same event is copied back to the event buffer. Host management software will issue commands to read the logs/events untill they get the class, subclass and detail all as zero in a response. If the very first log/event returned to the software tools contains all zeros, no further logs are read from the controller.
  - *Fix*: Do not copy back unimportant events from persistent memory to event buffer after a controller reset.
  - *Risk*: Low
- Fixed a controller hang problem during high priority rebuild on controller startup.
  - *Root Cause*: During controller startup sequence, there is a corner case where high priority rebuild logic to queisce host I/Os and cache read ahead gets into sequencing problem which results in controller being always in quiesce period rejecting I/Os other than rebuild.
  - *Fix*: Do not quiesce I/O when cache read ahead logic is alive during boot time.
  - *Risk*: Medium
- Fixed a controller hang problem during rebuild with heavy host IOs.
  - *Root Cause*: When rebuild IOs and host IOs overlap into same stripe region, controller gets into a dead lock situation when rebuild reads are failing due to medium errors and degraded performance optimization (DPO) setting is enabled on the controller. In this case, the rebuild code tries to allocate resources after acquiring the stripe lock but host IOs allocate necessary resources but cannot acquire the stripe lock.
  - *Fix*: Allocate all necessary resources ahead of a rebuild iteration before acquiring the lock. In case enough resources are unavailable, defer these rebuilding to next iteration that gives enough time for host IOs to get serviced.
  - *Risk*: Medium
- Fixed a data integrity problem on encrypted volumes when the scatter gather list has more than 512 entries.
  - *Root Cause*: If there are more SGL entries than the perceived capability (that is, 512), the firmware performs one DMA per SGL entry. If there is a SGL entry not aligned with a sector boundary, the controller cannot successfully encrypt data.
  - *Fix*: Even if there are more than 512 entries, perform the whole DMA transfer instead of one per SGL entry.
  - *Risk*: Medium
- Fixed a data integrity issue on RAID 1 volumes other than first volume in the array after a surface scan.

- *Root Cause*: If the volume's starting data offset is not aligned with volume's strip or chunk size (this happens typically when there are more than one logical drives in the same array), surface scan code does not lock necessary stripes. When host IOs fall in the same region, surface scan overwrites the host data.
- *Fix*: Aligned surface scan operations and stripe locks to the volume's strip or chunk size, even when the volume's starting data offset is not a multiple of strip or chunk size.
- *Risk*: Medium

- Fixed 0x27006 controller lockup issue during rebuild with medium errors.
  - *Root Cause*: Firmware code to track the rebuilding status has a strict check on the status bits, failing to which it asserts. In this case, both rebuild read error status and rebuild inconsistent status bits were set.
  - *Fix*: Removed the rebuild inconsistent check functionality. Inconsistency status during rebuild is not necessary since chance of a match between rebuild source and target is almost zero during a normal rebuild scenario.
  - *Risk*: Medium
- Fixed 0x01E30 lockup during logical drive transformation on controllers with no data preservation support.
  - *Root Cause*: Interrupts were disabled when trying to free up the memory buffers that require them to be enabled. These memory buffers are allocated only on controllers that do not support data preservation.
  - *Fix*: Enable the interrupts prior to freeing up memory in the transformation code.
  - *Risk*: Low
- Fixed 0x27006 lockup while handling I/O time outs.
  - *Root Cause*: LUN reset TMF sent during error handling never gets completed after device handle swap logic for SAS devices.
  - *Fix*: Complete all TMFs pending for a device after the device handle swap logic.
  - *Risk*: Low
- Fixed an issue where MCTP EID is set to zero for SMBUS support.
  - *Root Cause*: Firmware checks the EID only for PCIe VDM transport and on servers which supports only SMBUS, this is set always to zero.
  - *Fix*: Get the SMBUS EID if PCIe VDM EID is zero.
  - *Risk*: Low
- Fixed an issue where the drive write cache status is not in expected state for some of the drives.
  - *Root Cause*: Whenever a previous mode sense page for caching fails, the internal bookkeeping field is updated to the fail state. But, whenever this mode sense page for caching succeeds, the internal bookkeeping field is never set to no error. Failure in updating this field when it is cleared resulted in inconsistent behavior for some drives and prevented them from being configured for the drive cache.
  - *Fix*: Clear the internal bookkeeping field, whenever it successfully reads the caching information.
  - *Risk*: Low
- Fixed a lockup issue with a portion of firmware, which does error logging using an invalid address.
  - *Root Cause*: When ECC error is encountered, controller firmware attempts to print a few DDR registers as part of the crash dump for further analysis. Firmware was not reading the correct registers for DDR3 and DDR4 configurations.
  - *Fix*: Modified firmware to check the DDR type before printing debug info and dump the appropriate registers.
  - *Risk*:

- Fixed an issue where controller firmware does not robustly handle a drive behavior in which the drive violates SAS specification.
  - *Root Cause*: A SSD drive reported an NCQ error and completion in the same SDB FIS. That behavior is a SAS protocol violation which triggers two messages to lower level firmware. The first message was a completion with an underrun and firmware acted upon this and returned appropriate completion to the host. The second message for the error, reports that the SAS protocol hardware had set a bit called the Drain bit to block all I/Os. In this case, since firmware had already completed the request, it cannot find any context for the associated request for which this error was generated and drops the message. That action causes the Drain bit in the SAS protocol hardware to be left set and all the subsequent I/Os are flushed back (drained) by the SAS protocol hardware.
  - *Fix*: Modify the lower layer controller firmware to clear the drain bit when a SATA error is generated and if it is not entering NCQ error mode. The Drain bit is needed only for NCQ error mode.
  - *Risk*:
- Fixed a controller lockup during the suspend/resume operation.
  - *Root Cause*: During the suspend/resume operation, the scan devices process requires a large amount of memory space and under rare conditions when memory is not available, the suspend/resume operation is interrupted leading to a controller lockup.
  - *Fix*: During suspend/resume operation, if the memory allocation fails than use other internal memory pools.
  - *Risk*: Low
- Fixed an issue where the expander firmware update fails in dual domain configuration.
  - *Root Cause*: During expander firmware upgrade on dual domain configuration, drives behind the expander gets hot removed or hot added back but the internal firmware drive state does not revert to optimal state, leading to controller hang and expander firmware update failure.
  - *Fix*: The firmware will always add the device path even when the mode is set to disable hot plug events. This change will put a device from single path to multi path. This will in turn avoid queuing requests to the device in the internal queue and make it conducive to LUN resets from the host.
  - *Risk*: Low
- Fixed an issue where the MaxCache logical unit pair changed to 'DDR unsafe' when MaxCache volume and primary volume is changed (deletion/creation) in random order.
  - *Root Cause*: The MaxCache logical unit restoration logic assumes the data offset for its corresponding volume pair incorrectly, causing incorrect status to host tools.
  - *Fix*: Changes were made to MaxCache logical unit restoration logic to calculate the offset for its corresponding volume pair by matching a volume signature.
  - *Risk*: Low
- Fixed a controller lockup when waking up suspended threads during a memory pool freeing context.
  - *Root Cause*: During a memory pool free operation, interrupts are not enabled when waking up the suspended task, causing a controller lockup.
  - *Fix*: Set the correct interrupt bit before waking up task from a memory pool free operation.
  - *Risk*: Low
- Fixed a controller lockup during logical volume migration along with I/O workload.
  - *Root Cause*: A NULL pointer exception occurred when trying to access one of migration related queue using an unfilled logical volume data field.
  - *Fix*: Filled the logical volume data field before accessing it to index into migration related queue.
  - *Risk*: Low
- Fixed an issue where the cache status is reported as OK after 'clear config' operation is executed from host tools.
  - *Root Cause*: Cache status was not getting updated to 'Not Configured' after 'clear config' operation, as the check for the presence of cache enabled volumes was not present.

- *Fix*: Check for the presence of cache enabled volumes and then report appropriate cache status to host tools.
- *Risk*: Low

- Fixed an issue where the logical volume came back during reboot after clearing the logical volume.
  - *Root Cause*: When clear controller config command is used, the error handling mechanism does not check for error conditions and returns success always.
  - *Fix*: Updated error handling mechanism to check for any errors while clearing disk metadata and return appropriate status to host.
  - *Risk*: Low
- Fixed an issue where media read errors are not corrected by surface scan in RAID 1 logical volume.
  - *Root Cause*: When read error occurs on a sector that has media error, the regeneration logic incorrectly checks one of logical reads data structure and returns without correcting.
  - *Fix*: Logic to recover RAID 1's data using redundancy will now check the appropriate correct data structure and regeneration logic will start to fix the read media errors.
  - *Risk*: Low
- Fixed an issue where the SES status page returned to host gets delay up to 60 seconds.
  - *Root Cause*: When host requests SES status pages via Receive diagnostic command, firmware used a 60 second on demand polling mechanism to get the data and hence the delay.
  - *Fix*: An internal non blocking logical request callback mechanism will be used for SES Receive Diagnostic command to complete the request on time.
  - *Risk*: Low
- Fixed a hardware bug. When DRAM hits this bug, the result is DRAM-wide uncorrectable and correctable ECC errors leading to the controller locking up.
  - *Root Cause*: There is a hardware bug in the self-refresh logic in the Winbond DRAMs that are used in the 16 and 24 port SmartHBA 2100 and HBA 1100 adapters.
  - *Fix*: Disable Low-power mode which effectively disables auto-self refresh.
  - *Risk*: Low
- Modified the vendor identification as “Adaptec” instead of “MSCC” for HBA, SmartHBA, and SmartRAID controllers.
  - *Root Cause*: A request was made to have consistent sysfs information reported for the vendor text string.
  - *Fix*: Updated the board configuration data during firmware boot, so the vendor is set to "Adaptec".
  - *Risk*: Low

2.2.2 UEFI Fixes

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

2.2.2.1 Fixes and Enhancements for UEFI Build 1.3.8.2/Legacy BIOS Build 1.3.8.1

This release includes the following Legacy BIOS fixes and enhancements:

- Added support for blank family ID.
  - Root Cause: When the family ID was blank, there were alignment issues in displaying the controller name during the POST.
  - Fix: Corrected the alignment when the family ID is blank.
  - Risk: Low

This release includes the following UEFI fixes and enhancements:

- Added support for drive firmware update using the ATA microcode download method.

- Fix: Option added to update drive firmware using the ATA microcode download method.
- Risk: Low

- Fixed an issue where PCI attribute is not set to support addressing system memory above 4 GiB.
  - Root Cause: UEFI driver does not enable the `EFI_PCI_IO_ATTRIBUTE_DUAL_ADDRESS_CYCLE PCI` attribute in `EFI_DRIVER_BINDING_PROTOCOL.Start()`.
  - Fix: Enabled the `EFI_PCI_IO_ATTRIBUTE_DUAL_ADDRESS_CYCLE PCI` attribute in `EFI_DRIVER_BINDING_PROTOCOL.Start()`.
  - Risk: Low
- Fixed an issue where incorrect driver health message is displayed for error codes 1769 and 1779.
  - Root Cause: Generic message “Drive(s) Disabled due to Write-Back Cache Data Loss.” used for error codes 1769 and 1779 instead of error specific message.
  - Fix: Defined new unicode strings for error codes 1769 with message “Drive(s) Disabled due to Failure During Expansion.” and error code 1779 with message “Logical drive(s) previously failed.”
  - Risk: Low
- Fixed an issue where PollMem function of PCI I/O protocol is taking longer time than expected.
  - Root Cause: PollMem implementation of the platform is taking a longer time than the expected timeout.
  - Fix: Replaced PollMem with event timer functions for calculating command timeout.
  - Risk: Low
- Fixed an issue where unsupported option 128 is provided for cache line size selection while creating maxCache from HII.
  - Root Cause: Missing supported check for cache line size.
  - Fix: Added supported check for providing cache line size options.
  - Risk: Low
- Fixed an issue where the Controller Family ID is displayed as blank in the consolidated display for controller name.
  - Root Cause: White space gets added to the controller name if the family ID is blank.
  - Fix: Ignore family ID if blank while creation of consolidated controller name.
  - Risk: Low
- Fixed an issue where the tape autoloader device reported twice in the disk utilities menu.
  - Root Cause: Device enumeration function consider tape device under disk enumeration as well as non disk enumeration.
  - Fix: Ignore other disk types while enumerating physical drives.
  - Risk: Medium
- Fixed an issue where platform boot hang is observed while booting with a multi lun device connected.
  - Root Cause: Invalid memory access due to mismatch in actual memory allocated and requested size for a SCSI command.
  - Fix: Corrected SCSI request size to match allocated size.
  - Risk: Medium

2.2.3 Driver Fixes

2.2.3.1 Fixes and Enhancements for Linux Driver Build 1.2.8-026

The fixes and enhancements in this release.

- Fixed an issue during the force reboot (reboot -f), with outstanding commands that causes a controller firmware lockup.
  - *Root Cause*: Driver does not drain all commands before issuing a PQI reset.
  - *Fix*:
    - During system shutdown, the driver waits until all the outstanding I/O completes before issuing a PQI reset.
    - During kdump, the driver performs a soft reset of the controller before proceeding with normal driver initialization.
  - Risk: Medium
- Fixed an issue where deleting all logical RAID volumes may result in a call trace being output in the Linux system log file.
  - *Root Cause*: When an "arcconf delete ..." command is executed, the firmware sends multiple event notifications and the driver schedules a rescan operation. If the rescan operation takes longer than 120 seconds, and multiple events/threads are waiting for the rescan to complete, contention for a mutex can occur. This contention causes a call trace and deadlock issue.
  - *Fix*: Use mutex_trylock(), and if there is contention for the mutex, do not wait for rescan completion and schedule a rescan to occur after another 10 seconds.
  - Risk: Low
- Fixed an issue where there is inconsistent systool/sysfs output between SmartPQI and AACRAID drivers.
  - Root Cause: Request for consistent sysfs information to be exposed by the SmartPQI and AACRAID drivers.
  - Fix: Modified/added sysfs entries in such a way that they are consistent. In SmartPQI vendor, model, and serial number is added. The firmware version has been made a separate sysfs attribute.
  - Risk: Low
- Added a module parameter "expose_ld_first" to cause the driver to expose logical drives before physical drives to the OS.
- Added bay and enclosure identifier fields by populating the `pqi_sas_get_enclosure_identifier` and `pqi_sas_get_bay_identifier` routines so that they can be digested by sysfs output.
- Added a module parameter “hide_vsep” to hide the controller VSEP.
- Fixed an issue where CCISS_REGNEWD returns good completion, but device files for physical devices were not created.
  - *Root Cause*: When device scan is going on, if CCISS_REGNEWD is received from the application, driver queues the rescan worker and returns SUCCESS.
  - *Fix*: When the rescan worker is queued, return -EINPROGRESS instead of returning SUCCESS, if a rescan worker is in progress.
  - Risk: Low
- Fixed an issue where driver writes corrupted timestamp to controller log in certain Linux variants.
  - *Root Cause*: When driver is loaded, `pqi_write_current_time_to_host_wellness` is executed. Timestamp is corrupted for the kernel that does not support the `ktime_get_real_seconds` API.
  - *Fix*: Read 32-bit variable through timeval structure for the kernels that do not support the `ktime_get_real_seconds` API.
  - Risk: Low
- Fixed an issue where during the force reboot (reboot -f), there are outstanding commands while processing PQI reset, that causes firmware controller lockup.
  - *Root Cause*: Driver does not block driver initiated RAID path requests before issuing PQI reset.
  - *Fix*: During system shutdown, driver will disable the events and block synchronous commands before issuing a PQI reset.

- Risk: Low

2.2.3.2 Fixes and Enhancements for FreeBSD Driver Build 1.0.3-2323

Following are the fixes and enhancements in this release.

- Added FreeBSD 12.0 support for the SmartPQI driver.
- Fixed an issue when allocating memory using malloc, the tag name and description does not match with the driver name.
  - Root Cause: While allocating memory using malloc, the tag name and description did not match with the driver name.
  - Fix: Replaced the tags SMARTRAID with SMARTPQI when allocating memory using malloc.
- Fixed an issue where a low memory condition causes the system to freeze.
  - Root Cause: The low memory condition was not handled properly in a particular scenario.
  - Fix: Handled the low memory condition by unmapping the request.
- Fixed an issue of controller lockup due to outstanding I/O during PQI reset.
  - Root Cause: Before issuing PQI reset, the host should ensure that there is no pending I/O. Any outstanding I/O during PQI reset will lead to firmware lockup.
  - Fix: Wait for outstanding I/O to finish, before issuing a PQI reset.
- Fixed an issue where inquiry command was failing during device discovery.
  - Root Cause: The firmware was returning an aborted response for inquiry commands and the driver did not retry the commands.
  - Fix: Retry inquiry three times before returning a failure.
- Fixed a PSOD issue that occurs due to incorrect inbound queue selection logic for Task Management requests.
  - Root Cause: The driver was using an incorrect inbound queue to submit a Task Management request for an outstanding command. If that outstanding command had already completed, then the index for the inbound queue would be NULL and that results in a PSOD.
  - Fix: Always use inbound queue 0 for submitting a Task Management request.

2.2.3.3 Fixes and Enhancements for Solaris Driver Build 1.0.3-2323

Following are the fixes and enhancements for this release.

- Added support for Solaris 11.4 in this release.
- Fixed an issue of controller lockup due to outstanding I/O during PQI reset.
  - Root Cause: Before issuing PQI reset, the host should ensure that there is no pending I/O. Any outstanding I/O during PQI reset will lead to firmware lockup.
  - Fix: Wait for outstanding I/O to finish before issuing PQI reset.
- Fixed an issue where inquiry command was failing during device discovery.
  - Root Cause: The firmware was returning an aborted response for inquiry commands and the driver did not retry the commands.
  - Fix: Retry inquiry three times before returning a failure.
- Fixed a PSOD issue that occurs due to incorrect inbound queue selection logic for Task Management requests.
  - Root Cause: The driver was using an incorrect inbound queue to submit a Task Management request for an outstanding command. If that outstanding command had already completed, then the index for the inbound queue would be NULL and that results in a PSOD.
  - Fix: Always use inbound queue 0 for submitting a Task Management request.

- Fixed an issue with warning message after driver Installation: “Signature verification failed for SmartPQI"
  - Root Cause: Unsigned driver binary will show this message when the driver is loaded.
  - Fix: The driver will be signed by the elfsign utility.
- Fixed an issue where the system crashes when the Solaris driver is loaded.
  - Root Cause: While passing the `scsi_device` structure to `scsi_hba_probe()` to get the SCSI inquiry data, the following pointer `dev_info_t *sd_dev` is NULL. Device is still not added to the OS. During the INQUIRY packet completion this pointer is accessed by the upper layer. This created kernel panic in Solaris 11.4.
  - Fix: Instead of calling `scsi_hba_probe()` to get the SCSI inquiry data, now the function calls internal PQI SCSI inquiry function.

2.2.3.4 Fixes and Enhancements for Windows Builds 106.100.0.1014/6.100.0.1014

The fixes and enhancements in this release.

- Fixed an issue where the Windows PNP WHQL tests are failing.
  - Root Cause: Unwanted PQI reset triggering the controller post in SIS mode and due to that all the PNP WHQL test cases failed.
  - Fix: PQI reset is moved out from inappropriate place.
- Fixed an issue where the system freezes during repetition of DC Off/On test.
  - Root Cause: There is a polling routine that may fail if the controller takes too long to respond, causing a potential race condition in MemAlloc.
  - Fix: Rearranged the order of these host commands and made them all use polling instead of three of them using interrupts.
- Fixed an issue where the dump file (Memory.dmp) gets corrupted upon reboot.
  - Root Cause: During the SmartPQI Dump mode, Power SRB is completed before the flush cache completion. This causes the system to reboot before the cache flush completes and eventually data in the dump file does not get written to the drive.
  - Fix: Shutdown SRB is completed after the successful Flush Cache operation.

2.2.3.5 Fixes and Enhancements for VMware Driver Build 1.0.3-2323

This release provides the following fixes and enhancements:

- Fixed an issue where a LUN reset completion with a service response failure results in a PSOD.
  - Root Cause: When a "LUN reset" completed with service response failure, the driver was treating it as a SCSI command completion rather than a Task Management Function (TMF) completion. The driver then tried accessing the SCSI command structure, which is not valid for a LUN reset request and caused the PSOD.
  - Fix: Set the TMF flag for LUN reset request.
- Fixed an issue of controller lockup due to outstanding I/O during PQI reset.
  - Root Cause: Before issuing PQI reset, the host should ensure that there is no pending I/O. Any outstanding I/O during PQI reset will lead to a firmware lockup.
  - Fix: Wait for outstanding I/O to finish before issuing a PQI reset.
- Fixed an issue during device discovery where an inquiry command was failing.
  - Root Cause: The firmware was returning an aborted response for inquiry commands and the driver did not retry the commands.
  - Fix: Retry inquiry three times before returning a failure.
- Fixed a PSOD issue when a TMF request timed out.

- Root Cause: When a TMF request timed out, the driver completed the TMF as failed and cleared the internal request structure for that TMF. If the firmware completed that TMF at a later point in time, the driver will try accessing the internal request structure members and that leads to a page fault.
- Fix: Do not free the internal resources if the TMF request times out.

- Fixed an issue where I/O tags are exhausted when TMF fails.
  - Root Cause: Tags were not freed during TMF failures, which leads to tag exhaustion condition.
  - Fix: Free the tag when a TMF fails.
- Fixed a PSOD issue that occurs due to incorrect inbound queue selection logic for Task Management requests.
  - Root Cause: The driver was using an incorrect inbound queue to submit a Task Management request for an outstanding command. If that outstanding command had already completed, then the index for the inbound queue would be NULL and that results in a PSOD.
  - Fix: Always use inbound queue 0 for submitting a Task Management request.
- Fixed an issue where controller locks up when "`arcconf getconfig`" is executed.
  - Root Cause: The current smartPQI driver has maximum of 10 minutes timeout for all internal and TMF commands. If the firmware does not complete the request within 10 minutes, the driver forcefully completes it and frees up the resource. This resulted in re-using the tags. Also, the driver was issuing more than what the controller's maximum outstanding I/O request was.
  - Fix: Kept infinite timeout for all internal requests (the driver initiates device discovery, cache flush, management requests, and so on) and fixed 10 minutes timeout for the Task Management functionality).
- Fixed an issue where the following warning messages were observed in ESXi 6.7 while creating Raid 0 array.
  - WARNING: ScsiScan: 2329: Skipping out-of-range LUN vmhba3:C1:T0:L512 Max LUN ID supported by the driver is =256
  - WARNING: ScsiScan: 2336: Skipping out-of-range LUN vmhba3:C1:T5:L256 Max LUN ID configured for ESX is =1024
  - Root Cause: ESXi send Report LUN command to the devices with LUN id 0 and parse the response to derive the LUN number. For logical devices, this will generate LUN number greater than 255 and ESXi report it as an "Skipping out-of-range LUN" warning.
  - Fix: The following changes are implemented.
    - Swap target/LUN assignments for logical drives. This will make sure non zero LUN id is assigned for an LD.
    - Expose the controller device entry with non-zero positive LUN number (number should be less than maximum LUNs supported by the driver).
- Fixed an issue where interrupt ACK was not done properly for legacy interrupt.
  - Root Cause: Incorrect implementation of interrupt ACK.
  - Fix: Corrected interrupt ACK.

2.2.4 Management Software Fixes

2.2.4.1 Fixes and Enhancements for Arcconf/maxView Build B23600

This release includes the following fixes and enhancements for arcconf/maxView:

- Added support for Central Management of Systems in maxView.
  - *Root Cause*: Implement feature to have Central Management of Systems in maxView.
  - *Fix*: Implemented the exporting and importing of remote systems to be managed with maxView.
  - *Exposure*: All previous releases

- *Risk*: Low.

- Fixed an issue where arcconf shows incorrect value for SSD I/O BYPASS in SAS SSD Array.
  - *Root Cause*: Array interface type ‘SAS SSD’ was not considered while displaying SSD I/O Bypass which resulted in incorrect value being displayed.
  - *Fix*: Added changes to consider interface type ‘SAS SSD’, while displaying SSD I/O Bypass value.
  - *Exposure*: All previous releases.
  - *Risk*: Low.
- Fixed an issue where maxView does not display the disk partition information under logical drive/resources correctly.
  - *Root Cause*: Datatypes used for size conversion was not holding the correct value for sizes larger than 1TB, which resulted in invalid value for disk partition information.
  - *Fix*: Added changes to use correct data types to ensure disk partition information is displayed.
  - *Exposure*: All previous releases.
  - *Risk*: Low.
- Fixed an issue where SMART stats are not displayed in maxView in Standalone mode.
  - *Root Cause*: maxView in Standalone mode, was referring to the wrong address location for consuming the SMART stats buffer that resulted in a blank page.
  - *Fix*: Added changes to maxView's Standalone mode to refer to the right address location for consuming SMART stats buffer.
  - *Exposure*: All previous releases
  - *Risk*: Low.
- Fixed issue where remote management in maxView is not working when there is no network interface card is present.
  - *Root Cause*: maxView was always looking for local IP address for communication with Redfish server which caused the problem when no network interface card is configured.
  - *Fix*: Added changes to fallback to loop-back address where Redfish is not reachable via local IP address.
  - *Exposure*: All previous releases
  - *Risk*: Low.
- Fixed an issue where Active/Redundant Path Information not updated under runtime in maxView when enclosure is hot-plugged/removed.
  - *Root Cause*: maxView is not updating the configuration of active/redundant path settings when enclosure is hot-plugged/removed.
  - *Fix*: Added changes to generate a refresh event to update the Active/Redundant path information in maxView.
  - *Exposure*: All previous releases
  - *Risk*: Low.

2.3 Limitations

2.3.1 Firmware Limitations

2.3.1.1 Limitations for Firmware Release 2.30 B0

This release includes the following limitations:

- Firmware 2.30 B0 could potentially become unresponsive when a SCSI pass-through commands directed towards a drive/SEP device (or) LUN reset command from driver directed towards a logical drive gets lost in 'Loss of synchronization'/'Loss of Signal'
  - **Workaround**: None
- SATA drives attached to a non-Microsemi expander may get into a failed state when upgrading the controller firmware from previous releases to this release due to the expander not clearing STP affiliation.
  - **Workaround**: Power cycle the expanders to clear the STP affiliation.
- 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
- 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.
- A controller hang might occur when the controller is in Low Power mode and I/Os are running on RAID volumes that are undergoing migration or transformation.
  - **Workaround:** Ensure that the controller is in max performance mode (Default mode) prior to running I/Os, and does not change to Low Power mode.
- When running I/Os on multiple logical drives of various RAID levels, there is a corner-case timing issue where DMA and COMPLETION threads are dead locked and controller triggers 0x1ABD lockup since it does not see requests being completed even after recovery mechanisms.
  - **Workaround**: None
- When I/Os are performed on drives that respond slowly or which do not respond to READ or WRITE commands, and when Secure Erase is performed on other SATA drives, I/Os become stalled for a period of time. The time the I/Os are paused depends directly on the amount of unflushed data in the cache and speed with which the device responds to error recovery.
  - **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**: Try to maintain the same number of logical drives across each controller SAS port.
- Clear controller configuration may not successfully complete when a drive part of maxCache unit is getting reset and maximum supported maxCache units are in use.
  - **Workaround**: None

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 (ESC-2170577),* 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 1.3.8.2 /Legacy BIOS Build 1.3.8.1

There are no known limitations for this release.

2.3.3 Driver Limitations

2.3.3.1 Limitations for Linux Driver Build 1.2.8-026

There are no known limitations for this release.

2.3.3.2 Limitations for Windows Driver Builds 106.100.0.1014

There are no known limitations for this release.

2.3.3.3 Limitations for FreeBSD Driver Build 1.0.3-2323

There are no known limitations for this release.

2.3.3.4 Limitations for Solaris Driver Build 1.0.3-2323

This release includes the following limitations:

- UEFI Secure boot is not supported in this release.
- Deleting a logical drive through arcconf when I/O is running on the logical drive, might still result in the logical drive being listed by the OS when the `# format` or `# echo | format command` is executed. When the deleted logical drive is still listed, the OS API `ndi_devi_offline()` returned with a failure.
  - **Workaround**: None

2.3.3.5 Limitations for VMware Driver 1.0.3-2323

There are no limitations for this release.

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 B23600

There are no known limitations for this release.

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 or newer, 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 18.
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